PEOPLE'S DEMOCRATIC REPUBLIC OF ALGERIA
MINISTRY OF HIGHER EDUCATION AND SCIENTIFIC RESEARCH
UNIVERSITY MOHAMED BOUDIAF - M'SILA

FACULTY: MATHEMATICS AND INFORMATICS

DEPARTEMENT: CMPUTER SCIENCE

N° :……..

DOMAIN: MATHEMATICS AND COMPUTER SCIENCE

FIELD: COMPUTER SCIENCE
OPTION: RTIC

Dissertation submitted in partial fulfillment of the requirements for The degree of MASTER

By: ZAIDI BELQASSIM

Subject

Virtual Network Functions placement

Presented publicly to the jury the: 13/07/2019

……	University of M'sila	Chairman
Dr. LAMRI SAYAD	University of M'sila	Supervisor
……	University of M'sila	Examiner

Academic year: 2018 /2019

PEOPLE'S DEMOCRATIC REPUBLIC OF ALGERIA
MINISTRY OF HIGHER EDUCATION AND SCIENTIFIC RESEARCH
UNIVERSITY MOHAMED BOUDIAF - M'SILA

FACULTY: MATHEMATICS AND INFORMATICS

DEPARTEMENT: CMPUTER SCIENCE

N° :……..

DOMAIN: MATHEMATICS AND COMPUTER SCIENCE

FIELD: COMPUTER SCIENCE
OPTION: RTIC

Dissertation submitted in partial fulfillment of the requirements for The degree of MASTER

By: ZAIDI BELQASSIM

Subject

Virtual Network Functions placement

Presented publicly to the jury the: 13/07/2019

……	University of M'sila	Chairman
Dr. LAMRI SAYAD	University of M'sila	Supervisor
……	University of M'sila	Examiner

Academic year: 2018 /2019

Acknowledgment

First and foremost, heartfelt gratitude and praises go to the Almighty Allah who guided me through and through. This work could not have reached fruition without the unflagging assistance and participation of so many people whom I would never thank enough for the huge contribution that made this work what it is now.

I would like to thank profoundly Mr. SAYAD LAMRI for his scientific guidance and corrections, suggestions and advice, pertinent criticism and pragmatism, and particularly for his hard work and patience. I am very grateful to him, for without his help an important part of this work would have been missed. I would also like to thank all the Jury Members, who have agreed to review this work.

I would also thank those who helped me to finish this work whom without their assistance it would not have been done I thank all the teachers who guided us throughout our journey.

Thanks to all who helped me.

Thanks.

Dedication

This dissertation is dedicated to My parents for their encouragement, prayers, motivations and being there. May God bless them. My precious brothers and sisters who've been there for me in my entire journey. All of my friends and colleagues.

TABLE OF CONTENTS

FIGURE LIST

TABLE LIST

ABBREVIATIONS LIST

BnB	Branch-and-Bound
BPP	Bin Packing Problem
CaaS	Crypto as a Service
CI/CD	Continuous Integration/Continuous Deployment
CPVNF	cost-efficient Proactive VNF
DHCP	Dynamic Host Configuration Protocol
DNS	Domain System Name
ETSI	European Telecommunications Standards Institute
GWO	Grey wolf Optimizer
IaaS	Infrastructure as a Service
IEGWO	Integer Encoding GWO
ILP	Integer Linear Programming
IoT	Internet of Things
IT	Information Technology
MANO	Management and Orchestration
MILP	Mixed Integer linear Programming
MIP	Mixed Integer Programming
MVs	Virtual Machine
NAS	Network Attached Storage
NFS	Network File System
NFV	Network Function Virtualization
NFVI	Network Function Virtualization Infrastructure
NFVIaaS	Network Function Virtualization Infrastructure as a Service
NTP	Network Time Protocol
PaaS	Platform as a Service
QoE	Quality of Experience
SaaS	Software as a Service
SANs	Storage-Area Networks
SecaaS	Security as a Service
SFC	Service Function Chain
SNMP	Simple Network Management Protocol
TFTP	Simple Network Management Protocol
TOs	Telecom Operators
vCDN	Virtual Content Delivery Network
VNFCs	VNF Components
VNF-P	Virtual Network Function Placement
VNFs	Virtual Network Functions and Services

GENERAL INTRODUCTION

Today's networks consist of a wide range of special devices used to support network functions such as firewalls, Network Address Translator (NAT), Quality-of-Service (QoS) analyzers, etc. With the proliferation of networks and access to e-mail, social networking or cloud computing for mobile devices. The demand for network services has increased, and it has become difficult to integrate and operate such appliances or to add new functionalities to keep up with the ever-increasing demand at a reasonable cost. To cope with this challenge, the Network Functions Virtualization (NFV) paradigm has been recently proposed.

Network Function Virtualization (NFV) is an emerging technique to improve the performance of enterprise networks were This technique utilizes standard IT virtualization technology to allow quick service development for Network Operators and Service Providers [1]. Most current networks are composed of different network appliances that are connected or chained in a particular way to obtain the required network service functionality. Network Function Virtualization seeks to replace these network appliances with virtualized network functions (VNFs) that can be consolidated onto industry standard high-volume servers, switches, and storage, which could be in data centers, network nodes, or in the end-user premises.

Virtual Network Functions (VNFs) are deployed to satisfy the service requirements of flows. The requirements of a VNF are similar to those of a VM in terms of computing and storage resources. In the present work, we study the Virtual Network Functions placement problem and we seek to find the optimal allocation of the services function chains on the physical servers while satisfying the constraints on the server resources (CPU, RAM, DISC).

This dissertation is structured as follows:

- **The first chapter:** the first chapter introduces the basic concepts of network functions virtualization (NFV) including terminologies, history, architecture, and their use cases.
- **The second chapter:** presents the related works of Virtual Network Function (VNF) placement, a summary of state-of-the-art VNF placement, problem formulation system model and proposed VNF Placement Algorithm.
- **The third chapter:** presents the environment of work and Python programming language, IDE develop Pycharm, proposed algorithm implementation, values, and results.

CHAPTER 01

NETWORK FUNCTION VIRTUALIZATION

1. Introduction

Many service providers go beyond simply providing network connectivity to their corporate customers. They also offer additional services and network functions, such as network address translation, firewalls, encryption, domain name services, caching, … etc.

The virtualization of network functions, known by the acronym NFV, separate network functions, such as firewall or encryption, from any dedicated hardware and moves them to servers. Instead of installing costly proprietary equipment, network service providers can simply offer these services by means of switches, storage and cheaper servers executing virtual machines performing network functions.

In this chapter, we present the basic concepts of network functions virtualization (NFV) including terminologies, history, architecture and their use cases.

2. Background

In this section, we will introduce some generalities about cloud computing, data centers, and virtualization.

2.1. Cloud computing

Cloud computing is a model of access through the Internet to a set of digital resources that can be allocated and released on demand and for which the service provider provides all maintenance and support activities [2]. Consequently, it is possible to put on remote-located servers storage data or software that is usually stored on a user's computer, or even on servers installed in a local area network. This virtualization of resources allows the company to access its data without having to manage an IT infrastructure, which is often complex and represents a certain cost.

There are three main types of Cloud computing services: Software as a Service (SaaS), Platform as a Service (PaaS) and Infrastructure as a Service (IaaS), as illustrated in Figure 1.1. [3]

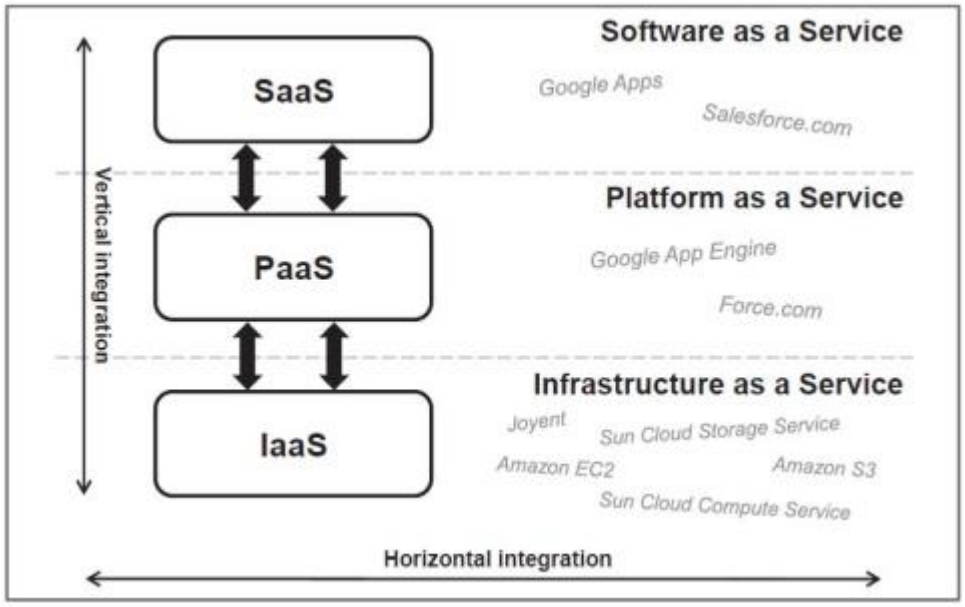

Figure1. 1: The three layers of Cloud Computing: SaaS, PaaS, and IaaS [3]

2.2. Data center

Data Centers contain critical computing resources in controlled environments and under centralized management. This allows companies operate 24 hours. These computing resources include computers central, web and application servers, file and print servers, servers messaging software, application software, and the operating system, as well as the storage, and network infrastructure, Storage-Area Networks (SANs). A number of servers support network operations and network applications. Applications operational network include Network Time Protocol (NTP), FTP, DNS (Domain System Name), Dynamic Host Configuration Protocol (DHCP), SNMP (Simple Network Management Protocol), TFTP, Network File System (NFS), and applications based on network, including IP telephony, IP video, IP videoconferencing, etc. [4]

2.3. Virtualization

Virtualization is a computer mechanism that consists of running multiple systems, servers or applications on the same physical server or machine which leads to share the same physical infrastructure. In other words, it’s an approach to pooling and sharing technology resources to optimize the total cost, simplify management and increase asset use so that IT resources can more readily meet business demand.

There are three main types of server virtualization: virtualization of the operating system, hardware emulation (Figure 1.2), and paravirtualization (Figure 1.3), designed to provide application size, a more efficient virtualization approach. [5]

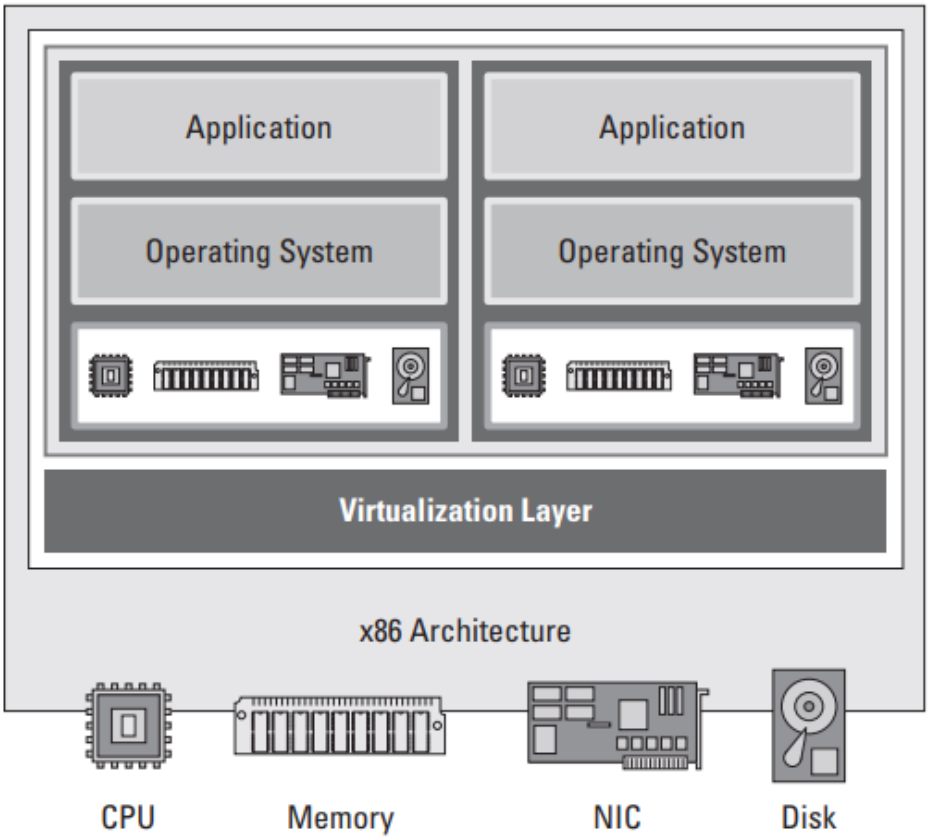

Figure1. 2: Hardware emulation virtualization [5]

