State of the Art

2.1 Introduction

Software-Defined Networking (SDN) and Network Function Virtualization (NFV) are enabling network programmability and the automated provisioning of virtual networking services. Combining these new paradigms can overcome the limitations of traditional clouds and networks by enhancing their dynamic networking capabilities. Since these evolutions have motivated this thesis and our investigations, this chapter on the state of the art will provide an overview of NFV architecture, SDN, resource allocation challenges and reflect the convergence trend between cloud computing, software networks, and the virtualization of networking functions.

This chapter provides in the first part an overview of the NFV and SDN architectures. The convergence between cloud computing and both SDN and NFV is discussed in the second part. Finally, the third part describes the relation between the VNF placement and chaining problem and the traditional VNE problem and surveys some existing models and algorithms of resources mapping in network environments.

It is important to clarify that the two sub-problems of VNF-FG embedding: i) Initial VNF-FG placement, and ii) Elastic VNF-FG placement (Dynamic resource management) are investigated.

2.2 Network Function Virtualization (NFV)

2.2.1 Network Services Before NFV

Communication Service Providers (CSPs) go beyond simply providing network connectivity for their enterprise customers. They also offer additional services and network functions like Network address translation (NAT), Firewall, Encryption, Domain Name Service (DNS), Caching, etc. Traditionally, these network functions were deployed using proprietary hardware at the customer premises. This approach provides additional revenue but deploying multiple proprietary devices is costly and makes upgrades difficult (i.e., every time a new network function is added to a service, a truck roll is required to install the dedicated new hardware device). Consequently, service providers began exploring ways to reduce cost and accelerate deployments through Network Function Virtualization (NFV).

2.2.2 What is NFV?

Network Function Virtualization (NFV) [3], [9], [10] is an innovative emerging way to design, deploy, and manage networking services by decoupling functions (such as firewalls, DPIs, load balancers, etc.) from dedicated hardware and moving them to virtual servers.

Several use cases of NFV are discussed in [11]. Note that manageability, reliability, stability, and security are considered in [11] as the key performance parameters in both physical and in software based virtualized networks.

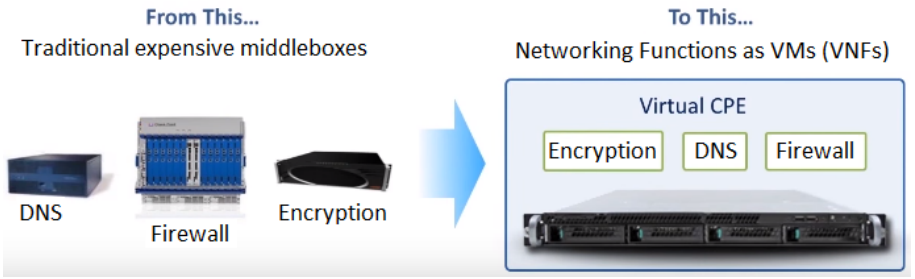

FIGURE 2.1: Network Function Virtualization

NFV provides a number of benefits to network operators, including:

- **Hardware Flexibility**: Because NFV uses regular Commercial-Off-The-Shelf (COTS) hardware, network operators have the freedom to choose and build the hardware in the most efficient way to suit their needs and requirements.

- **Faster Service Life Cycle**: New network services can now be deployed more quickly, in an on-demand and on-need basis, providing benefits for end users as well as the network providers.

- **Scalability and Elasticity**: New services and capacity-hungry applications keep network operators (especially cloud providers), on their toes to keep up with the fast-increasing demands of consumers.

- **Increased Revenue**: The combination of introducing new services faster and existing servers in a more dynamic fashion can jointly result in increased revenue.

- **Reduced Capital Expenditures (CAPEX)**: The use of industry-standard services, increased hardware utilization and adoption of open source software results in reduced capital expenditures.

- **Reduced Operational Expenditures (OPEX)**: Automation and hardware standardization can substantially slash operational expenditures.

- **Improved Customers' Satisfaction**: The combination of service agility and self-service can result in greater customer satisfaction.

