UNIVERSITAT POLITÈCNICA
DE CATALUNYA
BARCELONATECH

PhD Thesis

Power Aware Resource Allocation and Virtualization Algorithms for 5G Core Networks

Advisor: Dr. Xavier Hesselbach

Author: Khaled Hejja

Dept. Networks Engineering

Universitat Politècnica de Catalunya (UPC)

Barcelona, Spain, 2019

PhD Thesis

Power Aware Resource Allocation and Virtualization Algorithms for 5G Core Networks

A dissertation submitted in partial fulfillment of the requirements for the degree Doctor of Philosophy in Networks Engineering

Advisor: Dr. Xavier Hesselbach

Author: Khaled Hejja

Dept. Networks Engineering

Universitat Politècnica de Catalunya (UPC)

Barcelona, Spain, 2019

In the name of Allah, the most beneficent, the most merciful.

"And my success can only come from Allah. In Him I trust, and unto Him I return." Quran

"I will say of the LORD, He is my refuge and my fortress: my God; in him will I trust." Old Testament

Acknowledgment

I am humbly proud to acknowledge the unlimited support of my advisor, Prof Xavier Hesselbach. This dissertation would not have been completed without his deep thought contributions, invaluable guidance, support, many brain storming session, and continuous advice throughout my PhD training. Without his keen follow up and encouragements, this thesis may have not been finished.

Moreover, I am sincerely and deeply grateful for the generous support and assistance of the Centro de Cooperation para el Desarrollo of Universitat Politecnica de Catalunya for agreeing to cover my whole PhD fees. Also the I am deeply grateful to the unlimited help and guidance from the staff of the administration office of the Network Engineering department at UPC.

Finally, I would like to acknowledge the great help, support, and inspiration I got from my PhD colleagues at the Network Engineering department at UPC. They were my extended family abroad.

Dedication

I owe a great and special gratitudes to my wife, Sahar for her continued and unfailing love, support and understanding during my pursuit of my PhD studies that made the completion of the thesis possible. She was always around at times I thought that it is impossible to continue, and helped me to keep focused and work hard. I greatly value her contributions and deeply appreciate her belief in me. I appreciate my kids Noor and Omar for abiding my ignorance and the patience they showed during my absence, and my parents, mother and father in law, sisters and brothers and family for their prayers and support.

Words would never say how grateful I am to all of you. I consider myself the luckiest in the world to have such a lovely and caring family, standing beside me with their love and unconditional support.

Executive Summary

Solving the resource allocation problem to satisfy all requirements of a virtual network request in the virtual network embedding framework, or a service function chain in network function virtualization framework, is usually divided into two sub-problems that used to be solved with no, or very limited coordination between both of them. The first one deals with allocating virtual nodes onto physical nodes, which is known as virtual node mapping stage, and the other one deals with virtual edge mapping, which embeds virtual edges onto physical paths connecting the corresponding nodes in the physical network. Along such processes, the resource allocation problem usually trades between minimizing embedding costs; through utilizing less physical network resources, or maximizing revenues; through accepting as much as possible virtual network requests or service function chains, while maintaining acceptable quality of services.

However, most of the algorithms that solved the resource allocation problem, used to apply greedy algorithms to select the physical nodes and shortest paths to select the physical edges, which may include non necessary physical resources, but were used just because they were in that shortest path. These additional resources will consume more power and computational processing capacities, as well as cause more delays, and because of the lack or poor coordinations between selecting the physical nodes and edges, that may degrade the overall acceptance ratios and network performance as whole.

Therefore, the main objective of this PhD thesis is to develop power aware resource allocation and virtualization algorithms for 5G core networks, which will be achieved through developing a

virtualization resource allocation technique to perform virtual nodes and edges allocations in full coordination, and on the least physical resources. The algorithms will be general and solve the resource allocation problem for virtual network embedding and network function virtualization frameworks, while minimizing the total consumed power in the physical network, and consider end-to-end delay and migration as new optional features.

The proposed general modeling structure of the power aware resource allocation problem was modeled as an integer linear programming formulation. However, since these are well know NP-Hard problems [Chowdhury (2012)] [Botero (2013a)], this thesis suggested to solve the power aware resource allocation problem through brand new algorithms adopting a new technique called segmentation, which fully coordinates allocating the virtual nodes and edges together, and guarantees to use the very least physical resources to minimize the total power consumption, through consolidating the virtual machines into least number of nodes as much as possible. The proposed general power aware resource allocation and virtualization algorithms, solves virtual network embedding problem for offline and online scenarios, and solves resource allocations for network function virtualization environment for offline, online, and migration scenarios.

The evaluations of the proposed virtual network embedding algorithms were conducted against some of the most referenced algorithms in the literature. Accordingly, the overall results of the offline virtual network embedding algorithm, PaCoVNE, showed that it managed to save considerable amount of the physical network power consumption by 57% in average, through putting idle nodes into sleeping mode, while maintaining high acceptance ratios of the virtual network requests. Similarly, the online algorithm, oPaCoVNE, managed to minimize the average power consumption in the physical network by 23.54% in average. Regarding allocation times of PaCoVNE and oPaCoVNE, they were in the ranges of 20-40 ms, depending on the size of the physical network and virtual network requests. However, when end-to-end delay was factored in, the performance of both algorithms resulted on less saved power and acceptance ratios in comparison to the scenario when delay was not included during the resource allocation process. Suggesting that, introducing end-to-end delay had clear impact on the whole process.

For network function virtualization environment, the evaluations of the proposed offline power aware algorithm to solve the resource allocation problem within the frameworks of NFV, PaNFV, showed that on average it had lower total costs and lower migration cost by 32% and 65.5% respectively, compared to the state-of-art algorithms. In addition to that, when including virtual network functions consolidations together with physical servers consolidations, PaNFV had better total costs and servers' utilization by 27.88% and 38.48%, better than not consolidating the VNFs. Moreover, the evaluations showed that PaNFV allocation times without traffic migrations were faster by 9.59 times than when including migrations, without any significant impacts on the total power consumptions or servers' utilizations.

Regarding the online algorithm, oPaNFV, it managed to allocate the Network Services in average times of 60 ms, and it had very negligible migrations, mainly due to the efficient and careful

allocations of the online algorithm in the first place, suggesting that migrations in the online scenario will most likely be used for maintenance or emergency conditions. Moreover, when demands of service function chains of longer lifetimes were allocated, the overall performance of the online algorithm oPaNFV showed that, the power consumption in the physical network will, on average, increase by 34.2% more than the case of allocating demands of shorter lifetimes, suggesting to consider some trade-offs when allocating the demands. Moreover, when aggressive end-to-end delays where applied, oPaNFV performance in terms of acceptance ratio degraded by 19.3%, suggesting the significant implications of including delay during the resource allocation process in NFV frameworks.

Overall, the proposed power aware allocation algorithm for network function virtualization frameworks without including migrations, managed to significantly reduce the total power consumptions, and proved to be a very reliable choice for allocating networks' services in fractions of a second. For the migration version, they made the allocation algorithms to consume more time without providing any significant impact on reducing the total power consumptions, mainly because the proposed resource allocation and virtualization algorithms, PaNFV and oPaNFV, managed to use the least resources during the allocation process, and therefore the results recommend that migrations would better be used for emergency or maintenance purposes.

Nevertheless, this thesis suggests that future enhancements for the proposed algorithms, need to be focused around modifying the proposed segmentation technique to solve the resource allocation problem for multiple paths, in addition to consider power aware network slicing, especially for mobile edge computing, and modifying new algorithms for application aware resource allocations for very large scale networks. Moreover, future work can modify the segmentation technique and the proposed algorithms, by integrating machine learning techniques for smart traffic and optimal paths prediction, as well as applying machine learning for better energy efficiency, faster load balancing, much accurate resource allocations based on verity of quality of service metrics.

The structure of this thesis includes six chapters. Chapter 1 covers the main objectives of the thesis, and it points to some observations about power efficiency for virtualized physical networks. In addition, it highlights the main research questions, and concludes with summary of the main thesis contributions. Then chapter 2 provides detailed background review and coverage, for the related literature about the virtual network embedding problem, resource allocation problem in network function virtualization frameworks, and about 5G core network technologies, including software defined networks, network function virtualization, and some highlights about main directions for power efficiency in 5G core networks, such as virtual machines consolidations and traffic migrations.

Chapter 3 provides a detailed presentation about the proposed solution methodology for the power aware resource allocation problem, clarifying its general modeling, explaining the proposed segmentations technique to coordinate solving the power aware resource allocation problem, and most important is introducing and analyzing the proposed general power aware resource allocation and virtualization algorithm step-by-step, which will be the main building block for the works in chapters 4 and 5.

Chapter 4 applies the proposed segmentation technique, and the general power aware resource allocation and virtualization algorithm from chapter 3, to solve the virtual network embedding power aware resource allocation for online scenarios. The chapter introduces the problem formulation, explains the proposed new online algorithm, and concludes by showing the simulation results, and it includes evaluations about the proposed algorithms' power consumptions, demands lifetime, network topologies, and end-to-end delay.

Moreover, chapter 5 developed the segmentation technique from chapter 3 to work within the network function virtualization frameworks to solve the power aware resource allocation problem for offline, online and migration scenarios. The chapter explains how to apply the segmentation technique, introduces the offline and online scenarios, modeling, algorithms, and their simulations settings with migrations. Then it concludes by showing the evaluation results for each scenario.

The last chapter in this thesis is chapter 6, which provides a focused summary about the main findings and recommendation for suggested future research.

Contents

List of Figures

List of Tables

Table 1: Glossary of Notation

Notation	Description.
VNE	Virtual Network Embedding.
VNR	Virtual network request.
NFV	Network function virtualization.
SFC	Service function chain.
NS	Network service.
G^P	Directed graph of the physical network.
N^P	Substrate network nodes.
E^P	Substrate network edges.
N^{Sub}	Subset of physical network nodes.
E^{Sub}	Subset of physical network edges.
loc_i	Location of node i.
cpu_i^a	Available CPU capacity at node i.
cpu_i	Consumed CPU capacity at node i
CPU_i	Maximum CPU capacity at node i.
U_i	CPU utilization of node i.
bw_{ij}^a	Available bandwidth capacity of (i, j).
bw_{ij}	Consumed bandwidth capacity of (i, j).
BW_{ij}	Maximum bandwidth capacity of (i, j).
f_{ij}	Throughput flows in (i, j).
d_{ij}^a	Current end-to-end delay in (i, j).
P^P	All basic paths in physical network.
P^{ij}	Basic path of two nodes and one edge.
P^{sub}	Collection of basic paths.
G^V	Virtual network graph.
N^V	Virtual nodes of virtual network graph.
E^V	Virtual edges of virtual network graph.
VNR^r	Virtual network request r.

Table 2: Cont. Glossary of Notation

Notation	Description.
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.
E^r	All virtual edges in SFC^r.
R	Total number of VNRs of SFCs.
t, t^r_a, t^r_e	VNR^r duration, arrival, and expiry times.
loc_u	Preferred location to host virtual node u.
cpu^r_u	Demanded CPU capacity by u.
bw^r_{uv}	Demanded bandwidth capacity by (u, v).
d^r_{uv}	Demanded end-to-end delay in (u, v).
PC_i	Power consumption of i.
PC_i^{Busy}	Maximum power of node i.
PC_i^{idle}	Idle power of node i.
Seg_{ij}	Substrate pair i and j segment.
Seg^r_{uv}	Virtual pair u and v segment.
Adj^{sub}	Substrate set of segments adjacency matrix.
Adj^r	VNR adjacency matrix.
T^r_{down}	Average down time of migrating SFC^r.
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.
d^r	Demanded maximum end-to-end delay by SFC^r.

Table 3: Cont. Glossary of notation

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.
x_i^{ur}	1 if i is hosting virtual node u.
$x_{i^p}^{u}$	1 if i^p is hosting VNF u.
λ_{i^p}	1 if i^p is active.
$mig_{i^p}^{u}$	1 if VNF u hosted by i^p is migrating.
$mig_{ij}^{uv^r}$	1 if (u, v) hosted by (i, j) is migrating.
mig^r	1 if SFC^r is migrating.
$x_{ij}^{uv^r}$	1 if (i, j) is hosting (u, v).

CHAPTER 1

Introduction

Networks virtualization has become an integral component of future Internet, offering network operators a way to overcome ossification of the Internet, by consolidating many of their equipments onto standardized high volume components located at centralized datacenters [NGMN (2015)] [ETSI (2012)] [5GAericas (2018)]. More specifically, key advantageous of network virtualization are basically related to efficiently utilizing physical network resources through sharing them among several virtual networks requests (VNRs) in virtual network embedding (VNE) environment, or among several service function chain (SFCs) requests in network function virtualization resource allocation (RA-NFV) environment, which is a direct application of VNE concepts on the 5G virtualized core networks. In this manner, efficient resource allocations that consider 5G networks' requirements to be much faster and more reliable, with greater capacity and lower response times, will provide more flexibility to manage, expand, or shrink the physical network according to VNRs or SFCs characteristics.

However, allocating enough resources to satisfy all requirements of a VNR or SFC, on top of a physical network that has limited residual capacities, is a challenging task in virtualized networks [Fischer (2013)] [Mijumbi (2016)]. To realize that, VNE and RA-NFV processes are usually divided into two sub-problems, the first one is allocating virtual nodes onto physical nodes, which is known as virtual node mapping stage. The other one, is virtual edge mapping, which embeds virtual edges onto physical paths connecting corresponding nodes in the physical network. Along such processes, VNE or RA-NFV usually trade off between, minimizing embedding costs, through utilizing less physical network resources, and maximizing revenues, through accepting as much as possible VNRs and SFCs, while maintaining acceptable quality of services (QoS). These are proven NP-Hard problems to solve given the size and dynamic behavior of physical network [Herrera and Botero (2016)] [Chowdhury (2012)].

In the following sections, a high level introduction about the overall thesis objectives will be presented, as well as the main motivation behind this thesis, followed by listing the main questions that this thesis will ultimately answer.

1.1 Thesis main objectives

The main objective of this PhD thesis is to:

Develop power aware resource allocation and virtualization algorithms for 5G core networks.

Specifically, the following tasks will be performed to realize the main objective:

1. Define VNE and RA-NFV models and formulate their problems.
2. Develop a resource allocation technique to perform virtual nodes and edges allocations in full coordination on the least physical resources.
3. Develop a general resource allocation and virtualization algorithm to solve the VNE problem using the proposed new power aware resource allocations technique to minimize the total consumed power.
4. Generalize the algorithm to solve the power aware resource allocation problem for 5G virtualized core networks using NFV architecture for offline and online cases.
5. Evaluate impacts of including end-to-end delay as a main resource allocation constraint in the general algorithm.
6. Develop load migration strategy and evaluate its impacts on the performance of the general algorithm.

1.2 Observations about power efficiency for virtualized physical networks

The main building blocks of the physical network of today and future Internet, are composed of physical nodes and their connecting edges. In both, VNE and RA-NFV problems, the prime physical component that consumes most of the power is the physical network's nodes, which are typically datacenters equipped with computing devices, supporting unites, and transmission peripherals, all consuming various amounts of power. Computing examples are servers, switches, and storage, while supporting unites are cooling, lighting and office devices, and transmission unites are the end peripherals connecting nodes to transmission facilities [Google (2019)] [Dayarathna (2016)].

Moreover, literature review showed that virtual machine consolidation strategy is usually applied to save power in the datacenters. It allows virtual machines to be consolidated into a fewer physical servers, thus the idle ones can be shut down or turned into sleeping mode and that will minimize the overall power consumption in the physical network [Varasteh (2017)] [Eramo (2017a)].

In addition to virtual machines consolidations, literature studies considered virtual machines' migration technology in datacenters to aggravate and transfer the allocated VNRs or SFCs on least active virtual machines as much as possible, which will enhance the utilization efficiency of the

physical resources, and help in minimizing the power consumption in datacenters too [Silva (2018)] [Noshy (2018)].

Therefore, the conducted literature review by this thesis highlighted the importance of exploring efficient ways to minimize total power consumption of cloud datacenters, considering the benefits of network virtualization, virtual machines consolidations and migrations, as well as considering the following high-level observations:

1. The power consumption by computing devices of datacenters is mainly dominated by the power consumed by processors, memory, motherboards, etc.
2. The level of the consumed power by the processors, depends heavily on the workloads they handle, which are mostly of linear behavior, giving that more loads on the servers, results on more consumed power.
3. Even if the server is idle, it would still consume a constant amount of power, due to the motherboards, supporting peripherals, and fans working on these servers.
4. Also when the server is active and loaded, yet they were found underutilized, and typically operate at low to medium levels of their maximum capacities.
5. Virtual machines consolidation is usually applied as the main strategy to minimize the power consumption of the datacenters, but that may have a negative effect on the system reliability, and therefore, there will be a crucial trade-off between reliability and power efficiency.
6. Applying online migration strategy can result in performance degradation of the migrated virtual machines, or interrupt their services, and may add more delay on the allocation process as a whole.

1.3 Main research questions

In light of the above observations about the power consumption of datacenters in a virtualized environment, the main motivations behind this thesis are to answer the following questions:

1. Considering VNE and RA-NFV problems, what is the optimal formulation to minimize the total power consumption in the physical network?
2. How to design a power aware algorithm that includes full coordinated strategy to allocate resources for virtual nodes and edges at the same time?
3. How to identify idle servers, and how to apply virtual machines consolidation to eliminate idle power consumption in datacenters for offline and online scenarios?
4. How to validate the suggestion of turning off idle servers to minimize total power consumption within the frameworks of VNE and RA-NFV environments?

5. Compared to other algorithms in literature, how efficient and credible are the proposed power aware algorithms in identifying and consolidating low utilized servers to minimize total power consumption in the datacenter?
6. How to integrate and validate load migration strategy for virtualized environments?
7. Considering end-to-end delay as recommended for 5G systems, what are the impacts of including end-to-end delay on the allocation process, and how that would affect the extend of minimizing total power consumption?
8. Live migration can result in performance degradation of the migrated virtual machines, or even interrupting their services. Accordingly, which is most efficient: to apply live migrations, or limit it for emergency and maintenance applications?

1.4 Main contributions

1. Khaled Hejja, Xavier Hesselbach, "Evaluating Impacts of Traffic Migration and Virtual Network Functions consolidation on Power Aware Resource Allocation Algorithms," Journal of Future Generation Computer Systems (Impact Factor: 4.639), Elsevier, 2019. (SUBMITTED 13th-Feb-2019, first round revision was done, and expecting final decision during the writing of this thesis)

   a- This paper proposes power aware resource allocation and virtualization algorithm to solve RA-NFV problem in fractions of a second. The core part of the allocation algorithm is based on a modified version of the precise path construction and allocation strategy developed by the authors recently.

   b- The new algorithm supports physical servers' consolidations to minimize total costs in datacenters.

   c- As an optional feature, the new algorithm can support virtual network functions consolidations together with physical servers' consolidations to further reduce the operational and power consumption costs.

   d- 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.

2. Khaled Hejja, Xavier Hesselbach, "Offline and Online Power Aware Resource Allocation Algorithms with Migration and Delay Constraints," Journal of Computer Networks (Impact factor: 2.522), Elsevier, 2019. [Hejja (2019)].

   a- This paper proposes new offline and online power aware algorithms to solve RA-NFV problem in fractions of a second.

b- Each algorithm is integrated with a migration strategy.

c- To the best of our knowledge, this is the first time to include both migrations and delay in an online power aware algorithm to solve the resource allocation problem.

d- All algorithms used a new path construction methodology that facilitates allocating precise resources for the virtual nodes and edges, together and in full coordination, on the physical network.

3. Khaled Hejja, Xavier Hesselbach. "Power aware coordinated virtual network embedding with 5G delay constraint," Journal of Network and Computer Applications, Elsevier (Impact factor: 3.991), 2018. [Hejja (2018)]

   a- Impacts of end-to-end delay on the performance of the suggested online algorithm, oPaCoVNE, were deeply analyzed, representing direct application for virtualization in future 5G networks.

   b- To minimize the total power consumption in the whole physical network, segmentation approach was proposed to guarantee coordinating the embeddings of the virtual nodes and edges, while utilizing the least physical resources to the minimum.

4. Khaled Hejja, Xavier Hesselbach, "PaCoVNE: Power Consumption Aware Coordinated VNE with Delay Constraints," Polytechnic University of Valencia Congress, XIII Jornadas de Ingenieria Telematica - JITEL2017, 2017. [Hejja (2017)]

CHAPTER 2

Background and Literature Review

2.1 Introduction

ITU's visions for future 5G technologies incorporated the support and integration of software defined networking and network function virtualization technologies, in order to realize the promises of 5G technologies [ITUSDN3300 (2014)] [ITU-T T-REC-Y.3101 (2018)]. Great efforts are ongoing through many organizations, network operators and vendors, all collaborating together in formulating the required standards to allow for smooth integration between the technologies.

This chapter provides an introductory review about 5G technologies, introducing software defined networking, network function virtualization, and 5G technologies. Then the chapter summarizes the most related literature about resource allocations in virtual network embedding and network function virtualization, including the studies that focused on power aware algorithms, virtual machines consolidations and migrations.

2.2 Software defined networks

SDN decouples the control plane that defines the traffic routing and network topology, from the data plane which physically handles the traffic based on the configurations supplied from the control plane. Accordingly it provides a set of techniques enabling network operators to directly program, orchestrate, control and manage network resources to facilitate design, delivery and operation of network services, benefiting from the dynamic and scalable manner of the SDN architecture. This is done in SDN by relocating the control of network resources to a dedicated network element, called the controller, which provides an abstracted view of the network resources, and also provides means to program, orchestrate, control and manage the network resources through software [ONF (2012)] [SDNsurvey (2015)].

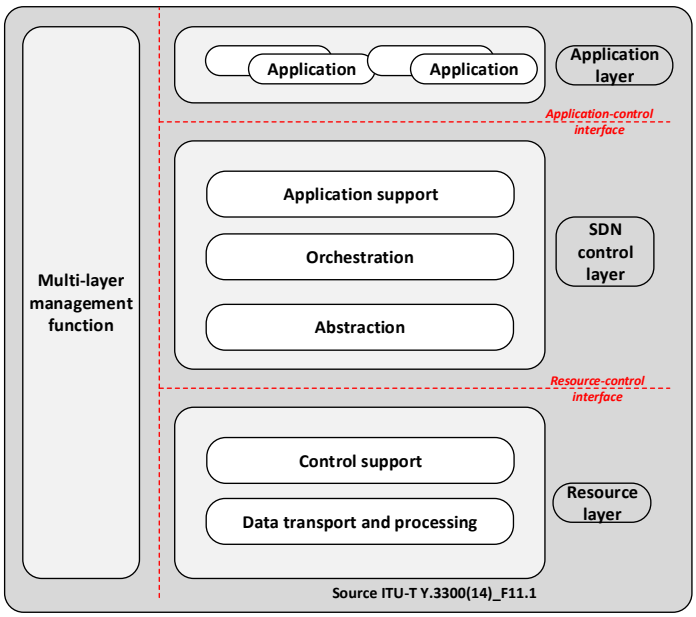

Figure 2.1: High-level architecture of SDN

2.2.1 High-level architecture of SDN

As shown in Fig.2.1 [ITUSDN3300 (2014)], the overall structure of an SDN architecture is composed of four main parts to be explained in brief as follows:

Application layer The application layer characterizes the required network services, in a software code style, which will guide SDN controller to customize the network resources according to the coded instructions inside the software.

SDN control layer The SDN control layer provides means to control network resources' behavior according to requirements of application layer. The main component in the control layer is the orchestration block, which follows the policies set by the management function to control and manage all network resources, and to coordinate the requests from the application layer to the network resources.

Resource layer This is the layer where network elements perform the transport and the processing of data packets according to the decisions made by the SDN control layer.

Multi-layer management functions This layer provides functionalities, such as, supporting fault, configuration, accounting, performance and security while managing all other layers.

2.3 Network functions virtualization (NFV)

In general terms, network function virtualization envisions to virtualize all network services that are usually carried out by vendor specific proprietary and dedicated hardware. More formally, based on the descriptions of ETSI [ETSI (2013c)] [ETSI (2014a)], network function virtualization is a principle that uses virtual hardware abstraction technologies to separate the functional blocks of the network infrastructure, such as network nodes or physical appliances, from the hardware components they run on them.

Therefore, the totality of hardware and software components offered by NFV, builds up the environment in which virtual network functions (VNFs), representing network services, are deployed. Thus, transforming the way network operators architect networks, by evolving standard IT virtualization technology to consolidate many network equipment types onto industry standard high volume servers, switches, and storage. These could be located in variety of NFV infrastructure datacenters, network nodes, and at end user premises [ETSI (2012)] [ETSI (2013a)].

2.3.1 NFV Architecture

In general, there are three main building blocks formulating the overall architecture of the NFV according to ETSI [ETSI (2013a)] as shown in Fig.2.2, which provides high-level architecture of the NFV framework.

Virtual network functions block which contains a set of software defined VNFs in a pool that can be chained and structured according to requested service function chain (SFC) [ETSI (2013a)].

NFV infrastructure block This is where the resource allocation takes place, it consists of hardware and software physical resources, called NFV infrastructure (NFVI), which will host the chained VNFs via a forwarding graph, that logically connects a set of softwarized virtual machines (VM), representing computing, storage, and networking resources of the physical infrastructure [ETSI (2013a)]. Forwarding graphs will define the connectivity between VMs based on the demands of SFC [ETSI (2013c)]. The virtualization process will be carried out through the algorithms residing within the virtualization layer, which will map each component of the forwarding graph onto the corresponding physical resources, physical compute, storage, and network infrastructure, as shown at the bottom of Fig.2.2.

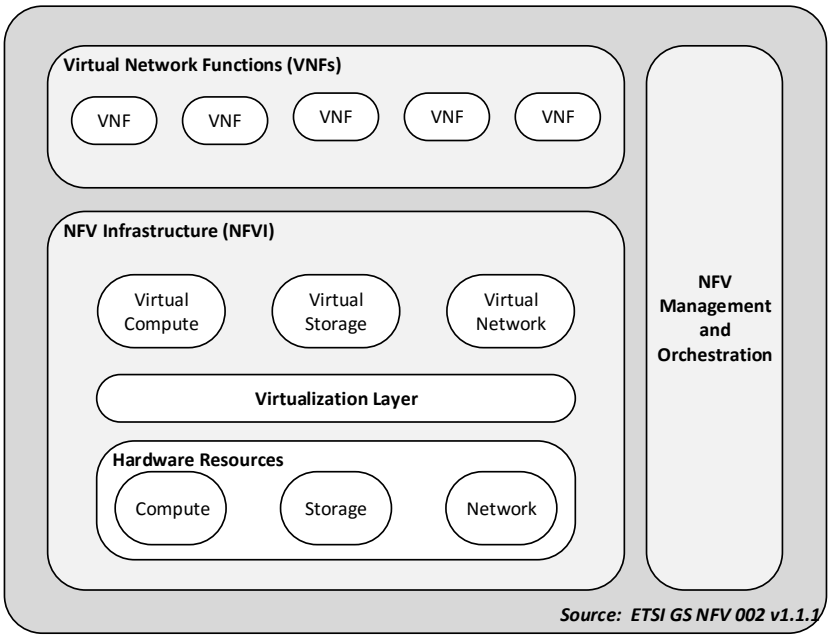

Figure 2.2: High-level architecture of NFV Framework

Management and orchestration (MANO) block It provides all necessary controls to handle orchestration tasks between physical and virtual resources. Basically, it manages the NFVIs through specific algorithms, and it orchestrates the allocation of resources requested by SFCs. In addition to that, the MANO handles the traditional networking management tasks, such as, fault, configuration, accounting, performance, and security management tasks [ETSI (2013a)].

2.3.2 Main benefits of NFV

The capability of NFV to separate software that defines the VNF from the generic hosting hardware infrastructure, which executes the VNF, helps network operators in the reassignment and sharing of the infrastructure resources in a much flexible way at anytime. This in turn would provide more leverage for network operators, in terms of time, to deploy and maintain new services on the same physical resources. Also would give wide possibilities to dynamically scale the actual performance of VNFs according to verity of metrics, such as traffic loads, power consumption, service delivery time, and others [Herrera and Botero (2016)] [Mijumbi (2016)].

2.4 5G core network technologies

As an overall view, 5G promises to provide users moving with high mobility and living on very dense areas with high quality services similar to the fixed networking technologies, with very minimal delay levels [ITU-T T-REC-Y.3101 (2018)] [NGMN (2015)]. Therefore, 5G will be key enabler for the Internet of Things (IoT), providing a platform to connect massive number of sensors, in addition, it will natively support mission critical services requiring very high reliability, global coverage, and very low latency.

Moreover, a detailed specifications and architecture of 5G are in progress, nevertheless, according to a recently released ITU specifications document [ITU-T T-REC-Y.3101 (2018)], 5G should be able to serve peak data rates, up to 20 Gbps, in the downlink of a serving network node, up to 100 Mbps, as downlink user experienced data rate, and provide user plan latency of 1 ms for ultra critical services, 20 ms control plan latency, and power efficient capabilities.

More broadly, 5G supposed to benefit from integrating networking, computing and storage resources into one programmable and unified infrastructure, through virtualization technologies, thus allowing the usage of all distributed resources, as well as facilitating convergence of fixed, mobile and broadcast services [5G (2017a)] [TR 21.915 3GPP (2019)].

2.4.1 5G and software defined networks

SDN is reshaping the network architecture to support the requirements of the forthcoming 5G ecosystem, by providing a dynamic framework for 5G networks' programmability to facilitate fixable design, build-up, and management of 5G networks. In particular, the 5G SDN based architecture separates the network's control and forwarding plans, makes the network control plan fully programmable, and the underlying infrastructure to be abstracted for applications and network services seamlessly. This will transform the future 5G architecture to be highly manageable and cost-effective [5GAericas (2018)].

Moreover, to provide far more efficient data flows as data traverses the 5G network, SDN controllers are designed to dynamically route traffic flows using both source and destination address information, which will improve end-to-end speeds and latency in 5G networks. Furthermore, since the SDN controller has end-to-end network visibility, that well allow it to lower network bandwidth requirements and overcome potential bottlenecks by intelligently recalculating data flow routes much faster and more efficiently, as well as allowing for seamless scalability and dynamic provisioning to expand the network, or add capacity to existing areas automatically [ETSISDN (2017)].

2.4.2 5G and network function virtualization

Network functions virtualization is a key enabler of the coming 5G infrastructure, where networks will be programmed in real time to introduce new services quickly based on service needs. It will help

to virtualize all the various appliances in 5G networks, and will lead to great benefits in bandwidth, mobility, resiliency, security, availability, and network coverage. Moreover, NFV will enable network slicing, which allows multiple virtual networks to be created atop a shared physical infrastructure, in addition to customizing these virtual networks to meet the needs of applications, services, devices, customers or operators, as well as offering the flexibility to allocate speed, capacity, and coverage in logical slices according to the demands of various use cases such as, mobile and fixed virtual network operations, Internet of things networks, automated cars infrastructure, content delivery networks, remote health care networks, and so on [ETSI (2013a)][ETSI (2013c)].

An additional advantage for the network function Virtualisation in 5G is the distributed cloud technology, which allows multiple datacenters to appear as a single, virtual data center, so that they will be more scalable, much resilient, smoothly optimized and manageable, fault-tolerant, and cost effectively maintainable.

2.4.3 5G and power efficiency

High power efficiency is a critical requirement of 5G systems, since it will facilitate reducing the total network's costs, as well as providing a sustainable and more resource efficient way when accessing 5G systems. Accordingly, one of the design principles that were identified for 5G networks as in [ITU-T T-REC-Y.3101 (2018)] and [ITU-T Focus Group (2017)], highlighted that the 5G-Management Plan will have a dedicated functionality that will provide power efficient operations of virtual and physical resources used for the implementation of all 5G architecture plans. The functionality can be realized by power-aware algorithms supported by resource status monitoring and analytics.

Focusing on the power efficiency of core network nodes located in datacenters, they used to handle and manage enormous amounts of aggregated traffic and users from a large number of access nodes, hence, the need for power saving mechanisms in these core network nodes as the load changes due to the higher degree of aggregation, are required to be cost efficient, flexible, adaptable and scalable.

In light of that, virtualized network functionality and cloud technologies are important tools when designing 5G systems, which will allow for better infrastructure scaling and less computational redundancy, and therefore improve the power consumption footprint. The management plan in the virtualized network of 5G systems will ensure that network resources are available when required, and non utilized resources remain in optimal power-saving modes. In this sense, the management plan allows the 5G systems to consolidate the running network services on a subset of the available network resources, which will run with higher utilization, while the rest can be turned off or put in sleeping mode. This in turn should enable more efficient utilization of the active physical resources, which leads to higher power performance during low load, while providing possibilities to quickly scale up or down to handle peak usage [ITU-T Focus Group (2017)].

2.4.4 Studies about virtualization for 5G

Virtualization and its customized version, the network slicing technology promising to revolutionize the fifth generation systems, providing a way of computing to allow Software-as-a-Service, Platform-as-a-Service, and Infrastructure-as-a-Service. The proliferation of these services among the users led to the establishment of large scale cloud datacenters that consume an enormous amount of power, resulting into high costs and carbon footprint. To touch on the recent trends, the following paragraphs summarize some of the most related studies about network virtualization and slicing in future generation 5G systems.

The article by [Adamuz-Hinojosa (2018)] provided an overview about the automation of the network service scaling operation in NFV, addressing the options and boundaries introduced by ETSI normative specifications. They described ETSI's reference NFV framework structure, focusing on how instantiation levels are constructed, then they discussed the different scaling procedures of the NFV framework, and concluded by proposing a detailed scaling procedure, clarifying the interactions and information exchanges between the functional blocks in the NFV framework when performing the scaling operation. Another survey study was conducted by [Sahrish Khan (2019)], which presented a survey on state of the art on the 5G integration with the SDN. The authors presented the different integrated architectures of 5G cellular network based on SDN and NFV, then they focused on the methods and techniques adopted for resource allocation, and highlighted the role of virtualization and provided an analysis of abstraction for resource allocation in SDN based cellular network.

Furthermore, the authors in [Lucena (2018)] presented an SDN/NFV architecture with multi-tenancy support which will enable a network slice provider to deploy network slice instances for multiple tenants on-the-fly, and simultaneously provision them with isolation guarantees. They addressed the mapping from high-level service requirements to network functions and infrastructure requirements, as well as the admission control and the specific information a network slice descriptor should have. Likewise, in [Ordonez (2017)], the authors presented the network slicing concept, focusing on its application to 5G systems. They summarized the key aspects that enable the realization of network slices, followed by a brief overview on the SDN architecture, and analyzed ETSI proposal that incorporated the capabilities of SDN into the NFV architecture. Finally, they presented an example scenario which combined SDN and NFV technologies to address the realization of network slices.

In[Habiba (2018)], the authors focused on the auction theory as a fundamental tool for designing business models for virtualization of 5G cellular networks in particular. They described the basic principles and solution approaches in auction theory for heterogeneous and multi-commodity scenarios, and concluded by outlining the open challenges and future research directions related to applications of auctions in 5G cellular networks' virtualization. In addition to that, the authors in [Verma (2018)] proposed three algorithm models utilizing the cost of the running cloud datacenter, cost of the optimization problem, and the resource utilization optimization problem. They developed a power consumption model for cloud computing environment focusing on linear relationship between power consumption and resource utilization, and they considered a virtual machine migration technique as

well.

Moreover, in the paper of [Xie (2018)], the authors considered a hierarchical caching architecture that allows core network and radio access network (RAN) to share their caching resources among mobile virtual network operators. They decomposed the optimization problem into two independent caching resource sharing problems in RAN and core network, respectively. The formulated optimization problem was designed as a competitive game targeting maximizing the expected revenues in the core and radio networks. In similar direction, [Abhishek (2018)] presented a framework for network survivability in next generation 5G networks. The authors proposed a 5G network architecture with network virtualization with multiple providers, in addition to that, they proposed a self-organizing ad-hoc network among the macro-sites that may use another provider for network resilience when the aggregation and back-haul networks fail. Finally, they presented an optimization formulation for network survivability enforcing a trade-off between using provider's own network or rely on auxiliary capacity from another provide.

Authors in [Massimo (2018)] analyzed and summarized the role of softwarization and virtualization paradigms, in order to derive and classify the requirements to be taken into account in the design process of 5G networks. The authors provided an overview on the recent advances by standardization bodies in considering the role of softwarization and virtualization in the next-to-come mobile systems, and they surveyed the recent proposals exploiting softwarization and virtualization for the network design and functionality implementation of 5G networks. In [Bordel (2018)], the authors presented a virtualization-based technique for the design, management and implementation of future 5G systems with network slicing. The proposed technique employs virtualization technologies such as Docker or Kubernetes in order to create, coordinate and manage slices, services and functional components in future 5G networks, and the authors in [Fendt (2018)] formalized a mathematical model for solving the offline network slice embedding problem as a standardized mixed integer linear program. They designed their objective function to be latency sensitive in order to guarantee the optimal network utilization as well as minimize latency in the network slice communication.

In [Agarwal (2019)], the authors formulated a queuing-based model and use it at the network orchestrator, to optimally match the network service clients' demands in an automated manner. They mapped their requirements into decisions concerning the network infrastructure, including VNF placement, resource assignment, and traffic routing to the available system resources. Another resource allocation model for 5G virtualized networks in a heterogeneous cloud infrastructure was proposed in [Halabian (2019)]. In the model, the author considered a system of collaborative slices and formulated the resource allocation as a convex optimization problem, maximizing the overall system utility function. Furthermore, the author introduced a distributed solution for the resource allocation problem by forming a resource auction between the slices and the datacenters.

In similar fashion, the authors in [Martin (2018)] provided a network resource allocator system which enables autonomous network management aware of quality of experience (QoE). The system predicts demand to foresee the amount of network resources to be allocated and the topology setup required

to cope with the traffic demand. Additionally, the authors in [Nasrin (2016)] discussed many of the implemented power aware resource allocation techniques for cloud environments, and presented a comprehensive review on the different power aware resource allocation and selection algorithms for virtual machines in the cloud.

A comprehensive overview about cluster frameworks was presented in [Wang (2018)], which were developed for data-intensive applications in datacenter networks to improve scheduling strategies and resource allocation algorithms. The authors observed that algorithms in cluster frameworks used to be implemented with different guidelines and can be classified according to scheduling granularity, controller management, and prior-knowledge requirement. In addition, the authors discussed the mechanisms for conquering crucial challenges in datacenter networks, including providing low latency and minimizing job completion time. Moreover, they analyzed the desirable properties of fault tolerance and scalability to illuminate the design principles of distributed systems.

Moreover, the authors in [Huaqing (2018)] studied the resource allocation problem when multiple datacenter operators and service subscribers coexist in the networks, and they modeled a multi-leader multi-follower hierarchical game in order to analyze the relationships among multiple datacenters and the subscribers, and in [Haneet (2019)] a spectrum sharing model has been proposed with an optimum resource allocation strategy to optimize the power and achieve high QoS using spectrum sharing for the next generation networks. The authors used Hidden Markov Model for resource allocation to achieve high spectrum efficiency and power efficiency targeting urban deployment scenarios in 5G networks. Finally, the authors in [Li (2019)] proposed a latency-optimal virtual resource allocation problem subject to the back-haul capacity and bandwidth constraints. The problem was formulated as an integer linear programming (ILP), and was solved with the branch-and-bound scheme. The authors characterized the latency by jointly taking into account the network capacity, datacenters' locations and link bandwidth. Furthermore, they employed a node importance metric to analyze the datacenters' availability and priority in the physical network.

Table.2.1 summarizes some of these recent studies.

2.5 Big picture connecting SDN, NFV and 5G

The big picture connecting the three technologies SDN, NFV and 5G, is shown in Fig.2.3. It shows in high-level the relationship between various supporting organizations, who are collaborating together towards developing an interworking system that includes SDN, NFV, and 5G technologies. Following paragraphs present the main defining components leading the way to integrate the three technologies together [ITUSDN3300 (2014)] [ETSI (2013a)]. .

Table 2.1: Studies about virtualization for 5G

Paper	Summary
[Adamuz-Hinojosa (2018)]	Provided an overview about the automation of the network service scaling operation in NFV.
[Lucena (2018)]	Presented an SDN/NFV architecture with multi-tenancy support.
[Ordonez (2017)]	Presented the network slicing concept, focusing on its application to 5G systems.
[Habiba (2018)]	Focused on the auction theory for virtualization of 5G cellular networks.
[Verma (2018)]	Proposed an algorithm utilizing the cost of datacenter, including resource utilization problem.
[Xie (2018)]	Considered a caching architecture so core and radio networks share resources.
[Abhishek (2018)]	Presented a framework for network survivability in next generation 5G networks.
[Massimo (2018)]	Analyzed softwarization and virtualization in 5G networks.
[Bordel (2018)]	Presented a technique for future 5G systems with network slicing.
[Fendt (2018)]	Formalized model for solving offline slice embedding problem.
[Agarwal (2019)]	Formulated a model to match service demands automatically.
[Halabian (2019)]	Considered collaborative slices to maximize overall utility function.
[Martin (2018)]	Provided a resource allocation and network management model considering QoE.
[Nasrin (2016)]	Discussed power aware resource allocation techniques for cloud environments.
[Wang (2018)]	Surveyed cluster frameworks developed for datacenter networks.
[Huaqing (2018)]	Studied resource allocation problem for multiple datacenters and services.
[Haneet (2019)]	Proposed a resource allocation strategy for next generation networks.
[Li (2019)]	Proposed a latency-optimal resource allocation problem subject to capacity and bandwidth.

Operating system for cloud

It works at the open northbound interface, mediating between the application layer of virtual network functions and the control layer. Typical example is the OpenStack cloud software [Openstack (2019)], which helps in controlling a pool of storage, compute and networking resources in a virtualized data center.

Controller

Which includes a set of common protocols for command and control of the physical hardware within the network. As an example, OpenDaylight platform is a typical controller, that enables network programmability, as well as adding intelligence and adaptability [Opendaylight (2019)].

Flow protocol

It works at the southbound interface, enabling the controller to determine how packets will travel through a network of software and routers. Typical example is the OpenFlow protocol [ONF (2012)], which enables network controllers to determine the path for network packets across a network of distinct switches.

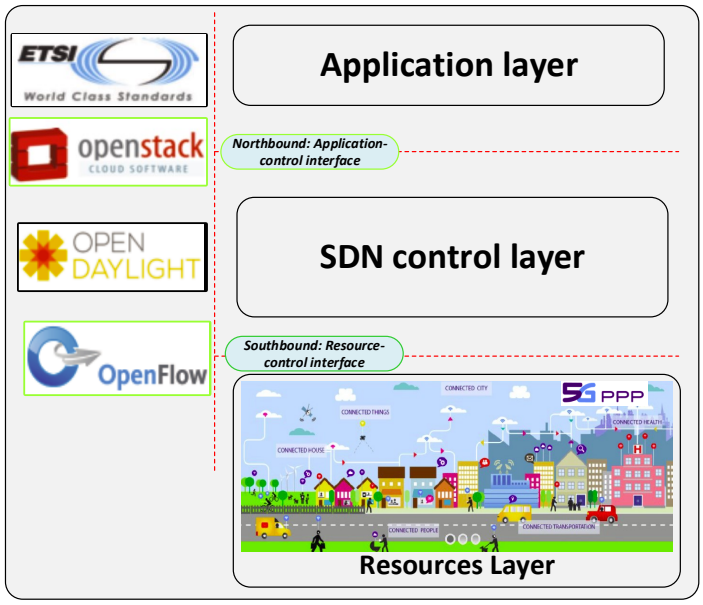

Figure 2.3: High-level relationships between SDN-NFV-5G

2.6 Introduction to virtual network embedding

Current Internet business model is based on Internet service providers (ISP) owning and operating the Internet infrastructure, while network vendors are usually responsible of developing, upgrading, and most of the time maintaining these networks. As such, ISP infrastructure is a vendor specific proprietary, built as silos of hardware entities, run by a proprietary software, mostly rigid, lacks flexibility, and usually get stuck due to limited cooperation between the various infrastructure stockholders of the Internet. Therefore, each time current ISPs plan to introduce new Internet services that require radical changes to the Internet, they suffer from time and operational difficulties. In addition, ISP will be forced to spend more on CAPEX and OPEX, which could be beyond their limited budget capabilities. This tendency is called the ossification of the Internet, reflecting rigidity to change the current Internet architecture as well as the business relationships among various stakeholders [ETSI (2012)].

To overcome these limitations, network virtualization (NV) revealed new sights of a new Internet business model spreading the rule of the current ISP among the following new players: service providers (SP), virtual network operator (VNO), virtual network provider (VNP), and infrastructure provider (InP) [ETSI (2012)]. Fig.2.4 clarifies the rule of each of them and the handshaking between them. The core property to realize NV, is the virtual network embedding, where virtual nodes are interconnected by virtual links, forming a virtual topology, which is then embedded on the physical network utilizing its physical nodes and links resources. However, the problem of embedding virtual networks in a physical network is the main resource allocation challenge in network virtualization,

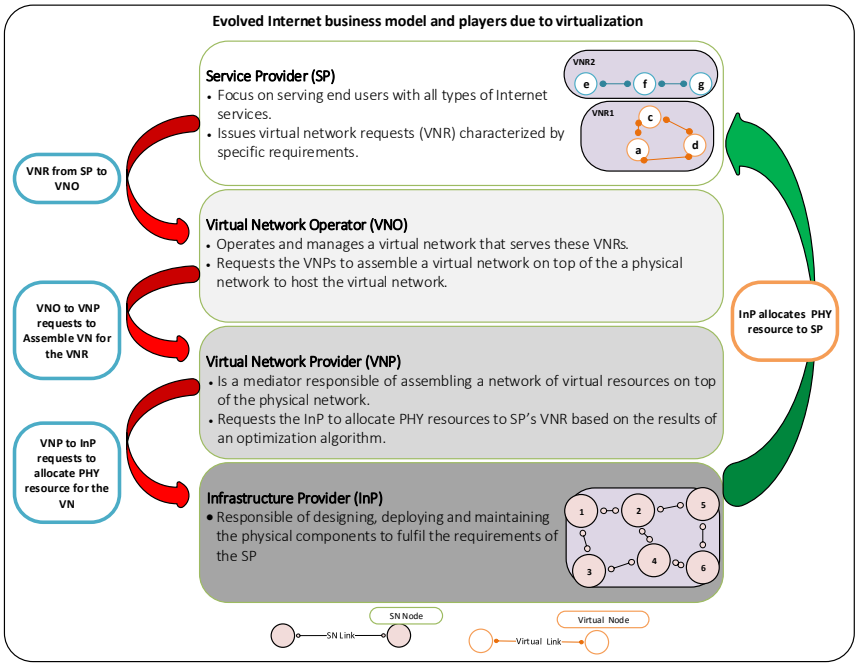

Figure 2.4: Internet business model

giving that for the embedding to happen, dynamic mapping and optimal dynamic resources allocation are required, optimality through an embedding algorithm that balances a successful embedding against verity of different objectives, such as quality of service, quality of network economics (QoNE), quality of power (QoEn) for example. This means that virtual resources are mapped to candidate physical resources, only if all virtual resources can be mapped, then the entire network is then embedded and physical resources are actually spent [ETSI (2014b)] [ETSI (2016)] [Fischer (2013)].