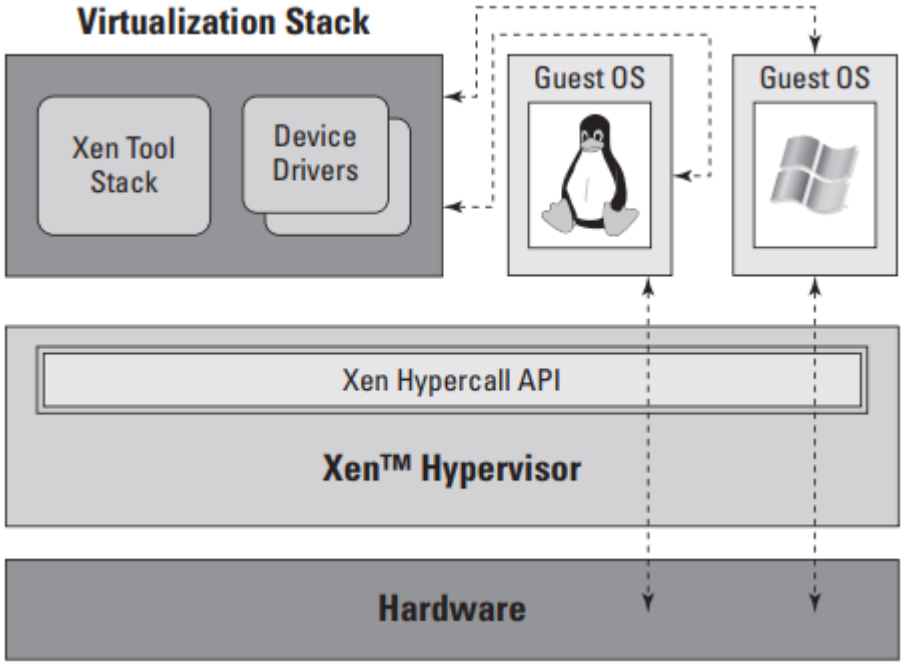

Figure1. 3: A look at paravirtualization [5]

3. Definition of Network Function Virtualization

The term NFV was originally proposed by over twenty of the world's largest Telecom Operators (TOs) such as AT&T, BT and DT. According to ETSI, the idea of NFV is recognized as a network architecture which transforms the way of building and operating networks by leveraging standard IT virtualization technologies to consolidate many network equipment types onto industry standard high-volume servers, switches and storage, which could be located in Data centers, Network Nodes and in the end user premises, as illustrated in Figure 1.4. It involves the implementation of network functions in software that can run on a range of industry standard server hardware, and that can be moved to, or instantiated in, various locations in the network as required, without the need for installation of new equipment. [6]

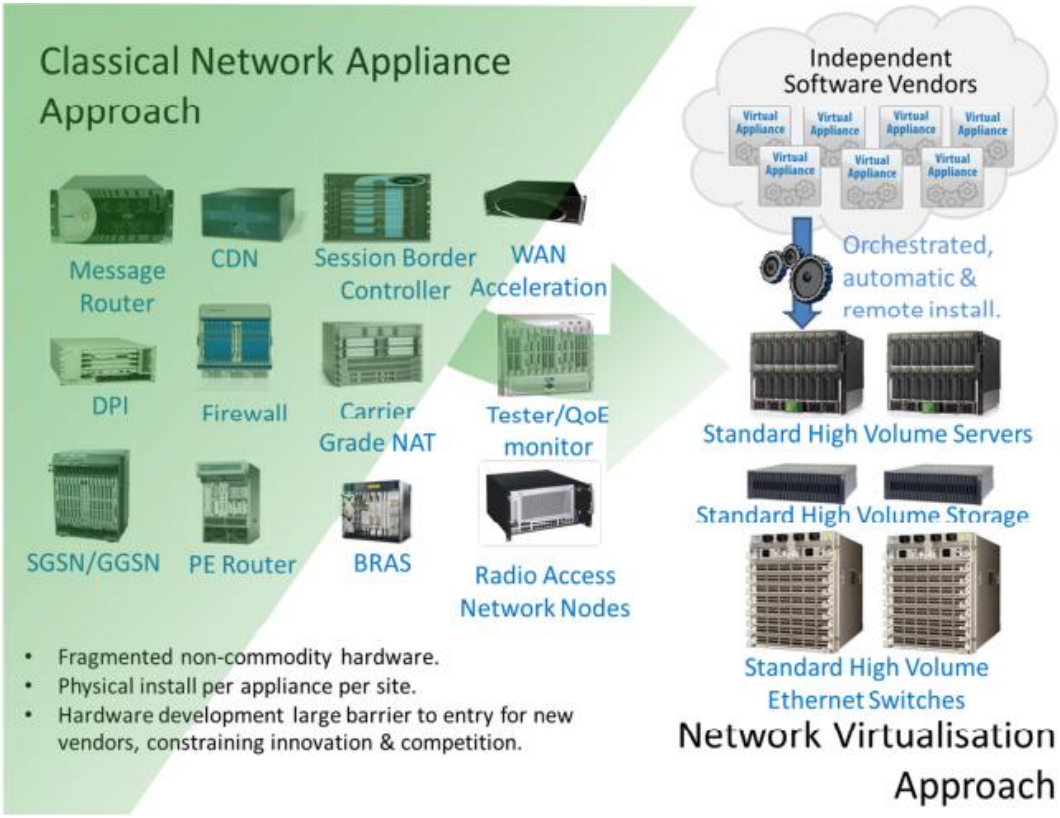

Figure1. 4: Vision for Network Functions Virtualization [6]

4. History

The concept NFV was born in October 2012 when a number of the world's leading Telecommunication Service Providers (TSPs) jointly authored a white paper [6]calling for industrial and research action. In November 2012 seven of these operators (AT&T, BT, Deutsche Telekom, Orange, Telecom Italia, Telefonica and Verizon) selected the European Telecommunications Standards Institute (ETSI) [7] to be the home of the Industry Specification Group for NFV (ETSI ISG NFV). [8]

ETSI ISG NFV has grown to 235 companies including 34 service provider organizations and has held seven plenary meetings spanning Asia, Europe and North America by January 2013 [9] . In October 2013, ETSI has updated the white paper [7] and published the NFV architecture framework which identified NFV system components.

After the first 2-year Phase, a lot of experts and companies (i.e., 289 companies exactly according to ETSI report in 2015) continually joined the working group (i.e., ETSI NFV ISG) to help promote the development of NFV standards and to share their thoughts about NFV. [10]

The Documents published over the first phase (2013-2014) are sometimes erroneously referred to as 'Release 1'and Subsequent tranches were referenced as 'Release 2', 'Release 3' etc. Release 2 development ended in 2016 when Release 3 started. [8]

5. NFV ARCHITECTURE

According to European Telecommunications Standards Institute (ETSI) the NFV Architecture is composed of three key elements: Network Function Virtualization Infrastructure (NFVI), VNFs and NFV MANO (Management and Orchestration), as illustrated in Figure 1.5. [11]

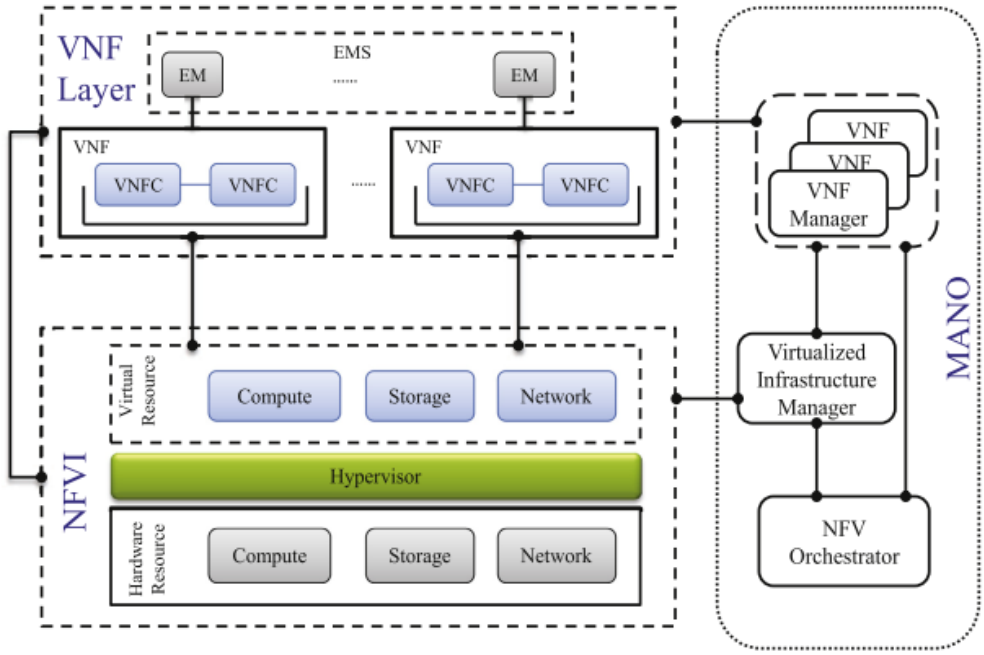

Figure1. 5: ETSI NFV reference architecture [10]

5.1. NFV Infrastructure (NFVI)

NFV infrastructure is a kind of cloud datacenter containing a set physical hardwares and virtual resources that build up the NFV environment including: servers, switches, virtual machines, and virtual switches.

- The first part of the NFV infrastructure is the hardware resources these include computing resources such as (servers, ram) storage resources such as (disk storage, Nass) and network resources such as (switches, firewalls, and routers).

- The second part of the NFV infrastructure is the hypervisor. It abstracts the hardware resources and couples the software from the hardware this enables the software to progress independently from the hardware for the virtualization layer we can use multiple open source and proprietary options such as KVM, QEMU, VMware and OpenStack.
- The third part of the NFV infrastructure is the virtualized resources these include virtual compute, virtual storage, and virtual network.

5.2. Virtual Network Functions and Services

The second component of the NFV architecture is the virtualized network functions, the VNFs are basic building blocks in the NFV architecture they are software implementation of network functions. VNFs can be connected or combined together as building blocks to offer a full-scale network communication service this is known as service chaining, examples of VNFs include vFirewall and vRouter.

5.3. NFV Management and Orchestration (NFV MANO)

The third component of the NFV architecture is the management and network orchestration unit also referred to as MANO has three parts:

- The first part of the NFV MANO is known as the virtualized infrastructure manager. the virtualized infrastructure manager controls and manages the interaction of VNF with NFVI compute storage and network resources it also has necessary deployment and monitoring tools for the virtualization layer.
- The second part of the NFV MANO is also the VNF manager the VNF manager manages the lifecycle of VNF instances it is responsible to: initialize, update, query, scale, and terminate VNF instances.
- The third part of the NFV MANO is known as the orchestrator the orchestrator manages the lifecycle of network services which includes instantiation, policy management, performance measurement, and KPI monitoring. [10]

6. NFV use cases

The use cases of interest for Network Functions Virtualization (NFV):

1) Network Function Virtualization Infrastructure as a Service (NFVIaaS):

The Computing Service user uses the NFV IaaS to manipulate a network accessible compute instance so that it is available for use.

2) VNF Forwarding Graphs:

The VNF Forwarding Graph provides the logical connectivity between virtual devices (i.e VNFs). The VNF forwarding graph use case has the following logical parts and actor-entity relationships as illustrated in the example of Figure 1.6

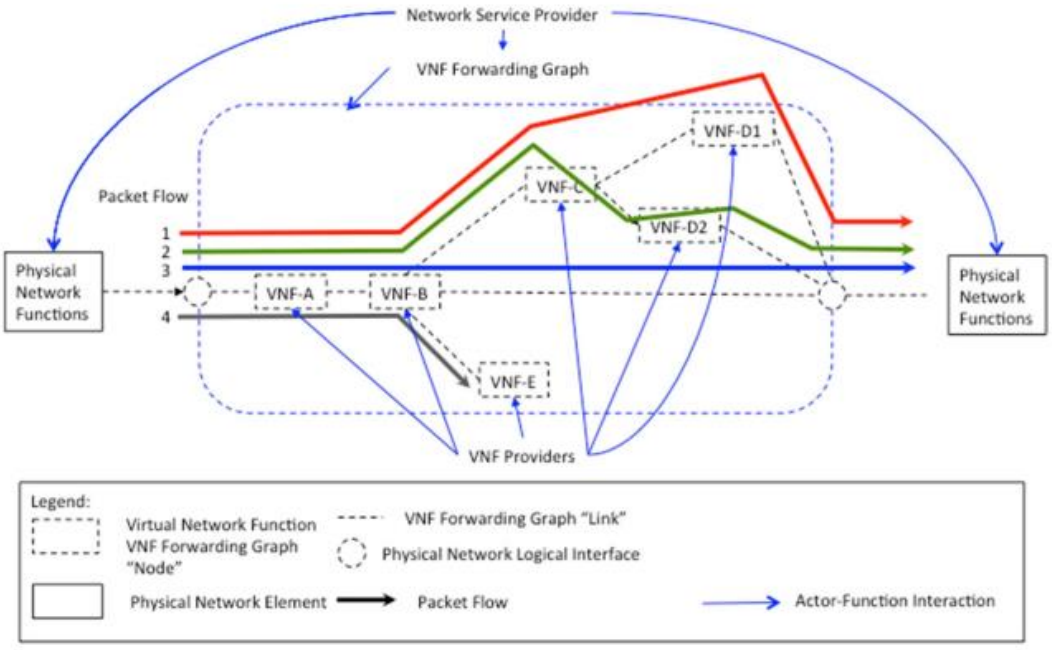

Figure1. 6: Logical View of Virtual Network Function Forwarding Graph (VNF FG) [12]

3) Virtualization of Mobile Core Network and IMS:

The virtualization of a mobile core network entails the capabilities that a virtual mobile network would provide to Network Providers to address customer's requests and associated network capacity demands.