- **Reduced Power Consumption and Complexity**: Efficiencies in space, power, and cooling. Communications Service Providers (CSPs) may have finite physical space, electrical power, and cooling capacity in a data center, so they will carefully select equipment to efficiently consume those finite and/or costly resources. NFV provides a better energy efficiency resulting from the consolidation of resources, as well as their more dynamic utilization.

2.2.3 NFV Architecture

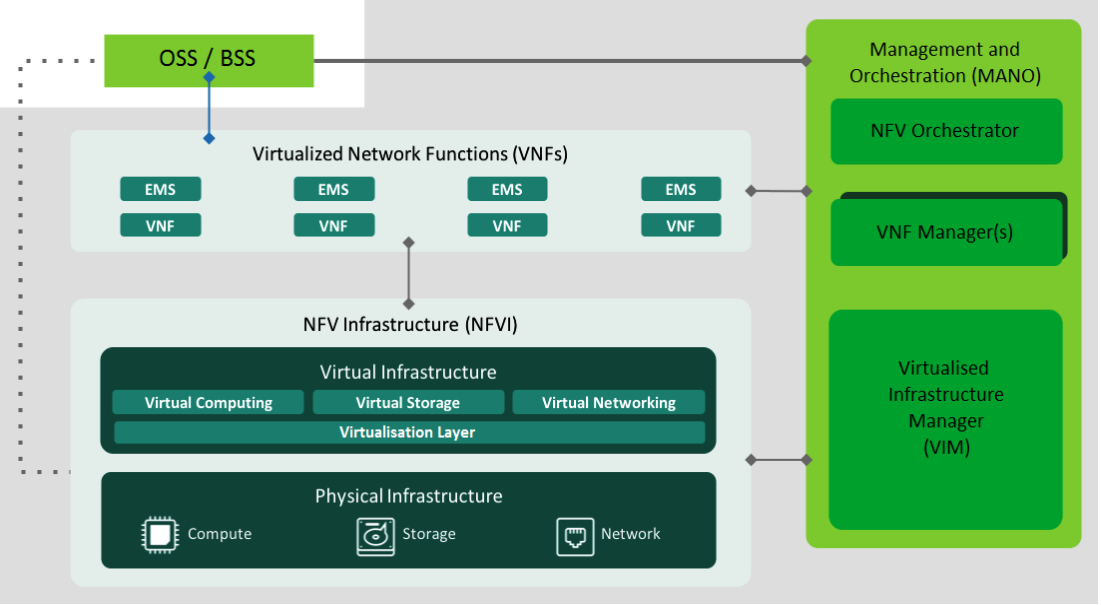

FIGURE 2.2: ETSI NFV reference architecture [2]

The main components of the NFV architectural framework are:

1. **NFV Infrastructure (NFVI)**: is a kind of cloud data center containing the totality of all hardware and software components that build up the NFV environment on which NFV services are deployed, managed and executed. NFVI includes:
   - *Physical Hardware*: this includes computing hardware (such as servers, RAM), storage hardware (such as disk storage, Network Attached Storage (NAS)), and network hardware (such as switches and routers).
   - *Virtualisation Layer*: abstracts the hardware resources and decouples the VNF software from the underlying hardware, thus ensuring a hardware independent lifecycle for the VNFs. We can use multiple open source and proprietary options for the virtualisation layer (such as KVM, QEMU, and VMware).
   - *Virtual Infrastructure*: this includes virtual compute (virtual machines or containers), virtual storage, and virtual networks (overlay networks).
2. **Virtualised Network Functions (VNFs)**: run on top of the NFVI and represent virtualized instances of different network functions. Each VNF has a corresponding

Element Management System (EMS) that provides management and control functionality for that VNF.