2.6.1 VNE taxonomy

Depending on the scenario, modification and relocation of virtual resources may be necessary. Accordingly, VNE approaches, either have to be static with unchanging infrastructures, or dynamic taking changes in virtual and physical infrastructure into account. In addition, VNE could be distributed, where virtual network might be spread over the physical infrastructure of multiple entities contributing to the mapping, or centralized, where VN might be centralized at one InP, as the only entity contributing to the mapping. Finally, VNE could also be performed in a concise manner minimizing the usage of the physical resources, or redundant, combining multiple physical resources to realize the virtual services [Fischer (2013)].

2.6.2 VNE parameters

VN nodes and links are characterized by individual capacities and qualities, which could be linear parameters requiring linear programming (LP) techniques to find the optimal solution, or nonlinear parameters, which are much harder to solve since LP cannot be used to solve them. Moreover, node and link parameters could be node related such as processing power and link related such as bandwidth and delay, and they could be, primary parameters directly assigned to a physical network resource so they can be explicitly required by a VNR, or secondary parameters depending on other primary parameters, which have to be calculated in advance. In addition, node and link parameters could be consumable resources when the virtual entity is mapped, and these are hard to match, or they could be static resources that do not depend on the number of virtual resources mapped. Finally, nodes and links parameters could be functional parameters to specify low-level functionality such as processing power and bandwidth, or nonfunctional parameters representing high level properties of respective entities such as security or resilience of an entity [Fischer (2013)] [Chowdhury (2012)].

2.6.3 VNE embedding objectives

In order to solve nodes and links mappings, VNE applies an objective criterion, then try to minimize or maximize it. For example, embedding based on certain QoS metrics such as enough bandwidth, low delay, and high processing power, or embedding based on maximizing the economical profit of the InP during embedding process to maximize their economic benefits when accepting VNs. Moreover, other embedding objectives could be based on providing survivability and resiliency, in terms of VNE guaranteeing integrating fall-back resources within the physical network for all or some of the nodes and links in case of failure [Fischer (2013)] [Chowdhury (2012)].

2.6.4 Solving VNE problems

Solving the VNE problem is usually performed either in uncoordinated manner, by mapping the virtual nodes first, then find the optimal paths to embed the virtual links, or mapping virtual nodes first while consequently coordinating mapping the virtual links afterwards.

In the uncoordinated strategy, VNE algorithms used to follow greedy algorithms to perform nodes' mapping, which chooses for each virtual node with bigger demands a set of eligible physical network nodes with bigger resources and assigns one of them based on its amount of available resources. For virtual link mapping, there could be two different ways: using k-shortest path solution, which requires that each virtual link is mapped just to a single path in the physical network, or multiple path mapping, where each virtual link demand can be carried by several paths in the physical network. In this later case the problem is reduced to the Multi-commodity Flow Problem (MCF) that provides a multi-path routing solution for each virtual link using optimal LP algorithm [Kolliopoulos (1997)] [Ilhem (2012)]. However, the lack of coordination between nodes and links mapping stages could

result on allocating nodes in a further away separated locations in the physical network topology, thus impacting acceptance ratio, and therefore low revenue [Fischer (2013)].

On other hand, in coordinated VNE strategy, coordination can be done in two stages, either by providing a virtual node mapping solution that optimizes the virtual link mapping, or can be done in one stage by solving them at the same time, giving that virtual link is mapped when the first pair of virtual nodes are mapped, which are then connected by a mapped virtual link [Chowdhury (2012)] [Botero (2013b)].

However, regardless of the strategy used to solve the VNE and due to the virtual links mapping stage, solving VNE problem is NP-hard, as it is related to the multi-way separator problem. Even for small number of nodes, the problem of optimally allocating a set of virtual links to single physical paths reduces to the un-splittable flow problem, which is also an NP-hard [Kolliopoulos (1997)] [Ilhem (2012)]. Therefore, for large problem sizes that has large physical networks and large virtual network request sizes, the time to find the optimal solution becomes critical issue. Nevertheless, truly optimal solutions can only be gained for small problems at the best, which lead the research community to solve the VNE problems through suggesting algorithms or meta-algorithms approaches [Fischer (2013)].

Algorithm-based solutions try to find a good solution while keeping execution time low, while meta-algorithms such as simulated annealing, genetic algorithms, ant colony optimization, particle swarm optimization or tabu search can be used to find near optimal solution by trying to improve candidate solution with regard to a given measure of quality [Fischer (2013)] [Chowdhury (2012)] [Botero (2013b)] [Mijumbi (2015)].

2.6.5 VNE metrics

There are variety of metrics that could be used to measure the behavior of VNE process, such as QoS metrics, resource spending metrics, resilience metrics, and others [ETSI (2014b)] [ETSI (2016)]. Fig.2.5 summarizes the most common used VNE metrics in literature.

QoS metrics these are used to evaluate the impact of the embedding process on the services delivered to end users by the virtual network. Typical examples are path length, stress level, utilization, throughput, delay, jitter, and packet loss. Path length, it is a measure of the number of links between two physical nodes that are mapped to two interconnected virtual nodes, the longer the path, the more resources required to perform the embedding. Stress level, representing how much loaded is a certain physical network resource when many virtual nodes are mapped on it. Same for physical link, that is if shared with multiple virtual links, this will result on stressing this physical link, causing delay, limitation on bandwidth availability, and less throughput. Utilization representing the percentage of used capacity from a specific resource, be it node related or link metric during a specific period of time [ETSI (2016)] [Fischer (2013)].

Category	Metric	General	Capacity	Speed	Reliability	Description
QoS	Processing power		•			Count of the number of cores per node.
	Memory		•			Memory size per node in Bytes.
	Throughput			•		Actual channel speed in bps
	Path length	•				a measure of the number of links between two mapped substrate nodes.
	Stress level	•				representing how much loaded is a certain substrate network resource.
	Utilization		•			representing the percentage of used capacity from a specific resource.
	Processing delay			•		representing the time a node takes to process the packets headers.
	Queuing delay			•		the time packet spend in the routing queues.
	Transmission delay			•		the time taken by a node to push the packet bits onto the link.
	Propagation delay			•		Travel time of a bit stream header from source node to the destination node.
	Jitter			•		Variation of a digital signal transitions from their ideal positions in time.
	Packet Loss				•	The discarded packets in a network at a given moment
Network economics	Cost	•				Amount of consumed resources in the virtual network embedding process
	Revenue	•				The sum of the virtual resources that were requested and embedded
	Cost to revenue ratio	•				Evaluates embedding algorithm by costs associated due to accepting a VNR
	Acceptance Ratio	•				Embedding efficiency, given by accepted VNRs by the total VNRs received
General	Run time of the algorithm	•				Compares algorithms with respect to the time they need to compute the VNE
	Active substrate nodes	•				Related to the average length of the substrate paths

Sources: Fischer2013, ETSI2014, ETSI2016

Figure 2.5: VNE Metrics

Delay can be divided into several types: processing delay representing the time a node takes to process the packets headers, queuing delay, which is the time a packet spends in the routing queues, transmission delay, is the time taken by a node to push the packet bits onto the link, and propagation delay, related to the amount of time it takes for the header of the bit stream to travel from source node to the destination node [RFC7679 (2017)].

While jitter [ETSI (2014b)] [ETSI (2016)], representing the variation of a digital signal transitions from their original positions in time. Finally, packet loss [ETSI (2016)], is the discarding of packets in a network when a router or other network device is overloaded and cannot accept additional packets at a given moment.

Network economics metrics while performing VNE, the virtualization process will consume the physical network resources. Therefore to evaluate the embedding process in terms of its consumption for the resource, cost, revenue, cost to revenue ratio, and acceptance ratio metrics are usually used. Regarding cost, it represents the amount of physical resources that were used in the virtual network embedding process, and is given by summing up all used resources in physical nodes such as processing power, and in links such as bandwidth that have been reserved for VNRs. Cost is directly related to the length of the embedded physical paths, which implies that the longer the path the higher the cost.

Next is revenue, which refers to the sum of the virtual resources that were requested and actually embedded on the physical network, then cost to revenue ratio, which evaluates and compares the embedding algorithm in terms of the costs associated due to accepting a VNR, the higher the ratio the poor the efficiency of the algorithm in embedding and consuming the physical network resources. Finally, Acceptance Ratio, which measures the overall efficiency of an embedding algorithm, and is calculated by dividing the total number of VNRs that could be completely embedded by the physical

network by the total VNRs received by the embedding algorithm [ETSI (2014b)] [ETSI (2016)].

General metrics Run time of the VNE algorithm, which compares the resource allocation algorithms with respect to the time they need to compute the VNE problem altogether [Fischer (2013)].

2.6.6 Uncoordinated VNE

In the uncoordinated VNE problems, the virtual node mapping stage is solved separately, usually without any coordination with the virtual edge stage. Consequently, virtual nodes could be mapped at physical nodes which are far away from each other, causing virtual edges to be mapped on longer physical paths. This may cause less virtual network request acceptance ratios, which in turn is translated into reduced revenues and increased costs.

In general, uncoordinated VNE strategies used to deploy greedy algorithms to assign virtual nodes of high demands to a candidate physical node that has the biggest resources. Accordingly, virtual edge mapping stage can be solved, either using single path mapping strategy, which uses shortest path algorithms to embed each virtual edge on a single path connecting the previously selected nodes on the physical network. Or using multi-path mapping algorithms, where each virtual edge can be carried out by several paths in the physical network [Fischer (2013)] [Yu (2008)].

2.6.7 Coordinated VNE

To overcome some of the shortcomings of uncoordinated VNE, coordinated VNE algorithms were proposed, and the are attracting increasing attention due to their capabilities to realize the VNE through imposing coordination between virtual nodes and virtual edges mapping stages, promising of leverage revenue through enhancing virtual networks' acceptance ratio. Table. 2.2 summarizes some recent studies that focused on coordinated VNE problem.

In coordinated VNE, the most referenced approach is the one suggested by [Chowdhury (2012)], which introduced a set of algorithms to correlate between node and edge embedding problems to solve the VNE. They embedded the virtual nodes onto physical network nodes based on their residual capacities, but they coordinated the edges embeddings using the multi-commodity flow algorithm to facilitate the embeddings of virtual edges onto physical network paths included the physical network hosting the virtual nodes. However, since nodes were embedded first then edges afterwards, longer paths could be used, causing additional costs and less accepted VNRs. In [Botero (2012a)], they developed a greedy algorithm using shortest paths of more residual resources to embed virtual demands of both nodes and edges together. They allowed using a number of hidden nodes in these shortest paths, which may lead to consuming more resources.

A novel methodology to solve the the embedding of virtual edges including their nodes and edges, was suggested by [Botero (2013a)]. They used a strategy using the mathematical frameworks of

path algebra, which finds all eligible paths between each pair of nodes in the physical network to embed a virtual network request suing any type of constraints. In this way, they managed to embed both nodes and edges simultaneously is a suitable path that has enough resources. A similar recent work using the same mathematical framework was proposed by [Nahid (2017)]. They ordered the physical network paths using an algorithm based on path algebra mathematics, which ranks and choses physical network nodes to host the virtual nodes, then, they search for the best physical network path connecting the already chosen nodes if it exists, by selecting the path with the highest order based on some determined metrics.

A VNE algorithm based on greedy approach was proposed by [Ogino (2017)], which prioritized the virtual edges assignment rather than the virtual nodes. They used the minimum-cost route algorithm to compute the optimum physical path for each virtual edge. The proposed approach focuses on selecting the optimum physical path that will minimize the increase in the demanded amount of edges' bandwidth and nodes' capacity. However, their methodology could allow to select longer paths containing more physical network resources than requested.

In addition to that, a study for the VNE problem under maximum latency constraint was developed by [Bianchi (2017)] to determine a set of physical network nodes and edges that could satisfy the demands of virtual network's end-to-end delay. Their algorithm used Markov chain reward metrics, coordinating virtual nodes embedding problem with virtual edges embedding problem. Virtual nodes were embedded on the physical nodes that has enough available capacity and not far away from each other, rewarding the nodes according to the availability of resources at their neighborhood nodes as well as the topological structure of the graph in that neighborhood. Then, in a second phase, they used the Dijkstra shortest path algorithm to embed the virtual edges to the physical paths connecting the previously assigned physical nodes, with sufficient bandwidth capacities. However, giving that the shortest path was selected based on delay constraint, and they used a proximity metric to avoid including other non utilized nodes in the selected path.

Moreover, the authors in [Hesselbach (2016)] introduced a new algorithm based on path algebra to solve both stages of the VNE, virtual node mapping and virtual link mapping in a coordinated methodology. They measured the availability for a path between a pair of nodes as a measure of the path's available resources to serve the VNR. They suggested to change the ranking equation using other metrics, also they suggested to use more thorough characterization of the physical network topology and to study nodes' closeness metric which measures the proximity of the nodes to each other, and nodes betweenness which evaluates which nodes are more traversed by paths connecting pair of nodes.

Motivated by the disadvantages of traditional greedy node mapping, [Jianxin (2014)] presented a new topology-aware approach, which measures the relative influence or importance of nodes and links from different aspects. They suggested algorithms to better coordinate node mapping and link mapping. They concluded that a node with high degree should have many connections, while a node with high strength should have strong connections, and a node with high closeness centrality is

on average at a short distance from other nodes in the network, but a node with high betweenness centrality will be part of many shortest paths between any other two nodes in the network, finally, a node with eigenvector centrality should have high probability of connecting to other nodes with high eigenvector centrality. In [Gong (2016)], the authors considered coordinated embedding for the virtualized SDN. They designed a novel online embedding algorithm which includes two stages: the node mapping stage, where the controller placement and virtual node mapping are tackled, and the link mapping stage. In the node mapping stage, they attached the controller to the physical node with the largest controller location selection factor, and they also considered in advance the subsequent virtual node and link mappings to obtain high revenue and low controller-to-switch delay.

The authors in [Guerzoni (2015)] presented a virtual link mapping formulation allowing the support of QoS, and to be suitable for the management of 5G network data plan, enriching the traffic engineering and virtual link mapping problems with constraints required to support, among others, delay critical services. Additionally, [Shuiqing (2016)] introduced five node attributes to quantify the embedding potential of the nodes from both the local and global views. They proposed a two-stage virtual network embedding algorithm, which maps the virtual nodes onto the physical nodes according to the node rank, and adopts a shortest path-based algorithm to map the virtual links.

Another VNE method uses Gomory-Hu tree transformation was proposed by [Soualah (2016)] which provided a compact representation of the network topologies. The initial VNE problem was transformed using successive cuts of the graphs to map the virtual nodes on the tree representing the physical network. The authors claimed that mapping virtual trees onto the physical tree fixes the mapping of the nodes and guarantees the mapping of virtual links in a splittable way, where one virtual link is dispatched to a set of physical paths. On the same direction, a near optimal solution to the unsplitable flow VNE problem using integer linear programing and path generation was proposed by [Mijumbi (2015)]. The proposed solution performs node and link mapping in two coordinated stages. The authors suggested to propose relaxations to the mathematical program to minimize time complexity using Tabu search and path relinking.

To solve the embedding problem, [Wen (2015)] suggested to use a algorithm which handles the interaction between a pair of nodes by correlating resources' capabilities and the distance or hops between them, and for virtual link embedding, they used the shortest path algorithm. Moreover, [Shashank (2014)] presented virtual network mapping approach that simultaneously maps virtual nodes and links onto the network infrastructure, considering bandwidth, processing power, and location constraints. The main idea was to formulate the virtual network mapping problem as a hub location problem, which deals with the virtual nodes as clients that need to be allocated to the physical nodes at a lowest cost, while ensuring balanced load across the physical network. Furthermore, the authors in [Shahriar (2016)] tackled the problem of ensuring virtual network connectivity in presence of multiple link failures in the physical network, ensuring different connectivity levels for each virtual node.

In order to supply dedicated virtual node to an end-user typified with a customized traffic, [Ilhem (2016)]

Table 2.2: Studies about Coordinated VNE

Paper	Summary
[Chowdhury (2012)]	Introduced algorithms to correlate node and edge problems to solve the VNE.
[Botero (2012a)]	Developed a greedy algorithm using shortest paths to embed nodes and edges together.
[Botero (2013a)]	Proposed a novel methodology to solve the the embedding of virtual edges using path algebra.
[Nahid (2017)]	Proposed a framework to order the physical network paths using path algebra mathematics.
[Ogino (2017)]	Proposed a VNE algorithm prioritizing edges assignment when selecting optimum path.
[Bianchi (2017)]	Provided a study for the VNE problem under maximum latency constraint.
[Hesselbach (2016)]	Introduced algorithm using path algebra to solve VNE in a coordinated methodology.
[Jianxin (2014)]	Suggested algorithms to better coordinate node mapping and link mapping.
[Gong (2016)]	Considered coordinated embedding for the virtualized SDN.
[Guerzoni (2015)]	Presented a virtual link mapping formulation for 5G network data plan.
[Shuiqing (2016)]	Proposed two-stage VNE algorithm to map virtual nodes and edges.
[Soualah (2016)]	Solved the VNE problem using successive cuts of the graphs.
[Mijumbi (2015)]	Proposed a solution that performs node and link mapping in two coordinated stages.
[Wen (2015)]	Handled interaction between a pair of nodes, and used the shortest path for virtual edges.
[Shashank (2014)]	Presented VNE that simultaneously maps nodes and links onto the network infrastructure.
[Shahriar (2016)]	Solved virtual network connectivity in presence of multiple link failures.
[Ilhem (2016)]	Proposed a new adaptive VNE algorithm to minimize the VNE cost.
[Gong (2016)]	Proposed two algorithms for achieving integrated node and link mapping.
[Yang Wang (2015)]	Proposed a framework to solve VNE problem employing a branch and bound process.

suggested a resource provisioning algorithm to guarantee an efficient and flexible share among all the instantiated VNs upon the underlying physical network. For that, the authors proposed a new adaptive VNE algorithm called (Adaptive-VNE), which adopts a backtracking algorithm in order to minimize the VNE cost. On the other hand, [Gong (2016)] proposed two algorithms for achieving integrated node and link mapping based on compatibility graph, where each node represents a candidate physical path for a virtual link, and if two physical paths are compatible with each other, the algorithm inserts a link to connect their corresponding nodes in the compatibility graph.

Finally, in [Yang Wang (2015)], the authors proposed a framework to address the VNE problem, employing a branch and bound process to resolve the integrity constraints, which depends on the efficient estimation of the lower and upper bounds of a branch in a rooted tree that represents a subset of the solution set. Then they applied the column generation process to effectively obtain the lower bound for a branch pruning. As the branch and price framework maintains the lower and upper bound of the optimal solution, the authors claimed that their proposed framework can obtain near-optimal solution with reduced computational time.

2.6.8 Power efficiency in VNE

One of the main benefits of network virtualization is its ability to consolidate network resources which allows for reducing power consumption and cost [Botero (2013b)][Chen (2016)]. In most cases, saving power in networks has been devoted to the reduction of power consumption in a single networking device or parts of a device, and not power saving in the whole network, where unused resources could be but into sleeping mode or turned off completely. Other approaches performed VNE on small parts of the physical network, then widen the area if no sufficient power resources could be found in the physical network. Moreover, virtual resources can be migrated to balance the overall load in a power efficient way, thus reducing the total power consumption of the network, without compromising QoS or VNRs' acceptance ratio. More details about most related and recent literature about power aware VNE approaches are summarized in the following paragraphs and listed in Table.2.3.

A modified VNE algorithm was presented by [Botero (2012b)], which prefers physical network nodes consuming less power and selects the edges in a power efficient path, then in [Botero (2013b)], they developed a scalable power-aware reconfiguration algorithm approach, including embedding cost and load balancing. The algorithm considers a set of embedded VNRs as input to perform a power efficient relocation of resources without impacting the acceptance ratio. [Su (2014)] proposed to maximize the accepted VNRs while minimizing the power cost of the whole system. They followed two observations: first they embedded virtual nodes on physical network nodes that has lowest electricity price, second they embedded the virtual nodes on an active physical network as much as possible, then put other nodes that has no load into sleeping mode.

Moreover, [Triki (2015)] developed an embedding algorithm that embeds a subset of VNRs into a subset of cleanest physical network resources in terms of CO2 emissions resulting from the power usage, while satisfying the VNR constraints. They constrained the VNE process by introducing link delay, packet loss, used power source, VNR priority and location. The authors showed that the embedding considered reducing the number of physical resources and cost, minimizing the embedding time, and reducing the carbon footprint of the VNE operation. While in [Nonde (2015)], the authors designed a real time algorithm that considers granular power consumption of all devices in the network. They tried to consolidate the nodes embeddings by filling the ones with the least residual capacity before switching on others, as well as consolidating more than virtual node at the same datacenter to minimize additional hop counts.

In [Chen (2016)], the authors considered a power efficient VNE in the network by adapting a feedback control approach performing the embedding on a smaller set of physical network resources. A limited mappable area consisting of a selection of candidate nodes was located first, then they checked if the virtual node embedding was successful, if not, then a feedback control approach was triggered to search for a wider mappable area, and the whole process repeats again. In this way, they managed to increase number of hibernated links and nodes, resulting on reducing the power consumption in the physical network.

Table 2.3: Studies about Power Efficiency in VNE

Paper	Summary
[Botero (2012b)]	Presented a VNE algorithm which prefers nodes consuming less power.
[Botero (2013b)]	Developed a scalable power-aware reconfiguration algorithm approach.
[Su (2014)]	Proposed to maximize accepted VNRs while minimizing power cost.
[Triki (2015)]	Developed algorithm to embed VNRs into cleanest resources.
[Nonde (2015)]	Consolidated nodes by filling those of least capacity before switching others.
[Chen (2016)]	Considered a power efficient VNE by adapting a feedback control approach.
[Lira (2017)]	Proposed an online embedding technique for power aware dependable virtual networks.
[Ghazisaeedi (2017)]	Re-embed virtual edge according to off-peak traffic, and shuts the least stressed edges.
[Hou (2016)]	Applied power efficient VNE algorithm using two phases.

A recent publication by [Lira (2017)] proposed an online meta-algorithm embedding technique for power aware dependable virtual networks, using stochastic Petri nets and reliability block diagrams. For redundancies, they embedded nodes first on two physical nodes, and edges on four physical edges. Another power aware algorithm using migrations was proposed by [Ghazisaeedi (2017)], which re-embed every virtual edge according to its off-peak traffic demand, by defining a stress rate for the physical edges, then after reallocating the virtual edges they put into sleeping mode the remaining least stressed edges. However, they do not re-embed the virtual nodes, but they make sure by applying shortest path algorithm of Dijkstra that the selected physical path used to migrate the virtual edges is also connecting the terminal nodes of the virtual edge. In [Hou (2016)] the authors applied load balancing and power efficient VNE algorithm using two embedding phases, one to embed the virtual nodes, and the other one to embed the virtual edges. They aimed to embed as much virtual nodes as possible on some ordered physical nodes based on the utilization of their connected edges and the residual processing capacity on them. They selected the physical nodes to be near to each other, and they used Dijkstra shortest path algorithm to embed the virtual edges on physical paths connecting the already selected nodes. By this way they consolidated the traffic from the least utilized nodes and put into sleeping mode their connecting edges.

2.7 Literature about NFV

A comprehensive survey of NFV is shown in [Mijumbi (2016)], providing an overview of the key research topics, standardization efforts, early implementation, use cases, and relationship between NFV, SDN and cloud computing. While in [Herrera and Botero (2016)] 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.

The following sections present a detailed review for the recent literature about NFV and power effi-

ciency, virtual machines consolidations and NFV, virtual machines migration in NFV environments, end-to-end delay in NFV, modeling the resource allocation problems for NFV architecture, and the most relevant NFV metrics.

2.7.1 Power efficiency in NFV

For related work covering power efficiency in NFV environment, a recent article by [Eramo (2017a)] 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 is the constrained resource. In addition, the authors proposed a new consolidation technique taking into account the reconfiguration costs, but with migration strategy. In [Pham (2017)] 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 [Soualah (2017)], 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, the authors in [Kim (2017)] 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 were allocated to the appropriate virtual machines representing the physical infrastructure component along the path that can minimize the power consumption. In [Kar (2018)], 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 [Meng (2018)] developed performance-aware placement algorithm which tracks the network elements to be consolidated, and places the SFCs compactly and optimizes the global packet transfer cost. They introduced a scaling up strategy to avoid degrading performance and taking up new CPU cores, while ensuring fairness when elements are consolidated on CPU core. Moreover, [Khan (2018)] proposed a dynamic virtual machine consolidation algorithm, which minimizes the power consumption of cloud datacenters through exploiting cloud service users' information, and in [Shuting Xu (2017)] 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.

On the other hand, some authors developed semi-algorithms to reduce networks' power consumptions and solved the resource allocation problem in NFV architecture using some artificial intelligence techniques. For example, the study by [Ranjbari (2018)] proposed a reinforcement learning algorithm,

Table 2.4: Studies about Power Efficiency in NFV

Paper	Summary
[Eramo (2017a)]	Investigated the consolidation problem in NFV based on vertical scaling techniques.
[Pham (2017)]	Formulated VNF placement problem for power and traffic-aware cost minimization.
[Kim (2017)]	Proposed a VNF placement algorithm to minimize power consumption.
[Kar (2018)]	Designed a power saving model for service chains placement.
[Meng (2018)]	Developed performance-aware placement algorithm for elements consolidation.
[Khan (2018)]	Proposed a dynamic VM consolidation algorithm to minimize power of datacenters.
[Shuting Xu (2017)]	Proposed a resource management scheme for VM consolidation.
[Ranjbari (2018)]	Proposed a reinforcement algorithm to reduce power of cloud infrastructure system.
[Guo 2018]	Suggested a game-based consolidation VM with power and load constraints.
[Akaki (2018)]	Provided VM consolidation Learning techniques.
[Mohammadi (2018)]	Proposed architecture to detect and manage under/over loaded VM.

which selects their current action based on past experiences to improve resource utilization and reduce power consumption inside a cloud infrastructure system. The proposed algorithm considers changes in the user demanded resources to predict the physical machine, which may suffer from overloaded utilization, reduces the number of migrations, and shuts down idle servers to reduce the power consumption of the datacenter. In addition, the authors in [Guo 2018] suggested a game-based consolidation method of virtual machines with power and load constraints, where they grouped all online physical machines by the number of virtual machines on them and their future load values. Based on the groupings, they proposed a pre-processing algorithm for selecting destination machines to determine a set of destination machines for a virtual machine awaiting migration, which was selected using game-based methods aimed at optimizing overall power consumption. In a similar trend, the paper of [Akaki (2018)] provided an intelligent solution, where the virtual machines were consolidated with a virtual machine clustering algorithm, utilizing Learning Automata techniques for mapping the clustering problem onto Graph Partitioning problem. Thereafter, the resulting clusters were assigned to the Server racks using a cluster placement algorithm that involves a completely different intelligent strategy that invokes Simulated Annealing. Finally, in [Mohammadi (2018)] a self-adaptive architecture is presented to detect and manage underloaded and overloaded virtual machines in reaction to workload changes in the data center, applying a probabilistic model of the datacenter using queuing theory.

Table.2.4 summarizes the recent studies that dealt with power efficiency in NFV frameworks.

2.7.2 Virtual machine consolidation

To save power in cloud datacenters, virtual machine consolidation is usually applied, targeting reducing (by consolidation) the number of active physical machines and using virtual machine migration to prevent inefficient usage of resources. However, high frequency of consolidations may

have a negative effect on the system reliability, and therefore, operators face crucial trade-off between reliability and power efficiency. Table.2.5 provids a summary for some selected related literature in virtual machine consolidation.

Consequently, to efficiently benefit from virtual machines consolidations and migrations technologies, this thesis leverages their usage in order to achieve the maximum power savings with the least impacts on the service levels in the system as much as possible. To review the recent trends in the virtual machines consolidations, the following paragraphs summarizes some of the most related literature in this topic.

A survey and taxonomy for the server consolidation techniques in cloud datacenters was presented in [Varasteh (2017)]. The authors covered the parameters and algorithmic approaches used to consolidate virtual machines onto physical machines, and they also discussed open challenges and suggested some areas for further research.

In [Cao (2019)], the author proposed a virtual machines dynamic consolidation approach to coordinate datacenter network topology and communication topology. The proposed approach adopted a multi-objective genetic algorithm to simultaneously minimize power consumption and communication traffic causing performance bottlenecks, while meeting service level agreement constraint on migration time. Moreover, the authors in [Monireh (2018)] proposed an approach that considers the reliability of each physical machine in the datacenter along with reducing the number of active physical machines simultaneously. To determine the reliability of physical machines, the authors designed a Markov chain model, and the physical machines were prioritized based on their CPU utilization level and the reliability status. In addition to that the authors presented a target physical machine selection criterion, which considers both power consumption and reliability in order to select the appropriate physical machine to be consolidated. Furthermore, the paper from [Heyang (2019)] investigated the migration cost-aware virtual machine consolidation problem and formulated the problem as a multi-constraint optimization model by considering migration cost and remaining runtime of virtual machines. The target of the proposed algorithm was to effectively decrease the migration cost and, at the same time guarantee the power consumption within a certain low level.

A multi-objective Boolean optimization encoding for virtual machine consolidation was proposed in [Miguel (2018)], and several approaches to solve it were described and compared. Moreover, the authors extended the encoding approach to consider anti-collocation constraints and the migration of virtual machines that were initially placed. The authors applied specific techniques to further boost the performance, namely search space reduction by symmetry breaking and algorithm reduction of the instance size. Similarly, in [Naik (2018)], the authors proposed a multi-objective Hybrid Fruit-fly Algorithm for virtual machine consideration in the cloud datacenter. The proposed algorithm is based on the virtual machine migration, which are mainly used to minimize the over provisioning of physical machine, by consolidating virtual machines from under-utilized physical machines.

The paper of [Deafallah (2018)] proposed virtual machine selection approach and virtual machine placement approach, considering multiple computing resources being used simultaneously. The

virtual machine selection approach selects a virtual machine with high CPU requirements and optimal memory requirement for reducing the workload of overloaded physical machines with minimum migration cost. The placement approach selects a physical machine at which to place the migrating virtual machine to reduce performance degradation because of the virtual machine migration. In addition, the authors in [Alsadie (2018)] presented a proposal to minimize the number of migrations in varying workload environments in datacenters. The proposed approach is fuzzy threshold-based used for adjusting the threshold values of physical machines in a cloud environment, which allows the number of migrations caused by overloading to be reduced.

In [Aryania (2018)] the authors proposed an algorithm based on Ant Colony System to solve the virtual machine consolidation problem. The aim was to save the power consumption of cloud datacenters, considering the power consumption during virtual machines migration and consolidation, as well as the number of migrations, number of sleeping physical machines, and number of service level agreement violations. Moreover, a comprehensive dynamic virtual machine consolidation strategy was proposed in [Zhou (2018)] to improve resource utilization and power efficiency for cloud computing datacenters. The strategy was based on time-series forecasting approach, using a specific adjustment of threshold to adapt the dynamic workload in the datacenter.

To optimize the consolidation process, effective virtual machine placement can be used as in the work of authors from [Yousefipour (2018)], who presented a mathematical model aimed at reducing power consumption and costs by employing virtual machine consolidation in the cloud datacenter. The authors proposed a power and cost aware virtual machine consolidation genetic based algorithm, and compared its results with the first fit, first fit decreasing, and permutation pack algorithms. The authors in [Ho (2011)] modeled the relocation virtual machines problem in cloud computing environments as a modified bin packing problem and proposed a server consolidation algorithm that guarantees server consolidation with bounded relocation costs. They conducted a detailed analysis on the complexity of the server consolidation problem, and compared their server consolidation algorithm with First Fit and Best Fit methods, and concluded that they had to trade-off between server consolidation quality and relocation cost. Finally, the study by [Luo (2014)] focused on the Infrastructure as a Service model, where custom virtual machines were launched in appropriate servers available in a datacenter. The authors presented a complete datacenter resource management scheme to ensure user quality of service and also considered power saving and green computing goals. In the paper, they modified shuffled frog leaping algorithm to solve the dynamic allocation and consolidation problem of virtual machines in the data enters.

2.7.3 Migrating virtual machines

In addition to virtual machines consolidations, this thesis studies live migration technology in datacenters. The target is to allow virtual machines to be consolidated and migrated into a fewer physical servers, thus the idle ones can be shut down or switched to low-power mode, which will be translated into reducing the power consumption of cloud datacenters. In addition, live migration

Table 2.5: Studies about Virtual machine consolidation

Paper	Summary
[Varasteh (2017)]	Provided a survey about consolidation techniques in cloud datacenters.
[Cao (2019)]	Proposed a dynamic VM consolidation approach.
[Monireh (2018)]	Proposed to reduce active machines simultaneously.
[Heyang (2019)]	Investigated migration cost-aware VM consolidation problem.
[Miguel (2018)]	Proposed an optimization encoding for VM consolidation.
[Naik (2018)]	Proposed algorithm consolidating VM from under-utilized machines.
[Deafallah (2018)]	Proposed VM selection and placement approach, considering migrations.
[Alsadie (2018)]	Proposed minimum migrations using fuzzy threshold.
[Aryania (2018)]	Proposed Ant Colony to solve VM consolidation problem.
[Zhou (2018)]	Proposed dynamic VM consolidation to improve power efficiency for datacenters.
[Yousefipour (2018)]	Reduced power consumption employing VM consolidation.
[Ho (2011)]	Proposed a server consolidation algorithm with bounded relocation costs.
[Luo (2014)]	Presented resource management scheme considering power.

can also aggravate the overheads of data transmissions and produce additional power consumption in cloud datacenters. However, live migration can result in performance degradation of migrated virtual machines, or even interrupting their services. To review the recent trends in literature about virtual machines migrations and their impacts on the overall performance of the datacenters, Table.2.6 summarizes some of the most relevant migration studies, which are discussed in more details in the following paragraphs.

In [Zhang (2018a)] paper, the authors 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 the three main challenges it faces: 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, the paper of [Noshy (2018)] 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 [Silva (2018)] 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.

In similar directions, live migration in datacenters and handover in cellular networks were discussed

in [Shangguang (2018)]. The authors reviewed the cutting-edge research efforts on service migration in mobile edge computing, presenting various research directions for efficient service migration, as well as summarizing the technologies in literature that focused on hosting services on edge servers, namely virtual machine, container, and agent. In [Surabhi (2019)], the authors proposed an analytical queuing model based approach for quality evaluation of cloud by exploring only rejection probability of jobs, and considering the variable buffer sizes and different distribution models. They concluded that changing buffer size proved to be gainful, as well as changing of queuing model as it leads to decrease in the rejection probability of the jobs.

A resource and route aware live virtual machine migration technique was proposed by [Paulraj (2018)], in which the dynamic resource utilization of the servers was predicted using combined forecasting technique. Based on that, the servers were clustered, then the virtual machines migrated from overloaded servers to the nearest underutilized server. Another work about migrations was presented in [Joseph (2018)], which classified the various migration techniques proposed in the literature, in addition to that the authors provided a survey of the various metrics that characterize the performance of a migration technique. Similar work can be seen in [Addya (2018)], which proposed a framework for secure live migration of virtual machines in a cloud federation. The authors analyzed the associated migration costs for serial, parallel, and improved serial strategies, considering communication overhead, migration time, and downtime.

The paper of [Heba (2019)] presented two cooperative algorithms to dynamically distribute the cloud system's physical resources, and to obtain a load-balanced consolidated system with minimal used power, memory, and processing time. The first algorithm arranges virtual machines in clusters based on their memory and CPU parameters' value, then it deals with the co-located virtual machines that share some of their memory pages and located on the same physical machine, as a group. The migration decision was made based on the evaluation for the entire system by the second algorithm, which minimizes the number of migrations among the system, saves the consumed power, and prevents performance degradation for the virtual machine while preserving the load-balance state of the entire system. Moreover, the authors in [Satpathy (2018)] proposed a two-tier virtual machine placement algorithm that has a queuing structure to manage and schedule a large set of virtual machines. They designed a multi-objective virtual machine placement algorithm to reduce the resources wastage and power consumption at the datacenters, utilizing and testing three different migration strategies namely serial, parallel, and improved serial for activities like maintenance, load balancing, and fault tolerance.

Finally, in [ZhongWang (2018)], the authors proposed an adaptive virtual machine monitoring and live migration strategy for IoT applications in edge cloud architecture. The basic idea of the proposed strategy was that the migration method of a virtual machine should be determined by its workload characteristics, based on the variation of current memory dirty page rate, so the strategy can adaptively select the most appropriate migration method to copy memory pages. And in [Namra (2019)] the authors incorporated an adaptive live task migration technique to prevent as many disasters as possible and to significantly reduce cost, and balance the load among available nodes.

Table 2.6: Studies about migrating virtual machines

Paper	Summary
[Zhang (2018a)]	Reviewed VM migration and discussed both its benefits and challenges.
[Noshy (2018)]	Reviewed live migration of VM in cloud computing environments.
[Silva (2018)]	Provided survey of VM placement and migration in the area of cloud computing.
[Shangguang (2018)]	Reviewed the research efforts on service migration in mobile edge computing.
[Surabhi (2019)]	Proposed a quality evaluation of cloud based on rejection probability.
[Paulraj (2018)]	Proposed a resource and route aware live VM migration technique.
[Joseph (2018)]	Classified the various migration techniques proposed in the literature.
[Addya (2018)]	Proposed a framework for secure live migration of VMs in cloud.
[Heba (2019)]	Presented cooperative algorithms to obtain balanced consolidated systems.
[Satpathy (2018)]	Designed VM placement algorithm to reduce wastage and power in datacenters.
[ZhongWang (2018)]	Proposed adaptive VM monitoring and live migration strategy for IoT.
[Namra (2019)]	Incorporated an adaptive live task migration technique to prevent disasters.

2.7.4 Delay and NFV

Regarding related work about delay and NFV, the authors in [Kim (2017)] 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 [Kar (2018)], 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, and used delay as a constraint. Moreover, [Luizelli (2015)] developed an optimization model to solve the chaining problem including end-to-end delays, while [Reddy (2016)] generalized an optimization model to cope with the arbitrary VNF chains and traffic demand uncertainties, fulfilling individual average round trip delay constraint for each VNF chain.

Moreover, [Qu (2016)] formulated the delay-aware virtual network function scheduling problem, considering both VNFs' processing delays at VMs, and service chain transmission delays at virtual links, and to solve the problem, the authors developed a algorithm using genetic algorithm to overcome the complexity of the original problem. Finally, in [Bhamare (2017)], the authors presented an analytical model for the placement of service function chains in multi cloud environments, and tried to optimize the end-to-end delays to end users with optimal placements of the service function chains in a multi cloud scenario. A summary for these studies is listed in Table.2.7.

Table 2.7: Studies about Delay in NFV

Paper	Summary
[Kim (2017)]	Proposed a VNF placement algorithm that allows users to meet their service latency requirements while minimizing the power consumption at the same time.
[Kar (2018)]	Designed power saving model for the placement of multiple service chains on the network, and used delay as a constraint.
[Luizelli (2015)]	Developed an optimization model to solve the chaining problem including end-to-end delays.
[Reddy (2016)]	Generalized a model to place VNF chains fulfilling individual average round trip delay constraint for each VNF chain.
[Qu (2016)]	Formulated the delay-aware virtual network function scheduling problem, considering both VNFs' processing and transmission delays.
[Bhamare (2017)]	Optimized end-to-end delays with optimal placements of the service function chains in a multi cloud scenario.

2.7.5 RA-NFV architectural model

In general terms VNE and network function virtualization resource allocation problems are similar, giving that both problems aim to find, the optimal allocations for certain virtual functionalities on top of physical resources that, have enough capacities to serve the demands of virtual network functionalities. Therefore, RA-NFV could be considered as a generalized structure of the traditional VNE problem [Herrera and Botero (2016)] [Mijumbi (2016)].

However, in NFV environment, a network service (NS) represents a collection of functionalities required to perform certain service, and is usually called service function chain. Which is then formulated as a set of defined number of chained and ordered VNFs to be allocated on the most suitable VNF infrastructure of the physical network.

Regarding resource allocation mathematical modeling in NFV environments, an early work targeting the complex scheduling problem was presented by [Riera (2015)], 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 [Riera and Grasa (2014)]. An optimal solution for orchestrating the VNFs in small scale networks is presented in [Bari (2016)], who proved the NP-hardness of the NFV orchestration problem, and suggested a algorithm to solve it faster for real-world synthetic topologies and traffic traces.

For the RA-NFV architectural modeling fig.2.6 presents an overall example about the architecture of a network representing a datacenter utilizing NFV architecture as envisioned by ETSI [ETSI (2013a)][ETSI (2013b)]. In block-1 a high level NFV architecture, where an SFC request composed of three VNFs that could be (Firewall, Intrusion Detection System, and Encryption VPN) is introduced to be allocated on the underlying physical infrastructure. Next block-2 in the figure shows the chaining structure of VNFs according to the requested VNF. It is part of a forwarding

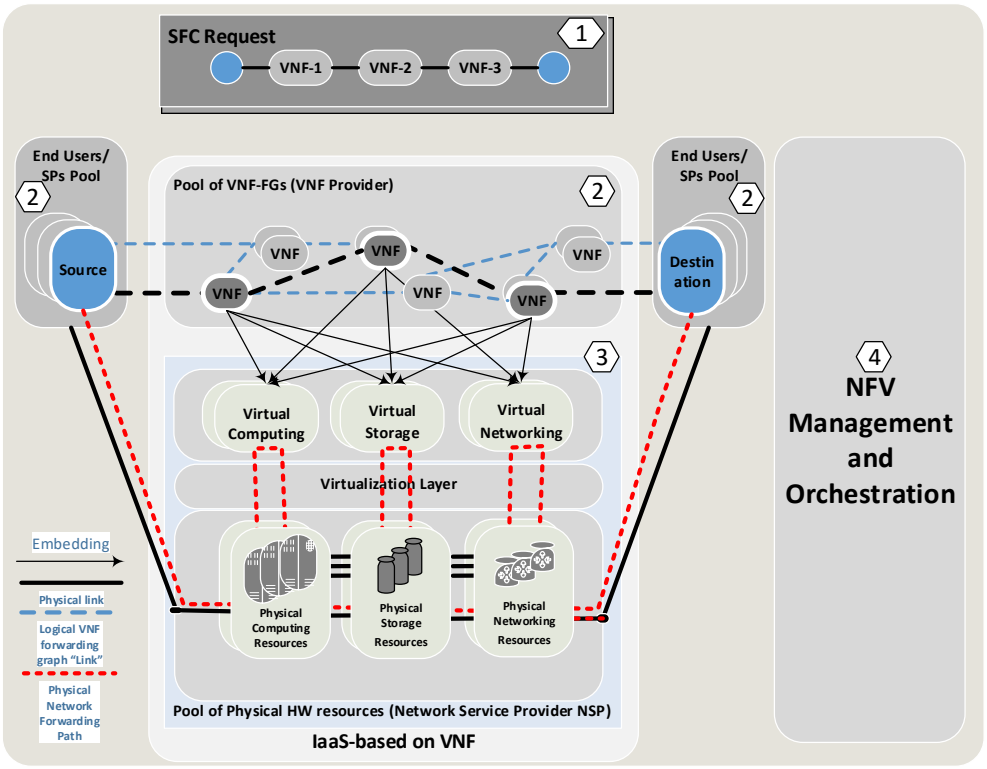

Figure 2.6: IaaS based on NFV

graph, that includes a pool of various SFC. Then in block-3, it is the resource allocation step, which assigns available physical resources to the corresponding VNF instances. And block-4 represents the NFV management and orchestration, which controls the whole process.

2.7.6 NFV metrics

Based on [ETSI (2014b)] and [ETSI (2016)], the suggested metrics that can be used to evaluate the overall performance of an NFV architecture, and validate its physical infrastructure can be divided into three categories as follows: compute, storage and networking sub groups, according to the types of the virtualized machines within the VNF architecture. Furthermore, each sub group can be organized in the following categories:

Performance and speed These are used to better get a understand what happens to the physical infrastructure, when virtual network functions are deployed on the NFV system, typical examples include: latency, processing speed, and throughput.

	Speed	Capacity
Compute	Latency for random memory access	Number of cores and threads
	Latency for cache read/write operations	Available memory size
	Processing speed (instructions per second)	Cache size
	Throughput for random memory access (bytes per second)	Processor utilization (max, average, standard deviation)
		Memory utilization (max, average, standard deviation)
		Cache utilization (max, average, standard deviation)
Network	Throughput per NFVI node (frames/byte per second)	Number of connections
	Throughput provided to a VM (frames/byte per second)	Number of frames sent/received
	Latency per traffic flow	Maximum throughput between VMs (frames/byte per second)
	Latency between VMs	Maximum throughput between NFVI nodes (frames/byte per second)
	Latency between NFVI nodes	Network utilization (max, average, standard deviation)
	Packet delay variation (jitter) between VMs	Number of traffic flows
	Packet delay variation (jitter) between NFVI nodes	
Storage	Sequential read/write input/output per sec	Storage/Disk size
	Random read/write input/output per sec	Capacity allocation (blockbased, object-based)
	Latency for storage read/write operations	Block size
	Throughput for storage read/write operations	Maximum sequential read/write input/output per sec
		Maximum random read/write input/output per sec
		Disk utilization (max, average, standard deviation)

Source: ETSI2016, ETSI2014)

Figure 2.7: NFV Metrics

Capacity and scale To get a feel about utilization and usage of the computing, network, and memory resources within the NFV physical infrastructure when handling VNF applications, capacity metrics are usually used. Typical examples such as: number of computing cores, maximum memory, maximum throughput, and utilizations of processor, network, and memory.

More metrics can be found in Fig.2.7, which summarizes NFV's metrics as recommended by ETSI. [ETSI (2014b)] [ETSI (2016)].

CHAPTER 3

Methodology for Power Aware Resource Allocation and Virtualization Problem

Introduction In networks' virtualization environments, such as virtual network embedding, VNE, or network function virtualization, NFV, allocating the virtual network requests (i.e network services) on the physical network is a proven NP-Hard problem [Fischer (2013)] [Bari (2016)]. The resource allocation problem is usually divided into two sub-problems, one for allocating the virtual nodes onto candidate physical nodes, and the other one for allocating the virtual edges onto physical paths connecting the corresponding nodes in the physical network. Along such process, the allocation problem used to trades off between minimizing the allocation costs, through efficiently utilizing less resources, versus maximizing the overall network's revenues, by allocating as much network services as possible for a given quality of service constraints.

Generally, the two sub-problems used to be carried out in two strategies, uncoordinated or coordinated [Fischer (2013)]. Regarding the uncoordinated case, virtual nodes and virtual edges allocations commonly solved independently without any coordination between the two stages, raising the possibilities of higher rejections or adding more costs. This is because virtual edges allocations could be mapped on longer physical paths, therefore, utilizing additional resources, consuming more power, and adding more delay due to passing through hidden hopes [Yu (2008)]. The other strategy, is performing both virtual nodes and virtual edges allocations in two separate, but coordinated stages, where virtual nodes allocations can be performed according to predefined virtual edges allocations' constraints to guide allocating the virtual nodes [Chowdhury (2012)].