4) Virtualization of Mobile base station:

This use case entails the capabilities that virtualization of mobile base station would provide to Network Providers expand and increase capacity through resource sharing to address customer request and associated customer QoE.

5) Virtualization of the Home Environment:

The NFV technologies give a flexible development of the home environment by reducing hardware-specific functionality with minimal cost, as illustrated in Figure 1.7.

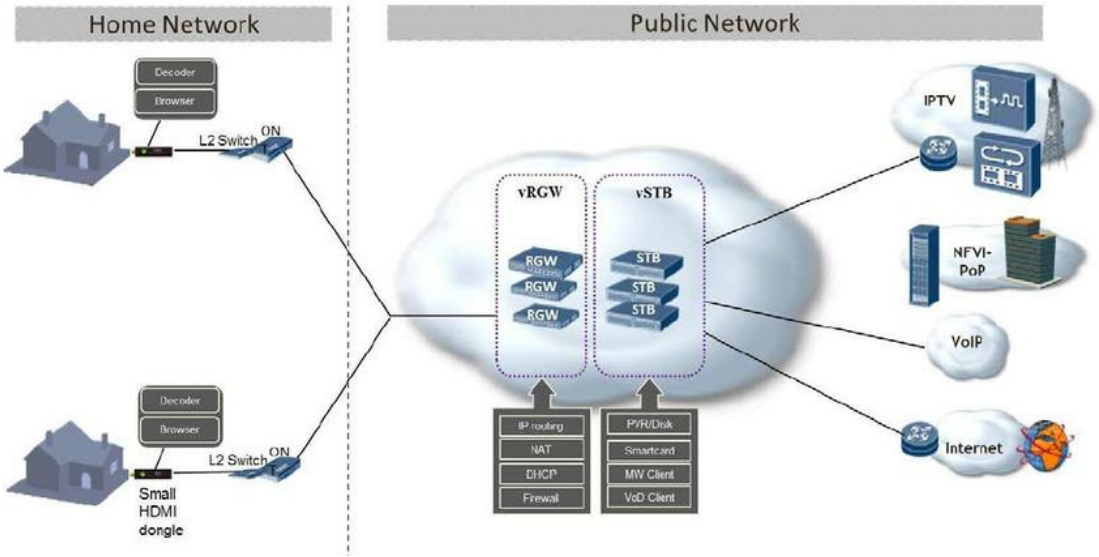

Figure1. 7: Home Virtualization functionality [12]

6) Virtual Content Delivery Network (vCDN) – Fulfilment:

The goal of This use case is extending and scaling Content Delivery Services more easily.

7) Fixed Access Network Functions Virtualization:

Access Network Functions Virtualization will be initially applied to access nodes and aggregation nodes typically located at the edge of the access network, and within multiple-occupancy buildings. These nodes in order to be economically viable are normally compact, with very low power consumption and very low maintenance costs. By applying NFV principles, hardware complexity at the remote node can be reduced both saving energy and providing an enhanced degree of future proofing through centralized software updates as services evolve.

8) Crypto as a Service (CaaS) :

The crypto as a service CaaS is used to offer services for VNF clients and same goes for network functions and encrypted traffics for apps.

9) Network Slicing:

Deploying and operating a network slice follows different steps depending if the definition of the network slice is based on existing resources or requires to provision new physical or virtual resources. This use case focuses on network slices across PNF and VNF within a single operator's domain.

10) Virtualization of Internet of Things (IoT):

The IoT applications include a large set of service types. Efficient delivery of services may require the deployment of IoT-related functions over NFV domains.

11) Rapid Service Deployment:

By deploying services using software components (VNFs) on a common NFVI, the time to deploy a service instance is dramatically reduced compared to physical installation. If the service is internal to the operator, it may be triggered electronically through some workflow business support system. If the service is customer facing, it may be triggered through a customer facing portal.

12) Devops/CI/CD:

By deploying services using software components (VNFs) on a common NFVI, the software development and upgrade processes permit rapid innovations through updates of the various components required for a network service. This assumes software is developed with an agile methodology driven by user stories and characterized by short development sprints delivering new versions of VNFs in a matter of weeks. Network services comprised of continuous testing, integration and deployment is then used to enable the latest service versions to be available.

13) A/B testing:

With more rapid software deployment, it becomes feasible to test the performance of alternative approaches in both sandboxed and live service environments. VNF alternatives could be VNF updates, VNF substitutes, etc. Network service updates could use alternate paths, VNFs, configurations etc. Using A/B testing approaches the performance of the alternatives could determine which of the variants is the better.

14) VNF composition across multiple administrative domains:

This use case considers the orchestration of VNFs in different administrative domains for serving a specific customer request. Certain access, control and management capabilities for the VNFs in a remote domain are granted to the local provider facing the customer in order to compose and end-to-end service, with the customer not being aware of the multi-domain nature of the service, and the local provider operating it as if it was formed completely by own resources.

15) Security as a Service (SecaaS):

To build enhanced SecaaS services, leveraging NFV technologies. Using this SecaaS model, the complexity of the security analysis can be hidden from ISP customers, to make deploying and manipulating specialized security equipment easier. [12]

Figure 1.8 summarizes this use case.

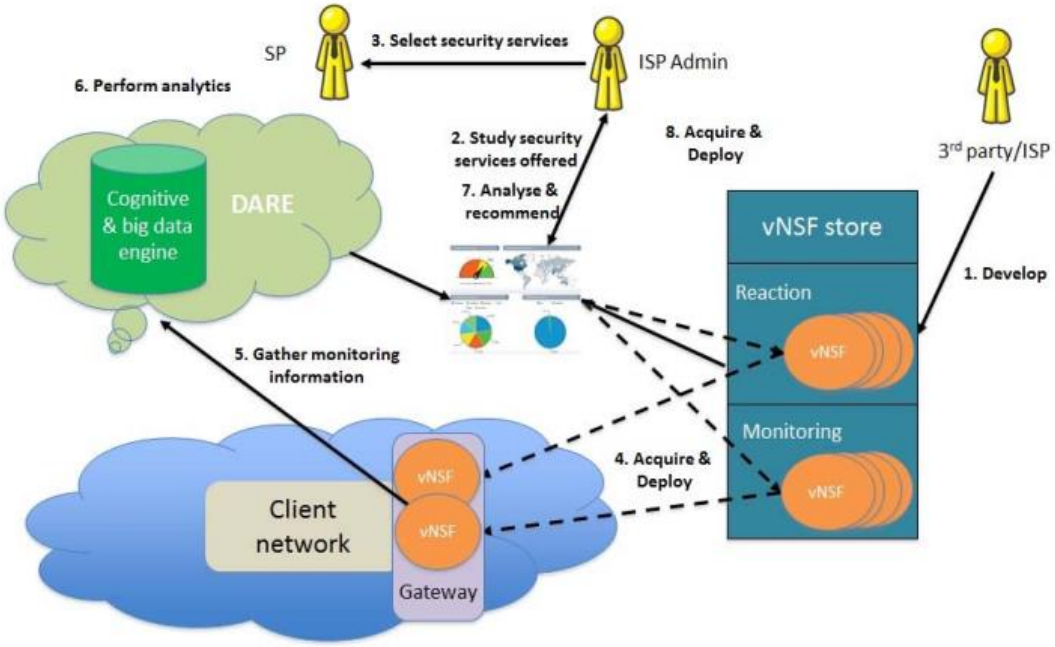

Figure1. 8: SecaaS based on NFV [12]

7. Conclusion

At the level of this chapter, we presented the basic concepts of network functions virtualization (NFV) including terminologies, history, architecture and their use cases. This led us to conclude that Network functions virtualization (NFV) is a fast-emerging technology area that is heavily influencing the world of networking.

CHAPTER 02

VIRTUAL NETWORK FUNCTIONS PLACEMENT PROBLEM

1. Introduction

A virtualized network function, or VNF, may consist of one or more virtual machines running different software and processes. Consequently, they are run on top of standard high-volume servers, switches, and storage, or even cloud computing infrastructure, instead of having custom hardware appliances for each network. On other hand, The Virtual Network Functions (VNFs) are deployed to satisfy the service requirements of flows. The requirements of a VNF are like to those of a VM in terms of computing and storage resources, i.e both demand CPU, RAM and disk.

In this chapter, we present the most of related works of Virtual Network Function (VNF) placement, then, a summary of state-of-the-art VNF placement, after that we give up the problem formulation system model and finally proposed VNF Placement Algorithm.

2. Related work

Regarding the VNF placement and routing problem or service chaining:

Mouhamad Dieye et al [13]

The placement and chaining of VNFs affect the desired QoS of value-added service and the cost of the CDN provider. In [13] the authors formulated this problem as an Integer Linear Program (ILP) and propose a cost-efficient Proactive VNF placement and chaining (CPVNF) algorithm. The objective is to be find the optimal number of VNFs along with their locations in such a manner that the cost is minimized while QoS is met.

Leivadeas et al [14]

In [14] the authors studied the problem of SC allocation in the MEC and cloud layers. To overcome this problem, they proposed two algorithms: i) an optimal solution formulated as a Mixed Integer Programming (MIP) problem and ii) a Tabu Search metaheuristic. The superior performance of the proposed solution was validated through the comparison with an existent algorithm that already found in the literature. The overall goal was to reconcile the conflicting goals of reduced deployment cost and latency.

Agarwal et al [15]

In [15] the authors focus on the allocation of computational and network resources, accounting for (i) the requirements of each VNF and vertical; (ii) the capabilities of the network operator's infrastructure; (iii) the capacity and latency of the links between network nodes. Thus, the authors propose a methodology to make the requirement decisions jointly and effectively, based on two pillars: a VNF placement heuristics called MaxZ, and a convex formulation of the CPU allocation problem given placement decisions.

Addis et al [16]

The authors [16] resolved the problem of VNF service chain placement using a mixed integer linear programming formulation (MILP) and provide insights into trade-offs between legacy and NFV-based Traffic Engineering, as well as, minimizing the cost of used links and network nodes, minimizing resource utilization subject to QoS requirements.

Khebbache et al [17]

In [17] focused on the placement of Virtualized Network Functions (VNFs) chains in the Network Function Virtualization (NFV) context where NFV Infrastructures (NFVIs) are used to host the VNFs. The authors propose a matrix-based optimization and a multi-stage graph method that achieved an efficient cost and improved scalability by finding solutions as polynomial times.

Moens et al [18]

The authors present an Integer Linear Programming (ILP) model to minimize the number of used servers (or compute resources) for resource allocation of virtualized network functions within NFV environments, a problem referred to as Virtual Network Function Placement (VNF-P). In order to figure out the larger instances issue, the authors are focused on a hybrid scenario Where a part of the services may be provided by dedicated physical hardware, and the other part by a virtualized one.

Luizelli et al [19]

In [19] authors formalized the network function placement, chaining problem and proposed a heuristic for efficiently guiding an Integer Linear Programming (ILP) model. They perform a binary search to find the lowest number of possible network functions instances that meet the current demand. Then, they run the ILP on this more constrained problem which allows reducing the ILP computational time.

Bari et al. [20]

In [20], the authors reveal an Integer Linear Programming (ILP) model for VNF orchestration. The situation is made up to find the number of required VNFs and allocating them to able to reduce the total network pertaining cost and the resources segmentation. In order to resolve bigger problems, the authors propose a dynamic programming-based heuristic.

Xing et al [21]

In [21] the authors studied the virtual network function placement (VNF-P) problem. The objective is to minimize the end-to-end SFC delay, with the compute, storage, I/O and bandwidth resources reasonably consumed.

To tackle this problem, the authors improve the basic grey wolf optimizer (GWO) by introducing an integer encoding method and a novel wolf position update mechanism, namely the integer encoding GWO (IEGWO).

Mohammadkhan et al [22]

The authors formulate the problem of network function placement and routing as a mixed integer linear programming problem (MILP) to place services optimally for flows and minimize network resource consumption.

3. Synthesis

The Table 2.1 sums up the most of relevant works that are closely related to VNF placement.