3. **NFV Management and Orchestration (MANO)**: NFV MANO does not act in isolation. It interacts with the Operational and Business Support Systems (OSS/BSS) components of the operator to manage the operational and business aspects of the network. MANO includes:

   - *Virtualized Infrastructure Manager (VIM)*: or cloud management software, e.g. OpenStack or Kubernetes. It is responsible for controlling and managing the computing, storage, and network resources, as well as their virtualization.
   - *VNF Manager(s)*: it is responsible for VNF life cycle management, including VNF instantiation, update, query, scaling, and termination.
   - *NFV Orchestrator*: it is in charge of the orchestration and management of NFV infrastructure and software resources, and realizing network services on NFVI. It utilizes resource allocation and placement algorithms to ensure optimal usage of both physical and software resources.

2.3 Software-Defined Networking (SDN)

Software-defined networking (SDN) [12], [13], is one of the most important architectures for the management of complex networks, which may require re-policing or re-configurations from time to time. SDN separates the control plane of traditional networking devices (e.g. switches, routers) from the data plane.

Before SDN, switches or routers were configured using routing protocols that did not allow fine-grained control. The control plane is known as an *SDN controller* and becomes directly programmable via an open interface (e.g., OpenFlow [14], [15] is the most popular SDN protocol/standard and has a set of design specifications). Thus, the underlying infrastructure (network routers/switches) just simply forwards packets by following the flow table rules set by the SDN controller.

SDN has the potential to dramatically simplify network management and to enable innovation and evolution through network programmability [16], [17], and [18].

The Open Networking Foundation (ONF) is taking the lead in SDN standardization, and has defined an SDN architecture model as depicted in Figure 2.3.

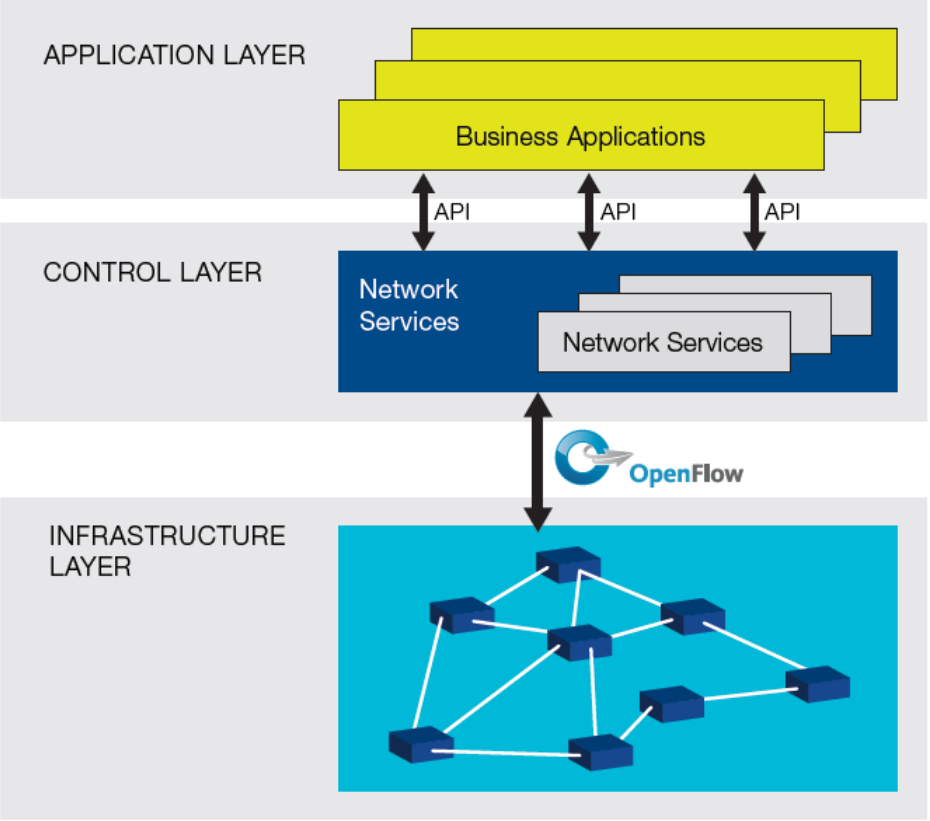

FIGURE 2.3: ONF/SDN architecture

The SDN architecture consists of three distinct layers that are accessible through open APIs:

- **Application Layer**: consists of the end-user business applications that consume the SDN communications services. The boundary between the Application Layer and the Control Layer is traversed by the northbound API.
- **Control Layer**: provides the logically centralized control functionality that supervises the network forwarding behavior through an open interface.
- **Infrastructure Layer**: consists of the network elements (NE) and devices that provide packet switching and forwarding.