However, in this sort of coordination approach, still, virtual nodes could be allocated at physical nodes that could be farther away from each other, enforcing edges to be allocated at longer physical paths, resulting on similar disadvantageous as in the uncoordinated scenario. Furthermore, regardless of the used strategy, solving the resource allocation problem used to be constrained by CPU and BW resources, but occasionally considering power consumption, and almost very seldom adding delay as an additional constraint. Thus, it could be possible that, the lack of considering more constraints throughout the allocation process could result on a degraded QoS for the whole allocation process, including raising operational costs, consuming more power, as well as generating less revenues.

In view of that, this thesis proposes a new power aware resource allocation approach, which performs

virtual nodes and edges allocations simultaneously, and in one stage on the least physical resources to minimize the total consumed power, while considering CPU, BW, and end-to-end delay constraints. The core of the proposed approach to minimize the total power in the physical network is based on formulating the network services' demands and the physical paths' resources into two separate sets, called (Segments Technique), one for the network service (could be a VNR or SFC), and another one for a precisely selected physical path. The segmentation technique will be the main building and fundamental block for the new proposed power aware resource allocation approach in this thesis. It will ensure efficient utilizing for the physical resources, as well as identifying other non utilized resources to be switched off.

This chapter will cover:

1. In section 3.1, it introduces the proposed general modeling structure of the power aware resource allocation problem, in an integer linear programming formulation, clarifying the problem's general mathematical formulation, and presenting the constrains that govern the relationships between the physical network resources and the network services' demanded resources.
2. In section 3.2, the chapter introduces the new proposed technique to coordinate allocating the virtual nodes and edges together in full coordination.
3. And section 3.3 introduces and explains the general structure of the new proposed power aware resource allocation and virtualization algorithm, including an illustrative example showing how the algorithm works, and evaluations of the algorithm .

3.1 General power aware resource allocation problem modeling

Following paragraphs explain the overall design model for the power aware resource allocation problem, starting by defining physical network model and introducing its notations. Then VN's model definition and notation will be explained, as well as defining the used power consumption model.

Fig.3.1 shows a general structure of the resource allocation problem in virtual network embedding framework. At the bottom of the figure it shows a sample physical network of nine interconnected physical nodes, and in the the middle of the figure, it shows a sample of two VNRs. Notations for the nodes and edges are shown at the top of the figure.

3.1.1 Physical network model

The physical network $G^P = (N^P, E^P)$ is modeled as a weighted directed graph, where: i and j $\in N^P$ are the physical network nodes, and $(i, j) \in E^P$ is an edge connecting nodes i and j in the physical network. Each node $i \in N^P$ is associated with pw_i^{idle} representing average power value

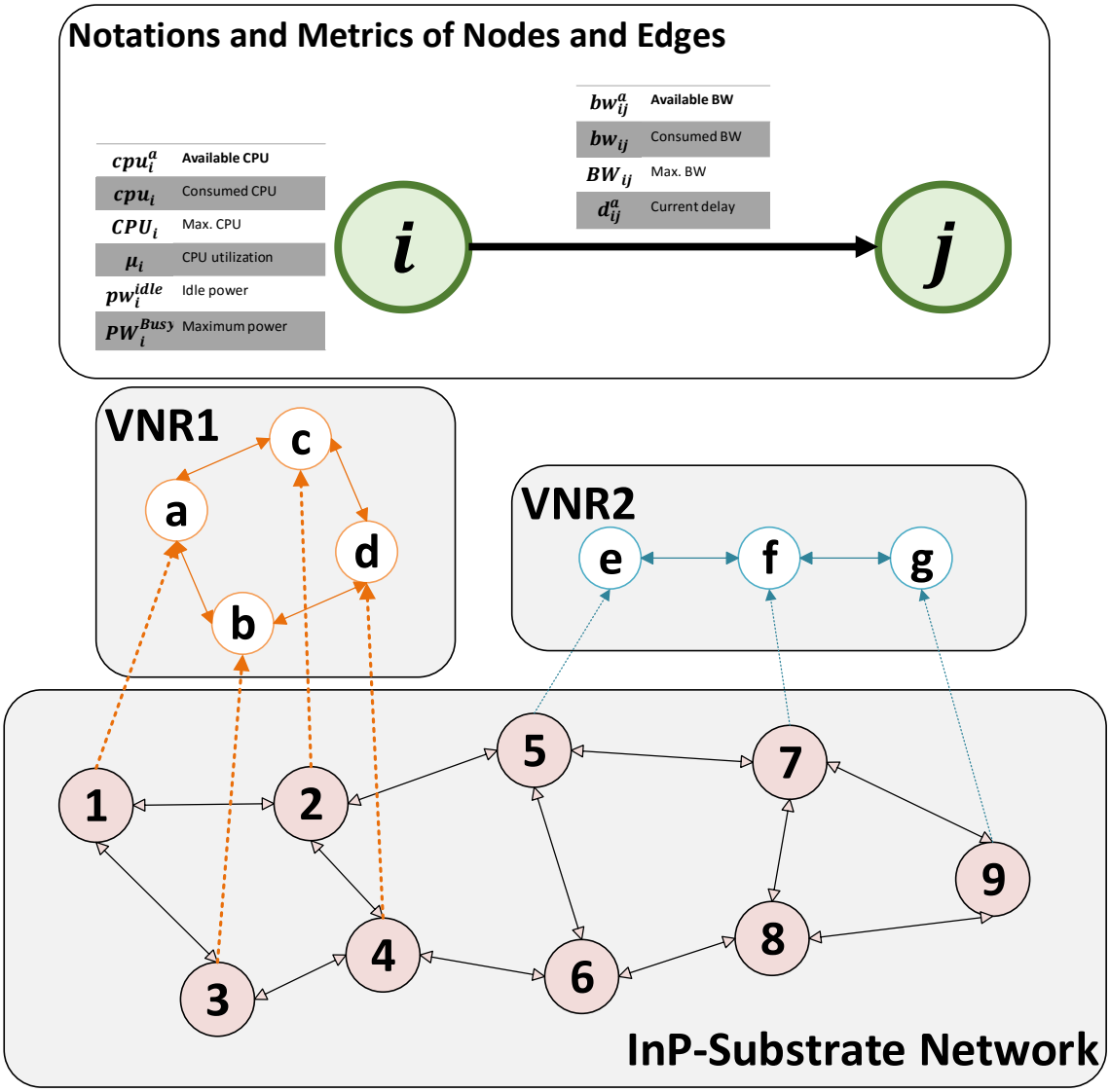

Figure 3.1: VNE Problem

when the sever is idle, PW_i^{Busy} average power value when the server is fully utilized, PC_i total power consumption of i, as well as cpu_i^a representing current available CPU capacity, cpu_i consumed CPU capacity, and CPU_i as the maximum CPU capacity at node i. μ_i is a fractional value (consumed to maximum CPU capacity, which could reach a value of 1 maximum) representing the processing utilization of node i defined in the range (0-1), zero if node i is not loaded, up to 1 if its 100% loaded. Each physical edge (i, j) is associated with bw_{ij}^a, representing current available bandwidth capacity, bw_{ij} as consumed bandwidth capacity, BW_{ij} for maximum bandwidth capacity, d_{ij}^a as current end-to-end delay in physical network edge (i, j), while $f_{i,j}^a$ is current traffic flow defined as the total throughput from physical network node i to j. $P^P = \{(i, j)\}$ represents a set of all directed paths connecting all pairs of the physical network nodes i and j with a set of edges $\{(i, j)\}$. And the physical path $P_{sd} = \{(s, n), ..., (k, l), ..., (m, d)\} \in P^P$ is an end-to-end path constructed of more than one physical edge, where (s, n) is the first physical edge connecting the source node s to its adjacent node n, (k, l) is an intermediate physical edges, and (m, d) is the last physical edge connecting destination node d to its previous node m. Finally, total end-to-end delay in P_{sd} is the sum of delays of each edge (i, j) between the source node s and the destination d, and is given by $d_{sd}^a = \sum_{\forall\ (i,j) \in E^P} d_{ij}^a$.

3.1.2 Virtual network model

Similar to the physical network, the virtual network is modeled as a weighted directed graph $G^V = (N^V, E^V)$, where u and $v \in N^V$ are virtual nodes, and $(u, v) \in E^V$ is a virtual edge. VNR^r is a virtual network request number r out of R total VNRs. Each virtual node $u \in N^V$ is associated with cpu_u^r, representing the demanded CPU capacity, and each virtual edge (u, v) connecting a pair of virtual nodes u and v is also associated with bw_{uv}^r as the demanded bandwidth capacity. Lastly, d_{uv}^r represents the maximum allowed end-to-end delay demanded by virtual edge (u, v).

3.1.3 Power consumption model

The authors in [Dayarathna (2016)] presented a comprehensive survey for the state of the art power consumption models. Accordingly, the linear power model introduced by [Fan (2007)] is identified, which defined a formula to estimate the power consumption of network's servers PC including its idle power. The model approximated the aggregate behavior of a server system while being active, by measuring the total power consumption of the server PC_i against it's CPU utilization. In addition to idle power, the model includes total power consumed by the server when loaded as shown in Fig.3.2. The formula is given as follows:

$\forall\, i \in N^P$

$$PC_i = pw_i^{idle} + [PW_i^{Busy} - pw_i^{idle}] \times \mu_i \tag{3.1}$$

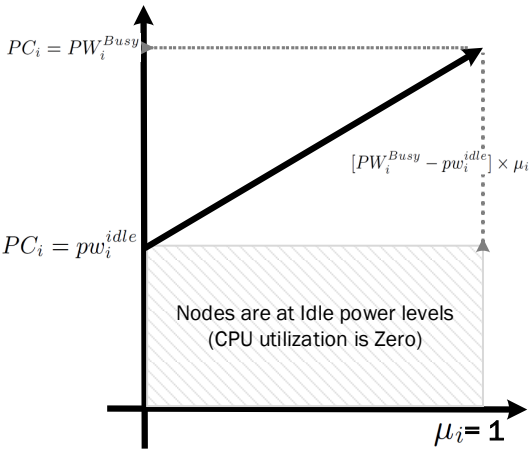

Figure 3.2: Power consumption model

$$\mu_i = \left(\frac{cpu_i}{CPU_i} \times 100 \right) \tag{3.2}$$

3.1.4 Objective function formulation

Following the same analogy of estimating the power consumption of physical network nodes, formula shown in Eq.(3.1) will be applied to formulate the objective function as an integer linear programming (ILP) optimization problem. The main target is to minimize overall power consumption in the whole physical network, by putting into sleeping mode all non utilized physical network resources that are at idle power consumption while accommodating VNRs demands. The rational behind that, is that for all physical network nodes that are at idle mode, still, they are consuming considerable amount of power, even if their consumed CPU were zero. This is because when nodes are at idle mode, the power consumed by chassis (backplane) and cooling systems still represent a considerable amount of the total power [TIA (2017)]. Accordingly, setting them into sleeping mode, will result on minimizing the total physical network power consumption.

To make sure that a specific physical network node is active and hosting at least one virtual node, variable x_i^{ur} is used in the ILP objective function formulation, which takes a binary value of "1" if physical node i is active and assigned to host the virtual node u, and zero otherwise. The objective function is shown bellow:

$\forall u \in N^V and \, \forall r \in R$

$$\min PC_i = \sum_{\forall i \in N^P} (pw_i^{idle} + [PW_i^{Busy} - pw_i^{idle}] \times \mu_i) \times x_i^{ur} \tag{3.3}$$

3.1.5 Constraints formulation

Objective function solution will be constrained by capacity, flow and domain constraints as shown bellow. However, power consumption constraint was intentionally omitted; since it relies on CPU utilization of each node, nevertheless, it will be satisfied if constraints Eq.(3.4) and Eq.(3.5) were satisfied.

Capacity constraints To ensure current available CPU processing power capacity in physical node i is greater than or equal to demanded capacity by virtual network node u.

$$\forall\, i \in N^P \quad cpu_i^a x_i^{ur} \geq cpu_u^r \tag{3.4}$$

To ensure total consumed CPU processing power capacity at physical network node i, is less than or equal to maximum CPU capacity at that physical network node:

$$\forall\, u \leftarrow i \quad \sum_{r \in R} cpu_u^r \leq CPU_i \tag{3.5}$$

Note: $u \leftarrow i$ means that virtual network node u is hosted at physical network node i.

To ensure that current available bandwidth capacity on physical network edge (i, j) is greater than or equal to demanded bandwidth capacity by virtual network edge (u, v):

$$\forall\, (i,j) \in P_{sd} \quad bw_{ij}^a \geq bw_{uv}^r \tag{3.6}$$

To ensure that total consumed bandwidth capacity in physical network edge (i, j), is less than or equal to maximum bandwidth capacity at that edge:

$$\forall\, (u,v) \leftarrow (i,j) \quad \sum_{r \in R} bw_{uv}^r \leq BW_{ij} \tag{3.7}$$

Note: $(u, v) \leftarrow (i, j)$ means that virtual network edge (u, v) is embedded on the physical network edge (i, j).

To ensure that current end-to-end delay in physical network path P_{sd} is less than or equal to maximum allowed delay d_{uv}^r by VNR^r:

$$d_{sd}^a \leq d_{uv}^r \tag{3.8}$$

Flow constraints To ensure that a flow getting in a physical node must go out, the following constraints has to be satisfied:

$$\sum_{\forall n \in N^P} f_{sn}^a - \sum_{\forall n \in N^P} f_{ns}^a = bw_{uo}^r \tag{3.9}$$

$$\sum_{\forall m \in N^P} f_{dm}^a - \sum_{\forall m \in N^P} f_{md}^a = -bw_{pv}^r \tag{3.10}$$

$$\sum_{\forall k,l \in N^P} f_{kl}^a = \sum_{\forall k,l \in N^P} f_{lk}^a \tag{3.11}$$

Constraint Eq.(3.9) ensures that the total flow getting out of source node s is the demanded flow bw^r_{uo}, while constraint Eq.(3.10) ensures that total flow getting into destination node d is the forwarded flow bw^r_{pv}, and constraint Eq.(3.11) ensures that all demanded flow is transfered from source to destination node, and nothing remains at any intermediate node within physical network path P_{sd}.

Domain constraints To solve the problem as ILP:

$$\forall\, i \in N^P\ x_i^{ur} \in \{0, 1\} \tag{3.12}$$

To ensure each virtual node is mapped only to one physical node:

$$\forall\, u \in N^V \sum_{\forall i \in N^P} x_i^{ur} = 1, \tag{3.13}$$

To ensure virtual nodes from the same VNR are mapped to different physical nodes:

$$\forall\, i \in N^P \sum_{\forall u \in N^V} x_i^{ur} \leq 1, \tag{3.14}$$

3.2 Proposed technique to solve power aware resource allocation and virtualization problem

The following sections will discuss the new proposed segmentation technique to solve the two subproblems of the resource allocation problem, by converting the structure of any virtual network request in VNE, or service function chain in NFV, into a collection of pairs of virtual nodes and their edges, where the demands of each pair will be listed together in a set format called segment, and converting the resources of an exact similar number of physical network pairs into segment format as well. The motivation behind using the segmentation approach, is to facilitate allocating all virtual pairs belonging to a specific VNR, on the corresponding physical pairs that have enough resources to host the demands of the virtual pairs' nodes and edges, together and in full coordination without using any additional hidden physical resources. This will guarantee high acceptance ratios for allocating the network services, in addition, segmentation technique will ensure using the least resources, therefore, minimize the power consumption in the whole physical network.

3.2.1 Segmentation technique

Segmentation technique is proposed in this thesis to guarantee full coordinated allocation for the virtual nodes and edges. The technique was recently published by [Hejja (2018)] and [Hejja (2017)] which provides direct way to check one-to-one, if each element in the physical pair's segment has enough resources to host the demands of its counterpart from the virtual pair's segment, precisely, by comparing the first parameter in the virtual segment to the first parameter in the physical segment, the second to the second, and so on for all the remaining parameters. If the results of 'ALL' checks are true, that is, each virtual node demand found a physical node to host it, 'AND', each virtual edge demand found a physical edge to host it, the embedding of

the virtual pair's nodes and edges will be realized, together in full coordination onto the corresponding physical pair.

The virtual pairs' u and v segment is denoted by Seg^r_{uv}, and physical pairs' i and j segment is denoted by Seg_{ij}. The structure of both segments must be similar in format, which means that the number of parameters representing the virtual nodes and edges in the virtual segment Seg^r_{uv}, are exactly similar to the number of parameters representing the physical nodes and edges in the physical segment Seg_{ij}, regardless the values of their parameters.

3.2.2 General definition of the basic segment

For any path P^{ij} belonging to a physical network or is part of a virtual network request, consisting of two nodes i and j, connected by a direct edge (i, j), and does not include any intermediate or hidden nodes, P^{ij} segment, denoted by Seg_{ij} is defined as a list including all parameters of the two nodes and the edge, grouped in a set format as shown in Eq.(3.15).

For example, P^{ij} could be represented by the following specific nodes' parameters, namely, nodes' locations denoted by loc_i and loc_j, processing power capacities cpu_i and cpu_j, and parameters of their connecting edge are bandwidth capacity bw_{ij} and delay d_{ij}. Consequently, the segment defining path P^{ij} can be written in set format as follows:

$$Seg_{ij} = \{loc_i, loc_j, cpu_i, cpu_j, bw_{ij}, d_{ij}\} \tag{3.15}$$

However, in general any physical or virtual directed graph can be reconstructed of one or multiple paths, where each path could be composed of multiples of the basic segment above representing the pairs formulating the corresponding paths. Therefore, to generalize the concept of the basic segment in Eq.(3.15) on any graph, let Set^r be a set of segments listing all demands of the virtual nodes and edges in a VNR^r graph, and let Set^P be a set of segments listing all resources of the candidate physical nodes and edges in a physical graph to host the virtual nodes and edges in the VNR^r.

The formulation of both segments is clarified in the following subsections, starting by the VNR segment:

3.2.3 Segment formulation for VNR

For a virtual network request VNR^r, consisting of a collection of pairs of virtual nodes associated with loc and CPU demands, and virtual edges associated with BW and delay demands, Eq.(3.15) can be generalized to list all these parameters together as shown in Eq.(3.16) as follows:

$$Set^r = \{Seg^r_{uo}, .., Seg^r_{wx}, .., Seg^r_{pv}\} \tag{3.16}$$

Where:

$$Seg^r_{uo} = \{loc^r_u, loc^r_o, cpu^r_u, cpu^r_o, bw^r_{uo}, d^r_{uo}\}$$

$$Seg^r_{wx} = \{loc^r_w, loc^r_x, cpu^r_w, cpu^r_x, bw^r_{wx}, d^r_{wx}\}$$

$$Seg^r_{pv} = \{loc^r_p, loc^r_v, cpu^r_p, cpu^r_v, bw^r_{pv}, d^r_{pv}\}$$

u, v, o, p, w, x are virtual nodes $\in$ VNR^r $graph$

An illustrative application of Eq.(3.16) is shown in Fig.3.3a, which presents two VNRs to be embedded, a line graph VNR^{r1} and a directed cycle graph VNR^{r2}. For example, VNR^{r1} is composed of three virtual nodes and two virtual edges forming two pairs.The first virtual pair includes virtual nodes 1.1 and 1.2, and virtual edge $(1.1, 1.2)$, while the second virtual pair includes virtual nodes 1.2 and 1.3, and virtual edge $(1.2, 1.3)$. Notice that the first virtual node location, $loc^{r1}_{1.1}$, demands being located at the location of physical network node a, while location of second virtual node $loc^{r1}_{1.2}$ at physical network node b, and $loc^{r1}_{1.3}$ at physical network node e. At the top of Fig.3.3b, the formulation of Eq.(3.16) was applied to construct the set of segments, Set^{r1}, representing VNR^{r1}, listing all demanded locs, cpus, bws, and the overall end-to-end delay. Accordingly, Set^{r1} is formulated as follows:

$$Set^{r1} = \{Seg^{r1}_{1.1,1.2}, Seg^{r1}_{1.2,1.3}\}$$

Where:

$$Seg^{r1}_{1.1,1.2} = \{loc^{r1}_{1.1}, loc^{r1}_{1.2}, cpu^{r1}_{1.1}, cpu^{r1}_{1.2}, bw^{r1}_{1.1,1.2}, d^{r1}_{1.1,1.2}\}$$
$$Seg^{r1}_{1.2,1.3} = \{loc^{r1}_{1.2}, loc^{r1}_{1.3}, cpu^{r1}_{1.2}, cpu^{r1}_{1.3}, bw^{r1}_{1.2,1.3}, d^{r1}_{1.2,1.3}\}$$

In this way, the demands of VNR^{r1} are translated from a line graph topology, into a set of segments representing the demands of the two pairs formulating VNR^{r1}. The same procedure can be repeated for VNR^{r2} as shown at the top of Fig.3.3c.

3.2.4 Segment formulation for a physical graph

For any physical graph consisting of similar pairs of nodes and edges as the VNR's pairs, without any hidden hopes or edges, reformulate its resources into a set of segments, listing the physical pairs' loc, CPU, BW, and delay values together as shown in Eq.(3.17).

$$Set^{P} = \{Seg_{sn}, .., Seg_{kl}, .., Seg_{md}\} \tag{3.17}$$

Where:

$$Seg_{sn} = \{loc_p, loc_n, cpu_p, cpu_n, bw_{sn}, d_{sn}\}$$
$$Seg_{kl} = \{loc_k, loc_l, cpu_k, cpu_l, bw_{kl}, d_{kl}\}$$
$$Seg_{md} = \{loc_m, loc_d, cpu_m, cpu_d, bw_{md}, d_{md}\}$$

s, d, k, l, n, m are physical nodes $\in$ $physical\ graph$

To complete the above example, the bottom of Fig.3.3b illustrates a direct application on how to use Eq.(3.17). Since the demanded virtual nodes' locations are the locations of the nodes on physical path P^{ae}, then the set of segments representing P^{ae}, denoted by Set^{ae}, represents a typical similar structure to that of Set^{r1}. Similar in this example means, since VNR^{r1} topology is a line graph, consisting of two pairs, with specific virtual

nodes' locations, then the selected physical graph topology must be a line graph, of two pairs fulfilling the demands of the virtual nodes' locations. Accordingly, applying Eq.(3.17) to formulate the set of segments, Set^{ae}, representing physical graph P^{ae} can be written as follows:

$$Set^{ae} = \{Seg_{ab}, Seg_{be}\}$$

Where:

$$Seg_{ab} = \{loc_a, loc_b, cpu_a, cpu_b, bw_{ab}, d_{ab}\}$$
$$Seg_{be} = \{loc_b, loc_e, cpu_b, cpu_e, bw_{be}, d_{be}\}$$

In this way, physical line graph path P^{ae} has been translated from a graph topology into segment format, fulfilling the virtual nodes' demanded locations and similar in structure to Set^{r1}. The same process can be followed to formulate the set of physical segments to allocate VNR^{r2} as shown in Fig.3.3c.

Now to embed all virtual nodes and edges from VNR^{r1} on the physical path P^{ae}, in full coordination, each element in the two formulated set of segments, Set^{r1} and Set^{ae} will be directly compared one-to-one, which means for each parameter value in Set^{r1} representing a virtual node in VNR^{r1}, check if its counterpart in Set^{ae} can fulfill its demands, and at the same time, for each parameter value in Set^{r1} representing a virtual edge in VNR^{r1}, check if its counterpart in Set^{ae} has enough residual capacity to host it as well. This process can be achieved ***if and only if*** each comparison statement in the following (if-AND) conditions is true, as follows:

$$\textbf{\textit{if}}\ loc_a = loc^{r1}_{1.1}\ \textbf{AND}$$
$$\textbf{\textit{if}}\ loc_b = loc^{r1}_{1.2}\ \textbf{AND}$$
$$\textbf{\textit{if}}\ loc_e = loc^{r1}_{1.3}\ \textbf{AND}$$
$$\textbf{\textit{if}}\ cpu_a - cpu^{r1}_{1.1} \geq 0\ \textbf{AND}$$
$$\textbf{\textit{if}}\ cpu_b - cpu^{r1}_{1.2} \geq 0\ \textbf{AND}$$
$$\textbf{\textit{if}}\ cpu_e - cpu^{r1}_{1.33} \geq 0\ \textbf{AND}$$
$$\textbf{\textit{if}}\ bw_{ab} - bw^{r1}_{1.1,1.2} \geq 0\ \textbf{AND}$$
$$\textbf{\textit{if}}\ bw_{be} - bw^{r1}_{1.2,1.3} \geq 0\ \textbf{AND}$$
$$\textbf{\textit{if}}\ d_{ab} \leq d^{r1}_{1.1,1.2}\ \textbf{AND}$$
$$\textbf{\textit{if}}\ d_{be} \leq d^{r1}_{1.2,1.3}$$

Accordingly, if all the above conditions were proven true, this guarantees that each virtual node and edge in VNR^{r1} will be hosted by their counterparts in physical path P^{ae}. Therefore, segments formulation and the comparisons, made the embeddings of virtual nodes fully coordinated with the embeddings of the virtual edges, together on the same physical path. The same comparison procedure can be followed to check if the demands of VNR^{r2} can be hosted by the directed cycle physical graph P^{mno}.

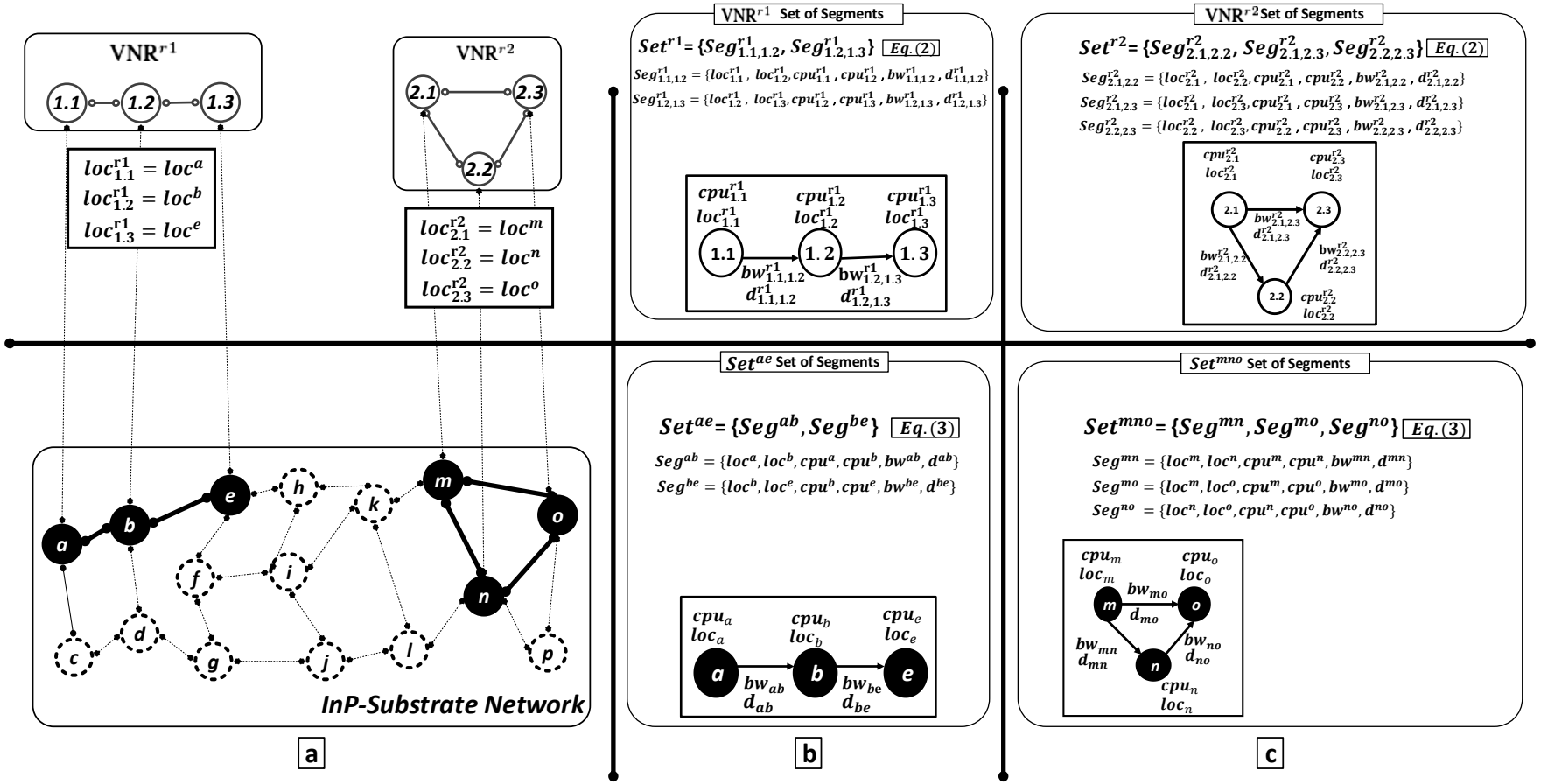

Figure 3.3: Example about constructing the physical path segment, Set^P, and VNR segment, Set^r.

3.3 Proposed power aware resource allocation and virtualization algorithm for offline, PaCoVNE

Optimal solution for the resource allocation problem is known to be NP-Hard and computationally intractable, since it can be reduced to multi-way separator problem, which is NP-Hard by itself [Chowdhury (2012)]. As a summary, [Ilhem (2012)] listed some of the main reasons highlighting why solving resource allocation problem as in the VNE environment is challenging, such as: randomness of the arrival of VNRs depending on users' demands, topology and resources constraints by each VNR, and limited physical network resources. However, the virtual edges embedding problem is what makes the VNE problem exceptionally an NP-hard, because it could be mapped to one or more physical edges that are not necessarily physically connected. Even for offline VNE case, given that all nodes were embedded, still virtual edge embedding stage can be reduced to the unsplittable flow problem, which is NP-hard [Bradley (1977)] [Kolliopoulos (1997)]. Consequently, solving VNE problem in polynomial time is not possible.

Therefore, majority of VNE approaches followed algorithm or meta-algorithms to solve VNE optimization problems in a reasonable polynomial time [Fischer (2013)]. For example, one of the most referenced VNE algorithm approaches is the algorithm presented by [Chowdhury (2012)]. It coordinates node and edge embedding, through mapping virtual nodes onto physical nodes in a way that facilitates mapping of virtual edges. Nevertheless, the authors performed virtual node allocation and virtual edge allocation in two interrelated stages. First they designed a node embedding algorithm to embed the virtual nodes on a suitable physical nodes, which could be separated a part from each other. Second, once node mapping was successful, they triggered another algorithm to embed the associated virtual edges on physical paths, noting that it mostly would include hidden nodes to be used as hops. However, other ideas could be explored to better coordinate embedding virtual node allocation and virtual edge allocation stages, and at the same time avoid including non necessary hidden hops and edges beyond VNRs needs.

The following sections will discuss the proposed algorithm called, Power Aware Coordinated Virtual Network Embedding, abbreviated as "PaCoVNE". PaCoVNE uses the segmentation technique to solve VNE optimization problem more efficiently, where it coordinates embeddings of the virtual nodes and edges in one step, based on matching each element in VNR^r segment, Seg^r, against their counterparts in the physical network path's segment, Seg^P, considering the following constrains, namely: CPU, BW, in addition to end-to-end delay constraints.

3.3.1 Explaining the algorithm

Pseudo-code for PaCoVNE algorithm is shown in Algorithm 3.1 bellow, and is explained by the following main four steps:

3.3.2 Initialization

It starts by generating physical network topology, lists all its possible paths, and categorizes them into types according to the number of nodes and edges per each physical network path. Notice that, the number of lists and paths per list varies depending on the size and topology of the physical network. Since physical network topology is physically fixed in real life, the main elements formulating any physical network path, namely number and connectivity of the physical network nodes and edges, are also fixed and does not change, but only

their capacities varies due to consumption. Therefore, to avoid searching for physical network paths while VNE algorithm is running, and in contrary to most available algorithms in literature, the proposed algorithm performs the initialization step in advance before treating the VNRs. This is will be a main advantage for PaCoVNE's speed when solving the VNE problem, given it mainly focuses on the actual mapping process itself. To facilitate recalling a specific list of physical network paths by PaCoVNE algorithm whenever it receives a new VNR, these lists will be saved and categorized per path type in a database repository, including number of nodes, edges, and connectivities for each path.

Algorithm 3.1, PaCoVNE Pseudo-Code

1. Input: G^P and G^V.
2. **for** each VNR$^r \in R$ **do**

   -Formulate VNR^r parameters into segment Seg^r according to Eq.(3.16).
3. For the set of all saved physical network paths P^P:

   List all physical network paths matching VNR^r size.

   Rank them in descending order based on μ_i according to Eq.(3.2)
4. For top ranked physical network path P_{sd}, formulate its segment Seg^p according to Eq.(3.17).
5. **Compare** Seg^r against Seg^P

   Check for CPU, BW and Delay constraints according to Eq.(3.18).
6. **If** satisfied,

   embed VNR^r on P_{sd}.

   else go to next ranked physical network path, step-4.
7. **for** all physical network nodes and edges **do**

   Update CPU and BW resources.

   Remove the embedded VNR^r from VNRs list.
8. **for** idle physical network nodes **do**

   Turn them off to save power.
9. Evaluate Metrics.
10. **If** VNRs list not empty, **go** to next VNR step-2.

3.3.3 Segmentation and ranking

This is the differentiating feature of PaCoVNE algorithm compared to others, mainly because it facilitates accommodating VNRs one by one and embed their nodes and edges in one step and in full coordination between virtual node allocation and virtual edge allocation. First the algorithm formulates VNR^r segment Seg^r. Then, to formulate the candidate physical network path segment Seg^P, it recalls the appropriate list of physical network paths that has similar number of nodes and edges as those in the VNR^r. Next, it ranks them according to their CPU utilization, and ends by formulating the physical network segment for the top ranked path.

3.3.4 Embedding decision

It compares each element in the physical network segment Seg^P to its counterpart in the Seg^r, one-by-one. Accordingly, if the physical network segment has enough resources to accommodate all demands of the VNR^r, PaCoVNE selects the path of physical network segment Seg^P to host that VNR^r. Decision matrix for the embedding process is shown in Eq.(3.18) bellow:

$$
\begin{aligned}
if\ cpu_i^a - cpu_u^r \geq 0\ and \\
if\ bw_{ij}^a - bw_{wx}^r \geq 0\ and \\
if\ d_{sd}^a \leq d_{uv}^r
\end{aligned}
\tag{3.18}
$$

3.3.5 Updating

Once a successful embedding occurs, the algorithm updates all changed physical network resources and moves to the next VNR. However, in case that the physical network segment does not have enough resources to accommodate the VNR demands, the algorithm jumps to the next ranked path, and follow on from step 5. This process keeps on going until no more VNRs to be handled.

3.3.6 Computational time complexity

Regardless the number of VNRs and based on the adjacency matrix of the physical network, searching and listing all types of paths will consume $O(|N^P| + |E^P|)$ processing time, depending on total number of nodes N and edges E formulating the physical network [Kleinberg (2009)]. This step is performed and saved only once before the arrival of any VNR. Therefore, it will not have any impact on the real computational time complexity of the VNE process.

However, the actual VNE process starts when the first VNR is prepared to be allocated on the physical network. Therefore, in order to evaluate computational time complexity of PaCoVNE at worst case, the focal computational component of the algorithm is determined based on the time consumed while sorting all listed physical network paths that has the same number of nodes and edges as the VNR^r. The larger the number of the listed paths, the more computational time is consumed by the working machine.

Accordingly, for each VNR^r, the PaCoVNE adopted (Bubble Sort) algorithm to rank all the physical network paths in descending order [Kleinberg (2009)]. Thus, at the worst case, the PaCoVNE algorithm will have a quadratic computational time complexity in the order of $O(n^2)$, where n is number of paths.

3.3.7 Understanding PaCoVNE through example

A detailed example to explain the proposed algorithm is shown in Fig.3.4. It applies PaCoVNE on a physical network of four nodes as shown in stage A, then it evaluates how to accommodate VNR^1, by sorting all listed physical network paths based on the total sum of CPU utilizations 'μ' for each path. As shown in stage B, the PaCoVNE concludes by embedding VNR^1 on nodes 2 and 3, along path P_{23}, which had enough resources to accommodate its demands. In this case, the algorithm managed to save 21% of the total consumed power in the whole physical network, by turning-off nodes 0 and 1, since they were idle. Stage C introduced VNR^2,

the PaCoVNE decides that even though P_{23} is still the top ranked path, based on its CPU utilization, but since it does not have enough BW resources to accommodate the demanded BW by VNR^2, it jumps to next ranked physical network path, P_{02}, which satisfies all demands of VNR^2. Therefore, PaCoVNE assigns P_{02} to accommodate VNR^2, then it keeps node 1 turned-off, since its the only idle node, resulting on saving 12% of the total consumed power by the whole physical network.

3.3.8 PaCoVNE performance evaluation

In this section, offline version of PaCoVNE algorithm was tested using homogeneous and heterogeneous settings, with end-to-end delay and without it, and the results were published in [Hejja (2017)]. The homogeneous version was compared to one of the most referenced algorithms, the energy aware relocating algorithm 'OCA/EA-RH' developed by [Botero (2013b)]. Then, for the heterogeneous scenario, PaCoVNE was compared to its homogeneous version.

For the offline homogeneous scenario without delay, PaCoVNE was compared to OCA/EA-RH algorithm, which only used VNRs of 15 nodes; denoted as ($VNRs_{15}$), and therefore, the same simulation settings will be applied for PaCoVNE as well [Botero (2013b)]. Specifically, the physical network will handle a set of 80 VNRs, for different average loads, denoted by $\rho \in \{0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9\}$. The value of each load ρ, reflects a ratio between VNRs demands to physical network capacities, and therefore, loading the physical network with X%, means this is the average of loading each node by X% load as well. The values of physical network's nodes cpu_i^a and bw_{ij}^a resources were set to uniformly distributed values equal to 100. For each VNR, cpu_w^r and bw_{wx}^r values were estimated from the average embedding cost figure of [Botero (2013b)], and are given as follows: $cpu_w^r = 2.1$ and $bw_{wx}^r = 2.3$. Finally, maximum power consumption by each physical network node PW_i^{Busy} was set to 524 watts, while its idle power pw_i^{idle} was set as ($PW_i^{Busy} * 0.4$) [TIA (2017)]. For physical network edges, end-to-end delay d_{ij}^a, was set equal to $250ms$ as a limit [ITU-T T-REC-Y.3101 (2018)],[RFC7679 (2017)]. While virtual network delays d_{wx}^r, was selected pseudorandomly between $100 - 250ms$. Table.3.1 summarizes all simulation settings to compare PaCoVNE against OCA/EA-RH for the offline and Homogeneous scenario.

Physical network topologies were generated as directed graphs, through Waxman algorithm according to the following parameters: $\alpha = 0.6$, $\beta = 0.23$, and mean probability of creating an edge between any two physical network nodes, denoted as p_{wax} was set equal to 0.2. Important to notice that, these parameters differ from what [Botero (2013b)] used, since the aforementioned parameters will provide average edges at each physical network node of 6, instead of 12 as used by [Botero (2013b)]. This caused PaCoVNE algorithm to rank much less number of paths, yet, it produced better results compared to OCA/RA-EH. To overcome the probabilistic nature of Waxman topology generation, the set of 80 VNRs were run for 50 times per each ρ load value.

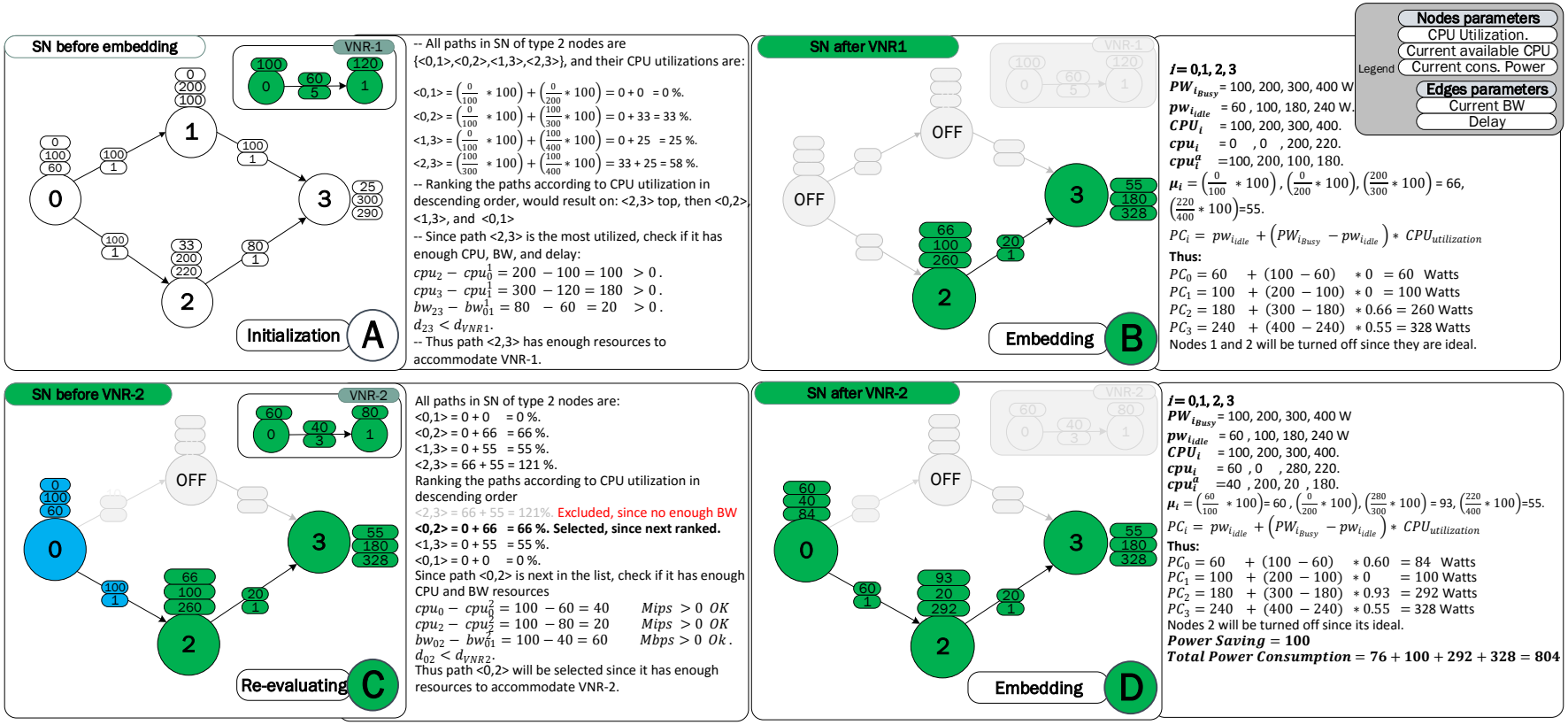

Figure 3.4: Numerical example showing basics of PaCoVNE

Table 3.1: Simulation settings for Offline Homogeneous Scenario

Parameter	Physical network	VNR
Nodes	50	15
$CPU\ max$	100	2.1
$BW\ max$	100	2.3
$Delay\ max$	250	$100 - 250$
PW^{Busy}	524	
pw^{idle}	$PW^{Busy} * 0.4$	
Loads	$0.2 - 0.9$	
Runs/load	50	
α	0.6	
β	0.23	
p_{wax}	0.2	

3.3.9 PaCoVNE work-flow tracing

Initialization based on the physical network adjacency matrix, the algorithm lists all physical network paths of 15 nodes, denoted as $P_{15} \in P^P$, this is only performed once and saved at the beginning. These paths can then be used for any number of $VNRs_{15}$. This is important, since the algorithm will focus on the actual embedding process itself, and not on searching for the best path at the arrival of each new VNR, which saved PaCoVNE's run time considerably. Moreover, its important to mention that PaCoVNE algorithm can generate all types of physical network paths at the beginning, and accordingly, it can handle any type of VNRs regardless of how many nodes they may contain.

VNR segment formulation Since OCA/EA-RH algorithm used VNRs_{15}, then PaCoVNE formulates $Seg^{r_{15}}$ following Eq.(3.16).

Physical network path segment the algorithm selects the path of highest μ, denoted as $P^a \in P_{15}$, and formulates its segment $Seg^{a_{15}}$ following Eq.(3.17).

Ranking: For each path $P^a \in P_{15}$, PaCoVNE calculates and sums its μs, then it ranks the paths based on the value of its μ from highest to lowest.

Embedding The algorithm compares both segments according to Eq.(3.18), if the conditions are satisfied, then it embeds $\text{VNR}^{r_{15}}$ on physical network path P^a.

Turning off idle physical network nodes Once $\text{VNR}^{r_{15}}$ is embedded successfully, PaCoVNE identifies all idle physical network nodes and turns them off to save power consumption. Next it updates all physical network elements based on that.

3.3.10 Evaluation metrics

The PaCoVNE algorithm will be evaluated according to following metrics:

• *Average total power consumption, PW:* defined as the total power consumed by all physical network nodes after each VNR embedding, and averaged over the total number of VNRs R [Botero (2013b)],

$\forall \rho \in \ Loads$,

$$PW = \frac{1}{R}(\sum_{\forall r \in R} \sum_{\forall i \in N^P} PC_i) \tag{3.19}$$

• *Average saved power, PS:* the amount of saved power after embedding each VNR, using the proposed power reduction strategy. Calculated by subtracting total power consumed by all physical network nodes without power reduction strategy PW^-, from total power consumed by all active physical network nodes after applying power reduction strategy PW^+. The results will be averaged over the total number of VNRs R.

$\forall \rho \in \ Loads$,

$$PS = \frac{1}{R} \sum_{\forall r \in R} (\sum_{\forall i \in N^P} PW^- - \sum_{\forall i \in N^P} PW^+) \tag{3.20}$$

•Average acceptance ratio, AR is a ratio to represent how PaCoVNE algorithm is performing, calculated for each load value ρ, by dividing number of successfully embedded VNRs by total number of VNRs R [Chowdhury (2012)],[Botero (2013b)].

$\forall \rho \in \ Loads$,

$$AR = \frac{1}{R} \ Total \ Number \ of \ Embdded \ VNRs * 100 \tag{3.21}$$

• *Average cost of embedding VNRs, EC:* is the sum of total consumed physical network resources CPU and BW while embedding each VNR. Tuning parameters to represent relative costs per each physical network resource, denoted as α for physical network nodes' cost, and β for physical network edges, were both set equal to one [Chowdhury (2012)],[Botero (2013b)].

$\forall \rho \in \ Loads$,

$$EC = \frac{1}{R} \sum_{\forall r \in R} (\sum_{\forall (i,j) \in E^P} (\beta * bw_{ij}) + \sum_{\forall i \in N^P} (\alpha * cpu_i)) \tag{3.22}$$

•Average CPU utilization, CPU_{util} it represents physical nodes' utilization trend after all simulation iterations. Its defined as ratio between consumed CPU cpu_i, and maximum CPU resources, averaged overall VNRs for each load ρ [Chowdhury (2012)].