Entitled	Year	Problem	Objective	Methods	Tools	Simulator	Traffics?	Multi-clouds ?
Mobile operators and content providers in next-generation SDN/NFV core networks: Between cooperation and competition	2017	VNF placement and traffic steering problems	Minimizing the total cost	Online algorithm	-	CPLEX	Yes	-
Optimal virtual network function placement in multi-cloud service function chaining architecture	2017	Deploying service function chains over network function virtualized architecture	Reducing the total delays to the end users and total cost of deployment for service providers in inter-cloud environments.	Make a comparison between the proposed heuristic and simple greedy approach (SGA) used in the state-of-the-art systems	-	GUESS	Yes	Yes
On service-chaining strategies using Virtual Network Functions in operator networks	2018	VNF service-chain placement and traffic routing problem	Minimizing network-resource consumption	Integer Linear Programming, ILP	-	CPLEX	Yes	Yes
A fix-and-optimize approach for efficient and large-scale virtual network function placement and chaining	2016	VNF placement and chaining	Minimize required resource allocation. (Minimizing energy)	A fix-and-optimize-based heuristic algorithm	Java	CPLEX	Yes	-

Entitled	Year	Problem	Objective	Methods	Tools	Simulator	Traffics?	Multi-clouds ?
Joint optimization of function mapping and preemptive scheduling for service chains in network function virtualization	2017	Function placement and scheduling	To minimize the total completion time	Integer Linear Programming, ILP	-	Gurobi Optimizer	-	-
Low-latency orchestration for workflow-oriented service function chain in edge computing	2018	Deployment of Service Function Chaining (SFC)	Reducing the latency for provisioning the workflow-like service request	Design DMRT_SL and DMRT_NSL algorithms to efficiently solve the studied problem	Java	-	Yes	-
Toward Superfluid Deployment of Virtual Functions: Exploiting Mobile Edge Computing for Video Streaming	2017	Deployment of Virtual Functions	Optimizing the network efficiency and/or the performances	Exploiting Mobile Edge Computing for Video Streaming	-	OpenStack Sys: Ubuntu 16.04.1	Yes	-
MOSC: a method to assign the outsourcing of service function chain across multiple clouds	2018	Service function chain outsourcing	Minimizing the operational cost	Formulate the service function chain outsourcing problem as an ILP model and propose deviation algorithm based MOSC,	Python	runs on a server with Intel Xeon E5-2620 CPU, 32 GB memory.	-	Yes
Scalable and coordinated allocation of service function chains	2016	NFV resource allocation	Minimize energy	Proposing CoordVNF, a heuristic method to coordinate the composition of VNF chains	-	ALEVIN simulation framework	Yes	-

Entitled	Year	Problem	Objective	Methods	Tools	Simulator	Traffics?	Multi-clouds ?
Efficient virtual network function placement strategies for Cloud Radio Access Networks	2018	VF placement over distributed virtual resources spread across multiple clouds	Minimize the overall costs	Proposing a combinatorial optimization model and the use of two heuristic approaches, which are, branch-and-bound (BnB) and simulated annealing (SA) for the proposed optimal placement	-	NS3	-	Yes
VNF Placement for service chaining in a distributed cloud environment with multiple stakeholders	2018	Optimal locations for a chain of VNFs	Reducing the costs	Proposing an integer linear program (ILP) and design two heuristic algorithms	Matlab	CPLEX	Yes	Yes
Optimal virtualized network function allocation for an SDN enabled cloud	2017	VNF placement	Minimizing the costs	Evaluate a set of algorithms: * Mixed integer programming formulation * Baseline approach * Load balancing approach * Worst Performance Approach	Java-based simulator	Simulator for Controlling Virtual Infrastructures (CVI-Sim)	Yes	-

Entitled	Year	Problem	Objective	Methods	Tools	Simulator	Traffics?	Multi-clouds ?
A fast-robust optimization-based heuristic for the deployment of green virtual network functions	2017	Robust VNF placement and network embedding	* Minimizing the power consumption of their NFV infrastructure * Minimizing the resulting traffic	Apply the theory of robust optimization	Matlab	CPLEX	Yes	-
Towards load-balanced service chaining by Hash-based Traffic Steering on Softswitches	2018	Loading a balancing system for chaining VNFs in a data center environment	Addressing hash collision issues, to ensure the efficient system performance	The flow-hashing technique applied to soft switches to carry out server and network load balancing without triggering the controller. By exploiting the advantages of HATS , derive two algorithms, HATS with Flowcell-based Multipathing (HATS-Flowcell) and Dynamic Weight Adjustment for HATS (D-HATS)	-	- Mininet network emulator - vSwitch 2.5.0, and ODLBeryllium SR2	Yes	-
Latency-aware cost optimization of the service infrastructure placement in 5G networks	2018	The service infrastructure placement	Minimization of costs	Mathematically models the placement problem in a Fog Computing/NFV environment as a Mixed-Integer Linear -	Python	-	Yes	-

Entitled	Year	Problem	Objective	Methods	Tools	Simulator	Traffics?	Multi-clouds ?
				programming problem and proposes a heuristic-based solution considering 5G mobile network requirements.				
Virtualized network functions chaining and routing algorithms	2017	VNF Chain Placement	Minimizing the energy consumption	Propose a matrix-based optimization and a multi-stage graph method that are cost-efficient and improves scalability by finding solutions in polynomial times	C++	CPLEX	Yes	-
On the energy cost of robustness for green virtual network function placement in 5G virtualized infrastructures	2017	How to find a robust strategy that place virtual network functions (VNF) inside vDC impact the energy savings and the protection level against resource demand uncertainty	Minimize the energy consumption and costs	Proposing a heuristic that can be used to warm-start the solution process of the solver, accelerating the convergence towards the optimum	-	CPLEX	Yes	-
Optimal orchestration of virtual network functions	2018	VNF placement and routing (VNF-PR)	Cost optimization	Proposing a versatile linear programming formulation	C++	CPLEX	Yes	-

Entitled	Year	Problem	Objective	Methods	Tools	Simulator	Traffics?	Multi-clouds ?
Throughput optimization for admitting NFV-enabled requests in cloud networks	2018	The throughput maximization in a cloud network	Cost minimization	Approximation algorithms Online algorithms	-	-	Yes	-
On the complexity of a Virtual Network Function Placement and Routing problem	2018	VNF placement and routing problem	Service costs, energy consumption	study the complexity of the capacitated and the uncapacitated versions of the problem and the impact of network topology on the problem complexity	-	-	Yes	-
Hardware Acceleration Resource Allocation Mechanism for VNF	2018	Acceleration Resource Allocation Mechanism in VNF	Minimize energy	This paper adopted the algorithms based on Greedy and Tabu Search (TS) to complete the deployment of network functions, and uses the acceleration resource utilization as the performance evaluation indicator	Matlab	-	Yes	-
Network Function Virtualization in 5G	2016	The potential of NFV in enhancing 5G	Reducing the latency, cost, and achieve considerable energy savings	Describe NFV implementation with network overlay and SDN technologies	-	-	Yes	-

Entitled	Year	Problem	Objective	Methods	Tools	Simulator	Traffics?	Multi-clouds ?
SDNFV-based Dynamic Network Function Deployment: Model and Mechanism	2017	SDNFV-based Dynamic Network Function Deployment	* Improve the network resource utilization * Performance enhancement	Propose a dynamic network function deployment model,	-	ClickOS	Yes	-
On the optimal NFVI-PoP placement for SDN-enabled 5G networks	2017	NFVI-PoP placement	Minimize related capital and operational expenditures	Proposing an approach for the NFVI-PoP placement problem solution considering 5G mobile network requirements in a Fog Computing (FC) and Software Defined Networks (SDN) ecosystem.	-	-	Yes	-

Table 2. 1: Summary of state-of-the-art VNF placement

4. problem formulation system model

4.1 Service Chaining

The term “service-chaining” is used “to describe the deployment of such functions, and the network operator’s process of specifying an ordered list of service functions that should be applied to a deterministic set of traffic flows” [23] . Figure 2.1 depicets a set of chained VNFs

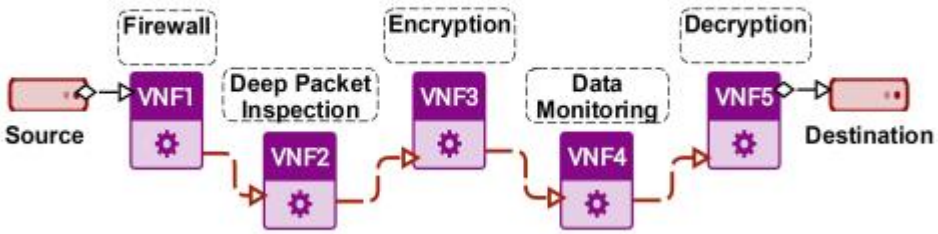

Figure 2. 1: Service Chain [24]

In NFV, to offer a network service is to place an ordered sequence of VNFs on one or more commodity servers. The ordered sequence of VNFs is also known as service function chain (SFC). [25]

Figure 2.2 shows an example of VNF-P. A target flow from node 1 to node 4 requests to receive a network service containing three ordered network functions, namely, VNF1, VNF2 and VNF3, where VNF1 =>VNF2 => VNF3 is an SFC.

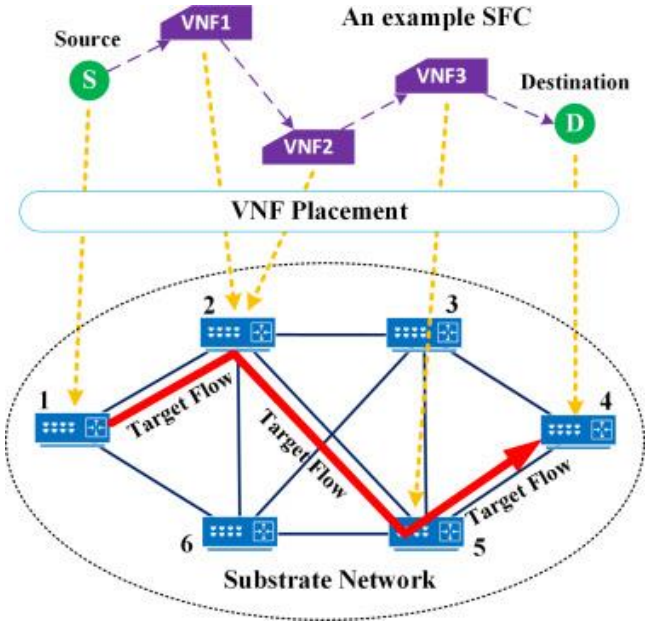

Figure 2. 2: An example SFC and its placement to a substrate network [25]

4.2 Problem Formulation

As shown in Figure2.3, the network topology is represented as an undirected weighted graph G(N, L), wherever N is that the set of network nodes and L is that the set of links. every link l ∈ L refers to a combine (i, j) with i, j ∈ N: i ≠ j. for every server s ∈ S, we have a tendency to represent by n(s) ∈ N the network node to that s is connected to. V is that the set of VNFCs we would like to position on the hardware resources of the VNI. C could be a family of sets representing the set of service chains. every C ∈ C is an ordered set of V that represent the sequence of VNFCs enclosed in an exceedingly service chain. Each C contains couples (v1, v2) with v1, v2 ∈ V and is connected with its own demands and latency bounds.

The objective of the problem is to find the optimal allocation of all the service chains on the physical servers and consequently, We seek to find the best placement of SFC in a way that it leads to optimize the number of servers , while satisfying the constraints on the server resources (CPU, RAM, DISC).