2.4 Integration of NFV with other technologies

Over the past years, the integration of NFV with other technologies, such as SDN, Cloud computing, and 5G [19] has attracted significant attention from both the academic research community and industry.

NFV integration with SDN and Cloud computing is beneficial due to the complementary features and distinctive approaches followed by each technology toward providing solutions to today's and future networks [20], [21]. For instance, NFV provides **function abstraction** (i.e., virtualization of network functions) supported by ETSI [22], SDN provides **network abstraction** supported by Open Networking Foundation (ONF) [23], and Cloud computing provides **computation abstraction** (i.e., a shared pool of configurable computing resources (e.g., networks, servers, storage, applications, and services)) supported by the Distributed Management Task Force (DMTF) [24]. Abstraction is one of the core features of cloud computing which allows abstraction of the physical implementation to hide the background (technical) details from users and developers. To summarize the relationships between NFV, SDN, and Cloud computing, we use Figure 2.4.

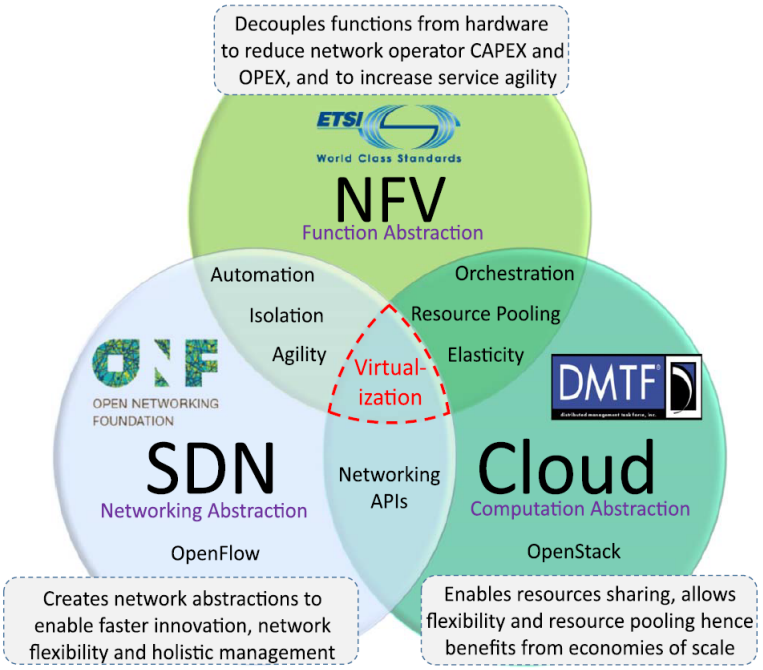

FIGURE 2.4: Relationships between NFV, SDN and cloud computing [3]

SDN, NFV, and Cloud computing technologies are complementary to each other but are independent and can be deployed alone or together. A combination of these technologies

together in a network architecture is more desirable [25]. In fact, the advantages that accrue from each of them are similar: agility, cost reduction, dynamism, automation, resource scaling, etc.

2.5 Resource Allocation in NFV

In NFV, a service is defined as a chain of software functions named Service Function Chain (SFC) [26], [27]. The process of allocating the resources of servers to the services, called Service Placement (or Resource Allocation), is one of the most challenging mission in NFV. The dynamic nature of the service arrivals and departures as well as the meeting Service Level Agreement (SLA) make the service placement problem even more challenging.

This section presents a survey of relevant research in the literature related to resources mapping in network environments. These studies are classified into two related topics. The first topic is the Virtual Network Embedding (VNE) problem [28] and the second one is the Virtual Network Function (VNF) placement and chaining problem [29].