$\forall \rho \in \ Loads$,

$$CPU_{util} = \frac{1}{R} \sum_{\forall r \in R} (\sum_{\forall i \in N^P} \frac{(CPU_i - cpu_i^a)}{CPU_i} * 100) \tag{3.23}$$

•Average BW utilization, BW_{util} it represents utilization of physical edges after all simulation iterations. And is defined as ratio between consumed bw_{ij}, and the maximum BW, averaged overall VNRs for each load ρ [Chowdhury (2012)].

$\forall \rho \in \ Loads$,

$$BW_{util} = \frac{1}{R} \sum_{\forall r \in R} (\sum_{\forall (i,j) \in E^P} \frac{(BW_{ij} - bw_{ij}^a)}{BW_{ij}} * 100) \tag{3.24}$$

3.3.11 PaCoVNE results and discussion

Offline homogeneous scenario

Simulation results in Fig.3.5a shows that PaCoVNE performed similar or better than EA-RH in terms of acceptance ratio for lower loads, and is much better for higher loads, thanks to the one stage, full coordinated embedding and segment formulation of PaCoVNE, which makes sure to allocate virtual nodes and their associated edges together, and at the same time, thus, increasing the acceptance ratio. In comparison, the OCA/EA-RH relocates least stressed virtual nodes and their associated edges to other suitable active physical nodes that are more stressed, using the cost-based VNE approach of [Chowdhury (2012)]. Then in a separate phase, OCA/EA-RH relocates least stressed edges to shortest energy path.

However, in terms of power consumption at physical nodes, PaCoVNE can not be compared to OCA/EA-RH, since both algorithms used different formulas to calculate the consumed power per each physical node. Nevertheless, Fig.3.5b shows that physical network power consumption is still high, giving that PaCoVNE model includes idle power in addition to power consumption when physical nodes were loaded according to their CPU utilization. This entails the importance of considering idle power as a main component for increasing power consumption of physical nodes, even if they do not process any data.

Moreover, regarding saved power results shown in Fig.3.5c clarifies that, when the load was 0.2 PaCoVNE managed to save 65% of physical network's total power, and when the load was much increased to 0.9 it saved 49%, implying that, in a range of loads between 0.2 to 0.9, PaCoVNE would save in average 57% of physical network's total power consumption, by putting idle nodes into sleeping mode, while maintaining high VNE acceptance ratios across almost all loads. These results highlights the benefits of using PaCoVNE's new segmentation strategy to fully coordinate VNE, also pinpoints the obvious impact of idle power consumption on the overall physical network's power consumption, thus, reducing it would ultimately reduce physical network costs. Indeed, important to point out that in real life conditions, putting idle nodes into sleeping mode as a power reduction strategy, could affect service maintainability of physical network, especially considering on-line scenarios. Therefore, other strategies could be explored as well.

In the case of including end-to-end delay, Fig.3.5a,b,c shows the obvious impact of end-to-end delay. In comparison to PaCoVNE homogeneous without delay, acceptance ratio was degraded by 24% in average for all loads, increased power consumption by 57%, and reduced saved power by 51%. These results implies the significance of including end-to-end delay as a main constraint to embed VNRs, and how negatively it would impact the whole VNE process.

Offline homogeneous against heterogeneous scenarios

The rational behind comparing PaCoVNE using homogeneous to heterogeneous configuration is to give some insights about how PaCoVNE would behave on semi-real life conditions, where physical network resources usually differ in size and capacity, in addition to including end-to-end delay. Table.3.2 summarizes the heterogeneous simulation settings.

Table 3.2: Simulation settings for Offline Heterogeneous scenario

Parameter	Physical Network	VNR
Nodes	50	15
CPU_{max}	$Random$ 40 – 100	$Random$ 1.5 - 2.1
BW_{max}	$Random$ 40 – 100	$Random$ 1.6 - 2.3
$Delay_{max}$	Random 100 – 250	Random 100 – 250
PW_{Busy}	524	
pw_{Idle}	PW_{Busy} * 0.4	

Fig.3.5d,g,h,i shows simulation results considering heterogeneous conditions, indicating the outstanding performance of PaCoVNE in homogeneous scenario in terms of acceptance ratio, saved power, CPU and BW utilizations with and without end-to-end delay. In terms of power consumption and embedding cost as shown in fig.3e and 3f, PaCoVNE in heterogeneous scenario performed much worse than in homogeneous scenario, mainly due to PaCoVNE's rapid tendency to utilize the physical network resources. This is clearly translated into less accepted VNRs as loads increases. In addition to that, almost same conclusion can be deduced when end-to-end delay was applied, showing that the resultant metrics for the heterogeneous performed even much worse than the homogeneous across all metrics and loads, doubling down the significance of including end-to-end delay as a main VNE constraint.

3.4 Conclusions

This chapter introduced the proposed general modeling structure of the power aware resource allocation problem, in an integer linear programming formulation, clarifying the problem's general mathematical formulation, and presenting the constrains that govern the relationships between the physical network resources and the network services' demanded resources. Then the chapter introduced the new proposed technique to coordinate allocating the virtual nodes and edges together in full coordination. Finally, the general structure of a new proposed power aware resource allocation and virtualization algorithm, PaCoVNE, was detailed in depth, which will be generalized in chapter 4 to solve an online VNE problem, and in chapter 5 to solve the resource allocation problem for NFV environment. Simulation results of PaCoVNE showed that it maintained high acceptance ratios in averages of 96%, and managed to save 57% of the physical network's total power consumption by putting idle nodes into sleeping mode. In the case of including end-to-end delay, the acceptance ratio of PaCoVNE in homogeneous scenario without delay was better by 24% in average for all loads, decreased power consumption by 57%, and increased saved power by 51%. These results implies the significance of including end-to-end delay as a main constraint to embed VNRs, and how negatively it would impact the whole VNE process.

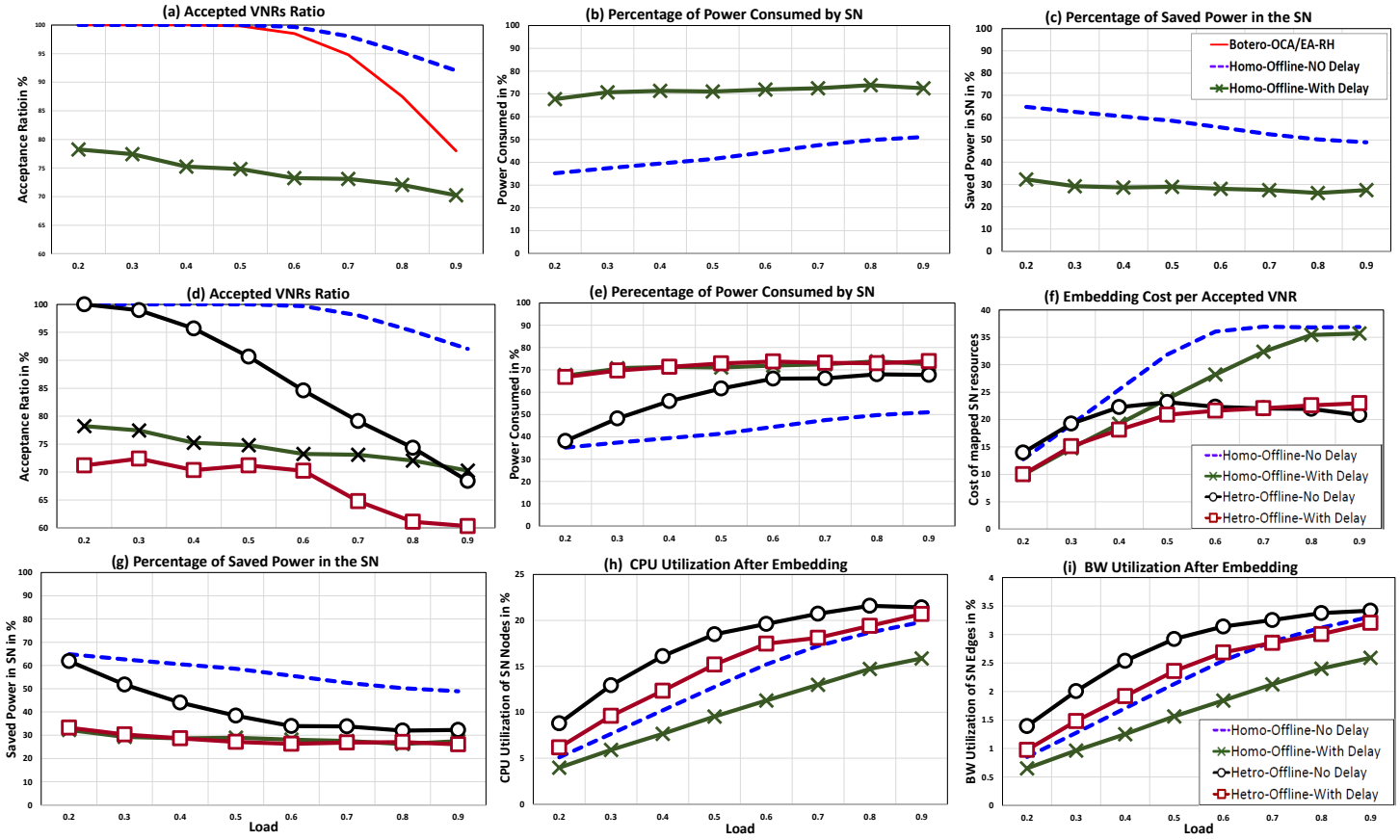

Figure 3.5: Comparison Results of PaCoVNE Homogeneous against OCA/EA-RH and PaCoVNE Heterogeneous

CHAPTER 4

Online VNE Power Aware Resource Allocation

4.1 Introduction

The main motivation of this chapter is to introduce a new efficient embedding algorithm to solve the generic power aware resource allocation VNE grand problem as in Chapter-3, on real-time bases and including end-to-end delay constraint, while minimizing the overall power consumption in the physical network. The proposed online power aware and fully coordinated VNE algorithm, denoted by (oPaCoVNE), allocates the virtual nodes and the virtual edges during one stage according to more realistic constraints, such as nodes' processing power, links' bandwidth and end-to-end delay. Moreover, the benefits of the new proposed algorithm will be reflected on realizing real time virtual network requests for 5G applications that are sensitive to delay, in addition to solving the VNE problem for large networks, using less resources, and generating lower costs and higher revenues.

The core idea of oPaCoVNE can be summarized as follows, for every virtual network request VNR^r number r, oPaCoVNE will apply the segmentation technique from Chapter-3 to reconstruct VNR^r into a collection of pairs, where each of the pairs will be composed of two virtual nodes and their edge, then the demands of each pair will be listed in a set format, representing all parameters of the two virtual nodes and their edge as in Eq.(3.15), including node's locations and their CPUs processing powers, in addition to their edge's demanded bandwidth BW and delay. As a result of that, oPaCoVNE will consider all pairs of any VNR as a set of segments to be handled collectively. To perform the embedding process, oPaCoVNE will afterwards select a similar number of physical pairs, each composed of two nodes and one edge only, which comply with the demanded locations of the virtual pair's nodes and edge, then it will reformulate their resources into segment format similar to the structure of the virtual segment format, representing the physical pairs' parameters based on Eq.(3.15). Ultimately, the algorithm will compare each parameter from the virtual segment to its corresponding parameter in the physical segment, and if all comparisons were true, oPaCoVNE will embed the virtual nodes and their edge from each virtual pair onto its physical counterpart.

In light of that, the proposed segmentation approach will allow oPaCoVNE to coordinate virtual nodes and edges' embeddings together, on a corresponding physical nodes and edges. This will guarantee high acceptance ratios, and efficiently utilizes the limited capacities in the physical network without any additional recourses, therefore, resulting on minimizing the total power consumption in the physical network. Subsequently, oPaCoVNE's use of the segmentation approached, differentiated it from the VNE's two steps approach adopted by literature, which used to apply greedy algorithm to embed all virtual nodes first, then run the shortest path algorithm to embed the virtual edges as a second step, with partial or no coordination with the nodes' embedding step.

General performance and evaluations of oPaCoVNE algorithm will be conducted against one of the most referenced online VNE algorithms in literature, the D-ViNE developed by [Chowdhury (2012)], 2012), and the power aware performance of oPaCoVNE was also compared against the online power aware VNE algorithm, EAD-VNE developed by [Zhongbao (2015)]. Finally, this chapter provides additional analysis regarding the impacts on oPaCoVNE's performance when end-to-end delay was used, in addition to analyzing power consumption, acceptance ratios and processing time of the algorithm, by varying number, lifetime, and size of the embedded VNRs, as well as varying number of edges in the physical network.

Main contributions:

1. As a main contribution by this chapter, impacts of end-to-end delay on the performance of the suggested online algorithm, oPaCoVNE, were deeply analyzed, representing direct application for virtualization in future 5G networks.
2. To minimize the total power consumption in the whole physical network, segmentation approach from Chapter-3 was proposed to guarantee coordinating the embeddings of the virtual nodes and edges, while utilizing the least physical resources to the minimum.
3. Additional evaluations for the proposed online algorithm, oPaCoVNE, were also introduced, showing the impacts of end-end-delay, by varying number and size of the embedded VNRs, as well as varying their lifetimes.

4.2 VNE evaluation scenario

In 5G networks, services, applications, and users interact with the infrastructure network instantly and on real-time. Therefore, in the context of networks' virtualization, the demands must be analyzed and allocated on the physical network, online, utilizing the shared resources efficiently, adhering to the required service qualities as demanded, and at the same time keep updating the statue of the physical network's resources regularly, in order to spontaneously evaluate the possibilities of allocating other virtualization demands when they arrive.

Based on that, the proposed algorithm by this chapter, oPaCoVNE is designed for online scenario, and its main objective is to successfully embed the VNRs, on real time, while minimizing the overall power consumption in the whole physical network considering end-to-end delay as a main embedding constraint. The algorithm handles the virtual network requests one-by-one, and keeps monitoring and updating the physical network frequently, to free up more resources for future usage by new virtual requests.

Detailed online problem formulations, simulation and evaluation for the online scenario are presented in the following section.

4.3 VNE online Problem formulation

To formally introduce the online power aware resource allocation VNE problem as an ILP, the following paragraphs will introduce network and power consumption models of the VNE, followed by the objective functions and its constraints. In addition to that, the proposed online algorithm that solves the VNE in

polynomial times will be explained in details, including an illustrative example, its computational complexity, as well as introducing the algorithm's evaluation metrics.

4.3.1 Physical network model

The physical network $G^P = (N^P, E^P)$ is modeled as a weighted directed graph, where i and $j \in N^P$ are physical network nodes, and $(i, j) \in E^P$ is an edge connecting nodes i and j. Each node $i \in N^P$ is associated with loc_i denoting the node's location, cpu_i^a representing current available CPU capacity, cpu_i for consumed CPU capacity, and CPU_i as the maximum CPU capacity of node i. Each physical edge (i, j) is associated with bw_{ij}^a, bw_{ij}, and BW_{ij} representing available, consumed, and maximum bandwidth capacities. It is assumed that the physical network is well established and stable, therefore it would be possible to extract all the physical paths in advance. Accordingly, $P^P = \{(i, j)\}$ represents the set of all directed basic paths in the whole physical network, where each basic path is connecting a pair of physical network nodes i and j with an edge. And physical path $P^{sub} \in P^P$ represents a candidate hosting physical topology, formulated from a collection of the basic segments P^{ij}, each constructed of a pair of two nodes and one physical edge, and its end-to-end delay is given by d_{ij}^a.

4.3.2 Virtual network model

The virtual network is modeled as a weighted directed graph $G^V = (N^V, E^V)$, where u and $v \in N^V$ are virtual nodes, and $(u, v) \in E^V$ is a virtual edge. VNR^r is a virtual network request number r out of R total VNRs. For online scenario, each VNR has a lifetime interval denoted by t, arrival time denoted by t_a^r, and expires at t_e^r. Each virtual node $u \in N^V$ is associated with loc_u denoting the preferred location to host the virtual node, cpu_u^r representing the demanded CPU capacity during time t, and each virtual edge (u, v) connecting a pair of virtual nodes u and v, is also associated with bw_{uv}^r, representing the demanded bandwidth capacity during time t, and d_{uv}^r representing the maximum allowed end-to-end delay demanded by virtual edge (u, v).

4.3.3 Power consumption model

For the online scenario, the linear power model introduced by [Dayarathna (2016)][Fan (2007)] is used to estimate power consumption of physical network servers (PC), including idle power as given in Eq.(4.1). Nodes' utilization, U_i, is a fraction between (0-1), reflecting the ratio of consumed cpu_i to the maximum CPU_i given by Eq.(4.2).

$\forall\, i \in N^P$

$$PC_i = pc_i^{idle} + [PC_i^{Busy} - pc_i^{idle}] \times U_i \tag{4.1}$$

$$U_i = \frac{cpu_i}{CPU_i} \tag{4.2}$$

4.3.4 Objective function formulation

The main target of the objective function for oPaCoVNE is to successfully accommodate all the demands of the arriving VNRs on online scenario, while minimizing overall power consumption in the whole physical

network, by putting into sleeping mode all idle resources. Demands in the online case arrive at certain t_a^r and expire at t_e^r, therefore, the objective function must consider embedding VNRs during the time intervals specified by each related VNR^r.

To make sure that a specific physical node is active, variable x_i^{ur} is used in the objective function formulation, which takes a binary value of 1 if physical node i is active and assigned to host the virtual node u ($u \leftarrow i$), during the time interval $t \in [t_a^r, t_e^r]$, and 0 otherwise. The objective function is shown in Eq.(4.3):

$\forall\, t \in [t_a^r, t_e^r],\ \forall\, u \in N^V and\, \forall\, r \in R$

$$\min PC = \sum_{\forall i \in N^P} (pc_i^{idle} + [PC_i^{Busy^t} - pc_i^{idle}] \times U_i^t) \times x_i^{ur} \tag{4.3}$$

Constraints formulation

Objective function solution will be constrained by location, capacity, flow, and domain constraints as shown bellow.

Location constraints

$$\forall\, u \leftarrow i \quad loc_u^r = loc_i \tag{4.4}$$

$$\forall\, (u,v) \leftarrow (i,j) \quad Adj_{(u,v)}^r = Adj_{(i,j)}^{sub} \tag{4.5}$$

Capacity constraints

$$\forall\, t \in [t_a^r, t_e^r],\ \forall\, u \leftarrow i \quad \sum_{r \in R} cpu_u^{r^t} \leq CPU_i \tag{4.6}$$

$$\forall\, t \in [t_a^r, t_e^r],\ \forall\, (u,v) \leftarrow (i,j) \quad \sum_{r \in R} bw_{uv}^{r^t} \leq BW_{ij} \tag{4.7}$$

$$\forall\, t \in [t_a^r, t_e^r],\ d_{ij}^a \leq d_{uv}^{r^t} \tag{4.8}$$

Flow constraints

$\forall\, t \in [t_a^r, t_e^r]$

$$\sum_{\forall m \in N^P} f_{sm}^t - \sum_{\forall m \in N^P} f_{ms}^t = bw_{uo}^{r^t} \tag{4.9}$$

$$\sum_{\forall m \in N^P} f_{dm}^t - \sum_{\forall m \in N^P} f_{md}^t = -bw_{pv}^{r^t} \tag{4.10}$$

$$k \neq s, d \sum_{\forall m \in N^P} f_{km}^t = \sum_{\forall m \in N^P} f_{mk}^t \tag{4.11}$$

Domain constraints

$$\forall\, t \in [t_a^r, t_e^r],\ \forall\, i \in N^P\ x_i^{ur^t} \in \{0,1\} \tag{4.12}$$

$$\forall\, t \in [t_a^r, t_e^r],\ \forall\, u \in N^V \sum_{\forall i \in N^P} x_i^{ur^t} = 1, \tag{4.13}$$

At each time interval $[t_a^r, t_e^r]$, Eq.(4.4) ensures that each virtual node is located at its demanded location, and Eq.(4.5) ensure that each virtual edge is only located on a single physical edge. Eq.(4.6) ensures that the total consumed CPU processing power capacity at physical node i is less than or equal to the maximum CPU capacity at that physical network node, while Eq.(4.7) ensures the same for bandwidth capacity. End-to-end delay in P^{ij} is controlled through Eq.(4.8) to be less than or equal to the maximum demanded delay by VNR^r. Eq.(4.9) ensures that the net flows getting out of the source node s is the demanded flow $bw_{uo}^{r^t}$ by virtual edge (u, o) during time interval t. While Eq.(4.10) ensures that, the net flow getting out of the destination node d is the forwarded flow $bw_{pv}^{r^t}$ by virtual edge (p, v). And Eq.(4.11) ensures that all flows are transfered through any intermediate node k between source node s and destination node d, and nothing remains at that intermediate node. Regarding network domain, Eq.(4.12) ensures to solve the problem as ILP, and Eq.(4.13) to guarantee each virtual node is mapped only to one physical node.

4.3.5 oPaCoVNE to solve VNE problem

To solve the objective function for online scenario, fully coordinating virtual nodes and edges' embeddings together, while minimizing the total power consumption in the whole physical network, this chapter proposes the online algorithm, oPaCoVNE, as a power aware resource allocation VNE algorithm, which can provide final results for the VNE problem in ms of time. Its main advantage in reducing the total power consumption relies on using only direct edges between any pair of nodes, then utilizing the segmentation approach for comparing each element in the VNR^r set of segments, Set^r, against their counterparts in the physical network path's set of segments, Set^P, considering the following constrains, namely: loc, CPU, BW, and end-to-end delay. If the elements of the physical segment has enough resources to accommodate the demands as stated by the elements of the VNR's segments, a successful embedding occurs.

4.3.6 oPaCoVNE explained

The oPaCoVNE methodology is shown in the flowchart shown in Fig. 4.1 and the pseudo-code is shown in Algorithm 4.1. At each time t, if a VNR^r arrives, the code structures its set of segments, and based on the demanded locations for the virtual nodes, the code will build a similar physical network topology to the VNR^r topology, then it formulates the physical network set of segments, and compares both sets to check if the candidate physical network segments, have enough resources to host all demands of VNR^r during the time interval. At each iteration, the algorithm keeps checking whether any VNR has expired, in oder to remove its demands from the hosting resources, and it updates the whole physical network accordingly. In parallel with that, at each iteration cycle, oPaCoVNE evaluates the power consumption of each physical network node if it is idle or loaded, and turns it off, if it has zero utilization, to save the overall power consumption in the physical network.

More elaborations about the steps of oPaCoVNE are discussed in the subsections.

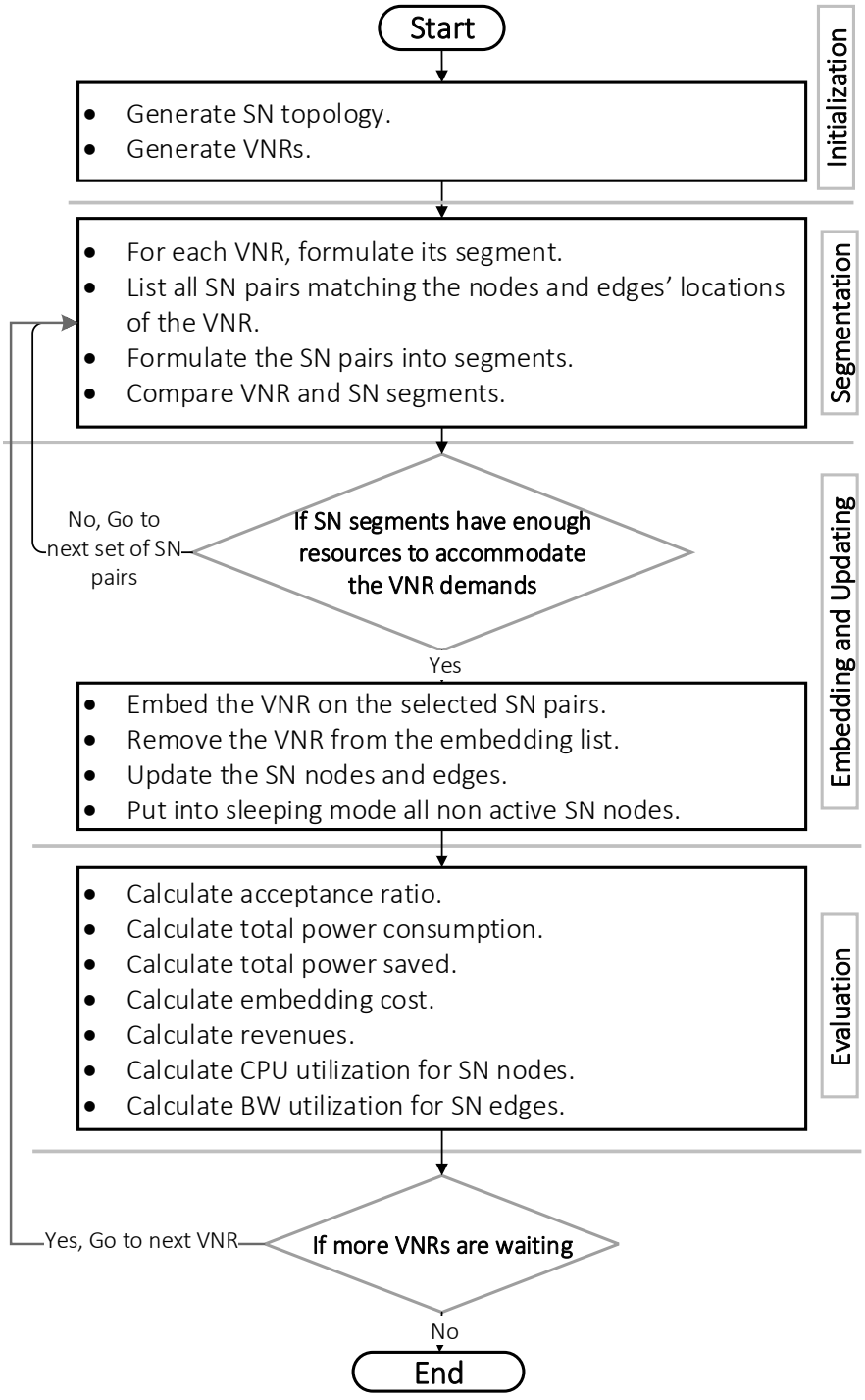

Figure 4.1: oPaCoVNE Flowchart.

Initialization

oPaCoVNE starts by listing all pairs of the VNR^r, then using the demanded locations of the virtual nodes and edges, the algorithm will allocate the physical pairs complying with the demanded locations' constraints, without using any hidden hops or edges, consequently, it will construct the candidate physical topology path P^{sub} similar to the topology structure of VNR^r. It is assumed that the physical network topology is physically fixed, therefore, the main elements formulating physical network, such as number and connectivity of the physical network nodes and edges, are also fixed and does not change, but only their capacities varies due to the consumption after each time interval t.

Algorithm 4.1, oPaCoVNE Pseudo-Code

1. Input: G^P and G^V.
2. **while** $t \neq 0$ **do**
3. **for** each VNR$^r \in R$ arriving the physical network randomly at time t
   Formulate Set^r as Eq.(3.16).
4. **for** the set of all physical network pairs $P^{ij} \in P^P$:
   - List all physical network pairs matching the nodes' and edges
   locations of VNR^r pairs, and ensure.
   they are connected by one physical network edge only.
   - Formulate Set^P representing P^{sub} as Eq.(3.17).
   - Compare Set^r against Set^P.
5. **- If** satisfied,
   - Main Output - Embed VNR^r on P^{sub}.
   - Update all parameters of physical network nodes and edges.
   - else go to next VNR, step-3.
6. **for** All embedded VNRs.
   - Check and offload expiring VNRs.
   - Update physical network's CPU and BW resources.
7. **for** All physical network nodes.
   - Calculate current U_i.
   - if $U_i = 0$
   - Turn-off physical network node i to save power.
8. Evaluate Metrics.
9. **If** VNRs list not empty, **go** to next VNR step-3.

Segmentation

This is the differentiating aspect of oPaCoVNE algorithm compared to others, mainly because it facilitates solving the two subproblems of VNE, virtual nodes' embedding problem and virtual edges' embedding problem, together, providing full coordination, while assuring not to use any additional physical resources, to minimize total power consumption in the physical network. To do that, first oPaCoVNE will reformulate all pairs of the arriving VNR^r into a set of segments, Set^r, following the format of Eq.(3.16). Then, it will construct the set of segments, Set^P, representing all pairs of the candidate physical network topology path P^{sub} following Eq.(3.17). This will prepare oPaCoVNE to check the possibilities of embedding each virtual pair of nodes and their edge, onto a corresponding physical pair of nodes and their edge, one-to-one, together, and without utilizing any hidden hopes or edges.

Embedding decision

This is the main step in making sure that each pair of virtual nodes and their edge are embedded together, resulting on solving the two subproblems of the VNE in full coordination altogether. To guarantee fully coordinated embeddings, oPaCoVNE compares each element in the physical network set of segments Set^P to its counterpart in Set^r, one-to-one, at the same time, and in one comparison step, as shown in the comparison inequalities given by Eq.(4.14). For example, each loc_i and cpu_i^a element in Set^P is compared to its counterpart loc_u^r and cpu_u^r in Set^r, and at the same time, do the same for the remaining elements representing BW and delay. Notice the use of '**AND**' in Eq.(4.14) to guarantee the full coordination ***if and only if*** every element in Set^P satisfies the needs of its counterpart in Set^r, while embedding each virtual edge on a single physical edge only, through using the adjacency matrices of the candidate physical path, Adj^{sub}, and Adj^r for the VNR^r.

Therefore, if the inequalities in Eq.(4.14) are '**ALL**' true for every candidate physical and virtual pair, which means that the demands of every virtual node and edge can be accommodated by the corresponding resources in the physical path represented by Set^P, accordingly, a successful fully coordinated embedding for the virtual nodes and edges is performed for both of them together. Decision matrix for the embedding process is given as follows:

$$
\begin{gathered}
i \in Set^P \; and \; u \in Set^r \; \textbf{\textit{if}} \; loc_i = loc_u^r \; \textbf{AND} \\
(i,j) \in P^{sub} and \; (u,v) \in E^V \quad dj_{(i,j)}^{sub} = Adj_{(u,v)}^r \; \textbf{AND} \\
i \in Set^P \; and \; u \in Set^r \; \textbf{\textit{if}} \; cpu_i^a - cpu_u^r \geq 0 \; \textbf{AND} \\
(i,j) \in Set^P \; and \; (w,x) \in Set^r \; \textbf{\textit{if}} \; bw_{ij}^a - bw_{wx}^r \geq 0 \; \textbf{AND} \\
\textbf{\textit{if}} \; d_{ij}^a \leq d_{uv}^r
\end{gathered}
\tag{4.14}
$$

Updating

Once a successful embedding occurs, the algorithm updates all changed physical network resources and moves to next VNR^{r+1}. However, in case that the selected physical network set of segments does not have enough resources to accommodate VNR^r demands, the algorithm rejects VNR^r, and jumps to the next VNR from step-3. This process keeps on going until no more VNRs to be handled.

Table 4.1: Comparing oPaCoVNE to D-ViNE algorithm

Item	oPaCoVNE	D-ViNE
Scenario	Online	Online
Goal	Minimize total power consumption	Minimize embedding cost
Strategy	Confirm locations' constraints	Confirm locations' constraints
	Check residual capacities then consolidate	Check residual capacities then embed
Coordination	Fully coordinated	Partially coordinated
Location const.	Yes	Yes
Nodes embedding	Based on location and residual capacities	Based on location and residual capacities
Edges embedding	Direct edge between each pair of nodes	Using multi-commodity flow algorithm
Power consumption	Yes	No
End-to-end delay	Yes	No

4.3.7 oPaCoVNE computational time complexity

Regardless the number of VNRs and based on the adjacency matrix of physical network, oPaCoVNE searches and lists all pairs in $O(|N^P| + |E^P|)$ processing time, depending on the total number of nodes N and edges E formulating the physical network. Once all pairs are listed, the actual embedding process starts, which consumes almost a negligible processing time in ms, since it is based on checking all elements in the comparison statement as in Eq.(4.14).

4.3.8 Evaluation metrics

The performance of oPaCoVNE algorithm will be evaluated based on acceptance ratio, total power consumption, saved power, total revenues, total cost, CPU and BW utilizations, and processing time.

Average acceptance ratio, AR

Is a ratio to represent how oPaCoVNE algorithm is performing, and is calculated by averaging and dividing the number of successfully embedded VNRs at each time interval by the total number of VNRs R [Chowdhury (2012)].

$$AR = \frac{1}{T} \sum_{\forall t \in T} \frac{1}{R} \, Total \; Number \; of \; Embedded \, VNRs * 100 \tag{4.15}$$

Average power consumption, PC

Calculated by averaging the total power consumed by all physical network nodes and edges after each time interval t, [Botero (2013a)],

Table 4.2: Simulation settings to test oPaCoVNE against D-ViNE

Parameter	physical network	VNR
$Nodes$	50	$2-10$
CPU	$50-100$	$0-20$
BW	$50-100$	$0-50$
$Delay$	$1-50$	$20-100$
PC^{Busy}	$15 * CPU$	
pc^{idle}	$165\ Watts$	
$time\ units$	$0-50,000$	
α	0.7	
β	0.9	
p_{wax}	0.5	
VNRs/100 $time\ units$	$4-8$	
VNR $lifetime$	1000 $time\ units$	

$$PC = \frac{1}{T} \sum_{\forall t \in T} \sum_{\forall i \in N^P} PC_i^t \tag{4.16}$$

Average saved power, PS

The average amount of saved power when the proposed power reduction strategy was used. It is calculated after each time interval t, given by subtracting the total power consumed by the whole physical network without power reduction strategy PC^{t-}, from the total power consumed by all physical network after applying the power reduction strategy PC^{t+}.

$$PS = \frac{1}{T} \sum_{\forall t \in T} (\sum_{\forall i \in N^P} PC_i^{t-} - \sum_{\forall i \in N^P} PC_i^{t+}) \tag{4.17}$$

Average cost, Co

The average amount of allocated virtual processing powers multiplied by the processing unite price μ, and throughputs multiplied by the flow unite price ρ, at each time interval t [Chowdhury (2012)].

$$Co = \frac{1}{T} \sum_{\forall t \in T} (\sum_{\forall u \in N^V} \mu \times cpu_u^{r^t} + \sum_{\forall (u,v) \in E^V} \rho \times bw_{uv}^{r^t}) \tag{4.18}$$

Average revenues, Rev

Representing revenues from all accepted virtual requests at each time interval t, calculated by adding the total demanded processing powers and throughputs [Chowdhury (2012)].

$$Rev = \frac{1}{T} \sum_{\forall t \in T} (\sum_{\forall u \in N^V} cpu_u^{r^t} + \sum_{\forall (u,v) \in E^V} bw_{uv}^{r^t}) \tag{4.19}$$

Average CPU **utilization,** CPU_{util}

It represents the average physical network nodes' utilization trend after all simulation iterations. Its defined as a ratio between consumed CPU cpu_i^t, and maximum CPU resources, averaged over all physical network nodes [Chowdhury (2012)].

$$CPU_{util} = \frac{1}{T} \sum_{\forall t \in T} \frac{1}{N^P} (\sum_{\forall i \in N^P} \frac{cpu_i^t}{CPU_i} * 100) \tag{4.20}$$

Average BW **utilization,** BW_{util}

It represents the average utilization of physical network edges after all simulation iterations. And is defined as a ratio between consumed bw_{ij}^t, and the maximum BW, averaged over all physical network edges [Chowdhury (2012)].

$$BW_{util} = \frac{1}{T} \sum_{\forall t \in T} \frac{1}{E^P} (\sum_{\forall (i,j) \in E^P} \frac{bw_{ij}^t}{BW_{ij}} * 100) \tag{4.21}$$

4.4 oPaCoVNE evaluations

To generally evaluate the new online power aware and fully coordinated virtual network embedding algorithm, oPaCoVNE, three sets of simulations were conducted in this chapter:

1. oPaCoVNE overall performance in terms of acceptance ratio, embedding cost, in addition to nodes and edges' utilizations will be compared against the most referenced online algorithm, D-ViNE by [Chowdhury (2012)].
2. To investigate oPaCoVNE's power aware performance in terms of average power consumption, average generated revenue, and number of switched off physical nodes, the performance of oPaCoVNE was compared against the state of art algorithm, EAD-VNE, by [Zhongbao (2015)].
3. Furthermore, four different experiments were also conducted to test the impacts of varying number, lifetime, and size of the embedded virtual requests, as well as the impacts of varying size of physical edges, on oPaCoVNE's average power consumption, acceptance ratio and processing time.

4.4.1 Evaluating overall performance of oPaCoVNE

The overall performance of oPaCoVNE was compared to D-ViNE, which targeted minimizing the embedding cost while improving acceptance ratio, by mapping the virtual nodes based on the total flow passing through the edges of a candidate physical network, and applies multi-commodity flow algorithm to map the virtual edges onto paths connecting the hosting physical network nodes. Table.4.1 provides a high-level comparison between oPaCoVNE and D-ViNE algorithms, listing their used strategies, and how they embed the virtual nodes and edges.

Simulation settings to test oPaCoVNE

Substrate network and virtual network topologies were randomly generated using Waxman algorithm, setting $\alpha = 0.7$, $\beta = 0.9$, and mean probability of a pair of two nodes being connected set equal to 0.5. Similar to [Chowdhury (2012)], the physical network includes 50 nodes, CPU and BW resources given as real numbers, uniformly distributed between $50 - 100$, and delay in each physical edge was randomly selected between $1 - 50$. Number of virtual nodes per VNR was randomly determined by a uniform distribution between 2 and 10, assuming that VNRs arrive according to Poisson process pattern, each having an exponentially distributed lifetime with an average of 1000 time units. Arriving rates of VNRs were varied between $4 - 8$ per 100 time units, over simulation time of $50,000$ units. The virtual CPU and BW resources were real numbers uniformly distributed between $0 - 20$ and $0 - 50$ respectively, while delay in each virtual edge was randomly selected between $20 - 100$. Substrate network and virtual network topologies were assumed located on the same grids, and locations of virtual nodes were selected based on the corresponding locations of the physical nodes. Table.4.2 summarizes all simulation parameters.

Discussing results of oPaCoVNE overall performance

Analyzing acceptance ratio: The average acceptance ratio of oPaCoVNE was 63.64% compared to 67.10% for D-ViNE as shown in Fig.4.2a, which indicates the close similarity in the overall performance of both algorithms. However, the fact that oPaCoVNE strategy in selecting only direct edges to minimize the used resources when embedding virtual edges, in order to minimize the total power consumption, has negatively impacted the algorithm's acceptance ratio, yet oPaCoVNE still managed to show solid and comparable results to D-ViNE in general, while leaving more free resources to host more demands. More precisely, this is due to applying the segment technique which insures that the virtual nodes and virtual edges were mapped together on the physical network path that, complies with the location constraints and has enough resources and acceptable end-to-end delay, without any hidden nodes or edges.

Similar to oPaCoVNE, D-ViNE maps virtual nodes on physical nodes having enough residual capacities, and maps virtual edges through applying multi-commodity flow algorithm to find the appropriate physical network edge between the already mapped virtual nodes. Nevertheless, the way D-ViNE selects the physical network nodes to host virtual nodes, may result on selecting physical network nodes that are further away from each other, which when applying the multi-commodity flow algorithm to map the virtual edges, could force using longer paths, therefore, consuming more resources, but would have better acceptance ratios. That was clear for the acceptance ratios of D-ViNE when the loads were between $4 - 7$. However, the results of oPaCoVNE started to converge with those of D-ViNE, most probably due to the multi-commodity flow algorithm's way of selecting more edges, and allowing for splitting the flows between more than one physical path, thus, raising

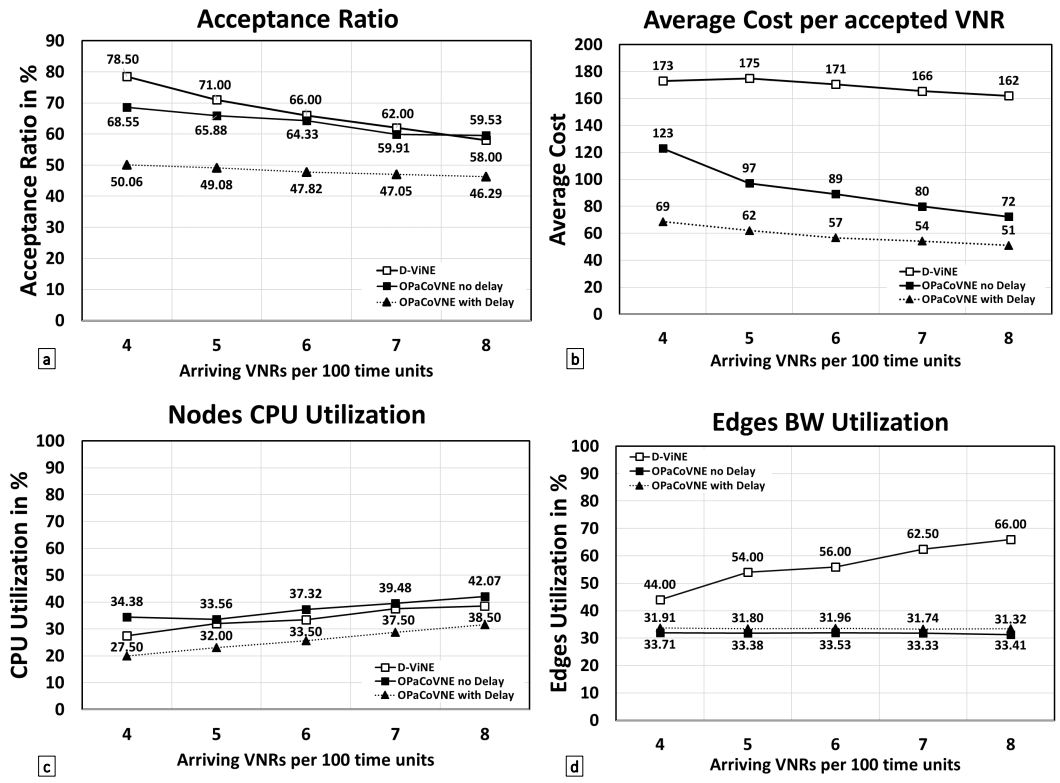

Figure 4.2: Overall Performance of oPaCoVNE.

the possibilities of congested edges, which in turn would cause degradation in the acceptance ratio. Moreover, referring to Fig.4.2a, and focusing on the load of 8 VNRs per 100 time units, OPaVoVNE acceptance ratio started to get better than that of D-ViNE, most likely since it would always stick to using less edges during the embedding process, therefore, keeping more free resources to be used to fulfill the demands of the new virtualization requests.

Average cost: Average cost per accepted VNR is shown in Fig.4.2b, confirms the successful strategy of oPaCoVNE in minimizing the use of physical network resources as least as possible while embedding VNRs, by selecting direct edges between the hosting physical nodes without any hidden hopes. Therefore, in average oPaCoVNE cost was around 92.28 compared to D-ViNE cost of 169.20, which most likely tends to use more edges when embedding VNRs than oPaCoVNE. The processing power unite price μ, and throughput unite price ρ were set equal to 1.

Physical network nodes utilization: Fig.4.2c shows nodes' average utilization. Physical network nodes in oPaCoVNE are in average moderately utilized resulting on 37.36%, which is very close when compared to 33.80% for D-ViNE. This is a reflection of the similarity between both oPaCoVNE and D-ViNE in utilizing physical network nodes having enough residual resources to embed virtual nodes. Also it is important to point out that both algorithms first select the physical network nodes according to the location constraint, then they check for the residual resources afterwards. In the case of including end-to-end delay, oPaCoVNE nodes' utilization is less than without delay, mainly since oPaCoVNE with end-to-end delay accepted less number of demands, thus has less utilized physical network nodes.

Table 4.3: Comparing oPaCoVNE to EAD-VNE algorithm

Item	oPaCoVNE	EAD-VNE
Scenario	Online	Online
Goal	Min. Power cons.	Min. Power cons., high revenues.
Strategy	Confirm locations' constraints	Confirm locations' constraints
	Use residual capacities then consolidate	Use residual capacities
Coordination	Fully coordinated	Partially coordinated
Location	Yes	Yes
Nodes embedding	Based on location and residual capacities	Based on location and residual capacities
Virtual edges embedding	Direct edge between each pair of nodes	Using shortest path algorithm
Power consumption	Yes	Yes
End-to-end delay	Yes	No

Physical network edges utilization: Average edges utilization in oPaCoVNE resulted on 31.75%, and was lower than D-ViNE of 56.5% as shown in Fig.4.2d. This is mainly because oPaCoVNE is allocating precisely the same number of physical network edges as the demanded edges without any hidden edges or nodes. However, the way D-ViNE embedding virtual edges using the multi-commodity flow algorithm, could have resulted on using longer paths including more physical network edge than requested, which raises its overall edges' utilization.

Impact of delay on oPaCoVNE: The impact of end-to-end delay on oPaCoVNE, was negative in general over all simulations as shown in Fig.4.2. However, since D-ViNE did not use delay as a constraint, the results of oPaCoVNE were shown just to reflect how much the performance of oPaCoVNE without delay is better than when it was included.

4.4.2 Evaluating power consumption of oPaCoVNE

This subsection compares the power aware performance of oPaCoVNE against EAD-VNE algorithm, which applies power aware with dynamic demands as well [Zhongbao (2015)]. Table.4.3 compares the two algorithms, giving that for virtual nodes' embedding, both algorithms first use location constraint to select the physical nodes, then embed the virtual nodes if enough residual resources are available. For the virtual edges, EAD-VNE uses shortest path, which may include additional edges, while oPaCoVNE uses direct edges connecting the pair of physical nodes, without any hidden nodes or additional edges. Tables.4.4 provides general settings to compare oPaCoVNE against EAD-VNE.

oPaCoVNE versus EAD-VNE

Exact simulation settings of EAD-VNE were applied on oPaCoVNE, assigning 4 VNRs to be embedded per 100 time units, each having average lifetime of 500 time units, and number of nodes per VNR were randomly distributed between $2-10$. As shown in Fig.4.3a, the average power consumption per physical network node using oPaCoVNE is 23.54% lower than EAD-VNE along the simulation duration, confirming the better power aware performance of oPaCoVNE. Fig.4.3b presents average revenue in the physical network, showing that

Table 4.4: Simulation settings to test oPaCoVNE against EAD-VNE

Parameter	physical network	VNR
$Nodes$	50	
CPU	$50-100$	$0-20$
BW	$50-100$	$0-50$
$Delay$	$1-50$	$20-100$
PC^{Busy}	$15 * CPU$	
pc^{idle}	$165\ Watts$	
$time\ units$	$0-50,000$	
α	0.7	
β	0.9	
p_{wax}	0.5	
Experiment-1		
VNRs$/100\ time\ units$	4	
VNR $lifetime$	$500\ time\ units$	
VNRs' $nodes$	Random $2-10$	

oPaCoVNE revenues are almost near to those of EAD-VNE, and in average were less by 4.4%. Noting that oPaCoVNE and EAD-VNE embed virtual nodes based on location and residual nodes' resources constraints, yet, in contrary to EAD-VNE, which uses shortest path to embed virtual edges, oPaCoVNE's way in embedding each virtual edge on a single physical edge, may cause some VNRs to be rejected, which would reduce the overall revenues of oPaCoVNE in general.