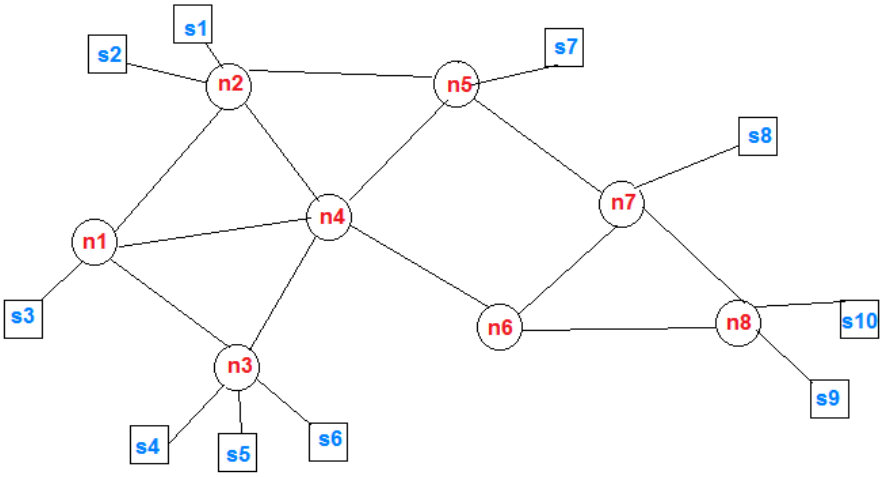

Figure 2. 3: Network topology

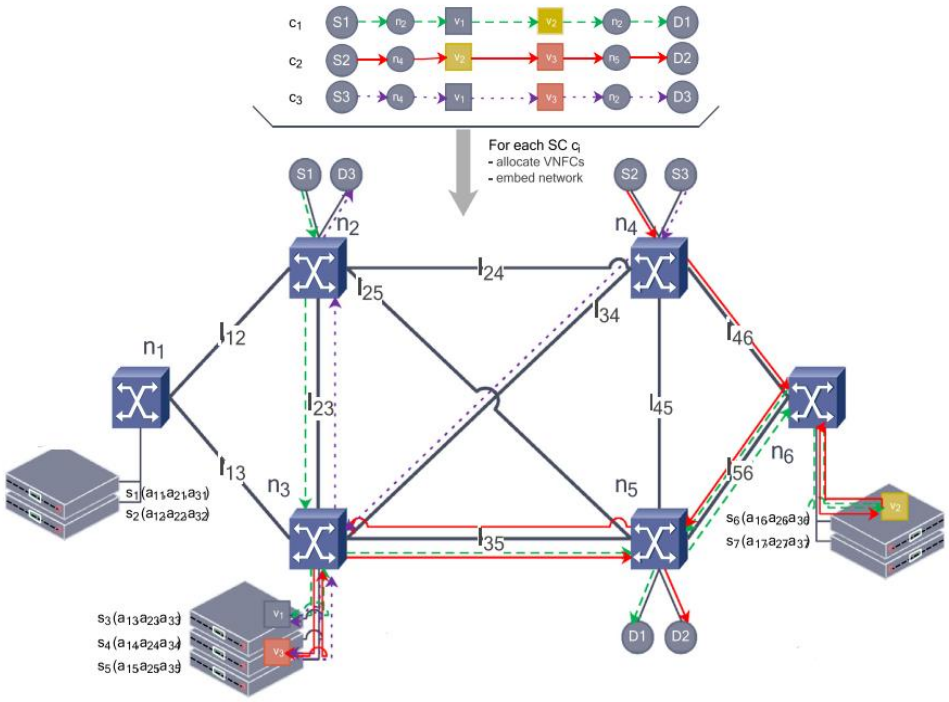

Figure 2. 4: The joint VNF placement and network embedding problem [26]

Figure 2.4 delineates the issue where we have in complete seven servers (s1 until s7), everyone with its own capacity profile individual CPU, memory and disc capacities. In the precedent given, server si has introduced a1i CPU, a2i RAM and a3i Disk. Every server is associated with a remarkable switch (for instance, s1 is associated with n1). Each link has a dedicated capacity and latency (for example, the latency for the link between n1 and n2 is

denoted as l12 we omit bandwidth from Figure 2.4 to maintain readability). The servers, their abilities, together with the network nodes and links with their abilities structure the NFV Infrastructure as far as Computing Power, Storage and Network. In this model, we ought to insert into this NFV Infrastructure three services chains (indicated as c1, c2 and c3), everyone with their own inactivity limits. Altogether, we have three distinctive VNFCs (v1, v2 and v3) and we accept that the traffic hotspot for c1 is the Sender S1, which is associated with switch n2 and infuses a specific volume of traffic into the administration chain towards v1. v1 forms the bundles (for which it needs CPU, memory and disc) and advances the prepared traffic (which may have an unexpected volume in comparison to the one infused) towards VNFC v2, which again forms it and advances a specific volume to the goal D1 that is associated with switch n2. The figure delineates a commendable VNF position and system installing into the physical substrate arrange. For instance, the VNFC v1 would be put onto server s3, v3 onto server s4, etc. Servers facilitating no VNFC would be controlled down (e.g. s1, s2 or s5) together with every one of the hubs not conveying any traffic (e.g. just n1 for this situation). [26]

In Table 2.2 all the parameters and the decision variable of the optimization problem are explained.

SYMBOLE	DESCRIPTION
G(N, L)	network graph (N, L are the set of nodes and links, respectively)
S	set of servers
V	set of VNFCs
C	set of service chains
R	set of resources
N (S)	is the network node to which server s is connected to
a_{rs}	is the amount of resource r available at server s
a_{vr}	is the amount of resource r requested by VNFC v
P_s	is the static power consumption of server s
P_s^{min} , P_s^{max}	are the idle and maximum power consumption of server s
l_{ij}	is the latency of the link (i, j)
DECISION VARIABLES:	
x_{vs}	is 1 if VNFC v is allocated to server s, 0 otherwise
y_s	is 1 if server s is active, 0 otherwise
z_n	is 1 if node n is active, 0 otherwise
g_{ij}	is 1 if the link (i, j) is used for transmitting any traffic
$x_{vs} \in \{0, 1\}$, $y_s \in \{0, 1\}$, $z_n \in \{0, 1\}$, $g_{ij} \in \{0, 1\}$	

Table 2. 2: Model parameters and decision variables according to [26]

$$\sum_{s \in S} x_{vs} = 1 \quad v \in V \qquad (1)$$

Constraint (1) ensures must be expresses that each VNFC must be allocated to exactly one server.

$$y_s \leq \sum_{v \in V} x_{vs} \quad s \in S \qquad (2)$$

Constraints (2) link the activation of a server and the allocation of a VNFC to it: if no VNFC is allocated to a server, then the server is not activated.

$$x_{vs} \leq y_s \quad s \in S, v \in V \qquad (3)$$

Constraints (3) introduce a further linking between the activation of a server and the allocation variables: if some VNFC is allocated to a server, then the server must be activated.

$$\sum_{v \in V} a_{vr} \cdot x_{vs} \leq a_{vr} \cdot y_s \quad s \in S, r \in R \qquad (4)$$

The constraints (4) present the resource capacity for every server and resource type, imposing that the entire usage of all VNFCs cannot override the capacity of every server.

$$g_{ij} \leq z_i \quad \text{and} \quad g_{ij} \leq z_j \quad (i,j) \in L \qquad (5)$$

The constraints (5) link the Boolean status of link activation variables to the status of the node activation variables: if a link is used, then its end-nodes must be activated; if a node is not activated, then a link ending in it cannot be used.

4.3 Server Power Model

An important aspect is that too many VNFs should not be allocated on the same server as we need to take into account the uncertainty in demand fluctuations of the components, in terms of compute resources. In order to derive such a model, we need to look into the power consumption models for the physical infrastructure in a data center. Most a part of the entire

VNI power consumption is due to physical servers that run the VNFCs. The servers power consumption depends on many factors like CPU load, memory and Disk. As stressed by many works within the literature, the foremost prestigious system on the server's power consumption is that the CPU. once a server is powered on however not experiencing any load, it consumes P_s^{min} . Hence, the power consumption will be simplified as: [27]

$$P_s(t) = P_s^{min} + \left(P_s^{max} - P_s^{min}\right).used_s^{cpu}(t) \qquad (6)$$

where $used_s^{cpu}$ is the usage (in percentage) of the processor (value between 0 and 1). At each time instant the power consumption is linearly increasing with the CPU utilization, due to the running tasks on the physical machine. If the physical server does not run any VNF, we power down that server.

4.4 Proposed VNF Placement Algorithm

Most of the heuristics for the placement of VMs or VNFs are based on a greedy approach, and BPP(Bin Packing Problem) heuristics are often investigated to obtain suitable algorithms for the placement, such as in [28]. In this section, we propose a method based on a greedy heuristic algorithm.

When allocating a VNF over a server, that server must have enough resources to answer the needs of all hosted VNFs. To resolve this problem, our proposed scheme is consisted of two main phases. The first phase estimates the resources available on the physical nodes and the resources requested by the VNFs. The second phase runs a greedy algorithm that takes as input the VNF requests as they arrive, and then it places each VNF on a server that have enough resources. The following Algorithm 1 shows the functionality of the heuristic approach.

Algorithm 1 Greedy heuristic procedure

Input: files of data (list VNFs, SFC, needs of VNFs, capacity of servers)
Output: placement of VNFs
extract data from files
sort vnfList in the order of increasing CPU utilization ;
placement ← ∅
for all r ∈ R | dr > 0 **do** ►Assignment of requests
    if ∃sˆ ←SELECTSERVER(dr, S, split) **then**
        create VNF fr if it does not exists in placement
        assign request r to server sˆ in placement
        demand dr is decreased
    else
        return infeasible
    end if
end for
do ►Add slaves
for all VNFs v ∈ placement do
    S^{-} ← servers of S without v and its slaves
    if ∃sˆ ←SELECTSERVER(dv, S^{-}, FALSE) **then**
        create slave of VNF v on server sˆ in placement
    end if
end for
while slaves are found
  return placement

Table 2. 3: Greedy heuristic procedure according to [29]

The greedy heuristic algorithm which is illustrated in Table 2.3 mainly runs in two steps. Firstly, the vnfList are sorted in the order of increasing CPU utilization, and a near-optimal VNF placement is obtained by running the greedy strategy, which prioritizes the server with maximum available resources. The selection of the server is performed by the procedure SELECTSERVER(S^{-}, d^{-}, split) which discards the servers without sufficient resources to satisfy demand d^{-}, and selects a server from nodes depending latency between nodes and depending on the chosen policy: the server whose capacity best fits the demand and the first server found and the server with the highest availability.

5. Conclusion

A survey of the previous work and the literature in this domain for the virtualized network functions has been made. The problem solution or the VNF placement aspect of the VNF remains unexplored and has motivated the present work. The work on the VNFs using Greedy heuristic method has also been described.

CHAPTER 03

IMPLEMENTATION AND RESULTS

1. Introduction

The objective of the VNF placement problem is to find the optimal allocation of all the service chains on the physical servers that leads to optimize the number of servers while satisfying the constraints on the server resources (CPU, RAM, DISC).in this chapter, we will perform experiment with the algorithm presented in the past chapter, and we will present the work environment, values, and the results.

2. Language and development environment

Implementation and testing in an environment have the characteristics following:

• A machine with an Intel (R) Cor (TM) processor i3-CPU speed 1.7 GH a 4GB memory capacity. The PyCharm IDE on Windows 7 64-bit.

• The JetBrains PyCharm Community Edition 2018.3.4 version with language Python.

• Python programming language.

2.1. The Python programming language

Python is a widely used high-level, general-purpose, interpreted, dynamic programming language. Its design philosophy emphasizes code readability, and its syntax allows programmers to express concepts in fewer lines of code than would be possible in languages such as C++ or Java. The language provides constructs intended to enable clear programs on both a small and large scale.

Python supports multiple programming paradigms, including object-oriented, imperative and functional programming or procedural styles. It features a dynamic type system and automatic memory management and has a large and comprehensive standard library. Python interpreters are available for installation on many operating systems, allowing Python code execution on a wide variety of systems. [30]

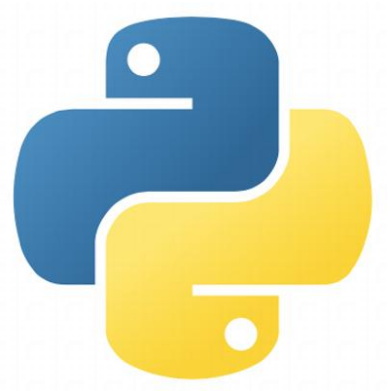

Figure 3. 1: python Language Logo [30]

❖ Why Do People Use Python **?**

The following primary factors cited by Python users seem to be these:

- Python is object-oriented
  - o Structure supports such concepts as polymorphism, operation overloading, and multiple inheritance.
- Indentation
  - o Indentation is one of the greatest future in Python.
- It's free (open source)
  - o Downloading and installing Python is free and easy
  - o Source code is easily accessible
- It's powerful
  - o Dynamic typing
  - o Built-in types and tools
  - o Library utilities
  - o Third party utilities (e.g. Numeric, NumPy, SciPy)
  - o Automatic memory management
- It's portable
  - o Python runs virtually every major platform used today
  - o As long as you have a compatible Python interpreter installed, Python programs will run in exactly the same manner, irrespective of platform**.**

2.2. IDE PyCharm

PyCharm is a dedicated Python and Django IDE providing a wide range of essential tools for Python developers, tightly integrated together to create a convenient environment for productive Python development and Web development.

With PyCharm you can develop applications in Python. In addition, in Professional Edition applications, Django, Flask, and Pyramid applications can be developed. It also fully supports HTML (including HTML5), CSS, JavaScript, and XML: these languages are grouped into the IDE via plug-ins and run by default. You can also add support for other languages and frames via plugins.

PyCharm is available in three editions: Professional, Community, and Educational (Edu). The Community and Edu editions are open-source projects and they are free, but they have less features. PyCharm is available for Windows, Mac OS, and Linux and can be expanded using dozens of plugins and integrations. [31]

3. Implementation

We present in Figure 3.2 process flow diagram for our algorithm proposed.