Before entering into details, we add a note on the relation between the VNF placement and chaining problem and the traditional VNE problem. In fact, the task of placing functions is closely related to virtual network embedding [30], [31], [32] and may therefore be formulated as an optimization problem, with a particular objective.

Despite some similarities, VNE and VNF-FG embedding are distinct and have different characteristics [1].

- First, while in VNE we observe one-level mappings (virtual network requests $\rightarrow$ physical network), in NFV environments we have two-level mappings (SFC requests $\rightarrow$ VNF instances $\rightarrow$ physical network). VNE requests are modeled by simple undirected graphs while VNF chains are more complex and contain both the VNFs to place and the traffic flows to steer between the VNF-FG endpoints.

- Second, in NFV environments, a VNF can be shared by multiple demands, while in VNE, different virtual networks are typically independent (i.e., a flow of a virtual network does not traverse through the virtual nodes of another virtual network).
- Third, while the VNE problem considers only one type of physical device (i.e., routers), a much wider number of different network functions coexist in NFV environments.

2.5.1 Virtual Network Embedding (VNE)

2.5.1.1 Initial VNE strategies

The initial VNE problem can be divided into two sub-problems: *Virtual Node Mapping* consists in mapping each virtual node of the VN to one physical node that has enough available resources, and *Virtual Links Mapping* where virtual links connecting the virtual nodes have to be mapped to paths connecting the corresponding nodes in the substrate network under bandwidth resource constraints.

In VNE, the most referenced approach is the one suggested by Chowdhury et al. [33], 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 Virtual Networks Requests (VNRs).

Ogino et al. [34] propose a VNE algorithm based on a greedy approach, 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.

Wang et al. [35] propose 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 solutions with reduced computational time.

Soualah et al. [36] propose another VNE method uses Gomory-Hu tree transformation which provides 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 claim 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.

It is important to note that VNE problem can be solved in either a *centralized* or in a *distributed* [37] way. Both approaches are fundamentally different [28], each having its own advantages and disadvantages. In a centralized approach, there will be one entity which is responsible for performing the embedding. The advantage of this approach lies in the fact that the mapping entity is at every step of the mapping aware of the overall situation of the network (i.e. it has global knowledge). Moreover, there may be scalability problems in large networks, where a single mapping entity may be overwhelmed by the number of Virtual Network Requests to handle. Contrary to centralized solutions, a distributed approach utilizes multiple entities for computing the embedding. The advantage of such an approach lies in its better scalability. However, one has to pay for this with synchronization overhead.

2.5.1.2 Dynamic/Adaptive resource management strategies

During their lifetime, virtual network resource requirements can evolve according to clients' fluctuating demands. Hence reserving a fixed amount of resources is inefficient to satisfy them. To cope with these problems, some dynamic resource management strategies where proposed. Migration has been also used in the VNE context. Several dynamic algorithms using reconfiguration optimize substrate resources for virtual networks.

Authors of [38] propose a path migration algorithm for reconfiguring and rerouting virtual links, unfortunately do not consider adaptation of the virtual nodes and thus find suboptimal solutions.

An iterative algorithm, called Virtual Network Reconfiguration (VNRe), is presented in [39] to ensure load balancing among substrate nodes and reduce the rejection rate caused by congested substrate links.

In order to supply dedicated virtual node to an end-user typified with a customized traffic, Fajjari et al. [40] suggest 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.

In [41], the authors propose an adaptive optimization algorithm which selects only critical VN requests for reconfigurations. Despite this selection of most critical VN requests, computational cost and service disruptions remain important at large scale.

In [42], the authors propose an adaptive fault-tolerant VN embedding algorithm, which relies on a multi-agent based framework to cope with failures and severe performance degradation.

Authors of [43] propose a dynamic adaptive virtual network resource allocation strategy to deal with the complexity and the inefficiency of resource allocation. The main idea behind the proposal is take advantage of unused bandwidth with respect to the occupancy rate of embedded virtual links. The unused bandwidth will be reassigned to incoming virtual network requests.