Nevertheless, the fact of using direct edges by oPaCoVNE is precisely why its power consumption is better than EAD-VNE, since oPaCoVNE strategy uses least physical network resources as much as possible, through using direct edges with no hidden hopes at all. This is confirmed referring to Fig.4.3c, which shows that oPaCoVNE strategy managed to turn-off more physical nodes than EAD-VNE by 22.2%.

4.4.3 Impacts of varying size of physical edges

Table.4.5 lists specific settings for experiments $2-5$ that were designed to evaluate oPaCoVNE in more details. The four experiments were specifically designed to evaluate the online power consumption and acceptance ratio behaviors of oPaCoVNE, when end-to-end delay was included as a major embedding constraint. In the second experiment, number of arriving VNRs were varied between $2-10$ each 100 time units, while the third experiment varied VNRs' lifetime between $200-1000$ time units, and the fourth experiment varied number of nodes per each embedded VNR between $2-10$. Finally, fifth experiment evaluated oPaCoVNE power consumption and acceptance ratios for five different physical network topologies.

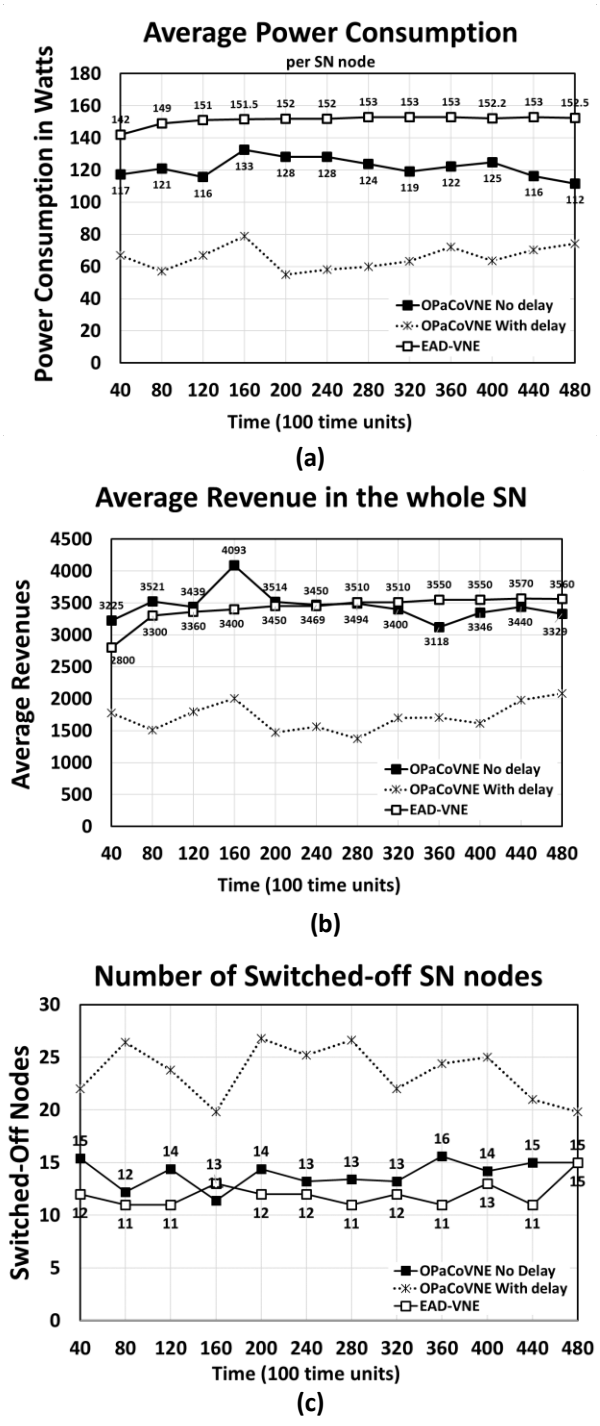

Figure 4.3: oPaCoVNE performance against EAD-VNE.

Table 4.5: Experiments 2-5 to test oPaCoVNE

Experiment-2	Varying number of arriving VNRs
VNRs/100 $time\ units$	$2-10$
VNR $lifetime$	500 $time\ units$
VNRs' $nodes$	Random $2-10$
Experiment-3	Varying VNRs' lifetime
VNRs/100 $time\ units$	4
VNR $lifetime$	$200-1000$ $time\ units$
VNRs' $nodes$	Random $2-10$
Experiment-4	Varying number of nodes per VNR
VNRs/100 $time\ units$	4
VNR $lifetime$	500 $time\ units$
VNR's $nodes$	from $4-10$
Experiment-5	Different physical network topologies
VNRs/100 $time\ units$	4
VNR $lifetime$	500 $time\ units$
VNR's $nodes$	Random $2-10$
Topologies	Average physical network edges (Edges/Node)
T1	380(7)
T2	530(10)
T3	630(12)
T4	770(15)
T5	860(17)

Average power consumption of oPaCoVNE

As shown in Fig.4.4 - Fig.4.7, average power consumption and impacts of adding delay constraint on oPaCoVNE's performance were reported using different settings according to experiments 2-5. From Fig.4.4a, varying number of embedded VNRs between $2-10$ per 100 time units, reflects the loading impacts on oPaCoVNE's power consumption, which showed increasing consumption trend in general from 76% when the number of embedded VNRs were as low as 2, to 96% for 10 loaded VNRs each 100 time units. Same increasing trends are also shown in Fig.4.5a, but in this case number of embedded VNRs were fixed on 4 per 100 time units, while varying their lifetime from $200-1000$ time units. In both experiments, impact of delay on the overall power consumption resulted on less values than the cases when delay was not included, yet it showed increasing trends in both settings.

Furthermore, Fig.4.6a presents the results when fixing number of embedded VNRs per 100 time units to 4, while varying number of virtual nodes per VNR between $4-10$, which resulted on slight increases in the average power consumption when delay was not included, suggesting that the change in VNRs' size has minor impact in general. However, this conclusion changed dramatically when delay was included, where

power consumption was higher for low sized VNRs, but decreased for larger VNRs. This result on power consumption with delay would most likely be due to high rejections of large sized VNRs.

Finally, Fig.4.7a reported the average power consumption of oPaCoVNE for five different physical network topologies, which were varied by increasing the number of edges per each of them. As shown, increasing number of connected edges per topology did not have any significant impact on oPaCoVNE's average power consumption in general, but when delay was include oPaCoVNE's power consumption showed slight decreasing trend for larger connected topologies, yet still it did not show any significant impact too. This is expected, since oPaCoVNE uses location constraint, suggesting that performance of oPaCoVNE will very slightly be affected, regardless of the size of connected edges in the infrastructure.

Average acceptance ratio of oPaCoVNE

oPaCoVNE's average acceptance ratios for experiments 2-4 had a decreasing trend with and without delay as shown in Fig.4.4b, Fig.4.5b, and Fig.4.6b. For experiment-2, number of arriving VNRs per 100 time units varied between $2-10$, the average acceptance ratio varied between 92.68% in the case of low number of arriving VNRs, and decreased to 78.39% when number of arriving VNRs was 10. Same trend was reported for experiment-3, when VNRs' lifetimes varied between $200-1000$, which resulted on average acceptance ratio between 94.49% for short lifetimes, down to 79.90% for lifetimes as long as 1000 time units. Moreover, the average acceptance ratio for experiment-4, when number of virtual nodes per VNR varied between $2-10$ resulted on 93.87% for small sized VNRs, decaying to 72.97% for larger VNRs.

However, in the case of experiment-5, which was designed to examine the impact of different physical topologies on oPaCoVNE's performance while using location constraint, acceptance ratio trend slightly increased, from 86.98% for small connected physicals, to 87.96% for larger connected configurations as shown in Fig.4.7b. This result implies that solving VNEs using location constraints was barely affected by the size of physical network topology, compared to the resources' capacities of physical network.

Average processing time of oPaCoVNE

The average processing time results for oPaCoVNE while performing experiments 2-5 are shown in Fig.4.4c to Fig.4.7c. In experiment-2, the average processing time was about 25.60 ms when the number of arriving VNRs were low, then increased as the number of arriving VNRs was increased up to 7 per 100 time units. Nevertheless, for larger number of arriving VNR between $8-10$, average processing time decreased again, most probably due to an already loaded and limited physical resources, which caused high rejected VNRs, therefore decreasing the processing time accordingly.

On the other hand, for experiment-3, average processing time was in the range of 37.96 ms as shown in Fig.4.5c, giving that the number of arriving VNRs was fixed to 4, but varying VNRs' lifetimes between $200-1000$ time units. This result suggests that oPaCoVNE's processing time performance was slightly affected when varying VNRs' lifetimes.

However, the general behavior of average processing time had an increasing trend when changing the size of VNRs or physical network topologies as shown in Fig.4.6c and Fig.4.7c. The processing time trend for smaller VNRs varied between $1.42-7.49$ ms, while for physical networks that has large number of connected pairs of nodes, the average processing time varied considerably between 15.59 ms for topology $T1$ where average number of physical edges was 380, to 238.23 ms for the case of topology $T5$, which had 860 edges.

In all previous experiments, oPaCoVNE code was developed using Eclipse IDE for Java Developers, version: Mars.2 Release (4.5.2). The used machine was Lenovo laptop, system model $20CLS2RG00$, processor Intel(R) Core(TM) i7-5600U CPU, 2.60 GHz, 2 Cores, 4 logical processors, RAM 8 GB, and the operating system was Microsoft Windows 10 Enterprise.

4.4.4 Impact of delay on oPaCoVNE performance

All simulation results showed that impact of delay on VNE process was clearly the most significant parameter among all varied variables while testing oPaCoVNE. Specifically, referring to Fig.4.4-Fig.4.7, oPaCoVNE's average acceptance ratios with delay, were less than when it was not included by $13.05\%, 14.38\%, 17.69\%$, and 17.39% for experiment-2 to experiment-5 respectively. Similar trends can be seen by referring to oPaCoVNE's results for average power consumption and processing time.

In all experiments, values of physical network delays were randomly assigned in the range $1 - 50$ ms [5GAericas (2018)] and between $20 - 100$ ms for VNRs. Accordingly, each time oPaCoVNE attempts to embed any VNR, it carefully tries to guarantee embedding on physical edges connecting directly the selected physical nodes, having enough bandwidth resources and end-to-end delay within the demanded ranges.

These values confirm the importance of including end-to-end delay as a major constraint when solving VNE problem, as a direct evaluation metric for real world 5G networks.

4.5 Conclusions

This chapter introduced a new online embedding algorithm, oPaCoVNE, solving the two subproblems of VNE process in full coordination using end-to-end delay as a main constraint. The algorithm restructures virtual nodes and edges' demands in one set of segments, and formulates an exact similar set of segments for a specifically selected physical path topology, which comply with the exact demanded locations to embed the virtual nodes. Consequently, the embeddings occurs by comparing the two segments, one-to-one, and checking if each element in the physical segment can accommodate the demands of their counterparts from the virtual segment. In addition to that, to minimize the total power consumption in the whole physical network, the proposed algorithm insures that each pair of the physical network nodes hosting the virtual nodes, are directly connected by a single edge with no hidden hops or edges, which guarantees utilizing the least physical network resource as low as possible.

The overall performance results of oPaCoVNE showed that, without delay, it managed to minimize the average power consumptions in the physical network by 23.54%, however, when end-to-end delay was factored in, performance of oPaCoVNE was degraded across all evaluation metrics, suggesting that, introducing end-to-end delay as a major constraint, had clear impact on the whole VNE process, and therefore, it has to be one of the main metrics when evaluating real world 5G networks.

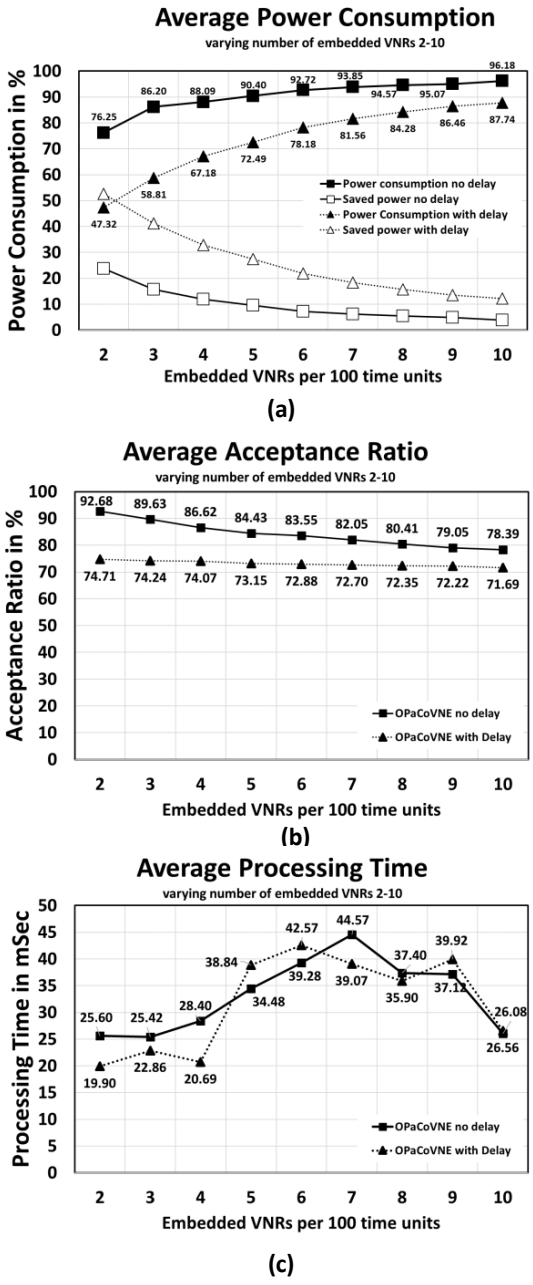

Figure 4.4: oPaCoVNE performance varying number of embedded VNRs.

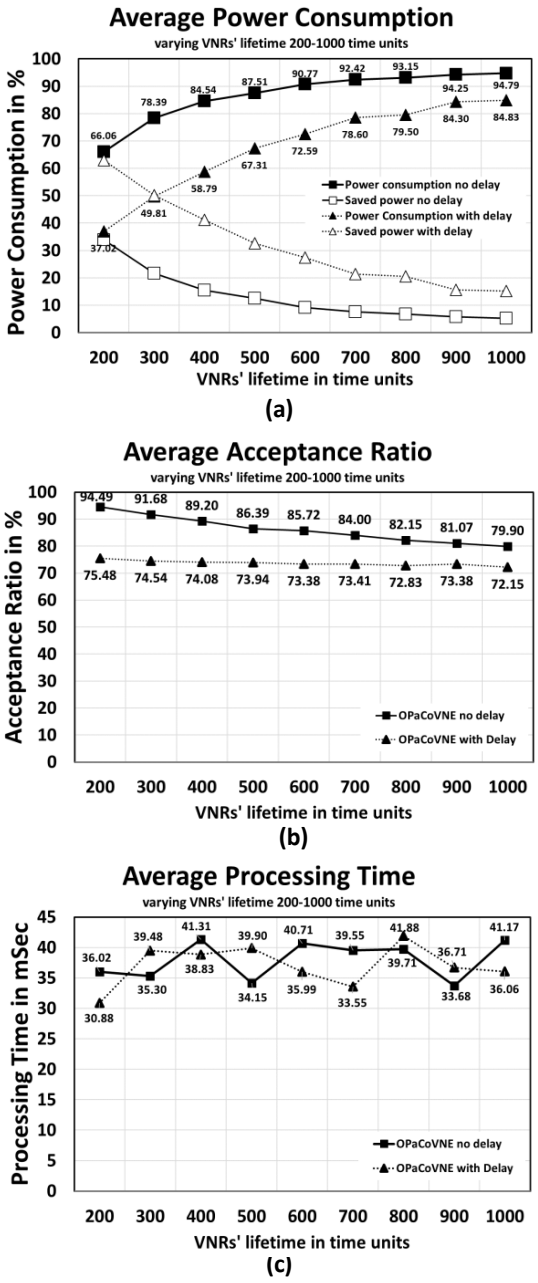

Figure 4.5: oPaCoVNE performance varying VNRs lifetime.

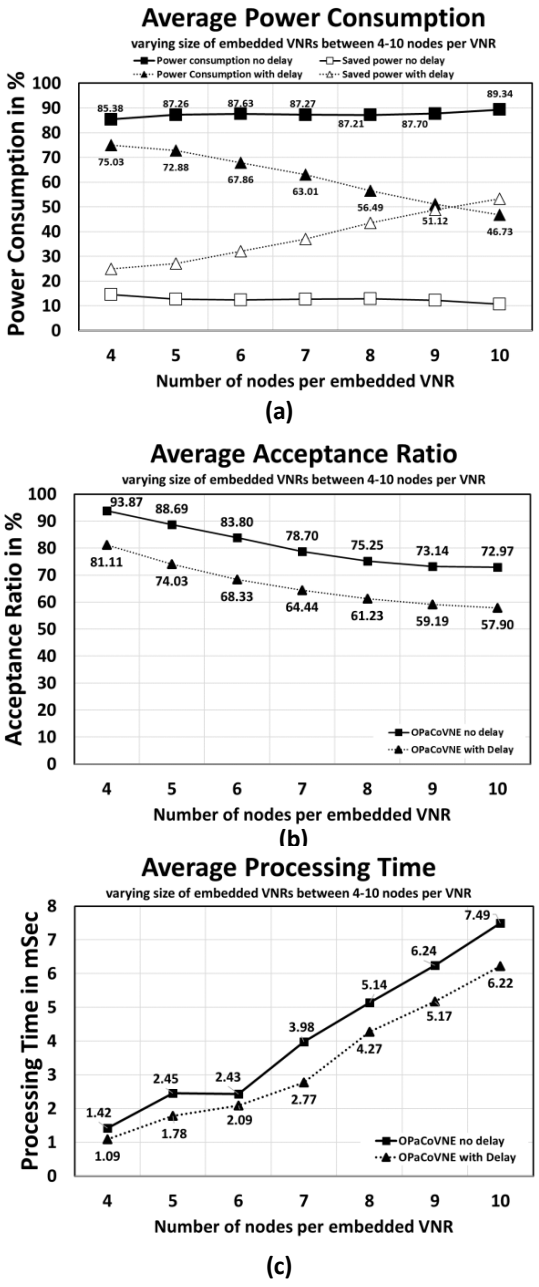

Figure 4.6: oPaCoVNE performance varying number of nodes per embedded VNR.

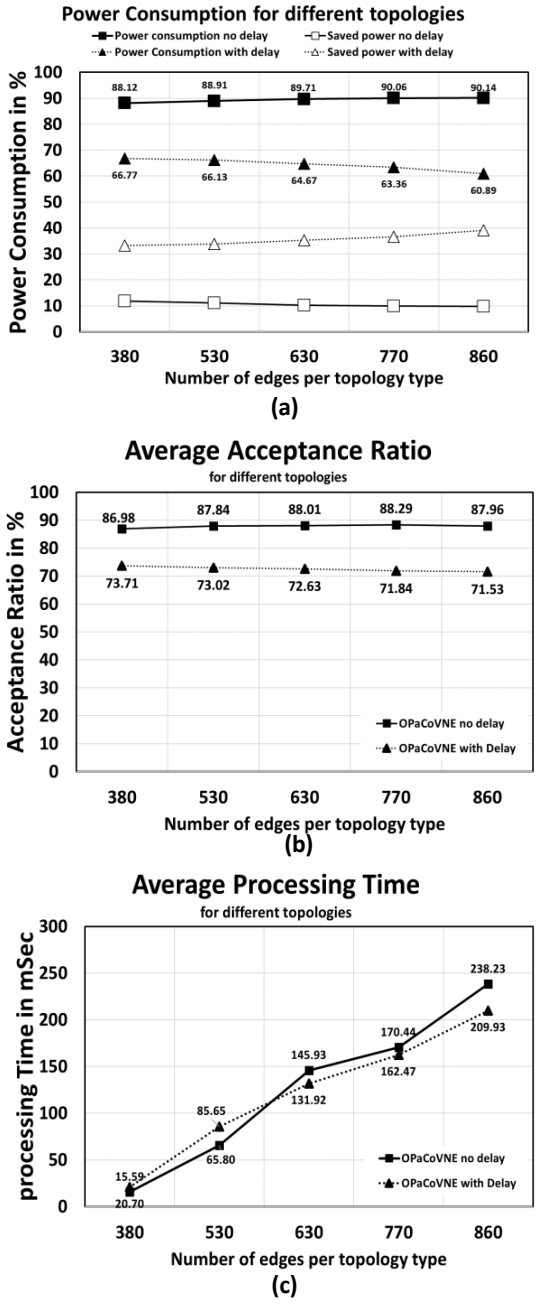

Figure 4.7: oPaCoVNE performance varying number of connected edges per topology.

CHAPTER 5

NFV Power Aware Resource Allocation

5.1 Introduction

Future Internet networks and 5G technologies are promising to offer enhanced mobile broadband services, connect massive number of sensors, and natively support mission critical services that require very high reliability and ultra low latency [ITU-T Focus Group (2017)]. To realize such networks, multiple design objectives were identified by ITU and 3GPP for 5G including the integration of software defined networking (SDN), networks' virtualization, and network slicing initiatives [5GAericas (2018)]-[ETSI (2013a)].

After all, the leading 5G technology that is captivating great deal of attention is network functions virtualization, which separates network functions, such as firewalls, intrusion detectors, and caching, to name a few, from proprietary hardware appliances so they can run in software [ETSI (2013a)]. However, according to [ITU-T Focus Group (2017)] and [ETSI (2013b)], they stressed on the significance of engineering these NFV based networks to be fully aware about their power efficiency. In practice, this means that NFV infrastructure providers (InP) need to consider the power efficiency of their infrastructure, while virtualizing their physical resources to allocate the demands of the virtual Network Services (NS). In NFV literature, the resource allocation process within NFV architecture is denoted by (RA-NFV) [Mijumbi (2016)][Herrera and Botero (2016)], in addition to that, virtual Network Services are usually called service function chains (SFC), which are represented as a collection of ordered virtual logical nodes, known as virtual network functions (VNFs) that are connected by a set of virtual logical edges.

On power efficient NFV infrastructure, ETSI in [ETSI (2013b)] highlighted that the NFV frameworks shall significantly reduce the energy consumption of network infrastructures, so that they support workload consolidations by scaling the traffic loads and concentrate them on a smaller number of servers during off-peak times, and consequently allow to turn-off or put on energy saving mode all other not loaded servers. In accordance with these requirements, several researchers started to apply the consolidation technique to minimize the total power consumption in the physical networks [Eramo (2017a)]-[Soualah (2017)] while allocating the service function chains.

This chapter modifies the general work from chapter 3 and the work from [Hejja (2018)], and proposes three new power aware RA-NFV algorithms for offline, online, and migration applications. The main objective of the offline coupled with migration algorithms is to allocate the demands of the Network Services that are known in advance, on the most appropriate physical components that have enough resources, while minimizing the costs of the power consumption in the physical network. The algorithms achieve that through consolidating the network's traffic into least number of active servers as much as possible, then putting other non utilized servers into saving mode. On the other hand, the online with migration algorithms are designed to solve the RA-NFV

problem on real time bases and minimize the total power consumptions in the whole physical network. For both scenarios, offline and online, the migration algorithm was kept on alert to be triggered whenever there is a need to transfer some traffic from low utilized resources or during maintenance and emergency conditions.

In general, the algorithms are differentiated from others in literature in the following aspects; first both of them relay on a fast and precise path construction methodology, called segmentation, which was developed in chapter 3. It allocates the service function chains, including the demands of their virtual network functions and their edges, together and in full coordination, on the most fit physical path. Second, up to the time of writing this chapter, it is the first time to include migrations and delay in an online power aware algorithm to solve the RA-NFV problem. The algorithms solve the RA-NFV problem in fractions of a second, can handle any number of demands, and can be customized to work for any type of physical networks.

Main contributions:

1. This chapter proposes new offline and online power aware algorithms to solve RA-NFV problem in fractions of a second.
2. Each algorithm is integrated with a migration strategy.
3. To the best of our knowledge, this is the first time to include both migrations and end-to-end delay in an online power aware algorithm to solve the resource allocation problem.
4. All algorithms used a modified version from the path construction methodology presented in chapter 3, which facilitates allocating precise resources for the virtual network functions and their virtual edges, together and in full coordination on the physical network.

5.2 Scenarios

The algorithms proposed by this chapter were designed to work for offline and online NFV scenarios. In offline, all SFCs are assumed known in advance, while in the online scenario they are assumed arriving on real time, and each has arrival and lifetime. The following paragraphs introduce the overall idea of each scenario as follows:

5.2.1 Offline scenario

The offline scenario is designed to evaluate the overall performance of the proposed algorithms under fully controlled conditions when allocating the service function chains. In this chapter, the offline algorithm from chapter 3 and [Hejja (2018)] is modified to work on NFV framework, and accordingly, two algorithms were proposed for the offline scenario, the first one is an offline power aware coordinated NFV algorithm, PaNFV, coordinating the allocations of the physical servers and edges' resources as demanded by the virtual nodes and edges of the service function chains, and the second algorithm is the migration extension of PaNFV, denoted by mPaNFV, which is triggered by the offline 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.

5.2.2 Online scenario

It represents real life conditions where the SFCs' arrival times and lifetimes are random and not known in advance, which means that the online algorithm has to deal with each SFC instantly and one-by-one. In the following paragraphs, a summary for some of the related work that focused on online resource allocation and migration strategies will be introduced.

For example, the recent work by [Qin (2019)] proposed a user distribution aware virtual machine redeployment mechanism and proposed a traffic redirection virtual machine migration scheme to keep the active flows from being interrupted. However, the authors did not evaluate the impact of their work on the power consumption of the network, and they did not include end-to-end delay as a main constraint.

Moreover, the authors in [Esfandiarpoor (2015)] considered rack, cooling structure and network topology when consolidating virtual machines inside a datacenter. They provided a dynamic placement method considering the utilization of racks, in addition to utilization of individual servers to improve the energy consumption in the cooling system and the datacenter network. They considered migrating the virtual machines of underutilized racks onto other utilized racks and turned off the idle racks in order to save more power. However, the authors did not consider resource allocations for distributed datacenters, and did not consider the impact of delay in their work. Another study was conducted by [Rachael (2019)] which focused on improving the virtual machines' placement problem inside a datacenter, by combining virtual machines for placement based on predicting the required processing power and bandwidth resource demands in order to reduce the amount of resources, and to improve both energy efficiency and performance. However, their work also did not include distributed datacenters, it treated migrations of individual virtual machines and not the network service as whole, and they did not consider end-to-end delay as a constraint in their work.

In [Ranjbari (2018)] the authors proposed a reinforcement learning algorithm that can determine the optimal action from a set of actions. The proposed algorithm considers changes in the demanded resources to predict the physical machine which may suffer from overload. To improve the utilization and to reduce the energy consumption in the datacenter, the proposed algorithm will turn off the idle physical machines after migrating their traffic to other machines. However, the proposed algorithm by the authors was designed for large datacenter, and did not include distributed networks of datacenters as well as not including end-to-end delay. Finally, the authors in [Zhihua (2018)] proposed an energy aware dynamic optimization approach using artificial bee colony, to find the mapping relation between physical machines and virtual machines inside a datacenter. The algorithm constrains the consolidations in a manner that limits the number of virtual machines' migrations, to minimize the negative impacts on quality of service. In addition to that, the algorithm attempts to minimize overloading the physical machines, and turns off underloaded physical machines to reduce the energy consumption of the datacenter. Also in this work, the authors did not consider distributed datacenters, and end-to-end delay as a resource allocation constraint too.

Accordingly, for the online scenario this chapter proposes a modified version of the online algorithm from [Hejja (2018)], which included online resource allocations, and considers end-to-end delay as a main allocation constraint. The proposed new algorithm is called, the online power aware coordinated NFV algorithm, oPaNFV, which ensures performing the allocation process based on minimizing the total power consumption in a distributed network of datacenters. It allocates the arriving SFCs once they arrive the physical network on real time, and terminates them from the physical network once their lifetimes get expired. oPaNFV treats the SFCs as first-arrived-first-served without any additional filtering, applies virtual machines consolidations to minimize total power consumption in the datacenters' network, and ensures selecting the physical path with the least end-to-end delay to meet the requirements of the SFCs.

Moreover, this chapter integrates the migration algorithm mPaNFV with oPaNFV to perform the migrations of the SFCs within a network of distributed datacenters for the online scenario, and it also considers the end-to-end delay while migrations to meet future 5G core networks.

5.3 Segmentation technique for NFV

In this chapter the segmentation technique developed in chapter 3 is modified to work for an NFV based physical networks when constructing the candidate paths for allocating the service function chains, and it was recently published in [Hejja (2019)]. 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.

5.3.1 Overview

Assume that the physical network is composed of datacenters having a collection of access, router, switch, and server nodes, connected by physical edges as the example shown in Fig.5.1b. Also assume that the edges connecting any pair of nodes in the physical network are fixed and can be identified in advance including their parameters. Moreover, assume that the Network Service (i.e SFC) is represented as a graph composed of source and termination access nodes, and orderly chained virtual network functions as shown in the example in Fig.5.1a. To solve the resource allocation problem, the path construction methodology from [Hejja (2018)] is applied to guarantee the coordinated allocations of the virtual functions and their connecting virtual edges, together, on a physical path between the datacenters that can fulfill the demands of the SFC. Each path could be constructed of one or multiple segments, giving that each segment represents a pair of nodes and their edge.

5.3.2 Definitions

Segmentation concept and notations

Segmentation concept is a representation of the information in any path. Assume that the set R of SFCs are required to be allocated on the physical network of datacenters, the proposed segmentation methodology will start converting the structure of the SFC number $r \in R$, listing all information about its nodes and edges together in a set format denoted by Seg^r, which represents the demands of SFC^r. Afterwards, to allocate this 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.

General definition of the basic segment

For any pair of nodes, a and b, and their edge (a, b), a basic segment lists the nodes' parameters such as their locations, a_{loc} and b_{loc}, processing powers of the nodes, cpu_a and cpu_b, and the edge's parameters such as, bandwidth bw_{ab}, and delay d_{ab}, all to be grouped in one set format denoted by Seg_{ab} as shown in Eq.(5.1).

$$Seg_{ab} = \{a_{loc}, b_{loc}, cpu_a, cpu_b, bw_{ab}, d_{ab}\} \tag{5.1}$$

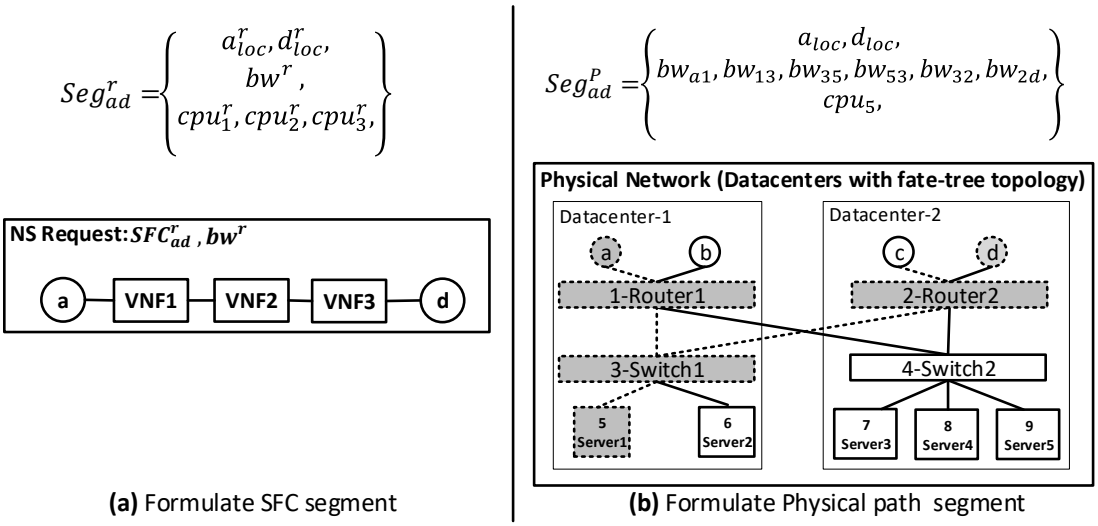

Figure 5.1: Segments' formulation demonstration.

Segmentation for a path of a single basic segment

Assume a path P_{ij} composed of two access nodes, source i and destination j, reaching node s, which could be a server if the path is inside a datacenter, or s could be a VNF if the path represents an SFC. Consequently, path P_{ij} segment denoted by Seg^P_{ij} is defined as a list of the basic segments $Seg_{i,s}$ and $Seg_{s,j}$, including all parameters of their nodes and edges grouped in a set format as shown in Eq.(5.2). It includes locations of the source and destination nodes, denoted by i_{loc} and j_{loc}, processing power capacities of node s in that path, cpu_p, and the parameters of the connecting edges, including bandwidth capacities, bw_{is}, bw_{sj}, and delay d_{ij}. Important hint, in this chapter, it is assumed that only the server nodes have computing resources, but access, router, and switch nodes do not.

The segment defining path P_{ij} can be written in set format as follows:

$$Seg^P_{ij} = \{i_{loc}, j_{loc}, cpu_p, bw_{is}, bw_{sj}, d_{ij}\} \tag{5.2}$$

5.3.3 Application of segmentation on NFV framework

Let the network shown in Fig.5.1b be used to generalize the concept of a single segment path shown in Eq.(5.2) on paths composed of multiple basic segments. Moreover, assume the network is using NFV framework, separating control plan from data plan, and the physical network includes datacenters (Datacenter-1 and Datacenter-2) of fat-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}, and the delay in each of them, d_{ij}. Finlay, assume that the VNFs constructing any SFC are of different types and the servers can treat anyone of them without any restrictions.

To construct the segments, following paragraph will introduce how to build the service function chain segment,

Seg^r, and based on that, the next paragraph will explain the segment formulation for a physical path, Seg^P_{ac}, matching the locations of the access nodes in Seg^r.

Formulating service function chain segment (Seg^r)

Assume SFC$^r_{ac}$ demanding allocating two different types of VNFs, VNF^1 of core processing power cpu^r_1, and VNF^2 of cpu^r_2. Moreover, assume that SFC$^r_{ac}$ demands to forward throughput capacity of bw^r at delay rates not exceeding d^r between access nodes a (at Datacenter-1) and c (at Datacenter-2). To construct SFC$^r_{ac}$ segment, the concept of path segment in Eq.(5.2) can be generalized on the graph of SFC$^r_{ac}$ shown in Fig.5.1a, and the resulting segment is formulated as in Eq.(5.3) bellow:

$$Seg^r_{ac} = \{a^r_{loc}, c^r_{loc}, cpu^r_1, cpu^r_2, bw^r, d^r\} \tag{5.3}$$

In this way, the demands of SFC$^r_{ac}$ are translated into a segment representing the demands of SFC$^r_{ac}$ as illustrated in Fig.5.1b.

Formulating physical path segment (Seg^P_{ac})

Assume that the NFV orchestrator has already the list of all pairs in the physical network shown in Fig.5.1b, and it can instruct the hypervisor layer (using the algorithms proposed by this chapter) to list all pairs that can be used to construct each physical path in the network in advance. Hence, the segmentation section in the algorithm will select a path fulfilling the locations of the demanded access nodes and all the pairs in between. To construct the path, it will connect the last node in the first pair with the first node in the next pair, if there is an edge between them, otherwise, it will go to the next pair, and do the same to the remaining pairs in the list of that path, until all pairs are connected properly formulating the path and its segment. Ultimately, the proposed algorithms residing at the hypervisor layer will follow this same methodology to construct and list all paths in the physical network.

To clarify that, refer to Fig.5.1b. Given that all physical paths and the list of their pairs are known in advance, and since SFC$^r_{ac}$ demanding access between nodes a and c, the proposed allocation algorithm will select the physical paths that match the demanded source and destination access nodes, then it will rank them according to the consumed cores of the servers in these paths. To test the possibilities of allocating SFC$^r_{ac}$, it will select the top ranked physical path, and formulate its segment.

For example, assume the physical path $P_{ac} = \{a, 1, 3, 5, 3, 2, c\}$ was the top ranked path, 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 $\{5, 3, 2, c\}$ from server 5 to destination access node c, 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 c. Accordingly, Seg^P_{ac} representing path P_{ac} can be constructed as shown in Eq.(5.4):

$$Seg^P_{ac} = \{a_{loc}, c_{loc}, bw_{a1}, bw_{13}, bw_{35}, bw_{53}, bw_{32}, bw_{2c}, cpu_5, d_{ac}\} \tag{5.4}$$

Given that:

$$d_{ac} = d_{a1} + d_{13} + d_{35} + d_{53} + d_{32} + d_{2c}$$

5.3.4 Segmentation technique for coordinated allocations

To allocate all demands of SFC^r_{ac} on the physical path P_{ac} in full coordination, allocating the virtual nodes together in the same step with the allocation of the virtual edges, each element in the two formulated set of segments Seg^r_{ac} and Seg^P_{ac} will be directly compared one-to-one, which means for each parameter value in Seg^r_{ac} representing a virtual function in SFC^r_{ac}, check if its counterpart in Seg^P_{ac} can fulfill its demands, and for the demanded bandwidth capacity and end-to-end delay by Seg^r_{ac}, check if each edge in Seg^P_{ac} has at least enough residual capacity to host that bandwidth without exceeding the demanded delay threshold.

This process can be achieved ***if and only if*** each comparison statement in the (**if-AND**) conditions shown in Fig.5.1c is true, which guarantees that all the demands of SFC^r_{ac}, including access nodes locations, processing powers of its virtual functions, bandwidth, and end-to-end delay, will be hosted together as one set by their counterparts in the physical path P_{ac}.

Therefore, segments formulation based on Eq.(5.2), and the check-up comparisons, made the allocations process of the SFCs' virtual functions, fully coordinated with the allocations of their virtual edges, together, on the same physical path.

5.4 Power aware RA-NFV on offline

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 with a simple example, and the final paragraph in this section will introduce the offline simulation settings and discuss the final results in more details.

5.4.1 Physical network model

The physical network is assumed following the network function virtualization architecture as proposed by [ETSI (2013a)]. 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, $i^a \in N^P_A$ 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, bw_{ij} current available bandwidth, and current propagation delay given by d_{ij} in ms. 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).

5.4.2 Service function chain model

In this chapter, the Authors formulated the SFCs segments in a very generic structure, so they can handle any type of VNFs. Accordingly, the ILP model does not include any constraints about the type of VNFs. Therefore, the SFCs model considers the set F, representing all types of 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^r representing the set of logical VNF server nodes belonging to SFC^r, and $u \in N^r$ 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 d^r is the maximum allowed end-to-end delay by SFC^r.

5.4.3 Offline problem formulation

In this chapter 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:

Offline RA-NFV objective function

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 [Eramo (2017a)]. 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 $mig_{i^p}^u$ is 1 if VNF u hosted by i^p is migrating, or 0 if not. Variable $mig_{ij}^{uv^r}$ is set to 1 if virtual edge (u, v) hosted by the physical edge (i, j) is migrating, variable mig^r is set to 1 if SFC^r is migrating, 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 [Eramo (2017a)] and are presented in Eq.(5.5), Eq.(5.6), and Eq.(5.7) as follows:

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

$$min\ C^{tot} = C^{pc} + C^{mig} \tag{5.5}$$

$$C^{pc} = \beta_e \Delta t (P^{idle} \sum_{i^p \in N^P} \lambda_{i^p} + \sum_{i^p \in N^P} \frac{\sum_{u \in N^r} x^u_{i^p} t^{proc}_u T^u / L^u}{CPU_{i^p}}) \tag{5.6}$$

$$C^{mig} = \beta_d \sum_{r \in S_R, u \in N^r} T^r_{down} T^u \mu^u_{i^p} \tag{5.7}$$

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 duration of a cyclic stationary interval, T^r_{down} 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 [Eramo (2017a)].

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.

Constraints formulation

The solution of the objective function Eq.(5.5) 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.(5.8) is formulated as follows:

   $$\forall\ i^p \in N^P \sum_{r \in R, u \in N^r} cpu_u x^u_{i^p} \leq CPU_{i^p} \tag{5.8}$$

   To ensure that a server is switched-on when it hosts at least one virtual function node Eq.(5.9) is formulated as follows:

   $$\forall\ i^p \in N^P, \sum_{u \in N^r} x^u_{i^p} \geq \lambda_{i^p} \tag{5.9}$$

2. *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.(5.10) is formulated as follows:

   $$\forall\ ij \in E^P\ \forall\ uv \in E^r \sum_{r \in R} bw^r x^{uv^r}_{ij} \leq BW_{ij} \tag{5.10}$$

   To ensure that the current end-to-end delay in the physical path P_{sd} is less than or equal to the

maximum allowed delay d^r by SFC^r, Eq.(5.11) is formulated as follows:

$$r \in R \sum_{\forall\ (i,j) \in P_{sd}} d_{ij} \leq d^r \tag{5.11}$$

3. *Migration constraints*

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.(5.12) is formulated as follows:

$$\forall\, i^p \in N^P \sum_{r \in R, u \in N^r} cpu_u x^u_{i^p} \leq cpu^{min} \tag{5.12}$$

To ensure that the maximum CPU capacity in physical server node i^p is greater than or equal to the demanded capacities by the already allocated VNFs v plus the migrating VNFs u on this server node, Eq.(5.13) is formulated as follows:

$$\forall\, i^p \in N^P \sum_{r \in R, v \in N^r} cpu_v x^v_{i^p} + \sum_{r \in R, u \in N^r} cpu_u mig^u_{i^p} \leq CPU_{i^p} \tag{5.13}$$

To ensure that the maximum BW capacity in physical edge (i, j) is greater than or equal to the demanded BW capacities by the already allocated edges (w, z) plus the migrating edges (u, v), Eq.(5.14) is formulated as follows:

$$\forall\, ij \in E^P \sum_{r \in R, \forall\ wz \in E^r} bw^r x^{wz^r}_{ij} + \sum_{r \in R, \forall\ uv \in E^r} bw^r mig^{uv^r}_{ij} \leq BW_{ij} \tag{5.14}$$

To ensure that the current end-to-end delay in the physical path P_{sd} is less than or equal to the maximum allowed delay d^r by the migrating SFC^r, Eq.(5.15) is formulated as follows:

$$r \in R \sum_{\forall\ (i,j) \in P_{sd}} d_{ij} \leq d^r mig^r \tag{5.15}$$

4. *Domain constraints*

To solve the problem as an ILP, Eq.(5.16)-Eq.(5.19) are defined as follows:

$$\forall\, i^p \in N^P\ x^u_{i^p} \in \{0, 1\} \tag{5.16}$$

$$\forall\, i^p \in N^P\ \mu^u_{i^p} \in \{0, 1\} \tag{5.17}$$

$$\forall\, i^p \in N^P\ \lambda_{i^p} \in \{0, 1\} \tag{5.18}$$

$$\forall\, ij \in E^P\ \forall\, uv \in E^r\ x^{uv^r}_{ij} \in \{0, 1\} \tag{5.19}$$

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.(5.20) is defined as follows:

$$\forall u \in N^r \sum_{\forall i^p \in N^P} x^u_{i^p} = 1, \tag{5.20}$$

5.4.4 PaNFV to solve RA-NFV in full coordination

Optimal solution to solve the RA-NFV objective function in Eq.(5.5) under the constraints in Eq.(5.8)-Eq.(5.20) 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 [Eramo (2017a)]. Moreover, the optimal solution for Eq.(5.5) 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 [Kleinberg (2009)].

Consequently, the majority of RA-NFV approaches followed heuristic or meta-heuristic algorithms to solve the optimization problem in a reasonable polynomial time [Mijumbi (2016)][Herrera and Botero (2016)]. Therefore, this chapter 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 for constructing the best path, which will be used to allocate the virtual functions and edges as will be explained in the following paragraphs.

5.4.5 PaNFV code explained

Pseudo-code for PaNFV heuristic is shown in Algorithm 5.1, which is explained as follows:

Initialization

The PaNFV is designed to work on NFV environments, giving that the physical network is constructed of an interconnected datacenters using access, router, and switch nodes, and each datacenter hosts a collection of physical servers in the form of fat-tree topology. The access, router, and switch nodes do not have any processing cores, only the servers will be characterized by specific number of cores, and each edge connecting these nodes has specific bandwidth and delay values. 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.

The initialization step 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, as well as a detailed information about the nodes and edges of these paths, such as, number of nodes and edges in the path, types of the nodes (access, router, switch, or server), consumed capacities from the resources of server nodes or edges, as well as end-to-end delay per each edge in the path.

Segmentation and ranking

In this step, PaNFV algorithm formulates SFC^r segment Seg^r following the format of Eq.(5.3), then to formulate the physical segment Seg^P of a candidate physical path P_{sd} it recalls all physical paths that start and terminate with the access nodes that were requested by the SFC^r. Next, it ranks these paths according to

the utilization of their servers, which is a fraction between (0-1) calculating the ratio of consumed cpu_{ip} to the maximum CPU_{ip}. Accordingly, PaNFV selects the top ranked physical path P_{sd} and formulates its segment following the format of Eq.(5.4), and handover Seg^r and Seg^P to the allocation stage.

Algorithm 5.1, 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.(5.2).
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 according to Eq.(5.2).
5. **Compare** each element in Seg^r against its counterpart in Seg^P

   - Check for CPU, BW and Delay constraints according to Eq.(5.21).
6. **If** satisfied, allocate SFC^r on P_{sd}.

   - For P_{sd} server node 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**

Allocation decision

Similar to paths ranking, PaNFV can rank the SFCs based on their demanded processing powers, throughput, delay, and also can order them according to their revenue generation potentials. However, to speed up the allocation process, this chapter processes the SFCs based on first listed first allocated. Consequently, the offline algorithm, PaNFV, lists all SFCs without any ordering, and then it handles each SFC starting from the top of the list, and proceeds to allocate it. Once an SFC was successfully allocated, it directly moves to the next SFC in the list of SFCs to be allocated.

Consequently, to guarantee full coordinated allocations for all VNFs and virtual edges in SFC^r, 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.(5.21).

Allocating virtual network functions Generally, PaNFV can allocate the VNFs of any SFC on one server, or distribute them among multiple servers. However, since the main objective of PaNFV is to minimize the total costs, allocating all VNFs of an SFC on only one single server, will minimize the total costs as low as possible, by utilizing all active servers to their full capacities before activating new ones. This will allow the PaNFV algorithm to use the least physical resources, and efficiently utilize the network resources to their full capacities. Moreover, allocating all VNFs of any SFC on one server will help in speeding up the migration processes, minimize impacts of delays, and increase service quality.

Accordingly, when allocating the VNFs of any SFC on a single server, PaNFV can work in two modes:

1. The original basic mode called (consolidated VNFs) that allocates all demanded cores on the same server, one after another. This is used in case the VNFs were not ordered in the SFC, and for strict quality of service conditions. The expense of using this mode is going to be consuming more resources from the servers.
2. The flexible mode called (non-consolidated VNFs), which allocates cores on the physical server based on the total demanded cores by the largest VNF in the SFC. This is used in case the VNFs are strictly ordered, and to utilize the servers to their maximum capacities.