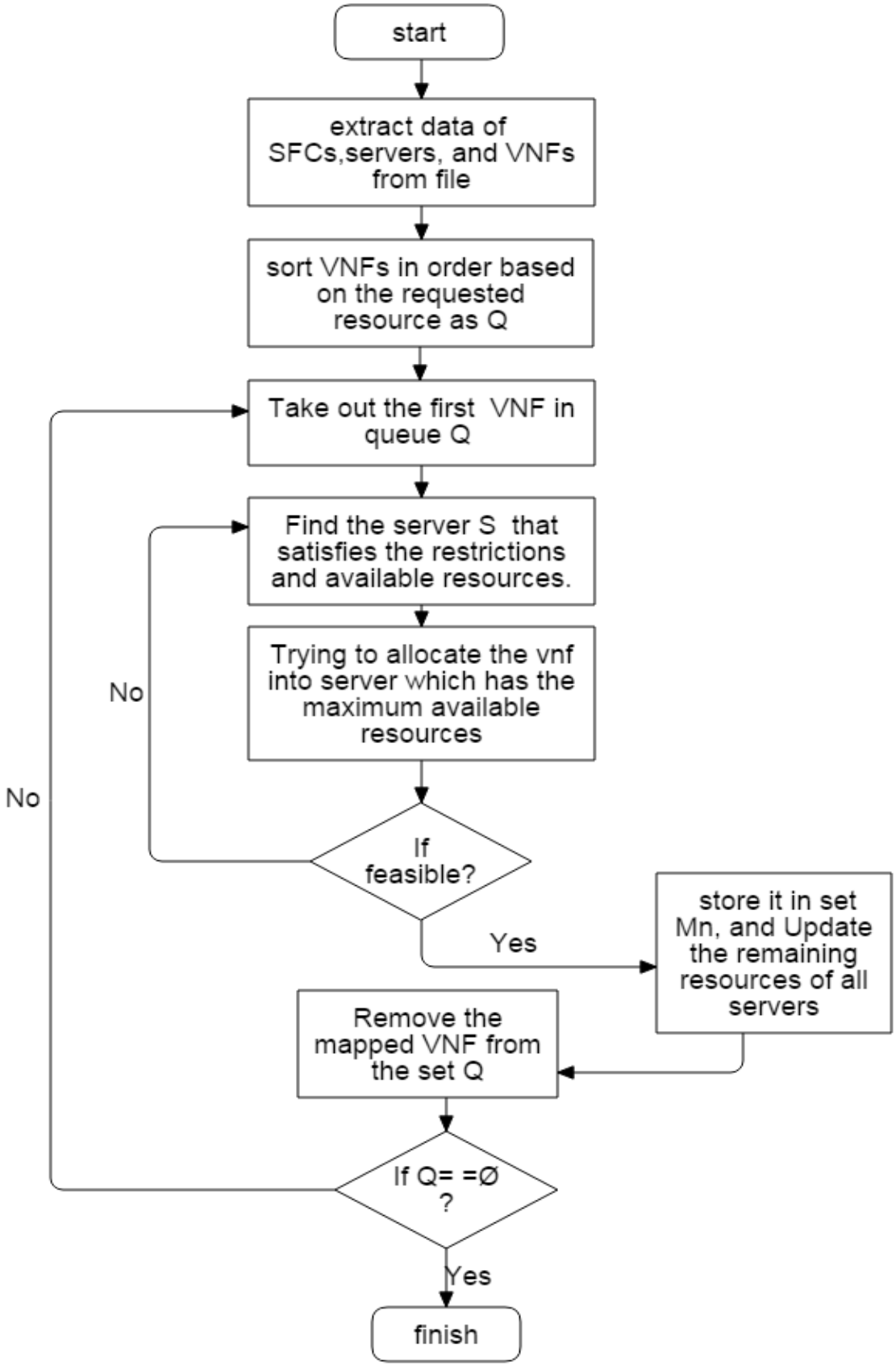

Figure 3. 2: Flow diagram

Q: the order of vnfs queue.

Mn: the set of vnfs which has allocated in servers successfully.

We save data of physical servers, VNFs, and SFCs in text documents then we extract information from files by functions as:

```
open("myfile.txt").read().split()
```

Figure 3. 3: The function of extract data of VNFs and servers from file

```
def Extact_Data_Servi(filepath):
    with open(filepath, "r") as f:
        l = [[int(num) for num in line.split(' ')] for line in f if line.strip() != ""]
        return l

service_01 = Extact_Data_Servi('1.txt')
service_02 = Extact_Data_Servi('2.txt')
service_03 = Extact_Data_Servi('3.txt')
service_04 = Extact_Data_Servi('4.txt')
```

Figure 3. 4: The function of extract SFCs from files

```
def Reservation(tab,val,tt):

    v = val
    for i in range(0, len(tab)):
        if np.any((v - tab[i]) >= 0):
            v = v - tab[i]
            tt.append(tab[i])
    return v
```

Figure 3. 5: Function Reservation

The function which is seen in Figure 3.5 it's responsible for booking the vnfs needs from server's capacity that is available.

The available servers that we want to allocate a set of VNFCs in, each connected to different routers in the network. The class **select_router** as shown in Figure 3.6 is responsible for specifying the routers that contain these servers.

```
class select_router:
    gg = [[None , None , 5, None, None, None, None , 1],
        [None, 0, None, None, None, None, None , None],
        [None, None, None, None,3, None, None, None],
        [None, None, 7, None, None, 9, None, None],
        [None, None, None, None, None, 5, 6, None],
        [None, 8, None, None, None, None, 2, None],
        [None, 4, None, None, None, None, None, None],
        [None, None, None,6, None, None, None, None],]
    tab=[];tab.append(0)
    line_1=gg[0]
    v1=line_1.index(min(x for x in line_1 if x is not None))
    tab.append(v1)
    line_2=gg[v1]
    v2=line_2.index(min(x for x in line_2 if x is not None))
    if v2 in tab:
    else:
    line_3=gg[v2]
    v3=line_3.index(min(x for x in line_3 if x is not None))
    if v3 in tab:
    else:
    line_4=gg[v3]
    v4=line_4.index(min(x for x in line_4 if x is not None))
    if v4 in tab:
    else:
    line_5=gg[v4]
    v5=line_5.index(min(x for x in line_5 if x is not None))
    if v5 in tab:
    else:
    line_6=gg[v5]
    v6=line_6.index(min(x for x in line_6 if x is not None))
    if v6 in tab:
    else:
        tab.append(v6)
    line_7=gg[v6]
    v7=line_7.index(min(x for x in line_7 if x is not None))
    if v7 in tab:
    else:
```

Figure 3. 6: Class select_router

The services functions chaining are saved as a matrix in document files .we generate this matrix after we extracted as graphical view (a directed graphs) .The figures(Figure3.7, Figure3.8, Figure3.9, Figure3.10) show this graphs.

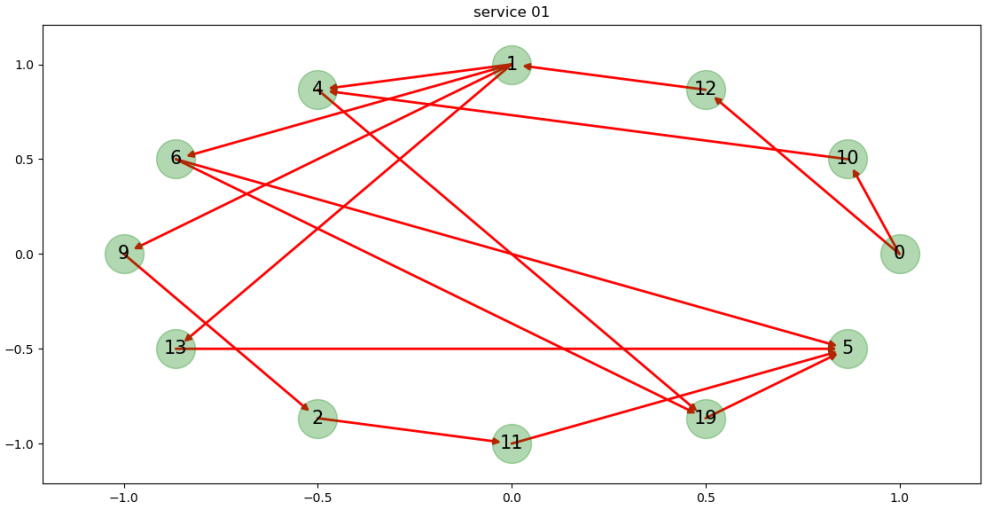

Figure 3. 7: Service function chain 01

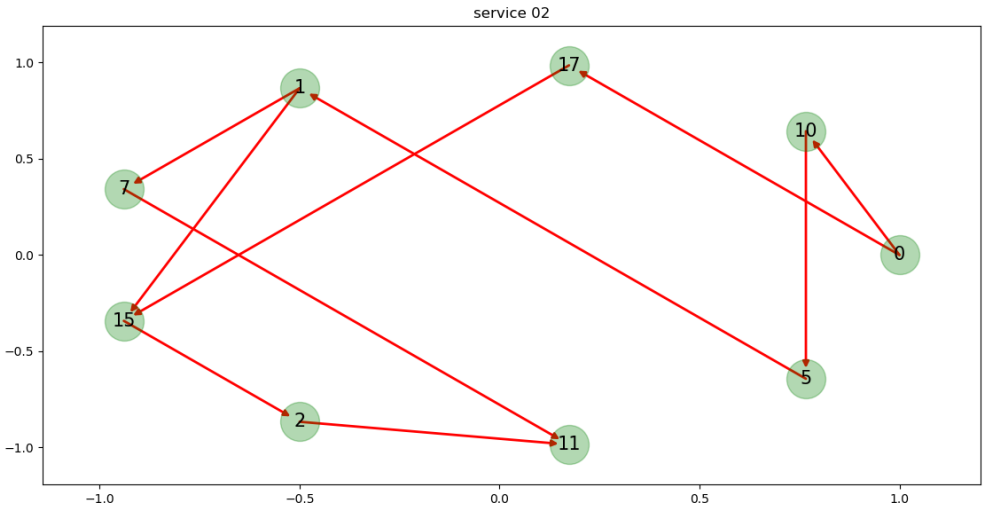

Figure 3. 8: Service function chain 02

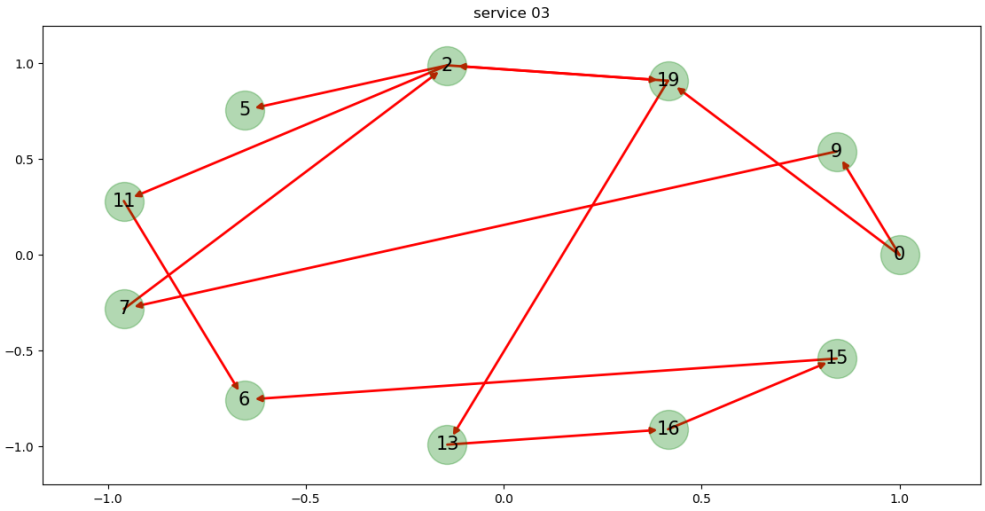

Figure 3. 9: Service function chain 03

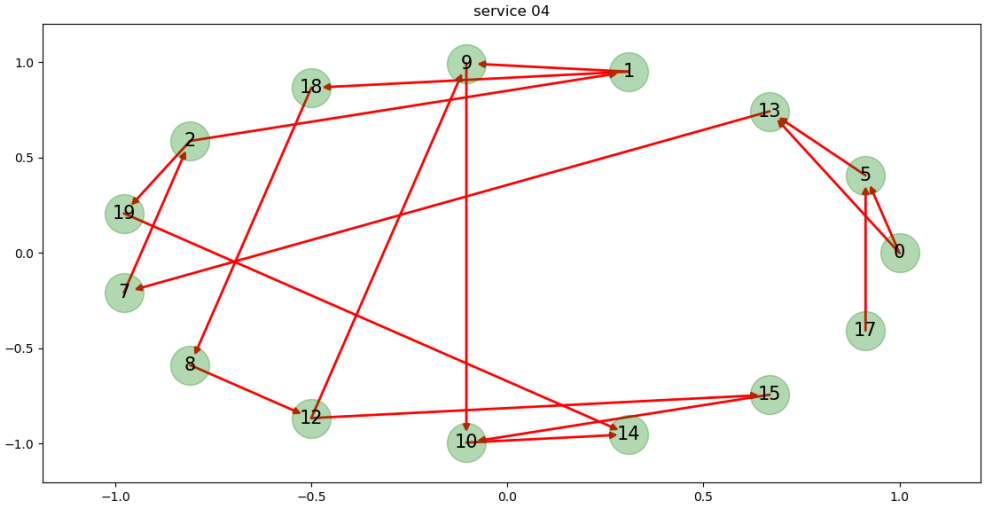

Figure 3. 10: Service function chain 04

4. Values

For the experimental evaluation, we considered the placement of service function chains, as discussed in the previous section.

For our use case we considered these values:

• 10 physical servers.