Inoue et al. [44] present an adaptive VNE method based on the biological "Yuragi" principle for software-defined infrastructure (SDI). The proposed method works with little information for large and complicated SDI frameworks. The term Yuragi is a Japanese word whose English translation means a small perturbation to the system. Yuragi is a mechanism of adaptability of organisms and is often expressed as an attractor selection model.

Shahriar et al. [45] address the connectivity-aware Virtual Network Embedding problem, which consists in embedding a virtual network (VN) on a substrate network while ensuring VN connectivity against multiple substrate link failures. The proposed method provides a weaker form of survivability incurring less resource overhead than traditional VN survivability models.

The more important body of work on scaling and migrating services in VNE is not directly applicable to the VNF-FG placement problem.

2.5.2 Virtual Network Function (VNF) placement and chaining

Several surveys are now available on NFV, see, e.g., [3], [10], [25], and [1] where the various NFV challenges are discussed.

2.5.2.1 Initial VNF-FG placement strategies

VNF-FG placement and chaining is formulated as an Integer Linear Programming (ILP) in [31], [32], [46], [47], and [48] to find exact solutions for hosting the VNFs of the requested service graph. These works propose partial and exact mathematical formulations for the SFC provisioning problem. Researchers consider different objective functions.

Hybrid mathematical formulations: Martini et al. [49] and Riggio et al. [50] solve the placement and routing for each request independently. Gupta et al. [51] propose a heuristic based on an ILP. They only consider the k-shortest paths for every request in the network and a simplified node capacity constraint, for which only one function per node can be deployed. There has been a recent attempt with a Column Generation model, by [52], and

[53]. Unfortunately, as the resulting column generation model was not well scaling, its ILP solution is not exact.

Exact mathematical formulations: Luizelli et al. [48] provide an exact model minimizing the number of instances of functions in the network. However, they consider only a couple of tens of requests. Savi et al. [54] propose an exact formulation in which the number of VNF nodes is minimized. Their model takes into account additional costs inherent to multi-core environment. However, they only provide results on a small network. Bari et al. [55] consider the operational expenditure (OpEx) for a daily traffic scenario as their objective function. Mohammadkhan et al. [56] propose an exact model along with heuristics aiming at minimizing the maximum usage of CPU and links. The scope of the experiments is limited to the case in which the number of cores per service is limited to one.

However, ILP is only suitable for the situation that all variables are integers. Thus, for some specific situations, the Mixed ILP (MILP) is used instead. For example, Addis et al. [57] proposed a VNF-P model, in which both the NFV goal (minimizing the number of CPUs used by instantiating VNFs) and the Traffic Engineering (TE) goal (minimizing the risk of a sudden bottleneck on network links) are considered. However, in order to jointly achieve the two goals, some non-integer variables have to be introduced. Thus, the VNF-P model proposed was actually a MILP model which described the relationship between VNF placement and traditional routing.

As the addressed problem is NP-Hard [55], the exact solutions do not scale with size and require an excessive amount of time to find the optimal solutions.

Heuristic algorithms: are typically and consequently proposed to scale better with problem size by solving the problem iteratively and to find good solutions much faster. Unfortunately, this is accomplished at the expense of quality of the solutions (proximity with the optimal solutions).

A baseline Greedy algorithm [58], using a bipartite graph construction and matching techniques, for VNF-FG placement and chaining, solves the problem in two steps: by first mapping the VNFs on physical hosts and second by steering the inter-VNF traffic across the hosts. This leads obviously to suboptimal solutions that are often far from optimal.

A heuristic algorithm in [59] uses "Simulated Annealing" (SA) to find solutions faster unfortunately by simplifying the overall problem by considering only one type of VNF and addressing rather small chains.

Authors in [60] focus on the placement of VNFs chains in the NFV context. They propose a matrix-based optimization and a multi-stage graph method that are cost efficient and improve scalability by finding solutions in polynomial times.

Note that some work study the SFC provisioning problem using game theory [61] or approximation algorithms [62], [63], and [64]. Some context-aware placement problems are also studied. Authors in [65] study VNF placement algorithms in virtual 5G network, with the goal of minimizing the path length and optimize the sessions' mobility.