Accordingly, if consolidated mode is allowed, cpu^u in Eq.(5.21) will be the sum of the demanded cores by all VNFs in the SFC, or if non-consolidated mode is allowed cpu^u will be represented by the demanded cores of the largest VNF in the SFC.

Allocating 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.(5.21) is the bandwidth of the minimum edge in the SFC, and for delay, PaNFV sums the current delays in all edges of P_{sd} and compares that to the demanded d^r in Seg^r.

Notice the use of '**AND**' in Eq.(5.21) to guarantee the full coordination, which is the tool that PaNFV uses to ensure that all allocations will be fulfilled ***if and only if*** every element in Seg^P satisfies the needs of its counterpart in Seg^r. Therefore, if the inequalities in Eq.(5.21) are '**ALL**' true, this means that the demands of SFC^r virtual function nodes and edges can be accommodated by the physical path represented by Seg^P. Consequently, a successful allocation is performed for both, VNFs and virtual edges of SFC^r together on the path of the physical segment Seg^P.

Decision matrix for the allocation process is given as follows:

$$\begin{gathered}\forall\, i^p \in P_{sd}\ and\ \forall u \in N^r\ \textbf{\textit{if}}\ cpu_{i^p} - cpu_u \geq 0\ \textbf{AND} \\ \forall\, ij \in P_{sd}\ and\ \forall r \in R\ \textbf{\textit{if}}\ bw_{ij} - bw^r \geq 0\ \textbf{AND} \\ \textbf{\textit{if}} \sum_{ij \in P_{sd}} d_{ij} \leq d^r \end{gathered} \tag{5.21}$$

Updating

Once a successful allocation occurs, the algorithm updates all changed resources and moves to the next SFC^r. However, in case that the candidate physical segment does not have enough resources to accommodate the demands of SFC^r, the algorithm jumps to the next ranked physical path and follows on from step-3. This process keeps on going until no more SFC^r to be handled.

5.4.6 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 5.2, 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.

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, if 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.

If all checks were true, 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, the 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.

5.4.7 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, considering the total number of nodes N and edges E formulating the physical network [Kleinberg (2009)]. 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 the PaNFV adopted (Bubble Sort) algorithm to sort and rank all physical network paths in descending order [Kleinberg (2009)], 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.

5.4.8 Understanding PaNFV through example

An illustrative example is shown in Fig.5.2. The physical network is composed of two datacenters, four access nodes $\{a, b, c, d\}$, two router nodes $\{1, 2\}$, two switch nodes $\{3, 4\}$, and four server nodes $\{5, 6, 7, 8\}$, each server has computing capacity of 16 core units. The maximum bandwidth capacity of each edge is 20 Gbps. Since all servers are assumed empty at the beginning, then their paths will be sorted based on the server number at the middle of each path, starting from server-1 to server-4 consequently.

Algorithm 5.2, mPaNFV Pseudo-Code

1. **Input:**
   - List all allocated SFC^r.
   - List all physical paths that are currently hosting SFC^{r}s.
2. **For** each allocated SFC^r, reformulate its Seg^r Eq.(5.2).
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 Eq.(5.2).
4. **Check if** all of the following conditions are satisfied:
   - If
   – The top ranked physical path does not currently host the SFC^r, AND
   – 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.
   - 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.(5.21).
   - If all inequalities are true, then, migrate SFC^r to the top ranked physical path.
   - 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.

In this example, PaNFV works on non-consolidated mode and is required to allocate 6 SFCs as shown at the top of Fig.5.2. SFC-1 requesting to be allocated between access nodes a and d, demanding 2 Gbps of bandwidth, having three VNFs, where the largest of them is demanding 8 core units. PaNFV will find all paths starting and terminating with access nodes a and d, then it selects and ranks those reaching server-1 (node number-5). Next,

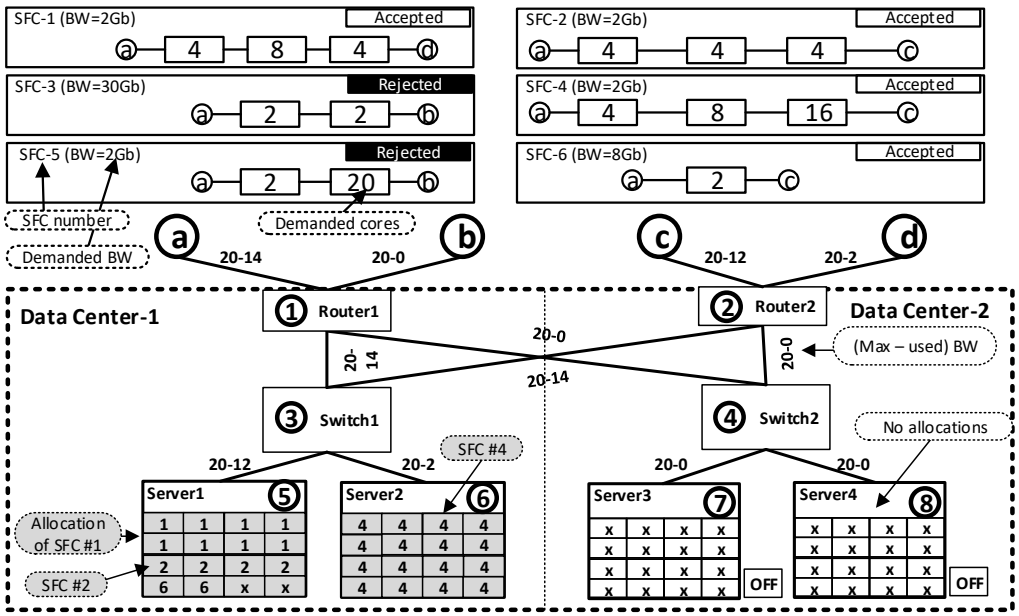

Figure 5.2: Numerical example about the basics of PaNFV.

PaNFV will select the top ranked path, which at the beginning will be path $\{a, 1, 3, 5, 3, 2, d\}$, reserve 8 core units on server node number-5, and at the same time will reserve 2 Gbps on each edge along this path from access node a to d. For SFC-2 demanding path between access nodes a to c, the largest VNF demanding 4 core units, and demanding 2 Gbps. Since server node number-5 is the most utilized among all other servers, PaNFV will rank all the paths outreaching to it and also reach to access node c, and it will select path $\{a, 1, 3, 5, 3, 2, c\}$, reserve the next available 4 core units from server node number-5, and reserve 2 Gbps along all the edges in this path. But since SFC-3 is demanding 30 Gbps, it will be rejected since PaNFV will not find any edge in the network to support it. Same case for SFC-5, PaNFV will not allocate it since the largest VNF is demanding 20 core units and no single server has such capacity.

In the case of SFC-4, the largest VNF is demanding 16 core units, PaNFV will decide to allocate them on server-2 (node number-6) through path $\{a, 1, 3, 6, 3, 2, c\}$, since server-1 does not have enough capacity to host the whole of SFC-4. Notice that PaNFV will always stick to allocate the demanded cores of any SFC on one server and will not split it between other servers at all. Finally, PaNFV will allocate SFC-6 on server-1 through path $\{a, 1, 3, 5, 3, 2, c\}$ since it is the next most utilized (server-2 is the most utilized, but it is full) server and has enough core capacity to host all of SFC-6. After each allocation attempt, PaNFV will evaluate the utilization of all server nodes in the network, and will turn them off if they are idle, such as server-3 (node number-7) and server-4 (node number-8) in datacenter 2.

5.4.9 Evaluation metrics

The following evaluation metrics were used by [Eramo (2017a)] to evaluate their RCPP/RLARCDP algorithms. Accordingly, this chapter used them as reference evaluation metrics as well.

Table 5.1: Comparing PaNFV and mPaNFV to RVPP and RLARCDP algorithms

Item	PaNFV / mPaNFV	RVPP / RLARCDP
Scenario	Offline	Offline
Goal	Minimize total cost (allocations/migration)	Minimize total cost (allocations/migration).
Strategy	Resources consolidation	Resources consolidation
Ranking	Rank paths based on utilization of their servers	Rank servers and edges based on their utilization efficiencies.
Allocating SFC's VNFs	All on one single server	All on one single server
Where to allocate VNFs	On the server of the top ranked path	On lightly loaded server with low edge stress.
Virtual edges' allocation	Together with VNFs on the top ranked path	After allocating VNFs on the shortest path.
Migration	At off-peak	At off-peak.
Migration condition	If the server's utilization below threshold	Migrate traffic from least utilized nodes.
servers' activation	One by one according to traffic load	All active with uniform loads.
Power minimization	Turn-off non utilized servers	Turn-off non utilized servers.
End-to-end delay	Yes (but not used in the simulations)	No

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

$$C^{tot} = \sum_{\forall r \in R} (C^{pc} + C^{mig}) \tag{5.22}$$

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

$$C^{pc^{tot}} = \sum_{\forall r \in R} C^{pc} \tag{5.23}$$

C^{pc} is given as in Eq.(5.6)

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

$$C^{mig^{tot}} = \sum_{\forall r \in R} C^{mig} \tag{5.24}$$

C^{mig} is given as in Eq.(5.7)

•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 [Eramo (2017a)]:

Table 5.2: Simulation settings to evaluate PaNFV and mPaNFV.

Scenario	Offline.	Network	As in Fig. 5.3.
servers' cores	48 cores/server.	Edges' bandwidth	10 Gbps server-switch, 40 Gbps else.
SFCs topology	As in Fig.5.6.	Number of SFCs	500 or 1500 for blocking simulations.
SFCs bandwidth	Random $(100, 150, 200, 250, 300)$ Mbps.	VNFs types	FW, IDS.
VNFs cores	4 cores for FW and 8 for IDS.	t_u^{proc} per VNF type	120, 160, 82.67 $\mu sec.$
Δt	1 hour.	T_{down}	2 sec.
β_d	$1.8E^{-7} - 2.7E^{-6}$	β_e	1
Simulation Runs	10	T^u	Equal to SFC bandwidth.
L^u	1500 Bytes.	P^{Busy}	1000 Watts.
P^{idle}	$a * P^{Busy}$.	a	0.3 and 1.

$$\forall\, ij \in E^P \;\forall\, uv \in E^r \; Blocking = \frac{\sum_{r \in R} bw^r - \sum_{r \in R} bw^r x_{ij}^{uv^r}}{\sum_{r \in R} bw^r} \tag{5.25}$$

5.4.10 Simulation settings for offline scenario

The simulations of PaNFV and mPaNFV were conducted against the best performing allocation and migration algorithms provided by [Eramo (2017a)] using the global policy. The same settings used to test the global policy were used for PaNFV and mPaNFV, as well as using SFCs of one VNF, and physical network topology, which includes 4 interconnected datacenters, each of them has 16 servers formed in fat-tree topology, and each server has 48 cores. The cost results of PaNFV / mPaNFV were compared to those of the global policy results in [Eramo (2017a)] when the fraction of server's idle to maximum powers was set equal to 0.3 and 1.

5.4.11 PaNFV and mPaNFV performance evaluation

To test PaNFV and mPaNFV the following sections compare their performance, against the routing and VNF placement problem algorithm (RVPP) and the revenue loss aware resource consolidation / de-consolidation problem algorithm (RLARCDP) suggested by [Eramo (2017a)]. Table.5.1, compares the main features of the algorithms in high level.

As an overall overview, the difference between PaNFV / mPaNFV and RCPP/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 [Eramo (2017a)] RVPP allocates the SFCs based on their demanded bandwidths in decreasing order and on the least stressed servers. While RLARCDP 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 that 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.

$$\alpha_h = \begin{cases} 1, & if\, h = 0 \\ 1 - 2\frac{h}{Q}(1 - \alpha_{min}), & if\, h = 1, ..., \frac{Q}{1} \\ 1 - 2\frac{Q-h}{Q}(1 - \alpha_{min}), & if\, h = \frac{Q}{2} + 1, ..., Q - 1 \end{cases} \tag{5.26}$$

Traffic model

To characterize semi-daily traffic profile, and since PaNFV is going to be compared to [Eramo (2017a)], it is assumed that the bandwidth demands of the SFCs are scaled according to the traffic patterns of a cyclic-stationary profile as suggested by [Eramo (2017a)], where the changes in traffic are predictable and reoccur on periodical and regular basis. Therefore, Eq.(5.26) that was suggested by [Eramo (2017a)] is also used in this chapter to make 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.

Based on this traffic profile, the ILP objective function in Eq.(5.5) and all the constraints starting by Eq.(5.8) to Eq.(5.20) will be checked at each traffic interval [$h \in 0, 1, ..., Q - 1$], and if satisfied, the evaluation metrics in Eq.(5.22) to Eq.(5.24) will be calculated in order to compare the results of PaNFV / mPaNFV against the results from RVPP / RLARCDP.

Table.5.2 lists all simulation settings used to analyze the performance of PaNFV and mPaNFV, and the results are discussed in some details in the following paragraphs.

5.4.12 Results and discussions of offline algorithms

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 problem algorithm (RLARCDP) for migrations as suggested by [Eramo (2017a)] who focused on the impact of using servers that has different idle to maximum power ratios on the datacenters' total costs.

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 [Eramo (2017a)] 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.

Impacts of different idle to maximum power ratios Referring to Fig.5.3 when idle to maximum power ratio increases, the only obvious impact on overall performance of PaNFV and PaNFV / mPaNFV was the increase in the levels of total costs as the a values go higher, which reflects normal behavior of the servers' increasing idle powers. 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.

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.5.3:a,d,g,j shows the total costs of PaNFV from [Hejja (2018)], PaNFV / mPaNFV, and RVPP / RLARCDP based on [Eramo (2017a)], 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 time outperformed [Eramo (2017a)] for the same a values by 32.1% points. For example refer to the results from Fig.5.3a and Fig.5.3j, where on average the total costs of PaNFV / mPaNFV were lower than those of RVPP / RLARCDP by 54% and 9% respectively. However when PaNFV without migrations was compared to RVPP / RLARCDP, it outperformed it by 58% and 13% respectively, and it outperformed PaNFV / mPaNFV by 9% and 5% respectively. Noting that in Fig.5.3j, 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, causing the total costs of [Eramo (2017a)] 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 migration algorithm was activated.

Power consumption costs Regarding the results of the power consumption costs as shown in Fig.5.3:b,e,h, they indicate that PaNFV and PaNFV / mPaNFV have nearly flat costs as a function of the cost of Gbits lost, and less costs than RVPP by 35.7%, thanks to less activated servers and very negligible migrations as well. However, in Fig.5.3k when $a = 1$, they 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.

Migration costs Referring to the migration costs shown in Fig.5.3:c,f,i,l, mPaNFV had almost negligible migration costs, mainly because of the precise allocations by PaNFV that consumed as least resources as possible in the first place. However, focusing on the results of RLARCDP migration algorithm as extracted from [Eramo (2017a)], it had much higher migration costs compared to those of mPaNFV, most probably

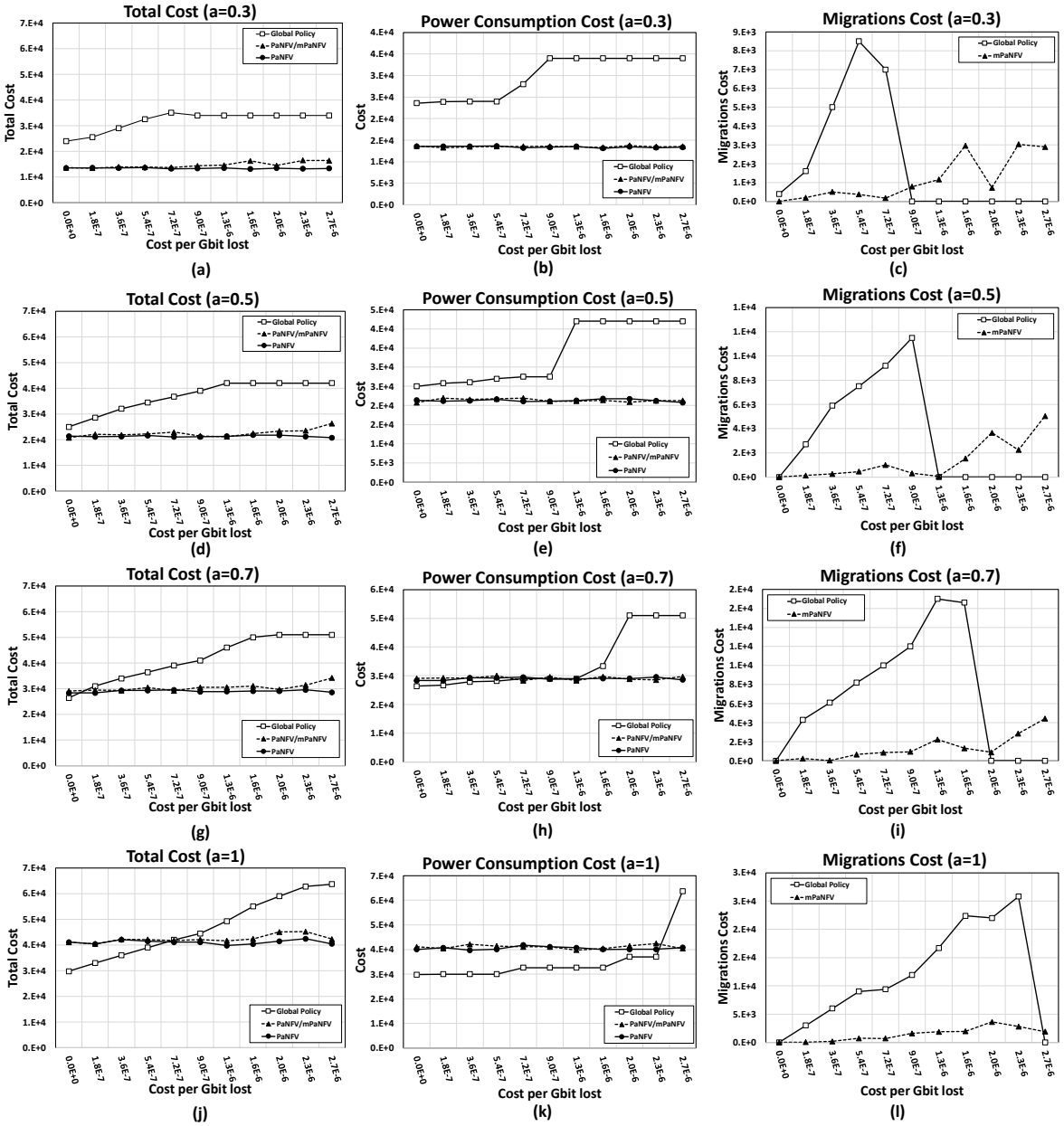

Figure 5.3: PaNFV / mPaNFV compared to RVPP / RLARCDP.

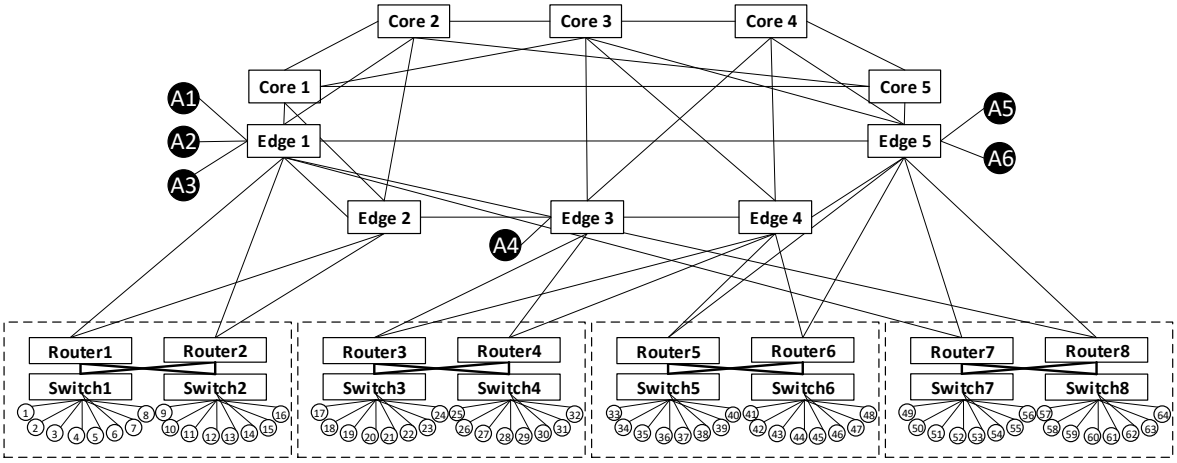

Figure 5.4: Physical network.

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, then allocates the virtual edges in another separate phase using the shortest path algorithm, that what most likely forced the migration algorithm, RLARCDP to increase the number of migrations to minimize the number of active servers, but 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 [Eramo (2017a)], 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.

Understanding blocking results To evaluate the blocking probability of PaNFV, it is performance was compared against the allocation algorithm RVPP by [Eramo (2017a)], and the VNF placement algorithm developed by [Bari (2016)] as extracted from [Eramo (2017a)]. 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 chapter.

The conducted simulations to test PaNFV blocking probability used the same network topology as in [Eramo (2017a)], 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.5.5, 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.

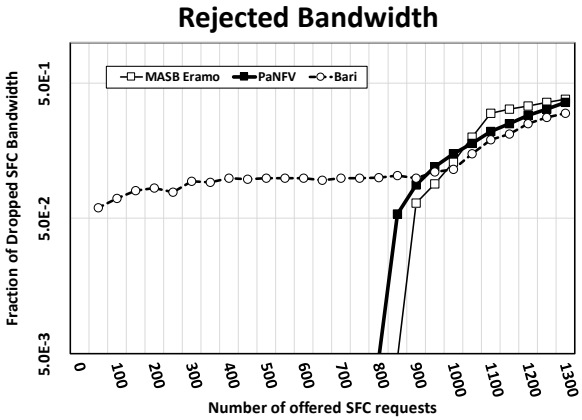

Figure 5.5: Fraction of blocked bandwidth.

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 [Bari (2016)] 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.5.5, it can be seen that, on average PaNFV performance was better than RVPP by 7% and 37% than the VNF Placement algorithm of [Bari (2016)], 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.

Impacts of VNFs consolidations on PaNFV allocation times 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.

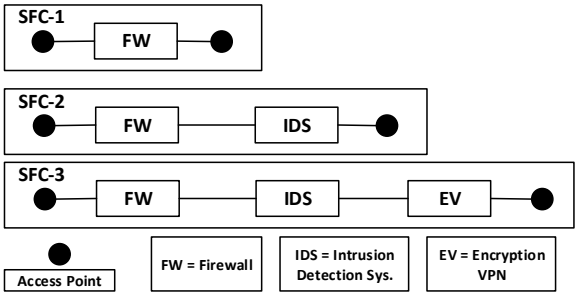

Figure 5.6: SFCs topologies for experiments 2 and 3

VNFs consolidation PaNFV strictly allocates all VNFs in SFC^r on the same Server only if there is enough capacity, otherwise if no other servers can host that SFC it will be 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, minimize impacts of delays, and guarantee 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 is about 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.

Accordingly, if non-consolidated mode is allowed, cpu^u in Eq.(5.21) 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.5.6, and for idle to maximum servers' power ratios of 0.3 and 1.

Discussions for the results of experiment-2 Referring to Fig.5.7: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.5.7: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.5.7: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 [ITU 5G Opportunities and Challenges (2018)]-[ITU (2011)]. 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, 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.

Impacts of migrations and VNFs consolidations on PaNFV 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.5.8: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.5.8: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.5.8: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.5.8: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.

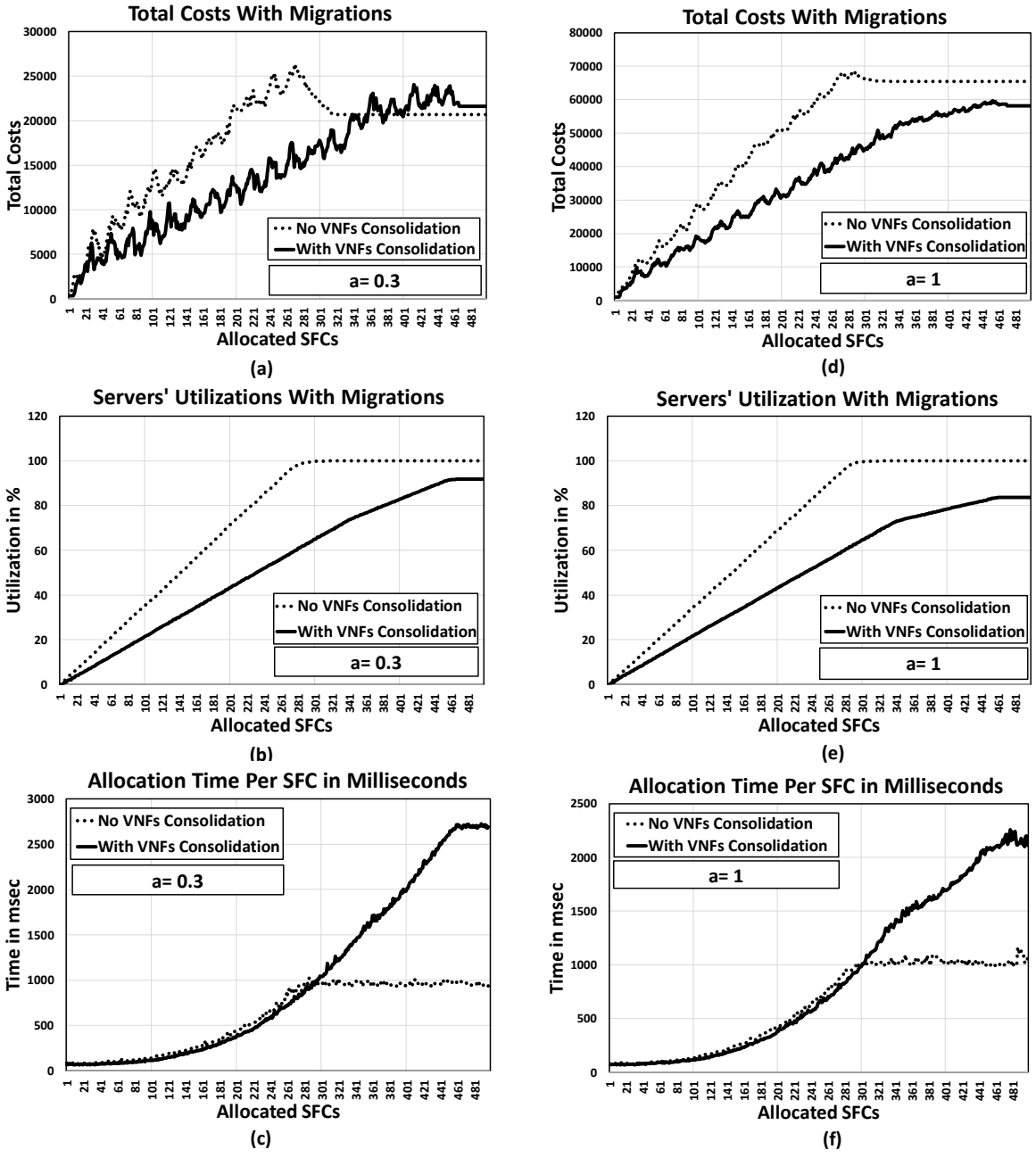

Figure 5.7: Impacts of activating VNFs consolidations when migrations are always used

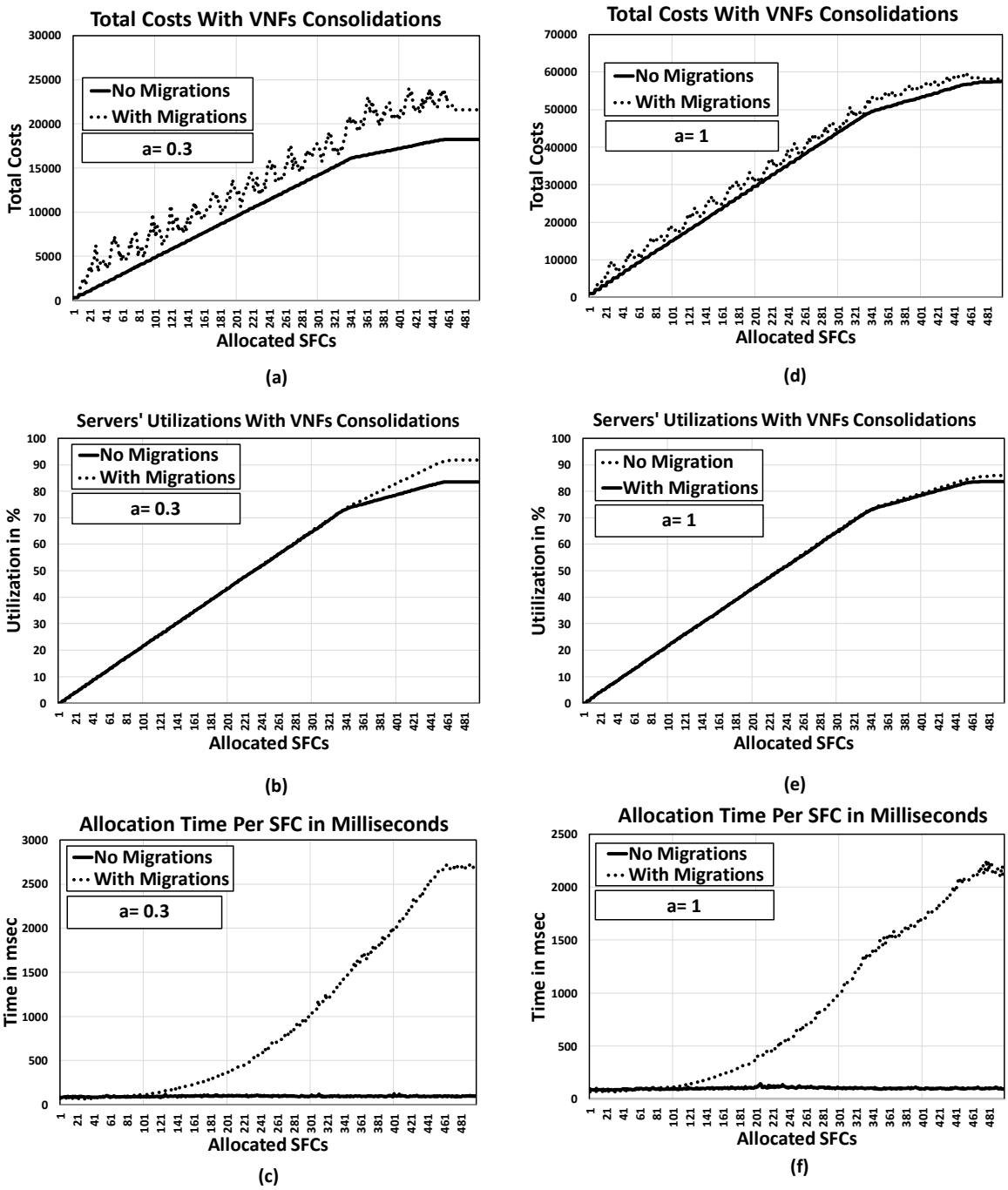

Figure 5.8: Impacts of activating Migrations when VNFs consolidations are always used

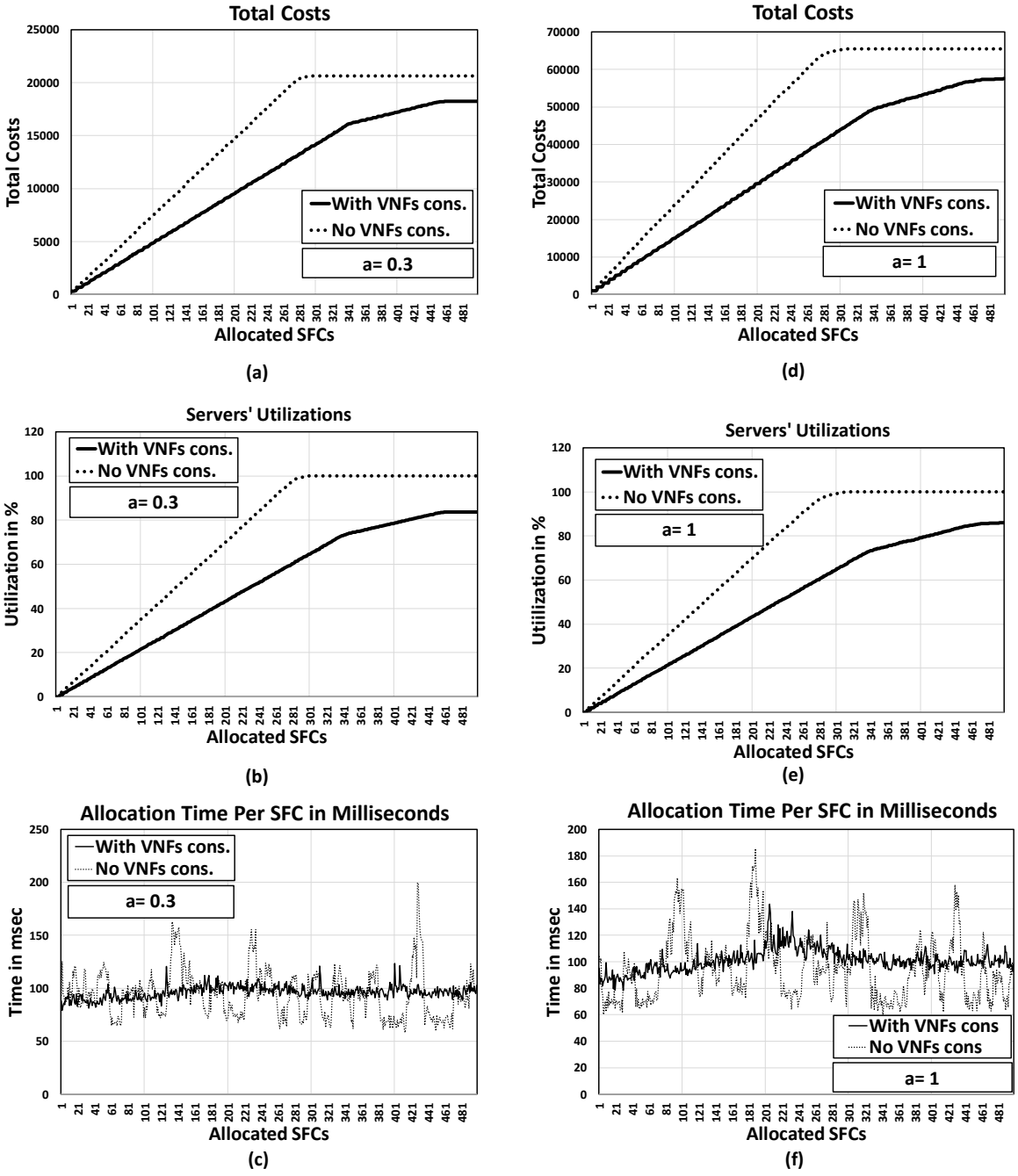

Figure 5.9: Impacts of VNFs consolidations on the original PaNFV with no migrations

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.

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.

Impacts of VNFs consolidations on PaNFV allocation times 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.5.9:c,f shows that the original PaNFV without migrations and without VNFs consolidations was on average faster than PaNFV with VNFs consolidations by slight margin of 2.49%. However, the total costs shown in Fig.5.9:a,d indicate that PaNFV without VNFs consolidations had on average much higher values by 26.65% than PaNFV with VNFs consolidations, and in terms of servers utilizations it resulted on higher utilizations by 29.04%.

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

5.5 Power aware RA-NFV on online

In this section, the online algorithm, oPaNFV, and the migration algorithm, mPaNFV, are introduced as a power aware coordinated NFV algorithms, allocating, migrating and removing SFCs online. It is assumed that the inter-arrival times of the SFCs reaching the physical network are normally distributed, and each SFC has arriving time and lifetime. oPaNFV allocates each SFC once it arrives and keeps monitoring until it's lifetime get expired, which would then trigger oPaNFV to terminate that SFC, by releasing its demands from their hosting physical resources in order to be used by other arriving SFCs.

Regarding migrations during online scenario, mPaNFV is inherited within oPaNFV and is basically triggered whenever the overall traffic profile in the network is at very low conditions. Moreover, mPaNFV is also kept on alert, always monitoring and evaluating the utilization of the active servers, and once it detects the utilization on one of them below the threshold migration level, mPaNFV will attempt to migrate all traffic from that low utilized server node to others, then turns it off to reduce the total power consumption in the network.

Following paragraphs explain the online objective function and its constraints, introduce the oPaNFV algorithm, and ends by discussing the simulation results.

5.5.1 Online objectives

The main target of the objective function for oPaNFV is to minimize the overall power consumption in the whole physical network, PC, by efficiently allocating the arriving SFCs, then putting into sleeping mode all idle servers that do not host any traffic. SFCs in the online scenario arrive at certain time defined by variable t^{ar}, and expire after consuming specific time units defined by variable t^{ex}. Therefore, the objective function must consider allocating these SFCs within these time boundaries.

5.5.2 Power consumption model

For the online scenario, the linear power model introduced by [Fan (2007)] cited by [Dayarathna (2016)], is used to estimate the power consumption of the physical server, PC_{i^p}, as given in Eq.(5.27), which sums the server's idle power and the power consumption due to the traffic load (Utilization) on that server. Server's utilization, U_{i^p}, is a fraction between (0-1), reflecting the traffic load as a ratio between the consumed cpu_{i^p} to the maximum CPU_{i^p} on the physical node i^p given by Eq.(5.28).

$$\forall\, i^p \in N^P \; PC_{i^p} = P^{idle} + [P^{Busy} - P^{idle}] \times U_{i^p} \tag{5.27}$$

$$U_{i^p} = \frac{cpu_{i^p}}{CPU_{i^p}} \tag{5.28}$$

Objective function formulation

For the online scenario Eq.(5.28) will be applied as an ILP objective function, targeting to minimize the total power consumption in the whole network after each allocated SFC. It will be constrained by capacity and domain constraints to guarantee that the optimal solution should be reached within the demanded time intervals by the SFCs.

To solve the objective function as an ILP problem, variable $x_{i^p}^{u^t}$ is used, which takes a binary value of 1 if server i^p is assigned to host SFC^r during the time interval $t \in [t^{ar}, t^{ex}]$, 0 otherwise. Variable $\lambda_{i^p}^{t}$ is 1 if the physical node i^p is active, or 0 if it is turned-off during time interval t. Variable $\mu_{i^p}^{u^t}$ is 1 if during time interval t, all VNF nodes hosted by i^p are migrating, or 0 if they remain. Variable $mig_{ij}^{uv^{r^t}}$ is set to 1 if virtual edge (u, v) hosted by the physical edge (i, j) is migrating, variable mig^{r^t} is set to 1 if SFC^r is migrating, and variable $x_{ij}^{uv^{r^t}}$ is set to 1 if physical edge (i, j) is hosting virtual edge (u, v).

The objective function is shown in Eq.(5.29):

$\forall\, t \in [t^{ar}, t^{ex}]$

$$\min PC = \sum_{\forall i^p \in N^P} (P^{idle} + [P^{Busy} - P^{idle}] \times U_{i^p}^{t}) \tag{5.29}$$

Constraints formulation

1. *Constraints on server nodes*

$$\forall\, i^p \in N^P \; \forall\, t \in [t^{ar}, t^{ex}] \sum_{r \in R, u \in N^r} cpu_{u^t} x_{i^p}^{u^t} \leq CPU_{i^p} \tag{5.30}$$

$$\forall\, i^p \in N^P,\ \forall\, t \in [t^{ar}, t^{ex}] \sum_{u \in N^r} x_{i^p}^{u^t} \geq \lambda_{i^p}^t \tag{5.31}$$

2. *Constraints on physical edges*

$$\forall\, ij \in E^P\ \forall\, uv \in E^r\ t \in [t^{ar}, t^{ex}] \sum_{r \in R} bw^{r^t} x_{ij}^{uv^{r^t}} \leq BW_{ij} \tag{5.32}$$

$$r \in R\, t \in [t^{ar}, t^{ex}] \sum_{ij \in P_{sd}} d_{ij}^t \leq d^{r^t} \tag{5.33}$$

3. *Migration constraints*

$$\forall\, i^p \in N^P\ \forall\, t \in [t^{ar}, t^{ex}]$$
$$\sum_{r \in R, u \in N^r} cpu_{u^t} x_{i^p}^{u^t} \leq cpu^{min} \tag{5.34}$$

$$\forall\, i^p \in N^P\ \forall\, t \in [t^{ar}, t^{ex}]$$
$$\sum_{r \in R, v \in N^r} cpu_{v^t} x_{i^p}^{v^t} + \sum_{r \in R, u \in N^r} cpu_{u^t} mig_{i^p}^{u^t} \leq CPU_{i^p} \tag{5.35}$$

$$\forall\, ij \in E^P\ t \in [t^{ar}, t^{ex}]$$
$$\sum_{r \in R, \forall\ wz \in E^r} bw^{r^t} x_{ij}^{wz^{r^t}} + \sum_{r \in R, \forall\ uv \in E^r} bw^{r^t} mig_{ij}^{uv^{r^t}} \leq BW_{ij} \tag{5.36}$$

$$\forall\, r \in R\, t \in [t^{ar}, t^{ex}]$$
$$\sum_{\forall\ (i,j) \in P_{sd}} d_{ij} \leq d^{r^t} mig^{r^t} \tag{5.37}$$

4. *Domain constraints*

   Similar to Eq.(5.16)-Eq.(5.20).

The above constraints behave similarly as when they were used for the offline scenario. However, the main difference in the online scenario, is that each constraint must also be true in each time interval $t \in [t^{ar}, t^{ex}]$.

5.5.3 Solution of the online objective function

The optimal solution to solve the online RA-NFV objective function in Eq.(5.29), implies introducing binary constraints to allocate all VNFs on one physical server node, which is similar to the multi-dimensional NP-Hard Bin Packing problem that is not solvable in polynomial times [Mijumbi (2016)][Herrera and Botero (2016)]. Therefore, this chapter proposes a modified version of the online power aware allocation algorithm developed in chapter 3 and recently published by [Hejja (2019)], called oPaNFV, to solve the online objective function, and fully coordinating allocating the VNF nodes and edges together on the best physical path.

Table 5.3: Simulation settings to test oPaNFV

Common		**Exp-3**	**Impacts of loading network**.
SFCs' lifetime	400 time units in all experiments.	Main objective	Test long/short SFCs' lifetimes.
Migration	Always active in all experiments.	Expiring SFCs	$(1-10)$ every $(5-10)$ time units.(Long)
Arriving SFCs	$(5-10)$ every $(5-10)$ time units.		$(5-10)$ every $(5-10)$ time units.(Short)
Consolidation	Non-consolidation mode.	Simulation time	5000 time units.
Exp-1	**Impact of migrations on oPaNFV**.	**Exp-4**	**Impacts of delay**.
Main objective	oPaNFV with/out migrations.	Main objective	oPaNFV with/out delay.
Expiring SFCs	$(1-10)$ every $(5-10)$ time units.	Expiring SFCs	$(1-10)$ every $(5-10)$ time units.
Simulation time	$100,000$ time units.	Simulation time	5000 time units.
Exp-2	**Impact of VNFs' consolidations**.	Physical delay	Random $(0.1-1)$ ms per edge.
Main objective	oPaNFV with/out consolidations.	SFC delay	Random $(1-20)$ ms, less sensitive SFCs.
Arriving SFCs	10 every 10 time units.		Random $(0.1-5)$ ms, Sensitive SFCs.
Expiring SFCs	10 every 10 time units.		
Simulation time	5000 time units.		

5.5.4 oPaNFV explained

The oPaNFV pseudo-code is shown in Algorithm 5.3. In the general sense it works similar to the online version from [Hejja (2018)], but for NFV environment. oPaNFV has four main steps as in the offline version PaNFV, namely, the initialization, segmentation, allocation, and the updating, where each of them works exactly as they did for the PaNFV. As a summary, at each time interval t, if an SFC arrives the network at t^{ar}, oPaNFV will construct its segment as in Eq.(5.3), and will formulate the physical segment for the top ranked physical path P_{sd} as in Eq.(5.4). Then oPaNFV will check if physical path has enough resources to host the arriving SFC^r during time interval $t \in [t^{ar}, t^{ex}]$. If yes, oPaNFV will allocate that SFC^r on the selected physical path and update the whole network accordingly.

To make oPaNFV work for online conditions as those in 5G networks, it includes a new part to monitor the lifetimes of the arriving SFCs, as well as to handle the termination procedure if any SFC is expiring. It works as follows, at each time unit the algorithm keeps checking if the lifetime of any SFC is about to be expired, oPaNFV will then recall the path hosting that SFC, and remove its demands from the hosting resources of that path. In parallel with that, and after each removal cycle, oPaNFV will always keep evaluating the power consumption of each server node if it is idle or loaded, and turns it off if it does not host any traffic to save the overall power consumption in the whole network accordingly.

Important to highlight that, the most significant addition by oPaNFV is the inclusion of migrations during online. At every moment of time oPaNFV will keep evaluating the utilization levels in the physical network, and if the three migration conditions (as clarified in mPaNFV) are met, oPaNFV will trigger the migration procedure, and accordingly will turn off any non loaded server to minimize the total power consumption. Notice that, oPaNFV can also trigger the migration algorithm for maintenance and emergency conditions at any time interval t.

Algorithm 5.3, oPaNFV Pseudo-Code

1. Input: G^P and G^r.
2. While t$\neq$0 **Do**
3. **Follow steps-(2-7)** as in PaNFV algorithm
4. **For** all allocated SFCs **Do**
   - Check if any SFC is expiring
     - Release the demands of the expiring SFCs (cpu_u and bw^r) from the hosting resources.
5. **Follow steps-(8-11)** as in PaNFV algorithm

5.5.5 Simulation settings to test oPaNFV

Because oPaNFV is an evolution of the offline algorithm PaNFV, which was evaluated against [Eramo (2017a)], then in the online scenario, all settings listed in Table.5.2 are also valid and used during the evaluation procedure of oPaNFV, including the same physical network topology and SFCs' configurations. In addition to that, since [Eramo (2017a)] does not include online algorithms, the authors could not compare their results to them, but they modified the online algorithms from [Hejja (2018)] to work for NFV frameworks and migrations including end-to-end delay.

Four different experiments were conducted to evaluate the impacts of migrations, VNFs' consolidations, SFCs' lifetimes, and delay on the overall performance of oPaNFV. Simulation settings for all these experiments are listed in Table.5.3, wich starts by summarizing the common parameters that were used in all experiments unless otherwise stated, assuming that the average lifetime for any SFC is 400 time units, migrations are always allowed, and the number of arriving SFCs selected randomly between $(5-10)$ SFCs every $(5-10)$ time units. Moreover, notice that in all experiments the types of SFCs are similar to those from [Eramo (2017a)], where the number of VNFs per SFC are randomly selected between $(1-3)$, but the VNFs will be allocated following the non-consolidation mode as follows: if the arriving SFC has one VNF, then 4 cores are allocated, while if the SFC is of two VNFs $4, 8$, then 8 cores are allocated, and if it is of three VNFs $4, 8, 4$, also 8 cores will be allocated only.