• 20 VNFs.

• 4 service chains.

The experimental values used for the servers and VNFs are shown in Table 3.1 and Figures (Figure3.11, Figure3.12).

Capacity of servers' CPU (GHz)	[5 , 20]
Capacity of servers' RAM_(Mb)	[10240 , 32222]
Capacity of servers' Disk (GB)	[60 , 250]
VNFs CPU (GHz)demands	[0.2 , 3]
VNFs RAM_(Mb)demands	[256 , 1024]
VNFs Disk (GB) demands	[5 , 22]
Server Idle Power Consumption (%)	20, 30, 20, 30, 40, 40, 30, 30, 30, 20
Server Max Power Consumption (W)	160, 160, 290, 270, 260, 220, 200, 200, 180,160

Table 3. 1: Experimental Values

```
[+] needs of all VNFs : ( CPU{GHZ}| RAM{MB} | Desk{GB} )

needs of VNF ( 0 ) : [0.2, 329.0, 10.0]
needs of VNF ( 1 ) : [0.7, 607.0, 13.0]
needs of VNF ( 2 ) : [0.5, 391.0, 12.0]
needs of VNF ( 3 ) : [0.3, 356.0, 19.0]
needs of VNF ( 4 ) : [0.9, 543.0, 8.0]
needs of VNF ( 5 ) : [1.2, 770.0, 5.0]
needs of VNF ( 6 ) : [0.23, 310.0, 9.0]
needs of VNF ( 7 ) : [1.4, 740.0, 11.0]
needs of VNF ( 8 ) : [1.7, 824.0, 23.0]
needs of VNF ( 9 ) : [0.4, 440.0, 15.0]
needs of VNF ( 10 ) : [0.8, 757.0, 7.0]
needs of VNF ( 11 ) : [0.25, 368.0, 18.0]
needs of VNF ( 12 ) : [0.6, 701.0, 14.0]
needs of VNF ( 13 ) : [0.37, 479.0, 8.0]
needs of VNF ( 14 ) : [1.5, 862.0, 22.0]
needs of VNF ( 15 ) : [0.28, 350.0, 6.0]
needs of VNF ( 16 ) : [1.38, 832.0, 17.0]
needs of VNF ( 17 ) : [1.1, 836.0, 16.0]
needs of VNF ( 18 ) : [2.5, 1006.0, 21.0]
needs of VNF ( 19 ) : [2.0, 902.0, 20.0]
```

Figure 3. 11: Needs of all VNFs

```
[+] Capacity of all servers : ( CPU{GHZ}| RAM{MB} | Desk{GB} )

capacity of server( 0 ) : [8.0, 18432.0, 150.0]
capacity of server( 1 ) : [5.0, 12288.0, 100.0]
capacity of server( 2 ) : [5.0, 10240.0, 80.0]
capacity of server( 3 ) : [12.0, 32222.0, 120.0]
capacity of server( 4 ) : [7.0, 32222.0, 250.0]
capacity of server( 5 ) : [15.0, 7168.0, 180.0]
capacity of server( 6 ) : [20.0, 25600.0, 200.0]
capacity of server( 7 ) : [10.0, 15360.0, 150.0]
capacity of server( 8 ) : [4.0, 18432.0, 60.0]
capacity of server( 9 ) : [5.0, 10241.0, 80.0]
```

Figure 3. 12: Capacity of all servers

5. The results

After we are implementing the algorithm presented in the past chapter, we got the following results:

- List of VNFs placement in physical servers (Figure 3.13).
- List of servers active (Figure 3.14).
- Total resources used in servers (Figure 3.16).
- Power consumption in the servers (Figure 3.17).

Figure 3.15 shows the servers and routers active as a graphical view.

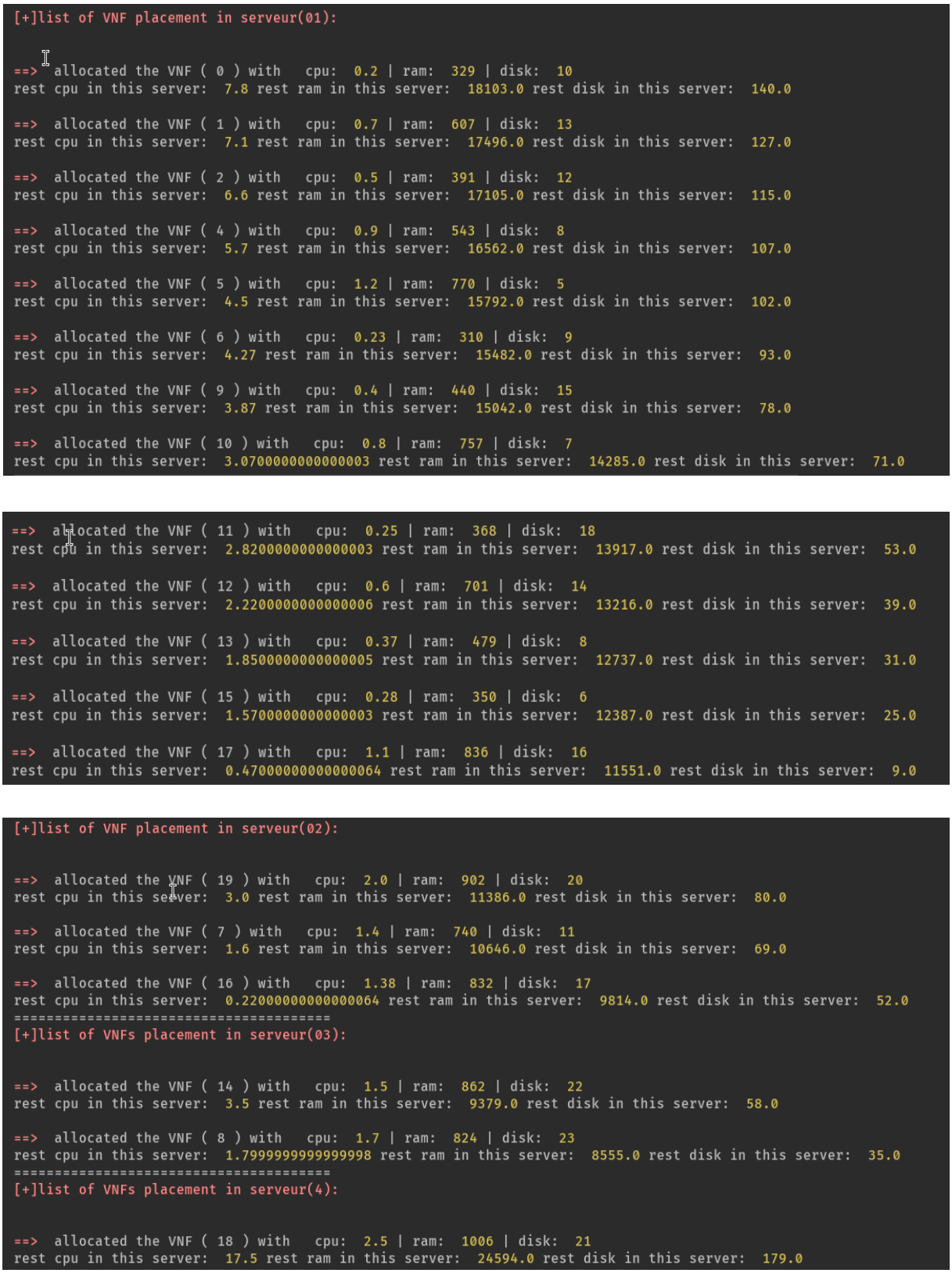

Figure 3. 13: Placement VNFs in servers

```
[+] List of servers active :
#  The server ( 10 )
#  The server ( 20 )
#  The server ( 17 )
#  The server ( 13 )
```

Figure 3. 14: List of servers active

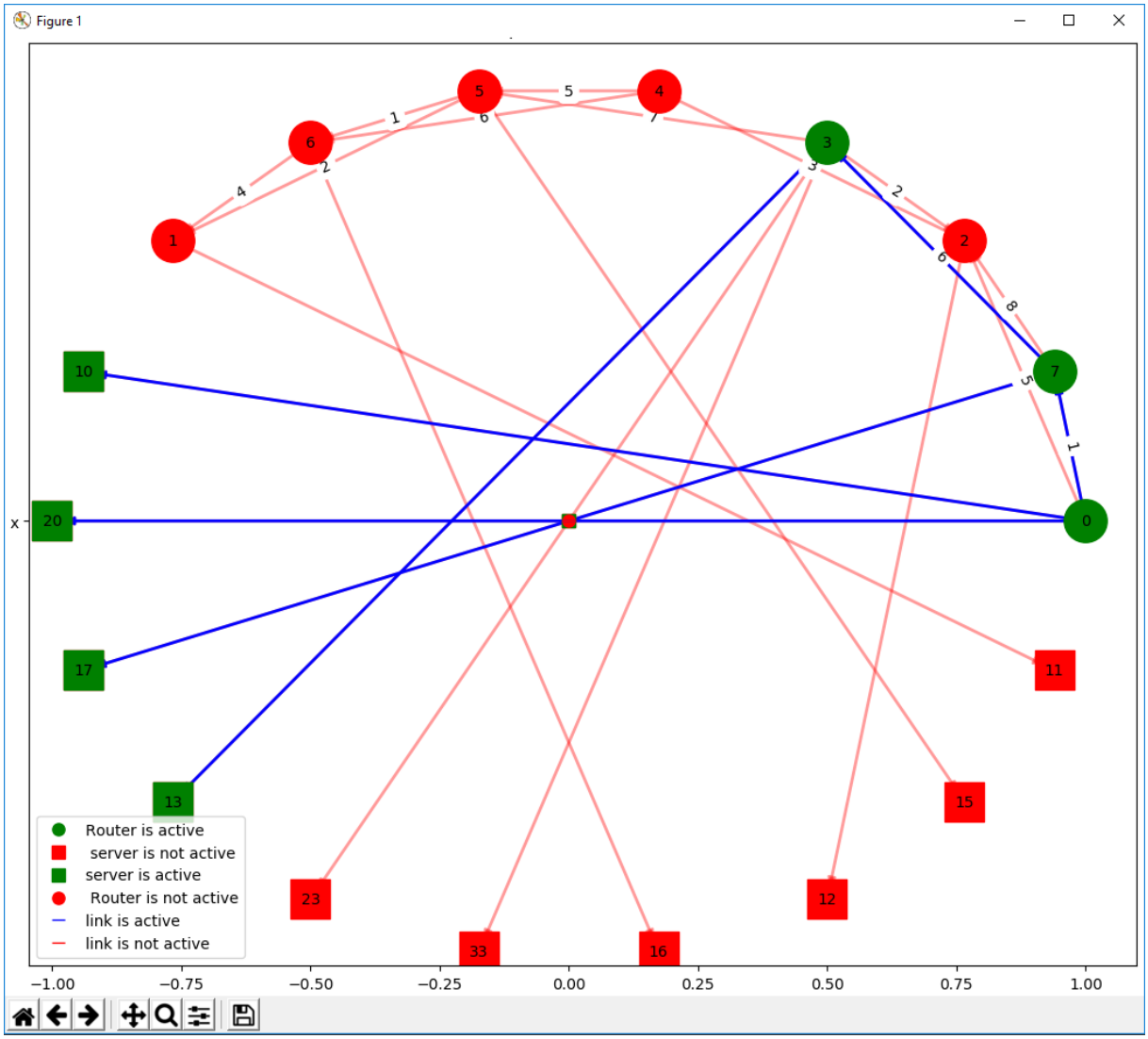

Figure 3. 15: Graph of servers active

```
[+] Total use of resources in servers active :

The total use of CPU in server (01) :  7.529999999999999
The total use of RAM in server (01) :  6881
The total use of DESK in server (01) :  141

The total use of CPU in server (02) :  4.779999999999999
The total use of RAM in server (02) :  2474
The total use of DESK in server (02) :  48

The total use of CPU in server (03) :  3.2
The total use of RAM in server (03) :  1686
The total use of DESK in server (03) :  45

The total use of CPU in server (04) :  2.5
The total use of RAM in server (04) :  1006
The total use of DESK in server (03) :  21
```