2.5.2.2 Elastic VNF-FG placement strategies

Elastic orchestration of virtual network functions (VNF) is a key factor to achieve NFV goals. However, most existing VNF orchestration researches are limited to offline policy, ignoring the dynamic characteristics of the workload. In fact, the resource that a VNF has may scale due to dynamic traffic (for example, a DPI need less computing resource when the traffic decreases). The QoS demand of a VNF may change also due to changes of service requests (for example, when an established service request asks for low latency, the re-allocation of VNFs is required) [66].

Therefore, in order to overcome those challenges, we should rethink the VNF-FG embedding problem and propose new solutions. Some authors do address virtual network function scaling and migration to ensure dynamic VNF chain placement for rising demand and traffic load.

Authors of [67] propose a consolidation algorithm called Simple Lazy Facility Location (SLFL) that optimizes the placement of the VNF instances in response to on-demand workload. SLFL chooses the VNF instances to be migrated on the basis of the instantaneous reconfiguration but does not assess the impact (induced benefits or penalties) of these decisions on future instants.

Bandwidth guaranteed VNF placement and scaling in Datacenter is considered in [68]. Leveraging the tree-like topology of Datacenter networks, this work proposes an on-line heuristic algorithm that achieves a near-optimal allocation. Note that this approach does not take into account migration cost.

Authors in [69] propose an architecture for the 5G core network using SDN and NFV technologies. Moreover, based on this architecture, they design a framework for determining the Network Functions (NFs) placement and optimizing the NFV Infrastructure (NFVI) resources and network response time during emergency situations. This framework is based on a two-stage method. In the first stage, the placement of the SDN Controllers and VNFs is determined considering not only latency requirements and load levels but also users' mobility and network functions type. Meanwhile, in the second stage, the NFVI resources are dynamically optimized according to variations on network conditions.

Toosi et al. [70] define a unified framework for building elastic service chains. They propose a dynamic auto-scaling algorithm called ElasticSFC to minimize the cost while meeting the end-to-end latency of the service chain. The experimental results show that the proposed algorithm can reduce the cost of SFC deployment and SLA violation significantly.

Gu et al. [71] propose an Elastic Virtual Network Function Orchestration (EVNFO) policy based on workload prediction. They adapt the online learning algorithm for predicting the flows rate of service function chains (SFC), which can help to obtain the VNF scaling decision. They further design the online instance provisioning strategy (OIPS) to accomplish the deployment of VNF instances according to the decision. The simulation proves that EVNFO can provide good performance with dynamic resource provision.

Luo et al. [72] propose a deep learning-based framework for scaling of the geo-distributed VNF chains, exploring inherent pattern of traffic variation and good deployment strategies over time. They novelly combine a recurrent neural network as the traffic model for predicting upcoming flow rates and a deep reinforcement learning (DRL) agent for making chain placement decisions. They adopt the experience replay technique based on the actor-critic DRL algorithm to optimize the learning results.

Pei et al. [73] study the SFC embedding problem (SFC-EP) with dynamic VNF placement in a geo-distributed cloud system. They formulate this problem as a Binary Integer Programming (BIP) model aiming to embed SFC requests with the minimum embedding cost.

Rankothge et al. [74] present an Iterated Local Search (ILS) based framework for automation of resource reallocation that supports the three scaling models. They use this framework to run experiments and compare the different scaling approaches, specifically how the optimization is affected by the scaling approach and the optimization objectives.

Ma et al. [75] present a load balancing methodology for the management of horizontal scaling of NF chains that does not require changes to the NF code. They also develop a prototype reference implementations to illustrate the feasibility of the proposed solution and conduct extensive simulations to assess the performance.

Gouareb et al. [76] aim to minimize latency, considering horizontal scaling of VMs by placing VNFs across physical nodes. Routing and placement of VNFs are linked and have a significant impact on latency especially for delay-critical services. They aim at optimizing the distribution and utilization of available resources and meet user requirements with the objective of minimizing the overall accumulated latency on both the edge clouds (where VNFs are running) and the flow routing paths.