The first experiment (Exp-1) does not include end-to-end delay, since it targeted evaluating the basic performance of oPaNFV with and without migrations, in terms of power consumption, utilizations of the physical servers, as well as the number of active servers. In Exp-1, a total of $(5-10)$ SFCs were assumed randomly arriving the physical network every $(5-10)$ time units, while $(1-10)$ SFCs assumed expiring every $(5-10)$ time units. The migration algorithm, mPaNFV on Exp-1 was always kept active, and depending on the migration conditions as defined in Algorithm 5.2, the migrations could be triggered at anytime during the whole simulation time. All demanded bandwidth values for the arriving SFC were randomly selected between $(100, 150, 200, 250, 300)$ Mbps, and the simulations were conducted over $(0-100,000)$ time units.

Second experiment (Exp-2) targeted evaluating the total power consumption performance of oPaNFV and other metrics, when the VNFs' consolidations on the physical servers were not allowed, and compare that to

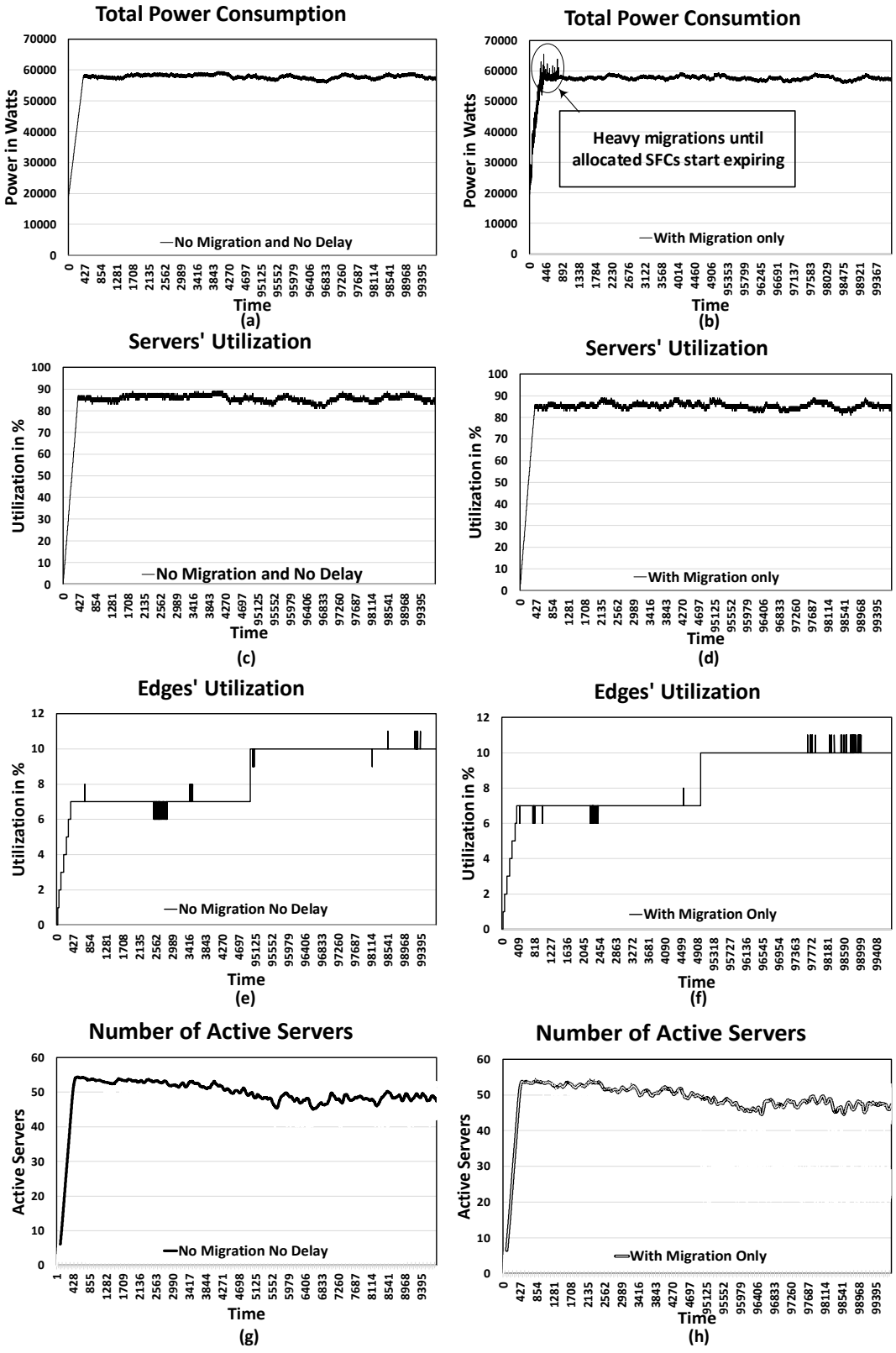

Figure 5.10: Overall Performance of oPaNFV with and without migrations.

the results when they were allowed. All other simulation settings are similar to Exp-1, including always active migrations, except that the simulation time was set equal to 5000 time units, assuming that 10 SFCs arriving, and other 10 expiring every 10 time units, and Exp-2 does not include end-to-end delay too.

Third experiment (Exp-3) targeted evaluating the performance of oPaNFV when the network is loaded, by varying the number of expiring SFCs, and analyzing how is that impacting the physical network. The experiment was conducted when the average lifetime per arriving SFCs was varied by setting the lifetimes equal to $400 + \delta^l$, giving that δ^l was selected random between $(1 - 10)$ time units for longer lifetimes (i.e. heavily loaded network), and between $(5 - 10)$ time units for shorter lifetimes (i.e. lightly loaded network).

Fourth experiment (Exp-4) targeted evaluating the performance of oPaNFV when end-to-end delay was used. Generally, it is possible to include any type of delay in the segmentation technique, giving that the authors did extensive review to the possible delays including propagation, transmission, processing and queuing delays, and concluded to follow the recommended end-to-end delay values for 5G technologies as stated in [ITU-T Focus Group (2017)], which includes all delay types. Accordingly, to simulate impacts of end-to-end delay on the allocations of SFCs on $5G$ like networks, Exp-4 will inject two types of SFCs to be allocated on the physical network, and evaluate the resultant end-to-end delays by comparing the results of both types to each other. The first type includes SFCs which are less sensitive to physical network delay, as is the case for most $5G$ applications, and the second type of SFCs are those very sensitive to end-to-end delay. Delays in the physical network edges were randomly selected between $(0.1 - 1)$ ms, and the demanded end-to-end delay by the first type of SFCs were randomly selected between $(1 - 20)$ ms, and between $(0.1 - 5)$ ms for the second type of SFCs.

5.5.6 Discussing results of oPaNFV performance

Impacts of migrations

Simulation results of Exp-1 are shown in Fig.5.10, presenting the performance of oPaNFV with and without migrations. The experiment was conducted for $100,000$ time units, but due to the large number of samples, the results are shown for the first and last 5000 time units using the real numbers from the simulations. Moreover, to show some details and to give real sense about the overall behavior of the online algorithm for a large number of demands, the results of the migration case were drawn in separate figures than those without migration.

The total power consumptions are shown in Fig.5.10a when migration was not included, and in Fig.5.10b with migrations, indicating that oPaNFV conducted some limited migrations in the early stages of the simulation time. However, the behavior of the algorithm stabilized when the allocated SFCs started expiring after passing the first 400 time units, where the total power consumption remains smoothly varying on average around 57.04 KWatts without migrations, representing (89.12%) of the network's maximum power, and 57.16 KWatts (89.31%) with migrations. These results reflect the online nature of oPaNFV and how it handles the arriving and expiring SFCs, with almost very negligible migrations along the simulation time frame. Servers' utilizations were in the averages of 83.98% when no migrations were allowed as in Fig.5.10c, and 83.62% with migrations Fig.5.10d.

Important to notice that the high values of power consumption and utilization were directly affected by the longer lifetimes of the arriving SFC, which is an indication about the overall network loads and the resources consumptions. Lastly, in Fig.5.10e and Fig.5.10f the average number of active servers was high, around 49 in

both cases, mainly due to the long lifetimes of the arriving SFCs, and that is why the total power consumptions were high too.

The overall conclusion from Exp-1 is that the performance of oPaNFV on a long run simulation times is stable and very smooth, but greatly depends on the lifetimes of the arriving SFCs. However, the bold conclusion from the previous observations is that, giving the careful allocation strategy of oPaNFV, migrations in online are mostly very minor or negligible, almost very few migrations at the early stage of the experiment. Therefore, it would be more efficient to use the migrations for maintenance or emergency cases, than to minimize the total power consumption in the physical network. This will further reduce the complexity in oPaNFV and make it even much faster.

Impacts of VNFs consolidations

Fig.5.11 shows the results of oPaNFV when VNFs' consolidations were allowed and not allowed. Remember, the consolidate case is when the reserved cores on the server are the sum of the total demanded cores by all VNFs of the SFC, and the non-consolidated is when the cores were reserved based on the demands of the largest VNF in the SFC. Note that it is only in Exp-2 the results were drawn using the moving average trends to show the curves more clearly.

To understand the overall impacts on oPaNFV, Fig.5.11a shows that the average power consumption of the whole physical network for the two cases were very close to each other. It was around 56.27 KWatts, that represents 87.92% of the network's maximum power in the case of non-consolidated VNFs, compared to 58.76 KWatts (91.81%) with VNFs allowed to consolidate.

The resulted savings in the total power consumption were around 4% on average, which means that when non-consolidation technique is used, the physical network will have additional room to allocate more SFCs, but at slightly lower power consumption rates than the consolidated case.

However, edges' utilizations results shown in Fig.5.11b did not follow the same trend as the previous results from other metrics, and they were 6.7% in the case of non-consolidated VNFs and 4.1% when consolidations were allowed. In non-consolidated VNFs case, the higher values in the edges' utilizations were due to oPaNFV's strategy of using the same paths more frequently for multiple SFCs until their servers get fully utilized. However, in the case of consolidated VNFs the servers will be utilized faster and their paths will be less used by multiple servers, accordingly, they will be less utilized. To some degree this is confirmed by the number of active servers, which were around 50 in the non-consolidated case and 49 with consolidations as shown in Fig.5.11c.

The overall conclusion from experiment-2 is that for some virtual network services, where the VNFs of a specific SFC are strictly ordered and not working on parallel, non-consolidation mode could be a good choice, since it allows the physical network to accommodate more SFCs at lower power consumption rates. However, if the VNFs are not ordered and my demand to allocate them in parallel, the consolidated mode could be applied, noting that it will slightly increase the total power consumption.

Impacts of varying SFCs' lifetimes

To understand the impacts of SFCs' lifetimes on the load of the physical network, Exp-3 varied the lifetimes of the arriving SFCs in two separate runs, long versus short living SFCs. For long living SFCs, the lifetimes

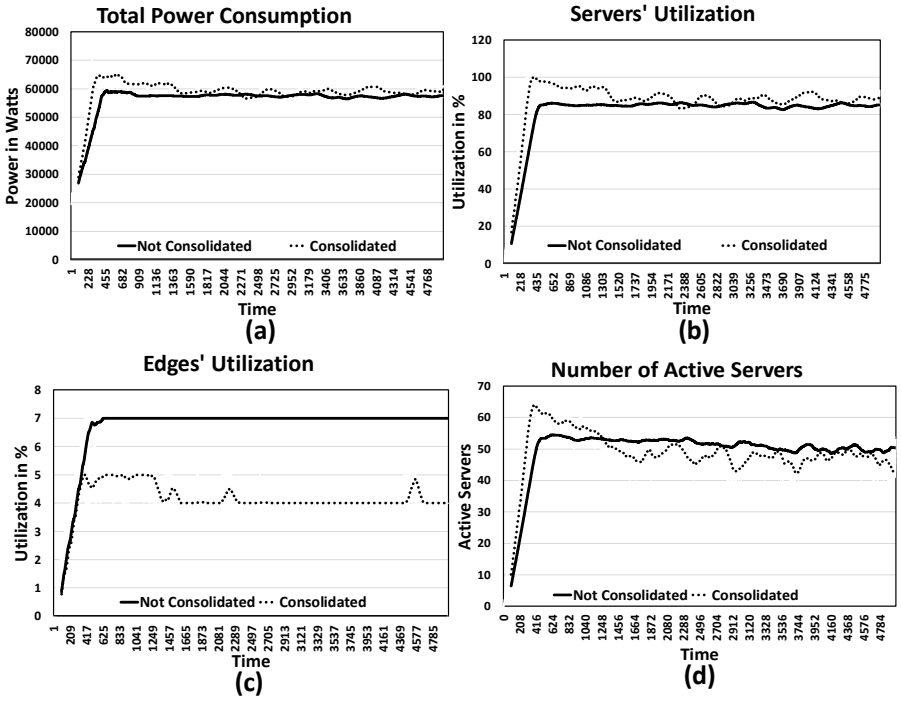

Figure 5.11: oPaNFV performance with/out consolidating the VNFs.

were varied between $(400 + \{1\ to\ 10\}))$ time units, and for short living SFCs it varied their lifetimes between $(400 + \{5\ to\ 10\}))$ time units. As shown in Fig.5.12a, when less number of SFCs were leaving (i.e. has longer lifetimes), the total power consumption in the whole network was almost hovering around 61.64 KWatts, representing 96.3% of the maximum network's power, but it was around 39.68 KWatts (62.1%) when more SFCs were leaving (i.e. having shorter lifetimes). Same trends can be seen for the servers' utilizations, which were around 93.56% and 44.65% as in Fig.5.12b. In addition, when considering the number of active servers in each case, Fig.5.12c shows that on average, 25 servers remain active when SFCs of the shorter lifetimes were hosted, compared to 58 active server when SFCs of longer lifetimes were allocated.

The overall conclusion from Fig.5.12 is that the longer the lifetimes of the allocated SFCs, the more the power consumption, and the higher the servers and edges' utilizations on the physical network. Most noticeable, is that even for a slight variation in the number of expiring SFCs, the impacts on the physical network metrics were immense and so obvious. Suggesting that, among all other physical network's parameters, SFCs' lifetime would most likely be the dominating factor on the overall performance of the physical network.

Impact of end-to-end delay

As a main contribution from this chapter, Exp-4 was designed to reflect the most realistic performance of oPaNFV, including end-to-end delay as a main constraint during the allocation process, in addition to keeping migrations as always active all the time. Delay results are shown in Fig.5.13, presneting the impacts on the overall performance of oPaNFV when less sensitive to delay SFCs were allocated compared to those having

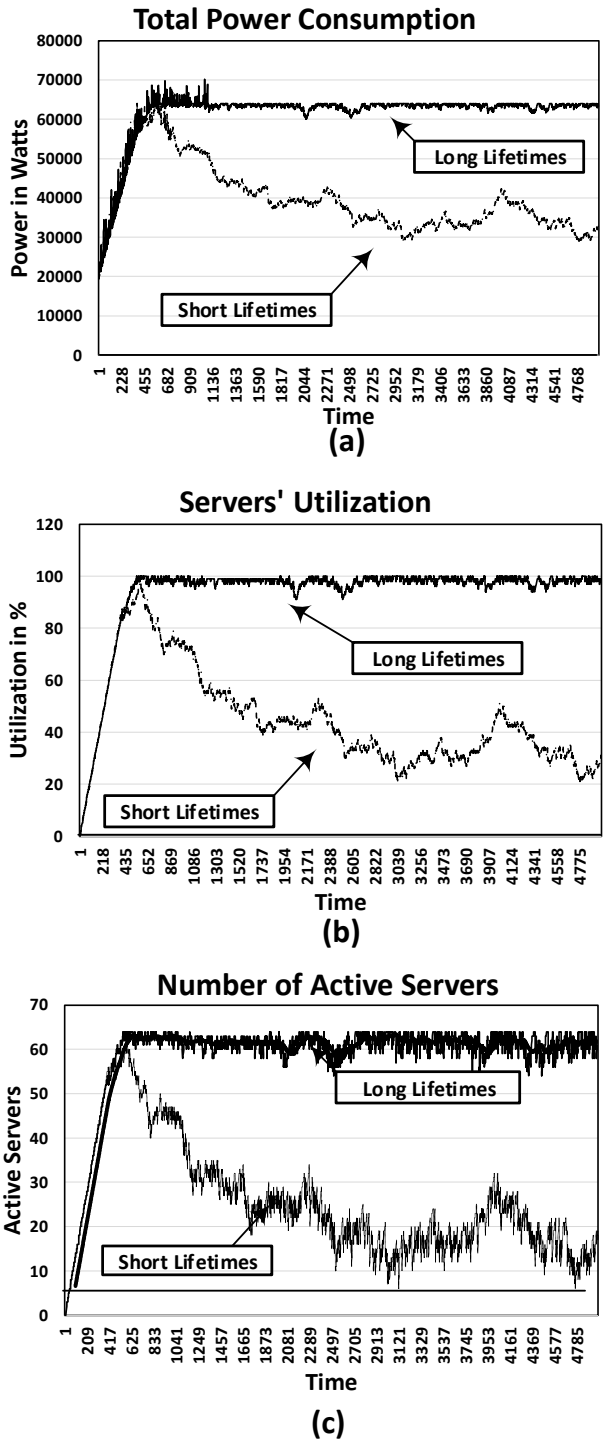

Figure 5.12: oPaNFV performance varying lifetimes.

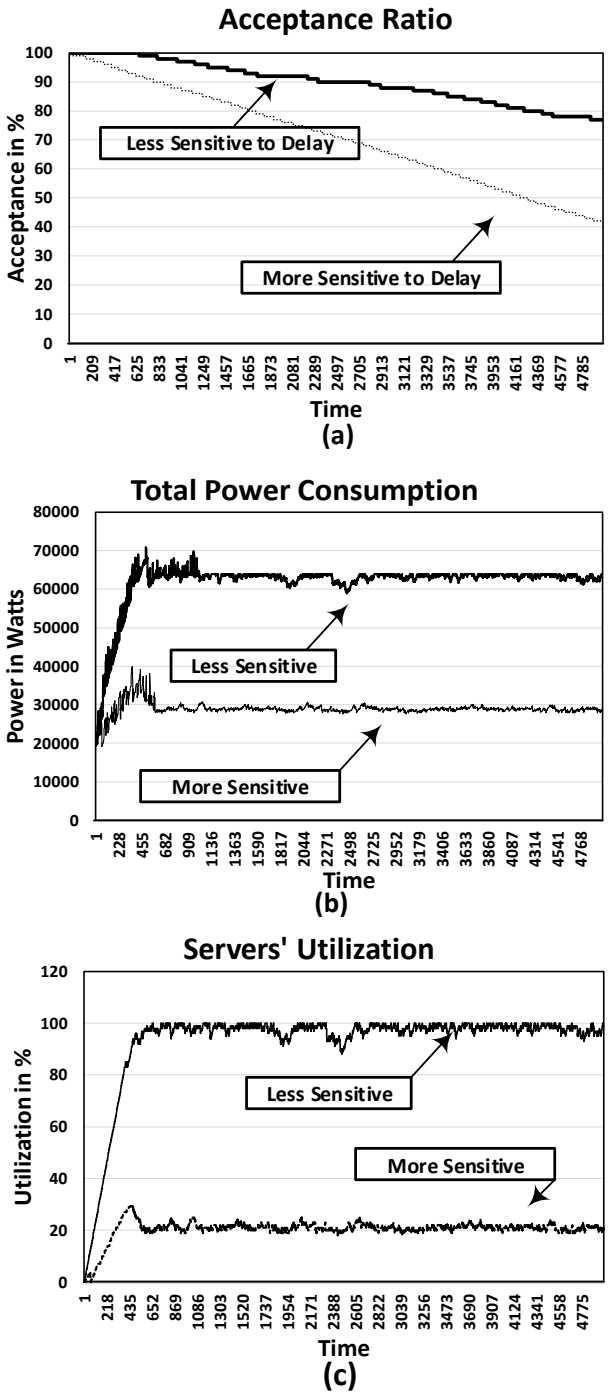

Figure 5.13: oPaNFV performance varying delay.

more strict and aggressive delay settings.

As a major evaluation metric about the impacts of delay on the resource allocation problem using oPaNFV, Fig.5.13a shows that the acceptance ratios were on the averages of 89.8% for the less sensitive to delay applications, compared to a reduced acceptance ratios in averages of 70.47% for the sensitive to delay SFCs, consequently, notice that in Fig.5.13b the total power consumption when sensitive to delay SFCs were allocated resulted on 53% lower consumptions than the less sensitive to delay case, mainly because of the less utilized resources, and less accepted demands. Moreover, Fig.5.13c show impacts on servers' utilizations when the less and the more sensitive to delay SFCs were allocated through oPaNFV, which recorded average values of 93.21% and 20.51%. The low values in the aggressive delay conditions were mainly due to the low numbers of the allocated SFCs.

Overall, results from Exp-4 underline the importance of including delay in the allocation process, also clearly point-out that, considering delay will have some serious consequences on the performance of the physical network, basically due to the physical characteristics and limited number of sufficient paths with accepted delay.

However, for 5G applications, still these delay results are not accepted and need to be enhanced big time to reach the minimum acceptance ratios hopefully above $98 - 99\%$. Moreover, the lower acceptance ratios in the aggressive scenario reflect that, much more additional enhancements are needed to support such sensitive to delay applications in the physical network layer to live up to 5G delay requirements.

Unfortunately, these are extremely hard problems to solve from the physical network point view. Nevertheless, using smart allocation algorithms such as oPaNFV, combined with edge computing, small mobile cells' designs, in addition to some advanced features, could be promising to enhance these acceptance ratios.

oPaNFV allocation times

The allocations' times shown in Fig.5.14 are oPaNFV consumed time in ms to embed each SFC, when the settings of first experiment (Exp-1) were applied. oPaNFV took on average 60.5519 ms to allocate a single SFC, giving that, the code was developed using Eclipse IDE for Java Developers, version Mars.2 Release (4.5.2). For all conducted experiments in this chapter, the used machine had an i7 processor, 5820K, 12 cores, 3.30 GHz, 16 GB RAM, and the operating system was Ubuntu 16.04.3 LTS.

5.6 Conclusions

This chapter proposed a modified power aware resource allocation and virtualization algorithm for NFV based networks. The new allocation algorithm solves the resource allocation problem in fractions of a second, supports physical servers 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 the state-of-art algorithm 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, this chapter proposed another optional feature in the new algorithm that can support virtual network functions consolidations together with physical servers consolidations to further reduce the operational and power consumption costs. Accordingly, simulation results of the total costs and servers' utilization when VNFs consolidations was allowed were 27.88% and 38.48%

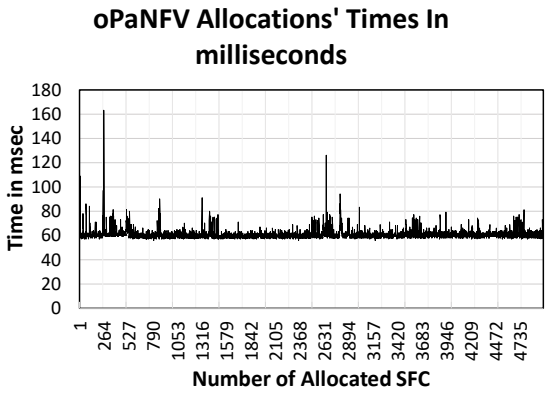

Figure 5.14: oPaNFV Embedding time in ms.

better than not consolidating the VNFs. Finally, to evaluate the resource allocation times, results from this chapter 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 chapter concluded that migrations would be better used for emergency or maintenance purposes.

CHAPTER 6

Findings and Future Research

This chapter will present the main findings from the proposed algorithms in this thesis, followed by some suggestions for future research that can leverage what was developed in the thesis. The findings summarized the benefits of implementing the segmentation technique to solve the power aware resource allocation problem. The chapter lists the impacts of segmentation on virtual network embedding and network function virtualization frameworks, for offline and online scenarios, as well as presenting the impacts of migration feature. Regarding future suggestions, the chapter summarizes some important additional research that can enhance the performance and applicability of the proposed algorithms, and then it provides some suggestions about utilizing machine learning algorithms combined with the segmentation technique, and apply them for better resource allocations and power minimization objectives.

6.1 Findings about VNE algorithms

6.1.1 Segmentation for efficient resource allocation

The segmentation technique was applied in the simulations in chapters 3 for offline VNE, chapter 4 for online VNE, and chapter 5 for offline, online, and migrations in NFV. The results confirmed the effectiveness of the proposed segmentation approach, which allowed constructing the best possible and direct paths that could be used by the proposed offline and online algorithms in fractions of a seconds, mostly in the range of $20 - 60$ ms. Accordingly, simulations on both VNE and NFV frameworks resulted on very solid allocations and efficient use of the resources from the early times of the allocation process. Consequently, that was translated into higher acceptance ratios in averages of 95%, faster allocations in fractions of second, as well as it provided lower power consumptions, negligible migration costs and blocking ratios overall.

6.1.2 Offline VNE algorithm, PaCoVNE

Simulation results in Fig.3.5 showed that PaCoVNE performed similar or better than other algorithms in terms of acceptance ratio for lower loads, and is much better for higher loads, thanks to the one stage, full coordinated embedding and segment formulation of PaCoVNE, which makes sure to allocate virtual nodes and their associated edges together, and at the same time to increase the acceptance ratio. In terms of power consumption at physical network nodes, PaCoVNE power consumption model included idle power in addition to power consumption when physical network nodes were loaded according to their CPU utilization, to ensure considering idle power as a main component in the power consumption of physical network's nodes, even if

they do not process any data. In the case of including end-to-end delay, simulations in Fig.3.5 showed that the acceptance ratio was degraded in average for all loads, increased power consumption, and reduced saved power. These results implies the significance of including end-to-end delay as a main constraint to embed VNRs, and how negatively it would impact the whole VNE process.

6.1.3 Online VNE algorithm, oPaCoVNE

Acceptance ratio and costs

The average acceptance ratio of oPaCoVNE as shown in Fig.4.2, were close and comparable to other algorithms in general. This is due to applying the segmentation technique for online scenario, which ensures that the virtual nodes and virtual edges are mapped together on the physical network path that complies with the location constraints, and that has enough resources and acceptable end-to-end delay without any hidden nodes or edges. Most important is the application of oPaCoVNE for higher loads and longer allocation times, where it had better acceptance ratios than other algorithms, most likely since it would always stick to using less edges during the embedding process, therefore, keeping more free resources to be used to fulfill the demands of the new virtualization requests. Regarding the average cost of oPaCoVNE, the results shown in Fig.4.2 confirm the successful strategy of oPaCoVNE in minimizing the use of physical network resources as least as possible while embedding VNRs, by selecting direct edges between the hosting physical nodes without any hidden hopes.

Nodes and edges utilization

Physical network nodes' average utilization in oPaCoVNE were in average moderately utilized and so close to other algorithms, but the average edges utilization in oPaCoVNE were much lower than others as shown in Fig.4.2d. This is mainly due to the tendency of oPaCoVNE ro allocate precisely the same number of physical network edges as the demanded edges without any hidden edges or nodes, and because it will alway try to utilize the active physical network nodes before activating new nodes.

Power consumption, revenues, and number of active nodes

As shown in Fig.4.3, the average power consumption per physical network node was lower than other algorithms, confirming the better power aware performance of oPaCoVNE, which uses least physical network resources as much as possible, through using direct edges with no hidden hopes at all, and this is confirmed referring to the number of turned off physical nodes by oPaCoVNE. Regarding the average revenues, oPaCoVNE resulted on almost very near values to those from the other algorithms, noting that oPaCoVNE way in embedding each virtual edge on a single physical edge may cause some VNRs to be rejected, which may slightly reduce the overall revenues of oPaCoVNE in general.

Impact of delay

All simulation results showed that impact of delay on VNE process was clearly the most significant parameter among all varied variables while testing oPaCoVNE. Specifically, referring to Fig.4.4-Fig.4.7, oPaCoVNE's

average acceptance ratios with delay, were less than when it was not included, and similar trends can be seen by referring to oPaCoVNE's results for average power consumption and processing time. The delay results confirm the importance of including end-to-end delay as a major constraint when solving VNE problem, as a direct evaluation metric for real world 5G networks.

Average processing time of oPaCoVNE

The average processing time results for oPaCoVNE are shown in Fig.4.4c-Fig.4.7c, which in general varied between 25.60 ms when varying the number of arriving VNRs, to 37.96 ms when varying VNRs' lifetimes. But the general behavior of average processing time had an increasing trend when the size of VNRs or physical network topologies were changed, where the processing time trend for smaller VNRs varied between $1.42 - 7.49$ ms, while when using larger physical networks, the average processing time varied considerably between $15.59 - 238.23$ ms depending on the number of edges in the network. That is the higher the number of edges, the more the processing time.

6.2 Findings about NFV algorithms

6.2.1 Offline NFV algorithms, PaNFV and mPaNFV

Total costs and migrations

The results of PaNFV costs as shown in Fig.5.3 were much lower than others on average, mainly because PaNFV carefully allocates the virtual functions and edges together using the segmentation technique, on the path of the most active (i.e. most 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. Referring to the migration costs, mPaNFV had almost negligible migration costs, mainly because of the precise allocations by PaNFV that consumed as least resources as possible in the first place, which allowed the migration algorithm to decide for very negligible migration attempts.

VNFs consolidations and migrations

Referring to Fig.5.7, when migrations were kept always active, the VNFs' consolidated mode provided lower average total costs, and higher server's utilizations. However, referring to Fig.7, 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.

6.2.2 Online NFV algorithm, oPaNFV

Overall oPaNFV performance and migrations

The performance of the online algorithm oPaNFV on a long run simulation times is stable and very smooth, but greatly depends on the lifetimes of the arriving SFCs as shown in Fig.5.10. However, giving the careful allocation strategy of oPaNFV, migrations in online are mostly very minor or negligible, almost very few migrations at the early stage of the experiment. Therefore, it would be more efficient to use the migrations for maintenance or emergency cases, than to minimize the total power consumption in the physical network. This will further reduce the complexity in oPaNFV and make it even much faster.

SFCs lifetimes

The overall conclusion from Fig.5.12 is that the longer the lifetimes of the allocated SFCs, the more the power consumption, and the higher the servers and edges' utilizations on the physical network. Most noticeable, is that even for a slight variation in the number of expiring SFCs, the impacts on the physical network metrics were immense and so obvious. Suggesting that, among all other physical network's parameters, SFCs' lifetime would most likely be the dominating factor on the overall performance of the physical network.

End-to-end delay

Overall, results from Fig.5.13 underline the importance of including delay in the allocation process, also clearly point-out that, considering delay will have some serious degradations in the performance of allocation algorithm on the physical network, basically due to the physical characteristics and limited number of sufficient paths with accepted delay. However, for 5G applications, still these delay results are not accepted and need to be enhanced big time to reach the minimum acceptance ratios.

oPaNFV allocation times

The online algorithm oPaNFV took on average fractions of ms in the range of $20 - 60$ ms to allocate a single SFC. However, when migrations are included the allocations times were significantly longer than without migrations, even though migrations did not result on any significant improvements in minimizing the total costs or power consumptions.

6.3 Future work - Enhancements

The following paragraphs provide some suggestions to modify the segmentation technique and the proposed algorithms in this thesis. The suggestions cover additional areas where the segmentation can be developed without using artificial intelligence techniques.

6.3.1 Future segmentation

The basic assumption in the segmentation is based on having a stable physical networks of fixed pairs in terms of connectivity, as explained in chapter 3 for VNEs, and in chapter 5 for NFVs. Segmentation in this sense is best fit for fixed core network nodes (i.e. distributed datacenters, servers, ect.), and fixed access nodes (i.e. edge datacenters, macro and micro base stations, transmission nodes, and repeater nodes). However, giving that future Internet using 5G for IoTs and vehicular terminals is going to be largely dynamic and mobile in nature, therefore, in future work, another version of the segmentation methodology supporting resource allocation in both the core network and the more dynamic access network could be developed as well.

6.3.2 Topology

Majority of literature that dealt with the resource allocation problem, assumed that the NFV architecture uses fat-tree datacenters, which are typically connected with special configuration of routers and switches. However, to generalize the simulation results of the used algorithms in this thesis it would be helpful to try other topology types and configurations, and test other evaluation metrics as well. For that, the work in chapter 4 was designed to handle any random physical network topology, and allocates any random virtual network request topology. Therefore, a future research can modify the algorithms from chapter 4 and make then work for any datacenter topology that uses NFV frameworks too.

6.3.3 Delay and 5G

In this thesis end-to-end delay was defined based on the recommendations form [ITU-T Focus Group (2017)] and [5GAericas (2018)], which is related to enhanced mobile broadband and the ultra reliable and low latency types of applications. However, giving that future networks are going to expand in using fiber optics in their topologies, then most likely they will suffer from the propagation delay than all other types of delays, namely processing, queuing, and transmission. 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.

6.3.4 Solving resource allocation for multiple paths

The allocation algorithms in this thesis used to allocate the virtual links on one single physical paths only. However, using multiple path mapping, where each virtual link demand can be carried by several paths in the physical network should be considered for further research. In this case the problem can be reduced to the Multi-commodity Flow Problem (MCF) that provides a multi-path routing solution for each virtual link using optimal LP algorithm [Kolliopoulos (1997)] [Ilhem (2012)]. In future research the suggested segmentation technique can be used to provide precise coordination between allocating the virtual nodes and the multiple paths hosting the virtual link.

6.3.5 Power aware network slicing

Network slicing is a key concept in 5G, which offers an independent connection of virtual resources and functions that are implemented and managed by software, and it allows the coexistence of multiple logically isolated network partitions over the same infrastructure [Wang (2018)] [Xiao (2018)] [Afolabi (2018)]. In future research, additional work can be conducted using the segmentation technique to create network slices, and allocate for them the required resources on top of the physical infrastructure in a manner that minimizes the total power consumption in the network. In addition, the future work should also focus on the real-world requirements for 5G of low latency and high transmission speeds, and be extended to support end-to-end resource allocation for services, as well as considering the composition of multiple slices over wireless and wired parts of the paths.

6.3.6 Power aware mobile edge computing

In 5G, mobile edge computing and small cells are key components to achieve the high throughputs and low latency, and they will be massively deployed. Consequently, utilizing their resources efficiently while preserving the total power consumption in the whole network is going to be a very challenging task. To minimize the power consumption in the network that use mobile edge computing technology, future research is required to apply the segmentation technique and to modify the suggested algorithms in this thesis to choose which cells to put into sleeping mode, while ensuring service maintainability, coverage, and good signal quality and reliability for the cell's users that could be migrated to other active cells.

In this scenario, more metrics need to be considered such as QoS classes for the different applications, detailed information about the edge network configuration, including information about the type of energy saving mode that, can be used depending on the QoS class for the different applications, as well as information about the current resource allocations and utilizations at the edge cloud, in addition to efficient and quick mobile edge topology discovery, path configuration, handover and other scalability challenges. Moreover, the algorithm should have predictive analysis capabilities using the historical profiles of the edge cloud to allow for efficient resource consolidations, and help choosing the best resources for the new allocations that will minimize total power consumptions and preserve service quality at the edge cloud [Bo Yi (2018)][Zhang (2018)][Lyu (2018)].

6.3.7 Application aware segmentation

Future research could be conducted to identify cloud applications' common behaviors, patterns, and explore load forecasting approaches that can potentially lead to more efficient resource provisioning and consequently higher energy efficiency. Furthermore, it is necessary to develop resource allocation consolidation algorithms that will use the information about the historical workload patterns and application behavior to select which applications will share physical resources, in order to minimize the overlapping of the resource usage by applications, and thus their influence on each other, as well as reducing the amount of migration needed when their demand for resources changes.

6.3.8 Segmentation for very large scale networks

A potential future research direction could be a development for the distributed online resource allocation and virtualization algorithms using the segmentation technique, as proposed in this thesis, taking into account handling huge traffic from large amounts of users in massive networks that include cloud and edge computing technologies. It would be interesting to consider other objective functions rather than the power minimization target which relies on servers utilizations, and use other objectives and constraints such as, targeting revenue maximization or cost minimizations, aided by constraints for power capping, faster migrations, allocations on multiple paths, utilizing edge datacenters, in addition to delay, processing power, and bandwidth.

6.4 Future work - Machine learning for 5G networks

Future 5G networks need to be fast and perform dynamic resource allocation to satisfy the demand of massive connections, ultra-low latency and ultra-high high reliability and capacity. Artificial intelligence learning methods that can adapt to the dynamic changes in the 5G networks and its environment, and exploit the past experience to improve the future performance of the whole system, could be helpful for future 5G systems [Latah 2019] [Xie 2019] [Pacheco 2018] [Cheng 2018] [Chapaneri 2019] [Liu 2018].

Generally, machine learning approaches have been widely used for solving various networking problems such as routing [Tang 2019], traffic classification [Saeed 2019], flow prediction [Baz 2018] intrusion detection [Sultana 2019], load balancing [Kim H 2017], fault detection [Park 2018], quality of service and quality of experience optimization [Qin 2018], and admission control and resource allocation [Wang 2018].

A comprehensive surveys for the applications of artificial intelligence and machine learning techniques in many networking problems were summarized in [Latah 2019] [Xie 2019] [Pacheco 2018] [Cheng 2018] [Chapaneri 2019] [Liu 2018]. The following paragraphs pinpoint some of the research suggestion where the works from this thesis, the segmentation for resource allocations and power aware networks, could be modified using some algorithms from the machine learning areas.

6.4.1 Machine learning for traffic prediction

Machine learning algorithms can be applied for traffic classification and prediction to better perform network management, and to solve the resource allocation problem for dynamic environments. Future research can be done utilizing the proposed segmentation technique in this thesis and deep neural networks specifically, and maybe other machine learning algorithms for traffic classifications and predictions, which will help the 5G network controllers to more precisely allocate network resources, and efficiently manage the power consumption in the network. Moreover, traffic predictions through proper machine learning algorithms in 5G networks can be used to perform sufficient resource allocations that avoid traffic congestions, improve management of QoS, as well as tackling the performance bottleneck and overload problems [Farshin 2019] [Changhe 2018] [Alawe 2018].

6.4.2 Machine learning for power efficiency

Future research that utilizes the segmentation technique, could use the deep Neural networks as a classifying tools, to identify which datacenters could be switched ON or OFF according to their internal traffic. The resulting research may rely on predicting the demands, so that each datacenter or cluster of them attempt to find the best suitable configuration that will reduce the power of its own group of datacenters. Moreover, other supporting machine learning algorithms can also be used to predict the minimum information about what other datacenters are doing to lead for better configuration for the active cluster of datacenters that will be kept active, and turning others off [Nasroollah 2019] [Meshkati 2019] [Aslam 2019].

6.4.3 Machine Learning for optimal paths

Routing optimization is a fundamental problem in network function virtualization and resource allocation paradigms. In 5G networks, the controller can control the routing of traffic flows by modifying flow tables in switches. For example, the controller can guide switches to discard a traffic flow or route it through a specific path. Inefficient routing decisions can lead to the overloading of network links and increase the end-to-end transmission delay, which will affect the total power consumption and the overall performance of the whole network.

Thus, constructing the best paths and optimizing the routing of traffic flows could be an important future research problem, which may leverage the use of the proposed segmentation technique in this thesis, and apply some machine learning algorithms, such as ant colony, reinforcement learning, or deep neural network. The application of machine learning to predict the shortest paths utilizing the segmentation technique will be a valuable added-on feature for the proposed algorithms in this thesis [Choudhury G 2018] [Sendra 2017] [Mao 2017] [Junfeng 2018] [Haider 2018] [Guo 2018] [Huamin 2017] [Eshaghnezhad 2018].

6.4.4 Machine learning for load balancing

The dramatic increase in data traffic and the demand for high data rates services that are bursty and random in 5G networks is a big challenge facing resources allocations in datacenters management. For example detecting traffic congestions in servers and links, measuring paths bottlenecks, dealing with servers and links failures, migrating traffic between datacenters in emergency conditions, handling ultra low latency and very sensitive services for end-to-end delays, and managing the power consumption of highly utilized servers, all of these can impose a significant computational overheads on datacenters, until they redistribute their loads in a cost effective manner, and on real time bases in fractions of microseconds to comply with 5G requirements.

Accordingly, a possible future research can be conducted using the segmentation technique from this thesis, combined with machine learning to design a new algorithm that considers load balancing mechanisms to distribute and classify the datacenters traffic, in a manner that increases overall network capacity, resources utilization and user throughput, while providing faster response time, lower costs, high network reliability and scalability, and efficient power consumption. Some examples for machine learning algorithms that are applicable to load balancing in datacenters such as neural networks, ant colony, genetic algorithms, and reinforcements learning algorithms, can be combined inside the proposed algorithms from this thesis to provide a centralized dynamic load balancing mechanism that considers servers and links utilizations, end-to-end delay, and total power consumption in the whole network [Gomez 2018] [Roshdy 2018] [Xu P 2018]

[Kanthimathi 2018] [Stefano 2015].

6.4.5 Machine learning for resource allocation

Efficient network resource management is one of the main contributions by the algorithms from this thesis, focusing on core networks' computing and networking resources. However, in 5G systems computing resources have been deployed closer to end users using edge computing technologies to overcome excessive end-to-end delays. Therefore, resources allocation problem is going to be much complex, specifically for online conditions, and because they will require more computational times, to better classify the best recourses and paths when allocating the network service that are sensitive to delay. Also in migration scenarios, lower migration time is a crucial key performance indicator about the efficiency of preforming the resource allocations for migrating services for real time scenario.

Consequently, another possible future research can deploy some machine learning techniques to predict and classify the available resources' capacities and paths that can be used to allocate the network services much efficiently. For example the classification features of neural networks and ant colony algorithms, and the optimization features of genetic and fuzzy algorithms, can be integrated inside the proposed algorithms by this thesis to enhance the segmentation technique when formulating the best paths to solve the resource allocation problem [Wang 2018] [Bong 2018] [Gudu 2018] [Murali 2018] [Azari 2019].

6.4.6 Machine learning for quality of service

Quality of service parameters such as servers and links' utilizations, packet loss rate, end-to-end delay, jitter and throughput, are network oriented metrics, which are usually used to manage and guide the resource allocation process, and to assess the overall network performance. Therefore, in order to offer better quality aware resource allocations, the proposed algorithms in this theses can deploy machine learning for traffic prediction and classifications purposes, and a modified version of the online algorithms can be developed using one or a combination of machine learning techniques such as neural networks, ant colony optimization, genetic algorithms, reinforcement learning, or decision trees and linear regression for quality based resource allocations. [Qin (2019)] [Amiri 2018] [Rahman 2018].

The modified algorithm can for example predict traffic congestions and other problems in advance, and correct errors during the resource allocations process, or identify which kind of action should be taken to guarantee the quality of service according to the service level agreement. Moreover, the modified algorithm can classify data traffic per server or link or per the datacenter as whole, and accordingly segregate network services' traffic flows into quality of service classes, or compress their flow entries with certain quality of service guarantees for more efficient management of the resource allocation process [Changhe 2018] [Luong-Vy 2018] [Li F (2017)] [Pasca (2017)].

References

References

[ITU-T T-REC-Y.3150 (2018)] T-REC-Y.3150, "Y.3150: High level technical characteristic of network softwarization for IMT-2020," https://www.itu.int/rec/T-REC-Y.3150/en.

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

[5G (2017a)] 5G PPP Architecture Working Group, "View on 5G Architecture," Version 2.0, 2017. https://5g-ppp.eu/, accessed on April-2019.

[5G (2017b)] 5G PPP White Paper: Software Networks WG, "View on 5G Vision on Software Networks and 5G," physical network WG, January 2017, final version 2.0. https://5g-ppp.eu/.

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

[Abhishek (2018)] R. Abhishek, D. Tipper and D. Medhi, "Network Virtualization and Survivability of 5G Networks: Framework, Optimization Model, and Performance," 2018 IEEE Globecom Workshops (GC Wkshps), Abu Dhabi, United Arab Emirates, pp. 1-6, 2018. DOI: 10.1109/GLOCOMW.2018.8644092

[Adamuz-Hinojosa (2018)] O. Adamuz-Hinojosa, J. Ordonez-Lucena, P. Ameigeiras, J. J. Ramos-Munoz, D. Lopez and J. Folgueira, "Automated Network Service Scaling in NFV: Concepts, Mechanisms and Scaling Workflow," in IEEE Communications Magazine, vol. 56, no. 7, pp. 162-169, 2018. DOI: 10.1109/MCOM.2018.1701336

[Addya (2018)] S. K. Addya, A. K. Turuk, A. Satpathy, B. Sahoo and M. Sarkar, "A Strategy for Live Migration of Virtual Machines in a Cloud Federation," in IEEE Systems Journal, pp. 1-11, 2018. DOI: 10.1109/JSYST.2018.2872580

[Agarwal (2019)] S. Agarwal, F. Malandrino, C. F. Chiasserini and S. De, "VNF Placement and Resource Allocation for the Support of Vertical Services in 5G Networks," in IEEE/ACM Transactions on Networking, vol. 27, no. 1, pp. 433-446, 2019. DOI: 10.1109/TNET.2018.2890631

[Akaki (2018)] Akaki Jobava, Anis Yazidi, B. John Oommen, Kyrre Begnum, "On achieving intelligent traffic-aware consolidation of virtual machines in a data center using Learning Automata," Journal of Computational Science,Volume 24, Pages 290-312, 2018. DOI:10.1016/j.jocs.2017.08.005

[Alawe 2018] I. Alawe, A. Ksentini, Y. Hadjadj-Aoul and P. Bertin, "Improving Traffic Forecasting for 5G Core Network Scalability: A Machine Learning Approach," in IEEE Network, vol. 32, no. 6, pp. 42-49, November/December 2018. DOI: 10.1109/MNET.2018.1800104

[Alsadie (2018)] D. Alsadie, E. J. Alzahrani, N. Sohrabi, Z. Tari and A. Y. Zomaya, "DTFA: A Dynamic Threshold-Based Fuzzy Approach for Power-Efficient virtual machine Consolidation," 2018 IEEE 17th International Symposium on Network Computing and Applications (NCA), Cambridge, MA, pp. 1-9, 2018. DOI: 10.1109/NCA.2018.8548162

[Amiri 2018] R. Amiri, H. Mehrpouyan, L. Fridman, R. K. Mallik, A. Nallanathan and D. Matolak, "A Machine Learning Approach for Power Allocation in HetNets Considering QoS," 2018 IEEE International Conference on Communications (ICC), Kansas City, MO, 2018, pp. 1-7. DOI: 10.1109/ICC.2018.8422864

[Aryania (2018)] A. Aryania, H. S. Aghdasi, L. M. Khanli, "Energy-Aware Virtual Machine Consolidation Algorithm Based on Ant Colony System," Journal of Grid Computing, September , Volume 16, Issue 3, pp. 477-491, 2018. DOI:10.1007/s10723-018-9428-4

[Aslam 2019] Aslam A.M., Kalra M. (2019) "Using Artificial Neural Network for VM Consolidation Approach to Enhance Energy Efficiency in Green Cloud," In: Kolhe M., Trivedi M., Tiwari S., Singh V. (eds) Advances in Data and Information Sciences. Lecture Notes in Networks and Systems, vol 39. Springer, Singapore. DOI: 10.1007/978-981-13-0277-0

[Azari 2019] A. Azari, M. Ozger and C. Cavdar, "Risk-Aware Resource Allocation for URLLC: Challenges and Strategies with Machine Learning," in IEEE Communications Magazine, vol. 57, no. 3, pp. 42-48, March 2019. DOI: 10.1109/MCOM.2019.1800610

[Afolabi (2018)] I. Afolabi, T. Taleb, K. Samdanis, A. Ksentini and H. Flinck, "Network Slicing and Softwarization: A Survey on Principles, Enabling Technologies, and Solutions," in IEEE Communications Surveys and Tutorials, vol. 20, no. 3, pp. 2429-2453, thirdquarter 2018. DOI: 10.1109/COMST.2018.2815638.