Figure 3. 16: The total use of resources in servers active

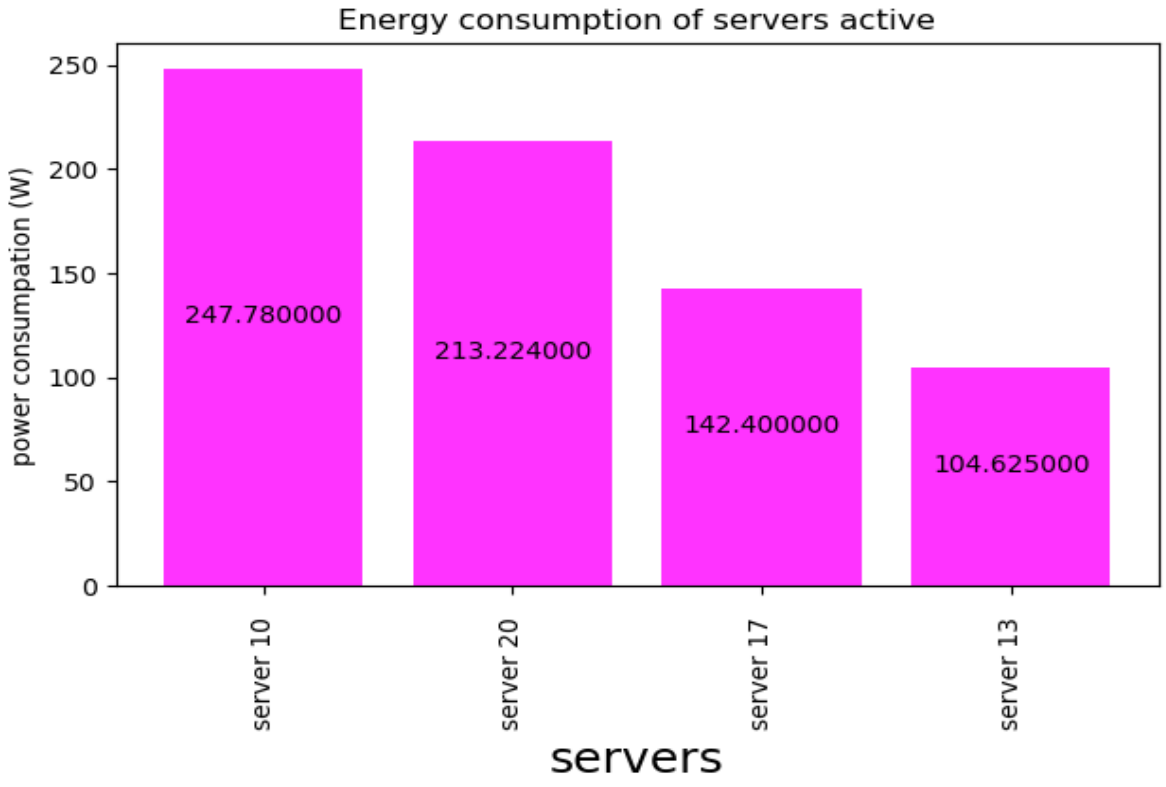

Figure 3. 17: Power consumption in the servers active

All results are saved in document file "Results.txt".

6. Conclusion

In this final chapter, we proceeded to the implementation of the proposed algorithm which is based greedy algorithm to find the best allocation placement of Virtual Network Functions (VNFs) that optimize network utilization and resource consumption. we presented the work environment and Python programming language, IDE of development PyCharm, values, and the results.

GENERAL CONCLUSION

Network functions virtualization (NFV) is transforming how networks are operated and how network services are delivered. By migrating NFs from dedicated hardware to virtualization platform, NFV can effectively improve the flexibility to deploy and manage service function chains (SFCs).

In this context, we have proposed a method based on a greedy heuristic algorithm which is typically very easy to describe and they have very fast running times. The contribution of our work is to place VNFs in a way that it leads to find the optimal allocation of the services function chains on the optimum number of physical servers while satisfying the constraints on the resources of the server, and so on enabling better use of network resources, minimizing the power consumption. These benefits also result in reduced operational and capital expenditure.

In this work, we have simulated and implemented the algorithm by the Python programming language and the experiments done on some of the servers which contain the different capacity of resources [CPU/RAM/DISK]. And after we insert input values, we got the following results:

- ✓ List of VNFs placement in physical servers.
- ✓ List of servers active.
- ✓ Total resources used in servers active.
- ✓ Power consumption in the servers active.

BIBLIOGRAPHICAL REFERENCES

[1] "NFV White Papers," European Telecommunications Standards Institute, Darmstadt-Germany, 2012.

[2] "Guide sur le Cloud Computing et les dataCenters à l'attention des collectivités locales," Juillet 2015.

[3] D. R. J. Shyam Patidar, "A Survey Paper on Cloud Computing," in *2012 Second International Conference on Advanced Computing & Communication Technologies*, 2012.

[4] M. P. Mauricio Arregoces, "Data Center Fundamentals," Cisco Press, 2004.

[5] B. Golden, Virtualization For Dummies, 3rd HP Special Edition, Indianapolis, Indiana: Wiley Publishing, Inc., 2011.

[6] "NFV White Papers," European Telecommunications Standards Institute, Darmstadt-Germany, 2012.

[7] "ETSI," European Telecommunications Standards Institute, Industry Specification Groups (ISG) - NFV, 2015. [Online]. Available: http://www.etsi.org/technologies-clusters/technologies/nfv. [Accessed 03 June 2015].

[8] "Technologies:Network Functions Virtualisation (NFV)," [Online]. Available: https://www.etsi.org/technologies/nfv. [Accessed 26 02 2019].

[9] "ETSI," Network function virtualisation-white paper1, SDN and openflow world, 2012. [Online]. Available: http://portal.etsi.org/NFV/NFV_White_Paper.pdf.

[10] X. W. ,. K. L. ,. S. k. D. ,. M. H. Bo Yi, "A comprehensive survey of Network Function Virtualization," *Computer Networks,* 2018.

[11] J. S. J.-L. G. N. B. F. D. T. R. B. Rashid Mijumbi, "Network Function Virtualization: State-of-the-art," *IEEE Communications Surveys & Tutorials,* 2015.

[12] "Network Functions Virtualisation (NFV);," European Telecommunications Standards Institute, 2017.

[13] S. A. J. S. E. A. R. G. Mouhamad Dieye, "CPVNF: Cost-Efficient Proactive VNF Placement and Chaining for Value-Added Services in Content Delivery Networks," *IEEE Transactions on Network and Service Management* , vol. 15, no. 2, 2018.

[14] A. Leivadeas, M. Falkner, I. Lambadaris, M. Ibnkahla and G. Kesidis, "Balancing Delay and Cost in Virtual Network Function Placement and Chaining," in *IEEE Conference on Network Softwarization and Workshops (NetSoft)*, Montreal, QC, Canada, 2018.

[15] S. Agarwal, F. Malandrino, C. F. Chiasserini and S. De, "VNF Placement and Resource Allocation for the Support of Vertical Services in 5G Networks," *IEEE/ACM Transactions on Networking* , vol. 27, no. 1, 2019.

[16] B. Addis, D. Belabed, M. Bouet and S. Secci, "Virtual network functions placement and routing optimization," in *IEEE 4th International Conference on Cloud Networking (CloudNet)*, Niagara Falls, ON, Canada, 2015.

[17] M. H. D. Z. Selma Khebbache, "Virtualized network functions chaining and routing algorithms," *Computer Networks*, vol. 114, 2017.

[18] H. Moens and F. D. Turck, "VNF-P: A model for efficient placement of virtualized network functions," in *10th International Conference on Network and Service Management (CNSM) and Workshop*, Rio de Janeiro, Brazil, 2014.

[19] M. C. Luizelli, L. R. Bays, L. S. Buriol, M. P. Barcellos and L. P. Gaspary, "Piecing together the NFV provisioning puzzle: Efficient placement and chaining of virtual network functions," in *IFIP/IEEE International Symposium on Integrated Network Management (IM)*, Ottawa, ON, Canada, 2015.

[20] M. F. Bari, S. R. Chowdhury, R. Ahmed and R. Boutaba, "On orchestrating virtual network functions," in *International Conference on Network and Service Management (CNSM)*, Barcelona, Spain, 2015.

[21] X. Z. X. W. S. L. P. D. K. L. H. Y. Huanlai Xing, "An integer encoding grey wolf optimizer for virtual network function," *Applied Soft Computing Journal*, vol. 76, pp. 575-594, 2018.

[22] A. Mohammadkhan, S. Ghapani, G. Liu and W. Zhang, "Virtual function placement and traffic steering in flexible and dynamic software defined networks," in *IEEE International Workshop on Local and Metropolitan Area Networks*, Beijing, China, 2015.

[23] "Network service chaining problem statement,," in *IETF*, 2013.

[24] J. G. Herrera and J. F. Botero, "Resource Allocation in NFV: A Comprehensive Survey," *IEEE Transactions on Network and Service Management* , vol. 13, no. 3, 2016.

[25] J. G. Herrera and J. F. Botero, "Resource allocation in NFV: a comprehensive survey," *IEEE Transactions on Network and Service Management* , vol. 13, no. 3, pp. 518 - 532, 2016.

[26] F. D. b. A. K. a. E. Z. Antonio Marotta a, "On the energy cost of robustness for green virtual network function placement in 5G virtualized infrastructures," *Computer Networks*, vol. 125, pp. Pages 64-75, 2017.

[27] A. K. Antonio Marotta, "A Power Efficient and Robust Virtual Network Functions Placement Problem," in *International Teletraffic Congress (ITC 28)*, Würzburg, Germany, Germany, 2016.

[28] P. Silva, C. Perez and F. Desprez, "Efficient Heuristics for Placing Large-Scale Distributed Applications on Multiple Clouds," in *IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (CCGrid)*, Cartagena, Colombia, 2016.

[29] P. F. ,. M. B. ,. S. Marco Casazza, "Securing virtual network function placement with high availability guarantees," in *IFIP Networking Conference (IFIP Networking) and Workshops*, Stockholm, Sweden, 2017.

[30] "tutorial," [Online]. Available: https://www.python.org/doc/. [Accessed 29 5 2019].

[31] [Online]. Available: https://www.jetbrains.com/help/pycharm/meet-pycharm.html. [Accessed 07 04 2019].

ملخص.المحاكاة الافتراضية لوظائف الشبكة (NFV) هي عبارة عن مفهوم جديد لشبكة الهندسة يعمل على محاكاة وظائف الشبكة مثل ترجمة عنوان الشبكة ، الجدران النارية ، التشفير ، خدمات أسماء النطاقات ، التخزين المؤقت ، ... إلخ ، بحيث يمكن دمجها في الخوادم ذات الأحجام الكبيرة القياسية ، والمفاتيح ، و التخزين ، والتي يمكن أن تكون في مراكز البيانات ، عقد الشبكة ، أو في أماكن المستخدم النهائي. في هذا السياق ، يتم استبدال الأجهزة الوسيطة التقليدية بأجهزة افتراضية تتضمن وظائف الشبكة الافتراضية (VNFs). يمكن ربط سلاسل VNF المختلفة مع بعضها البعض لتكوين سلاسل وظائف خدمة مختلفة (SFC) لخدمات شبكة مختلفة. وبالتالي ، بدأ مشغلو الشبكات في نشر وظائف الشبكة الافتراضية (VNFs) لتحقيق خدمات الشبكة. في هذه الرسالة ، نقترح نهجًا يستند إلى خوارزمية استرشادية جشعة لإيجاد أفضل موقع لـ SFC لتحسين عدد الخوادم المستضافة. وهذا يؤدي إلى تحسين استخدام موارد الشبكة وتقليل استهلاك الطاقة.

الكلمات المفتاحية : وظيفة الشبكة الافتراضية ، وظائف الشبكة الافتراضية، سلاسل وظائف الخدمة ، خوارزمية الجشع ، وضع وظائف الشبكة الافتراضية.

Abstract. Network Functions Virtualization (NFV) is a new network architecture concept virtualizes the network functions such as network address translation, firewalls, encryption, domain name services, caching, … etc. so they can be integrated onto industry standard high volume servers, switches, and storage, which could be in data centers, network nodes, or in the end-user premises. In this context, traditional middlebox appliances are replaced by virtual machines embedding Virtual Network Functions (VNFs). Different VNFs can be chained together to form different service functions chains (SFC) for different network services. Hence, network operators are starting to deploy VNFs to realize network services. In this dissertation, we propose an approach which is based on a greedy heuristic algorithm to find the best placement of SFC to optimize the number of servers hosted. That is leads to optimize network resources utilization and minimizing power consumption.

Keywords: Network Function Virtualization, Virtual Network Functions, service functions chains, greedy algorithm, Virtual Network Functions placement.

Résumé. La virtualisation des fonctions réseau (NFV) est un nouveau concept d'architecture réseau virtualise les fonctions du réseau telles que la traduction d'adresses réseau, pare-feu, cryptage, services de noms de domaine, la mise en cache, ... Etc. afin qu'ils puissent être intégrés dans les serveurs, les commutateurs et le stockage, qui pourraient se faire dans les centres de données, les nœuds réseau ou dans les locaux de l'utilisateur final. Dans ce contexte, les appareils middlebox traditionnels sont remplacés par des machines virtuelles intégrant des fonctions de réseau virtuel (VNF). Différents VNF peuvent être enchaînés pour former différentes chaînes de fonctions de service (SFC) pour différents services réseau. Par conséquent, les opérateurs de réseau commencent à déployer des VNF pour réaliser des services de réseau. Dans ce mémoire, nous proposons une approche basée sur un algorithme heuristique glouton pour trouver le meilleur placement de SFC pour optimiser le nombre de serveurs hébergés, qui permet d'optimiser l'utilisation des ressources réseau et de minimiser la consommation d'énergie.

Mots clés : Virtualisation des fonctions de réseau, fonctions de réseau virtuel, chaînes de fonctions de service, algorithme glouton, placement de fonctions de réseau virtuel.