[Bari (2016)] 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

[Bari (2016)] 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

[Baz 2018] A. Baz, "Bayesian Machine Learning Algorithm for Flow Prediction in SDN Switches," 2018 1st International Conference on Computer Applications and Information Security (ICCAIS), Riyadh, 2018, pp. 1-7. DOI: 10.1109/CAIS.2018.8441969

[Bhamare (2017)] Bhamare, Mohammed Samaka, Aiman Erbad, Raj Jain, Lav Gupta, H. Anthony Chan, 2017. "Optimal virtual network function placement in multi-cloud service function chaining architecture," Computer Communications, Volume 102, 2017, Pages 1-16. DOI:10.1016/j.comcom.2017.02.011

[Bianchi (2017)] F. Bianchi, F. Presti, 2017. "A Markov Reward based Resource-Latency Aware Algorithm for the Virtual Network Embedding Problem," SIGMETRICS Perform. Eval. Rev. 44, no. 4 pp. 57-68. DOI:10.1145/3092819.3092827.

[Bo Yi (2018)] Bo Yi, Xingwei Wang, Keqin Li, Sajal k. Das, Min Huang, "A comprehensive survey of Network Function Virtualization," Computer Networks, Volume 133, 2018, Pages 212-262. DOI: 10.1016/j.comnet.2018.01.021..

[Bong 2018] Bong Jun Ko, Kin K. Leung, Theodoros Salonidis, "Machine learning for dynamic resource allocation at network edge," Proc. SPIE 10635, Ground/Air Multisensor Interoperability, Integration, and Networking for Persistent ISR IX, 106350J (4 May 2018). DOI: 10.1117/12.2306095

[Bordel (2018)] B. Bordel, S. de Rivera, R. Alcarria, A. Rocha, H. Adeli, P. Reis Luís, S. Costanzo, "Virtualization-Based Techniques for the Design, Management and Implementation of Future 5G Systems with Network Slicing," Trends and Advances in Information Systems and Technologies, Springer. Pp. 133-143, 2018. DOI: 10.1007/978-3-319-77712-2

[Botero (2012a)] J. Botero, X. Hesselbach, A. Fischer, H. Meer, 2012a. "Optimal mapping of virtual networks with hidden hops," Telecommunication Systems, vol. 51, no. 4, pp. 273-282. DOI: 10.1007/s11235-011-9437-0.

[Botero (2012b)] J. Botero, X. Hesselbach, M. Duelli, D. Schlosser, A. Fischer, and H. de Meer, 2012b. "Energy efficient virtual network embedding," Communications Letters, IEEE, vol. 16, no. 5, pp. 756-759. DOI: 10.1109/LCOMM.2012.030912.120082.

[Botero (2013a)] J. Botero, M. Molina, X. Hesselbach, J. Amazonas, 2013a. "A novel paths algebra-based strategy to flexibly solve the link mapping stage of VNE problems," Journal of Network and Computer Applications, vol. 36. no. 6, pp. 1735-1752. DOI: 10.1016/j.jnca.2013.02.029.

[Botero (2013b)] J. Botero, X. Hesselbach, 2013b. "Greener networking in a network virtualization environment," Computer Networks, vol 57, issue 9, pp. 20121-2039. DOI: 10.1016/j.comnet.2013.04.004.

[Bradley (1977)] Bradley, Hax and Magnanti, 1977. "Applied Mathematical Programming," Addison-Wesley.

[Budzisz (2014)] L. Budzisz et al., "Dynamic resource provisioning for energy efficiency in wireless access networks: A survey and an outlook," IEEE Communications Surveys and Tutorials, vol. 16, no. 4, 2014 , pp. 2259-2285 DOI: 10.1109/COMST.2014.2329505.

[Cao (2019)] G. Cao, "Topology-aware multi-objective virtual machine dynamic consolidation for cloud datacenter," Journal of Sustainable Computing: Informatics and Systems, Volume 21, pp. 179-188, 2019. DOI: 10.1016/j.suscom.2019.01.004

[Changhe 2018] Changhe Yu, Julong Lan, JiChao Xie, Yuxiang Hu, "QoS-aware Traffic Classification Architecture Using Machine Learning and Deep Packet Inspection in SDNs," Procedia Computer Science, Volume 131, 2018, Pages 1209-1216. DOI: 10.1016/j.procs.2018.04.331

[Changhe 2018] Changhe Yu, Julong Lan, JiChao Xie, Yuxiang Hu, "QoS-aware Traffic Classification Architecture Using Machine Learning and Deep Packet Inspection in SDNs," Procedia Computer Science, Volume 131, 2018, Pages 1209-1216. DOI: 10.1016/j.procs.2018.04.331

[Chapaneri 2019] Chapaneri R., Shah S. "A Comprehensive Survey of Machine Learning-Based Network Intrusion Detection," In: Satapathy S., Bhateja V., Das S. (eds) Smart Intelligent Computing and Applications. Smart Innovation, Systems and Technologies, vol 104. Springer, Singapore, pp 345-356, 2019. DOI: 10.1007/978-981-13-1921-1

[Chen (2016)] X. Chen, C. Li, and Y. Jiang, 2016. "A feedback control approach for energy efficient virtual network embedding," Computer Communications, vol. 80, pp. 16-32. DOI: 10.1016/j.comcom.2015.10.010.

[Cheng 2018] Cheng, Yang, Geng, Jinkun, Wang, Yanshu, Li, Junfeng, Li, Dan, Wu, Jianping, "Bridging machine learning and computer network research: a survey," CCF Journal of Transactions on Networking, pp 1-15, 2018. DOI: 10.1007/s42045-018-0009-7

[Choudhury G 2018] Choudhury, G., Lynch, D., Thakur, G., Tse, S., "Two Use Cases of Machine Learning for SDN-Enabled IP/Optical Networks: Traffic Matrix Prediction and Optical Path Performance Prediction," J. Opt. Commun. Netw., 2018, 10, D52-D62. DOI: 10.1364/JOCN.10.000D52

[Chowdhury (2012)] M. Chowdhury, M. Rahman and R. Boutaba, 2012. "ViNEYard: Virtual Network Embedding Algorithms With Coordinated Node and Link Mapping," in IEEE/ACM Transactions on Networking, vol. 20, no. 1, pp. 206-219. DOI: 10.1109/TNET.2011.2159308.

[Dayarathna (2016)] 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

[Deafallah (2018)] A. Deafallah, T. Zahir, A. Eidah J., A. Ahmed, H. Hakim, C. Wojciech, W. Hua, P. Hye-Young, Z. Rui, "LIFE-MP: Online Virtual Machine Consolidation with Multiple Resource Usages in Cloud Environments," Journal of Web Information Systems Engineering, WISE, Springer International Publishing, pp. 167-177, 2018. DOI: 10.1007/978-3-030-02925-8

[Eramo (2016)] V. Eramo, A. Tosti, and E. Miucci, "Server Resource Dimensioning and Routing of Service Function Chain in NFV Network Architectures," Journal of Electrical and Computer Engineering, vol. 2016, Article ID 7139852, 12 pages, 2016. DOI:10.1155/2016/7139852.

[Eramo (2017a)] V. Eramo, E. Miucci, M. Ammar and F. G. Lavacca, 2017. "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. 2008-2025. DOI:10.1109/TNET.2017.2668470

[Eramo (2017b)] V. Eramo, M. Ammar and F. G. Lavacca, "Migration Energy Aware Reconfigurations of Virtual Network Function Instances in NFV Architectures," in IEEE Access, vol. 5, no. , pp. 4927-4938, 2017. DOI: 10.1109/ACCESS.2017.2685437.

[Esfandiarpoor (2015)] Esfandiarpoor, Ali Pahlavan, Maziar Goudarzi, "Structure-aware online virtual machine consolidation for datacenter energy improvement in cloud computing," Computers and Electrical Engineering, Volume 42, 2015, Pages 74-89. DOI:10.1016/j.compeleceng.2014.09.005

[Eshaghnezhad 2018] Eshaghnezhad, M., Rahbarnia, F., Effati, S., Mansoori, A., "An Artificial Neural Network Model to Solve the Fuzzy Shortest Path Problem," Neural Processing Letters, 2018. DOI: 10.1007/s11063-018-9945-y

[ETSI (2012)] ETSI, "Network Functions Virtualisation, Introductory White Paper," 2012. etsi.org/nfv/.

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

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

[ETSI (2013c)] ETSI, "Network Function Virtualisation (NFV); Use Cases," GS NFV 001 v1.1.1, 2013. etsi.org/nfv/.

[ETSI (2014a)] ETSI, "Network Function Virtualisation (NFV); Terminology for Main Concepts in NFV," ETSI GS NFV 003 V1.2.1, 2014. etsi.org/nfv/.

[ETSI (2014b)] ETSI, "Network Function Virtualization (NFV); Service Quality Metrics," GS NFV-INF 010 v1.1.1, 2014. etsi.org/nfv/.

[ETSI (2016)] ETSI, "Pre-deployment Testing; Report on Validation of NFV Environments and Services," GS NFV-TST 001 v1.1.11, 2016. etsi.org/nfv/.

[ETSISDN (2017)] ETSI, "Report on SDN Usage in NFV Architectural Framework," ETSI GS NFV-EVE 005 V1.1.1, 2015. etsi.org/nfv/.

[Fan (2007)] 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

[Farshin 2019] Farshin, Alireza and Sharifian, Saeed "A modified knowledge-based ant colony algorithm for virtual machine placement and simultaneous routing of NFV in distributed cloud architecture,",The Journal of Supercomputing, 2019. DOI: 10.1007/s11227-019-02804-x

[Fendt (2018)] A. Fendt, C. Mannweiler, L. C. Schmelz and B. Bauer, "A Formal Optimization Model for 5G Mobile Network Slice Resource Allocation," 2018 IEEE 9th Annual Information Technology, Electronics and Mobile Communication Conference (IEMCON), Vancouver, BC, pp. 101-106, 2018. DOI: 10.1109/IEMCON.2018.8615049

[Fischer (2013)] A. Fischer, J. Botero, M. Beck, H. de Meer and X. Hesselbach, 2013. "Virtual Network Embedding: A Survey," in IEEE Communications Surveys and Tutorials, vol. 15, no. 4, pp. 1888-1906. DOI: 10.1109/SURV.2013.013013.00155.

[Ghazisaeedi (2017)] E. Ghazisaeedi and C. Huang, 2017. "Off-Peak Energy Optimization for Links in Virtualized Network Environment," in IEEE Transactions on Cloud Computing, vol. 5, no. 2, pp. 155-167. DOI:10.1109/TCC.2015.2440246.

[Ghaznavi (2015)] M. Ghaznavi, A. Khan, N. Shahriar, K. Alsubhi, R. Ahmed and R. Boutaba, "Elastic virtual network function placement," 2015 IEEE 4th International Conference on Cloud Networking (CloudNet), Niagara Falls, ON, 2015, pp. 255-260. DOI: 10.1109/CloudNet.2015.7335318.

[Gomez 2018] Gomez, Cesar A.; Shami, Abdallah; Wang, Xianbin. 2018. "Machine Learning Aided Scheme for Load Balancing in Dense IoT Networks." Sensors 18, no. 11: 3779. DOI: 10.3390/s18113779

[Gong (2016)] S., Chen, J., Kang, Q. et al, "An efficient and coordinated mapping algorithm in virtualized SDN networks," Frontiers Inf Technol Electronic Eng (2016) 17: 701. DOI:10.1631/FITEE.1500387.

[Gong (2016)] L. Gong, H. Jiang, Y. Wang and Z. Zhu, "Novel Location-Constrained Virtual Network Embedding LC-VNE Algorithms Towards Integrated Node and Link Mapping," in IEEE/ACM Transactions on Networking, vol. 24, no. 6, pp. 3648-3661, December 2016. DOI: 10.1109/TNET.2016.2533625.

[Google (2019)] Google, "Google Data Centers: Efficiency," https://www.google.com/.

[Gudu 2018] Gudu D., Hardt M., Streit A., "Combinatorial Auction Algorithm Selection for Cloud Resource Allocation Using Machine Learning," In: Aldinucci M., Padovani L., Torquati M. (eds) Euro-Par 2018: Parallel Processing. Euro-Par 2018. Lecture Notes in Computer Science, vol 11014. Springer, Cham DOI: 10.1007/978-3-319-96983-1

[Guerzoni (2015)] R. Guerzoni, I. Vaishnavi, A. Frimpong and R. Trivisonno, "Virtual Link Mapping for delay critical services in SDN-enabled 5G networks," Proceedings of the 2015 1st IEEE Conference on Network Softwarization (NetSoft), London, 2015, pp. 1-9. DOI: 10.1109/NETSOFT.2015.7116163.

[Guo (2018)] L. Guo, G. Hu, Y. Dong, Y. Luo and Y. Zhu, "A Game Based Consolidation Method of Virtual Machines in Cloud Data Centers With Energy and Load Constraints," in IEEE Access, vol. 6, pp. 4664-4676, 2018. DOI:10.1109/ACCESS.2017.2787735

[Guo 2018] J. Guo, P. Cao, J. Wu, B. Xu and J. Yang, "Path Delay Variation Prediction Model with Machine Learning," 2018 14th IEEE International Conference on Solid-State and Integrated Circuit Technology (ICSICT), Qingdao, 2018, pp. 1-3. DOI: 10.1109/ICSICT.2018.8565726

[Habiba (2018)] U. Habiba and E. Hossain, "Auction Mechanisms for Virtualization in 5G Cellular Networks: Basics, Trends, and Open Challenges," in IEEE Communications Surveys and Tutorials, vol. 20, no. 3, pp. 2264-2293, thirdquarter 2018. DOI: 10.1109/COMST.2018.2811395

[Haider 2018] Haider K Hoomod and Tuka Kareem, Jebur, "Applying self-organizing map and modified radial based neural network for clustering and routing optimal path in wireless network," Journal of Physics: Conference Series, Volume 1003, conference 1, 2018. DOI: 10.1088/1742-6596/1003/1/012040

[Halabian (2019)] H. Halabian, "Distributed Resource Allocation Optimization in 5G Virtualized Networks," in IEEE Journal on Selected Areas in Communications, vol. 37, no. 3, pp. 627-642, 2019. DOI: 10.1109/JSAC.2019.2894305

[Haneet (2019)] K. Haneet, J. Rakesh Kumar, J. Sanjeev, K. Preetam. "Protocol design and resource allocation for power optimization using spectrum sharing for 5G networks," Journal of Telecommunication Systems. Springer, pp. 1-19, 2019. DOI: 10.1007/s11235-019-00550-2.

[Heba (2019)] N. Heba, A. Nesma, R. Rawya, "Smart elastic scheduling algorithm for virtual machine migration in cloud computing," The Journal of Supercomputing, pp. 1-24, 2019. DOI: 10.1007/s11227-019-02748-2

[Hejja (2017)] Khaled Hejja, Xavier Hesselbach, "PaCoVNE: Power Consumption Aware Coordinated VNE with Delay Constraints." Polytechnic University of Valencia Congress, XIII Jornadas de Ingenieria Telematica - JITEL2017, 2017. DOI:10.4995/JITEL2017.2017.6490.

[Hejja (2018)] Khaled Hejja, Xavier Hesselbach, "Power aware coordinated virtual network embedding with 5G delay constraint," Journal of Network and Computer Applications, Elsevier, 2018. DOI:10.1016/j.jnca.2018.10.005.

[Hejja (2019)] Hejja, Xavier Hesselbach, "Offline and Online Power Aware Resource Allocation Algorithms with Migration and Delay Constraints," Journal of Computer Networks, Elsevier, 2019. DOI:10.1016/j.comnet.2019.04.030

[Herrera and Botero (2016)] 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

[Hesselbach (2016)] X.Hesselbach, J.R.Amazonas, S.Villanueva, J.F.Botero, "Coordinated node and link mapping VNE using a new paths algebra strategy," Journal of Network and Computer Applications, 2016. Available online at: DOI: 10.1016/j.jnca.2016.02.025.

[Heyang (2019)] X. Heyang, L. Yang, W. Wei, X. Ying, "Migration Cost and Energy-Aware Virtual Machine Consolidation Under Cloud Environments Considering Remaining Runtime," International Journal of Parallel Programming, 2019. DOI: 10.1007/s10766-018-00622-x

[Ho (2011)] Y. Ho, P. Liu and J. Wu, "Server Consolidation Algorithms with Bounded Migration Cost and Performance Guarantees in Cloud Computing," 2011 Fourth IEEE International Conference on Utility and Cloud Computing, Victoria, NSW, pp. 154-161, 2011. DOI: 10.1109/UCC.2011.30

[Hou (2016)] W. Hou, C. Yu, L. Guo, X. Wei, 2016. "Virtual network embedding for power savings of servers and switches in elastic data center networks," Science China Information Sciences, vol. 59, no. 12, pp. 122307:1-122307:14. DOI:10.1007/s11432-016-5590-0.

[Huamin 2017] Huamin Zhang, Hua Che, Di Chen, "Neural-network-based Path Planning Optimization," 2nd International Forum on Management, Education and Information Technology Application (IFMEITA 2017), Atlantis Press. DOI: 10.2991/ifmeita-17.2018.31

[Huaqing (2018)] H. Zhang, Y. Xiao, S. Bu, R. Yu, D. Niyato and Z. Han, "Distributed Resource Allocation for Data Center Networks: A Hierarchical Game Approach," in IEEE Transactions on Cloud Computing. DOI:10.1109/TCC.2018.2829744

[Ilhem (2012)] Ilhem Fajjari, "Resource Allocation Algorithms for Virtual networks within Cloud Backbone Network," PhD Thesis, Pierre et Marie Curie University, France, 2012. DOI: 10.13140/RG.2.1.4549.6487.

[Ilhem (2016)] Ilhem Fajjari, Nadjib Aitsaadi, Boutheina Dab, Guy Pujolle, "Novel adaptive virtual network embedding algorithm for Cloud's private backbone network," Computer Communications, Volume 84, 15 June 2016, Pages 12-24. DOI: 10.1016/j.comcom.2016.03.019.

[ITU (2011)] 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.

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

[ITUSDN3300 (2014)] ITU-T, "Framework of software-define networking", Recommendation ITU-T Y.3300. 2014. https://www.itu.int/rec/T-REC-Y.3300/en.

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

[ITU-T T-REC-Y.3101 (2018)] T-REC-Y.3101, "Y.3101: Requirements of the IMT-2020 network," https://www.itu.int/rec/T-REC-Y.3101/en.

[Jianxin (2014)] Jianxin Liao, Min Feng, Tonghong Li, Jingyu Wang and Sude Qing, "Topology-aware Virtual Network Embedding Using Multiple Characteristics," KSII Transactions on Internet and Information Systems, vol. 8, no. 1, pp. 145-164, 2014. DOI: 10.3837/tiis.2014.01.009.

[Joseph (2018)] C. T. Joseph, J. P. Martin, K. Chandrasekaran, A. Kandasamy, E. B. Rajsingh, J. Veerasamy, A. H. Alavi, J. D. Peter, "Virtual Machine Migration A Perspective Study," Advances in Big Data and Cloud Computing, Springer, pp. 79-89, 2018. DOI: 10.1007/978-981-10-7200-0

[Junfeng 2018] W. Junfeng, Z. Tao, Z. Hongming, "Fuzzy neural network model construction based on shortest path parallel algorithm," Cluster Computing, pp 1-6, 2018. DOI: 10.1007/s10586-018-2188-x

[Kanthimathi 2018] M. Kanthimathi and D. Vijayakumar, "An Enhanced Approach of Genetic and Ant colony based Load Balancing in Cloud Environment," 2018 International Conference on Soft-computing and Network Security (ICSNS), Coimbatore, 2018, pp. 1-5. DOI: 10.1109/ICSNS.2018.8573608

[Kar (2018)] 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

[Khan (2018)] 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

[Kim (2016)] S. Kim, Y. Han and S. Park, "An Energy-Aware Service Function Chaining and Reconfiguration Algorithm in NFV," 2016 IEEE 1st International Workshops on Foundations and Applications of Self Systems (FAS*W), Augsburg, 2016, pp. 54-59. DOI: 10.1109/FAS-W.2016.24.

[Kim (2017)] 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

[Kim H 2017] Kim, H.Y., Kim, J.M., "A load balancing scheme based on deep-learning in IoT," Cluster Computing, 2017,20,(1), pp. 873-878. DOI: 10.1007/s10586-016-0667-5

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

[Kolliopoulos (1997)] S. Kolliopoulos and C. Stein, 1997. "Improved approximation algorithms for unsplittable flow problems," Proceedings 38th Annual Symposium on Foundations of Computer Science, Miami Beach, FL, pp. 426-436. DOI: 10.1109/SFCS.1997.646131

[Latah 2019] Latah, Majd; Toker, Levent: 'Artificial intelligence enabled software-defined networking: a comprehensive overview', IET Networks, 2019, 8, (2), p. 79-99, IET Digital Library. DOI: 10.1049/iet-net.2018.5082

[Li (2019)] W. Li, Y. Zi, L. Feng, F. Zhou, P. Yu, X. Qiu, "Latency-Optimal Virtual Network Functions Resource Allocation for 5G Backhaul Transport Network Slicing," Applied Sciences, 9(4):701, 2019. DOI: 10.3390/app9040701.

[Li F (2017)] Li, F., Cao, J., Wang, X., Sun, Y., "A QoS Guaranteed Technique for Cloud Applications Based on Software Defined Networking," IEEE Access, 2017,5, pp. 21229-21241. DOI: 10.1109/ACCESS.2017.2755768

[Lira (2017)] V. Lira, E. Tavares and M. Oliveira, "An Approach for Reducing Energy Consumption in Dependable Virtual Network Embedding," 2017 26th International Conference on Computer Communication and Networks (ICCCN), Vancouver, BC, 2017, pp. 1-9. DOI:10.1109/ICCCN.2017.8038459

[Liu 2018] Liu Q., Jiang Y. "A Survey of Machine Learning-Based Resource Scheduling Algorithms in Cloud Computing Environment," In: Sun X., Pan Z., Bertino E. (eds) Cloud Computing and Security. ICCCS 2018. Lecture Notes in Computer Science, vol 11063. Springer, Cham. DOI: 10.1007/978-3-030-00006-6

[Lucena (2018)] O. Lucena, O. Adamuz-Hinojosa, P. Ameigeiras, P. Munoz, JJ. Ramos-Munoz, J. Folgueira Chavarria, D. R. Lopez, "The Creation Phase in Network Slicing: From a Service Order to an Operative Network Slice," 2018 European Conference on Networks and Communications (EuCNC), Ljubljana, Slovenia, pp. 1-36, 2018. DOI: 10.1109/EuCNC.2018.8443255

[Luizelli (2015)] M. C. Luizelli, L. R. Bays, L. S. Buriol, M. P. Barcellos and L. P. Gaspary, 2015. "Piecing together the NFV provisioning puzzle: Efficient placement and chaining of virtual network functions," 2015 IFIP-IEEE International Symposium on Integrated Network Management (IM), Ottawa, ON, 2015, pp. 98-106. DOI:10.1109/INM.2015.7140281

[Luo (2014)] P. Luo, X. Li, M. Chen, "Hybrid shuffled frog leaping algorithm for energy-efficient dynamic consolidation of virtual machines in cloud datacenters," Journal of Expert Systems with Applications, Volume 41, Issue 13, pp. 5804-5816, 2014. DOI: 10.1016/j.eswa.2014.03.039

[Luong-Vy 2018] LE, Luong-Vy; LIN, Bao-Shuh; DO, Sinh, "Applying Big Data, Machine Learning, and SDN/NFV for 5G Early-Stage Traffic Classification and Network QoS Control," Transactions on Networks and Communications, [S.l.], v. 6, n. 2, p. 36, may 2018. DOI: 10.14738/tnc.62.4446

[Lyu (2018)] X. Lyu, H. Tian, W. Ni, Y. Zhang, P. Zhang and R. P. Liu, "Energy-Efficient Admission of Delay-Sensitive Tasks for Mobile Edge Computing," in IEEE Transactions on Communications, vol. 66, no. 6, pp. 2603-2616, June 2018. DOI: 10.1109/TCOMM.2018.2799937.

[Mao 2017] Mao, B., Fadlullah, Z.M., Tang, F., et al., "Routing or Computing: The Paradigm Shift Towards Intelligent Computer Network Packet Transmission Based on Deep Learning," IEEE Transactions on Computers, 2017,66,(11), pp. 1946:1960. DOI: 10.1109/TC.2017.2709742

[Martin (2018)] A. Martin et al., "Network Resource Allocation System for QoE-Aware Delivery of Media Services in 5G Networks," in IEEE Transactions on Broadcasting, vol. 64, no. 2, pp. 561-574, 2018. DOI: 10.1109/TBC.2018.2828608

[Massimo (2018)] M. Condoluci, T. Mahmoodi, "Softwarization and virtualization in 5G mobile networks: Benefits, trends and challenges," Journal of Computer Networks, Volume 146, pp. 65-84, 2018. DOI: 10.1016/j.comnet.2018.09.005

[Meng (2018)] 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

[Meshkati 2019] Meshkati, Jafar, Safi-Esfahani, Faramarz, "Energy-aware resource utilization based on particle swarm optimization and artificial bee colony algorithms in cloud computing," The Journal of Supercomputing, 2018, October 11. DOI: 10.1007/s11227-018-2626-9

[Miguel (2018)] T.N. Miguel, L. Ines, M. Vasco, "Virtual machine consolidation using constraint-based multi-objective optimization," Journal of Algorithms, 2018. DOI: 10.1007/s10732-018-9400-2

[Mijumbi (2015)] R. Mijumbi, J. Serrat, J. L. Gorricho and R. Boutaba, "A Path Generation Approach to Embedding of Virtual Networks," in IEEE Transactions on Network and Service Management, vol. 12, no. 3, pp. 334-348, Sept. 2015. DOI: 10.1109/TNSM.2015.2459073.

[Mijumbi (2016)] 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/

[Mohammadi (2018)] Mohammadi Bahram Abadi, R., Rahmani, A.M. Hossein Alizadeh, S. "Self-adaptive architecture for virtual machines consolidation based on probabilistic model evaluation of data centers in Cloud computing," Cluster Comput, 21: 1711, 2018. DOI:10.1007/s10586-018-2806-7

[Monireh (2018)] S. Monireh H., T. Haghighat, Abolfazl, R. Amir Masoud, "A reliable energy-aware approach for dynamic virtual machine consolidation in cloud datacenters," The Journal of Supercomputing, 2018. DOI: 10.1007/s11227-018-2709-7

[Murali 2018] A. Murali, N. N. Das, S. S. Sukumaran, K. Chandrasekaran, C. Joseph and J. P. Martin, "Machine Learning Approaches for Resource Allocation in the Cloud: Critical Reflections," 2018 International Conference on Advances in Computing, Communications and Informatics (ICACCI), Bangalore, 2018, pp. 2073-2079. DOI: 10.1109/ICACCI.2018.8554703

[Nahid (2017)] N. Nia, S.Adabi, M. Nategh, 2017. "A Coordinated Algorithm Approach for Virtual Network Embedding in Cloud Infrastructure," KSII Transactions on Internet and Information Systems, vol. 11, no. 5, pp. 2346-2361. DOI: 10.3837/tiis.2017.05.002.

[Naik (2018)] B. B. Naik, D. Singh, A. B. Samaddar and S. Jung, "Developing a Cloud Computing Data Center Virtual Machine Consolidation Based on Multi-objective Hybrid Fruit-fly Cuckoo Search Algorithm," 2018 IEEE 5G World Forum (5GWF), Silicon Valley, CA, pp. 512-515, 2018. DOI: 10.1109/5GWF.2018.8516947

[Namra (2019)] S. Namra Bhadreshkumar, T. Tirth Chetankumar, R. Shrey Manish, T. Harshal, S. Suresh Chandra, J. Amit, "Adaptive Live Task Migration in Cloud Environment for Significant Disaster Prevention and Cost Reduction," in Information and Communication Technology for Intelligent Systems, Springer, pp. 639-654, 2019. DOI: 10.1007/978-981-13-1742-2

[Nasrin (2016)] A. Nasrin, O. Mohamed, "Energy aware resource allocation of cloud datacenter: review and open issues," Journal of Cluster Computing, V 19, N 3, pp. 1163-1182, 2016. DOI: 10.1007/s10586-016-0579-4

[Nasroollah 2019] Nasroollah Z., Moonsamy I., Chuttur Y. (2019) A Cloud-Based Energy Monitoring System Using IoT and Machine Learning. In: Satapathy S., Bhateja V., Somanah R., Yang XS., Senkerik R. (eds) Information Systems Design and Intelligent Applications. Advances in Intelligent Systems and Computing, vol 863. Springer, Singapore DOI: 10.1007/978-981-13-3338-5

[NGMN (2015)] Rachid El Hattachi, and Javan Erfanian, "NGMN 5G White Paper," 2015. https://www.ngmn.org/.

[Nonde (2015)] L. Nonde, T. El-Gorashi, and J. Elmirghani, 2015. "Energy Efficient Virtual Network Embedding for Cloud Networks," Journal of Lightwave Technology, vol. 33, no. 9, pp. 1828-1849. DOI:10.1109/JLT.2014.2380777.

[Noshy (2018)] 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

[Ogino (2017)] N. Ogino, T. Kitahara, S. Arakawa, M. Murata, 2017. "Virtual network embedding with multiple priority classes sharing physical resources," Journal of Computer Networks, vol. 112, issue C, pp. 52-66. DOI: 10.1016/j.comnet.2016.10.007.

[ONF (2012)] Open Networking Foundation, "Software-Defined Networking: The New Norm for Networks," ONF White Paper, 2012. https://www.opennetworking.org/.

[Opendaylight (2019)] OpenDayLight Organization, "OpenDaylight Use Cases," Available online at: www.opendaylight.org, accessed on April-2019.

[Openstack (2019)] Openstack Organization, "What is OpenStack". Available online at: www.openstack.org, accessed on April-2019.

[Ordonez (2017)] J. Ordonez-Lucena, P. Ameigeiras, D. Lopez, J. J. Ramos-Munoz, J. Lorca and J. Folgueira, "Network Slicing for 5G with SDN/NFV: Concepts, Architectures, and Challenges," in IEEE Communications Magazine, vol. 55, no. 5, pp. 80-87, 2017. DOI: 10.1109/MCOM.2017.1600935

[Pacheco 2018] F. Pacheco, E. Exposito, M. Gineste, C. Baudoin and J. Aguilar, "Towards the deployment of Machine Learning solutions in network traffic classification: A systematic survey," in IEEE Communications Surveys and Tutorials. DOI: 10.1109/COMST.2018.2883147

[Park 2018] Park, Donghyun; Kim, Seulgi; An, Yelin; Jung, Jae-Yoon, "LiReD: A Light-Weight Real-Time Fault Detection System for Edge Computing Using LSTM Recurrent Neural Networks," Sensors 18, no. 7: 2110, 2018. DOI: 10.3390/s18072110

[Pasca (2017)] Pasca, S.T.V., Kodali, S.S.P., Kataoka, K., "AMPS: Application aware multipath flow routing using machine learning in SDN," In Proc. of IEEE Twenty-third National Conference on Communications (NCC), Chennai, India, Mar. 2017, pp. 1-6. DOI: 10.1109/NCC.2017.8077095

[Paulraj (2018)] G. J. L. Paulraj, S. A. J. Francis, J. D. Peter and I. J. Jebadurai, "Route Aware Virtual Machine Migration in Cloud Datacenter," 2018 Second International Conference on Inventive Communication and Computational Technologies (ICICCT), Coimbatore, pp. 363-367, 2018. DOI: 10.1109/ICICCT.2018.8472980

[Pham (2017)] 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

[Qin (2019)] J. Qin, Y. Wu, Y. Chen, K. Xue and D. S. L. Wei, "Online User Distribution-Aware Virtual Machine Re-Deployment and Live Migration in SDN-Based Data Centers," in IEEE Access, vol. 7, pp. 11152-11164, 2019. DOI:10.1109/ACCESS.2019.2891115

[Qin 2018] M. Qin, Q. Yang, N. Cheng, H. Zhou, R. R. Rao and X. Shen, "Machine Learning Aided Context-Aware Self-Healing Management for Ultra Dense Networks With QoS Provisions," in IEEE Transactions on Vehicular Technology, vol. 67, no. 12, pp. 12339-12351, Dec. 2018. DOI: 10.1109/TVT.2018.2877910

[Qu (2016)] L. Qu, C. Assi and K. Shaban, 2016. "Delay-Aware Scheduling and Resource Optimization With Network Function Virtualization," in IEEE Transactions on Communications, vol. 64, no. 9, pp. 3746-3758. 2016.DOI:10.1109/TCOMM.2016.2580150

[Rachael (2019)] Rachael Shaw, Enda Howley, Enda Barrett, "An energy efficient anti-correlated virtual machine placement algorithm using resource usage predictions," Simulation Modeling Practice and Theory, Volume 93, 2019, Pages 322-342. DOI:10.1016/j.simpat.2018.09.019

[Rahman 2018] S. Rahman, T. Ahmed, M. Huynh, M. Tornatore and B. Mukherjee, "Auto-Scaling VNFs Using Machine Learning to Improve QoS and Reduce Cost," 2018 IEEE International Conference on Communications (ICC), Kansas City, MO, 2018, pp. 1-6. DOI: 10.1109/ICC.2018.8422788

[Ranjbari (2018)] Ranjbari, Javad Akbari Torkestani, "A learning automata-based algorithm for energy and SLA efficient consolidation of virtual machines in cloud data centers," Journal of Parallel and Distributed Computing, Volume 113, 2018, Pages 55-62. DOI:10.1016/j.jpdc.2017.10.009

[Ranjbari (2018)] Ranjbari, Javad Akbari Torkestani, "A learning automata-based algorithm for energy and SLA efficient consolidation of virtual machines in cloud data centers," Journal of Parallel and Distributed Computing, Volume 113, Pages 55-62, 2018. DOI: 10.1016/j.jpdc.2017.10.009

[Reddy (2016)] V. S. Reddy, A. Baumgartner and T. Bauschert, 2016. "Robust embedding of VNF/service chains with delay bounds," 2016 IEEE Conference on Network Function Virtualization and Software Defined Networks (NFV-SDN), Palo Alto, CA, pp. 93-99.DOI:10.1109/NFV-SDN.2016.7919482

[RFC7679 (2017)] G. Almes, S. Kalidindi, M. Zekauskas, and A. Morton, "A One-Way Delay Metric for IP Performance Metrics (IPPM)," 2016. https://tools.ietf.org/html/rfc7679.

[Riera (2015)] 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

[Riera and Grasa (2014)] 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

[Roshdy 2018] A. Roshdy, A. Gaber, F. Hantera and M. ElSebai, "Mobility load balancing using machine learning with case study in live network," 2018 International Conference on Innovative Trends in Computer Engineering (ITCE), Aswan, 2018, pp. 145-150. DOI: 10.1109/ITCE.2018.8316614

[Saeed 2019] A. Saeed and M. Kolberg, "Towards Optimizing WLANs Power Saving: Novel Context-Aware Network Traffic Classification Based on a Machine Learning Approach," in IEEE Access, vol. 7, pp. 3122-3135, 2019. DOI: 10.1109/ACCESS.2018.2888813

[Sahrish Khan (2019)] T. Sahrish Khan, S. Munam Ali. "Resource allocation in SDN based 5G cellular networks," Journal of Peer-to-Peer Networking and Applications, Springer, V 12, N 2, pp. 514-538, 2019. DOI: 10.1007/s12083-018-0651-3.

[Satpathy (2018)] A. Satpathy, S. Kanti Addya, A. Kumar Turuk, B. Majhi, G. Sahoo, "Crow search based virtual machine placement strategy in cloud datacenterswith live migration," Journal of Computers and Electrical Engineering, Volume 69, pp. 334-350, 2018. DOI: 10.1016/j.compeleceng.2017.12.032

[SDNsurvey (2015)] D. Kreutz, F. M. V. Ramos, P. E. Veríssimo, C. E. Rothenberg, S. Azodolmolky and S. Uhlig, "Software-Defined Networking: A Comprehensive Survey," in Proceedings of the IEEE, vol. 103, no. 1, pp. 14-76, Jan. 2015. DOI: 10.1109/JPROC.2014.2371999.

[Sendra 2017] Sendra, S., Rego, A., Lloret, J., Jimenez, J.M., Romero, O., "Including artificial intelligence in a routing protocol using Software Defined Networks," In Proc. of IEEE International Conference on Communications Workshops (ICC Workshops), Paris, France, May 2017, pp. 670-674. DOI: 10.1109/ICCW.2017.7962735

[Shahriar (2016)] N. Shahriar et al., "Connectivity-aware virtual network embedding," 2016 IFIP Networking Conference (IFIP Networking) and Workshops, Vienna, 2016, pp. 46-54. DOI: 10.1109/IFIPNetworking.2016.749724.

[Shangguang (2018)] S. Wang, J. Xu, N. Zhang and Y. Liu, "A Survey on Service Migration in Mobile Edge Computing," in IEEE Access, vol. 6, pp. 23511-23528, 2018. DOI: 10.1109/ACCESS.2018.2828102

[Shashank (2014)] Shashank Shanbhag, Arun Reddy Kandoor, Cong Wang, Ramgopal Mettu, Tilman Wolf, "VHub: Single-stage virtual network mapping through hub location," Computer Networks, Volume 77, 11 February 2015, Pages 169-180, 2014. DOI: 10.1016/j.comnet.2014.12.006.

[Shuiqing (2016)] Shuiqing Gong, Jing Chen, Siyi Zhao and Qingchao Zhu, "Virtual Network Embedding with Multi-attribute Node Ranking Based on TOPSIS," KSII Transactions on Internet and Information Systems, vol. 10, no. 2, pp. 522-541, 2016. Available online at: DOI: 10.3837/tiis.2016.02.005.

[Shuting Xu (2017)] 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

[Silva (2018)] 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

[Soualah (2016)] O. Soualah, I. Fajjari, M. Hadji, N. Aitsaadi and D. Zeghlache, "A novel virtual network embedding scheme based on Gomory-Hu tree within cloud's backbone," NOMS 2016 - 2016 IEEE/IFIP Network Operations and Management Symposium, Istanbul, 2016, pp. 536-542. DOI: 10.1109/NOMS.2016.7502855.

[Soualah (2017)] 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/

[Stefano 2015] Di Stefano, A., Cammarata, G., Morana, G., Zito, D., "A4sdn-adaptive alienated ant algorithm for software-defined networking," In Proc. of 10th International Conference on P2P, Parallel, Grid, Cloud and Internet Computing (3PGCIC), Krakow, Poland, Nov. 2015, pp. 344-350. DOI: 10.1109/3PGCIC.2015.120

[Su (2014)] S. Su, Z. Zhang, A.. Liu, X. Cheng, Y. Wang, and X. Zhao, 2014. "Energy-Aware Virtual Network Embedding," IEEE/ACM Transactions on Networking, vol. 22, no. 5, pp. 1607-1620. DOI: 10.1109/TNET.2013.2286156.

[Sultana 2019] Sultana, N., Chilamkurti, N., Peng, W. et al, "Survey on SDN based network intrusion detection system using machine learning approaches," Peer-to-Peer Netw. Appl. (2019) 12: 493. DOI: 10.1007/s12083-017-0630-0

[Surabhi (2019)] S. Surabhi, G. Journal of Neeraj, L. Ashish Kumar, S. Dharm, H. Pao-Ann, H. Kamarul Bin Ghazali, L. Pawan, S. Pradeep Kumar, "Queueing Analysis of Migration of Virtual Machines," Journal of Advanced Informatics for Computing Research Springer, pp. 782-793, 2019. DOI: 10.1007/978-981-13-3140-4

[Tang 2019] Y. Tang, N. Cheng, W. Wu, M. Wang, Y. Dai and X. Shen, "Delay-Minimization Routing for Heterogeneous VANETs With Machine Learning Based Mobility Prediction," in IEEE Transactions on Vehicular Technology, vol. 68, no. 4, pp. 3967-3979, April 2019. DOI: 10.1109/TVT.2019.2899627

[TIA (2017)] TIA, "Telecommunications Infrastructure Standard for Data Centers," Available online at: http://www.tia-942.org/, accessed on April-2019.

[TR 21.915 3GPP (2019)] TR 21.915 V1.0.0 (2019-03) "Technical Report: Release 15 Description; Summary of Rel-15 Work Items (Release 15)," 3rd Generation Partnership Project; Technical Specification Group Services and System Aspects. https://www.3gpp.org/release-15.

[Triki (2015)] N. Triki, N. Kara, M. El Barachi, S. Hadjres, 2015. "A green energy-aware hybrid virtual network embedding," Computer networks, vol. 91, pp. 712-737. DOI: 10.1016/j.comnet.2015.08.016.

[Varasteh (2017)] A. Varasteh and M. Goudarzi, "Server Consolidation Techniques in Virtualized Data Centers: A Survey," in IEEE Systems Journal, vol. 11, no. 2, pp. 772-783, 2017. DOI:10.1109/JSYST.2015.2458273

[Verma (2018)] JK. Verma, S. Kumar, O. Kaiwartya, et al, "Enabling green computing in cloud environments: Network virtualization approach toward 5G support," Transactions on Emerging Telecommunications Technologies, 2018. DOI: 10.1002/ett.3434

[Violeta (2014)] Violeta Medina and Juan Manuel Garca, "A survey of migration mechanisms of virtual machines," ACM Compute Vol. 46, No.3, 33 pages, 2014. DOI: 10.1145/2492705

[Wang (2018)] K. Wang, Q. Zhou, S. Guo and J. Luo, "Cluster Frameworks for Efficient Scheduling and Resource Allocation in Data Center Networks: A Survey," in IEEE Communications Surveys and Tutorials, vol. 20, no. 4, pp. 3560-3580, 2018. DOI: 10.1109/COMST.2018.2857922

[Wang 2018] J. Wang et al., "A Machine Learning Framework for Resource Allocation Assisted by Cloud Computing," in IEEE Network, vol. 32, no. 2, pp. 144-151, March-April 2018. DOI: 10.1109/MNET.2018.1700293

[Wang (2018)] Q. Wang, J. Fu, J. Wu, B. Moran and M. Zukerman, "Energy-Efficient Priority-Based Scheduling for Wireless Network Slicing," 2018 IEEE Global Communications Conference (GLOBECOM), Abu Dhabi, United Arab Emirates, 2018, pp. 1-6. DOI: 10.1109/GLOCOM.2018.8647696.

[Wen (2015)] X. Wen, Y. Han, H. Yuan, X. Zhou and Z. Xu, "An Efficient Resource Embedding Algorithm in Software Defined Virtualized Data Center," 2015 IEEE Global Communications Conference (GLOBECOM), San Diego, CA, 2015, pp. 1-7. DOI: 10.1109/GLOCOM.2015.7417556.

[Xie (2018)] R. Xie and R. Wang, "Hierarchical Caching Resource Sharing in 5G Cellular Networks with Virtualization," 2018 IEEE International Conference on Communications Workshops (ICC Workshops), Kansas City, MO, pp. 1-6, 2018. DOI: 10.1109/ICCW.2018.8403700

[Xie 2019] J. Xie et al., "A Survey of Machine Learning Techniques Applied to Software Defined Networking (SDN): Research Issues and Challenges," in IEEE Communications Surveys and Tutorials, vol. 21, no. 1, pp. 393-430, 2019. DOI: 10.1109/COMST.2018.2866942

[Xiao (2018)] Y. Xiao and M. Krunz, "Dynamic Network Slicing for Scalable Fog Computing Systems With Energy Harvesting," in IEEE Journal on Selected Areas in Communications, vol. 36, no. 12, pp. 2640-2654, Dec. 2018. DOI: 10.1109/JSAC.2018.2871292.

[Xu P 2018] Xu, P., He, G., Li, Z., Zhang, Z., "An efficient load balancing algorithm for virtual machine allocation based on ant colony optimization," International Journal of Distributed Sensor Networks. 2018. DOI: 10.1177/1550147718793799

[Yang Wang (2015)] Yang Wang, Qian Hu, Xiaojun Cao, "A branch-and-price framework for optimal virtual network embedding, Computer Networks," Volume 94, 15 January 2016, Pages 318-326, 2015. DOI: 10.1016/j.comnet.2015.11.005.

[Yousefipour (2018)] A. Yousefipour, AM. Rahmani, M. Jahanshahi, "Energy and cost aware virtual machine consolidation in cloud computing," Wiley Journal of Software Practice Experience, V. 48, pp. 1758-1774, 2018. DOI: 10.1002/spe.2585

[Yu (2008)] M. Yu, Y. Yi, J. Rexford, and M. Chiang, 2008. "Rethinking virtual network embedding: Substrate support for path splitting and migration," ACM SIGCOMM CCR, vol. 38, noi. 2, pp. 17-29. DOI: 10.1145/1355734.1355737.

[Zhang (2018a)] 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

[Zhang (2018)] J. Zhang et al., "Energy-Latency Trade-off for Energy-Aware Offloading in Mobile Edge Computing Networks," in IEEE Internet of Things Journal, vol. 5, no. 4, pp. 2633-2645, Aug. 2018. DOI: 10.1109/JIOT.2017.2786343.

[Zhihua (2018)] Zhihua Li, Chengyu Yan, Lei Yu, Xinrong Yu, "Energy-aware and multi-resource overload probability constraint-based virtual machine dynamic consolidation method," Future Generation Computer Systems, Volume 80, 2018, Pages 139-156. DOI:10.1016/j.future.2017.09.075

[Zhongbao (2015)] Z. Zhang, S. Su, J. Zhang, K. Shuang, P. Xu, 2015. "Energy aware virtual network embedding with dynamic demands: Online and offline," Computer Networks, Volume 93, pp. 448-459. DOI: 10.1016/j.comnet.2015.09.036.

[ZhongWang (2018)] Z. Wang, D. Sun, G. Xue, S. Qian, G. Li, M. Li, "Ada-Things: An adaptive virtual machine monitoring and migration strategy for Internet of things applications," Journal of Parallel and Distributed Computing, 2018. DOI: 10.1016/j.jpdc.2018.06.009

[Zhou (2018)] H. Zhou, Q. Li, K. Raymond Choo, H. Zhu, "DADTA: A novel adaptive strategy for energy and performance efficient virtual machine consolidation," Journal of Parallel and Distributed Computing, Volume 121, pp. 15-26, 2018. DOI: 10.1016/j.jpdc.2018.06.011