Chen et al. [77] formulate the SFC migration problem as a minimization problem with the objective of total network operation cost under constraints of users' quality of service. They design a deep Q-network based algorithm to solve single SFC migration problem, which can adjust migration strategy online without knowing future information. A novel multi-agent cooperative framework is also proposed to address the challenge of considering multiple SFC migration based on single SFC migration.

Tang et al. [78] propose a real-time VNF migration algorithm based on the deep belief network to predict future resource requirements. But training the practicable neural network is extremely eager for large amounts of training data.

In a recent work, Fei et al. [79] study the proactive VNF provisioning problem for NFV providers, considering the fluctuations of traffic traveling service chains. They formulate

the problem that minimizes the cost incurred by inaccurate prediction and VNF deployment. They first employ an online learning method which aims to minimize the prediction error, to predict the upcoming traffic flows. Then, when launching new instances based on the prediction outcomes, an adaptive scaling strategy is adopted for saving resources and decreasing deployment cost. They also propose two algorithms that are called by the complete online algorithm, for new instance assignment and service chain routing.

2.6 Conclusion

This chapter describes the state of the art that relates directly to the NFV resource allocation problem and that is directly connected to our contributions. In recent years, the problem has received some interest from the community and has produced several valuable contributions. However, the outcome is still incomplete. There are yet important aspects that should be investigated to efficiently manage and allocate the use of the resources in NFV-based network architectures. The next chapters describe the thesis contributions in terms of dynamic resource management strategies, mathematical models, and both exact and heuristic optimization algorithms.

Chapter 3

Virtual Network Function Scaling

3.1 Introduction

In NFV based environment, one of the most important challenges for providers is to efficiently allocate hosting resources to dynamic virtualized network services demands while increasing revenue. Elastic mechanisms and scaling algorithms are essential to improve adaptation and deployment of Virtualised Network Functions (VNFs) in NFV infrastructures to support increasing traffic load and customer demands.

As introduced in [80] and [67], Virtual Network Function scaling is triggered by new client requirements and/or rising traffic load due, for example, to an increase in the user plane traffic or the need of allocating more resources to a VNF to avoid service interruption.

To enhance initial VNF placement with dynamic adaptation, three scaling mechanisms are defined in [67] and [81]:

- *Horizontal scaling (scale out/in)*: Add/remove virtualized resources (e.g., VNF Components (VNFCs)).
- *Vertical scaling (scale up/down)*: Reconfigure the capacity/size of existing virtualized resources.

- *VNF Migration*: move VNF components from one hardware platform onto a better platform while still satisfying the service continuity requirement.

To address this NP-Hard elasticity problem [82], we propose an exact algorithm and a Greedy heuristic. An Integer Linear Programming (ILP) formulation with a search space reduction is adopted to improve scalability. A Greedy algorithm searching for a scaling solution in the neighborhood of an initial placement is also presented. Comparisons between the two approaches show that an adequately tuned and devised ILP can outperform Greedy solutions in terms of proportion of successful scalings.

In [82], authors propose an Integer Linear Programming (ILP) model to solve the network function migration problem. In the rest of this chapter, we denote this competitor algorithm by (ILP_C). It proposes a migration cost model and a heuristic algorithm to decrease the migration cost. Note that these algorithms do not consider scaling and elasticity to optimize resource utilization.

The system model is described in Section 3.2. The proposals are introduced in Section 3.3. Section 3.4 reports the performance evaluation results and finally, we summarize the results with some future research directions in Section 3.5.

3.2 Problem Formulation

This section models the VNF and the VNF-FG scaling problem and derives an ILP model that ensures placement and scaling for increasing demands with minimal cost and service interruptions.

3.2.1 Substrate and virtual network models

The physical network (commonly known as substrate or physical infrastructure. The terms are used interchangeably), as defined by the ETSI NFV Infrastructure (NFV-I) [83], is modeled as an undirected weighted graph, denoted by $G_p = (N_p, E_p)$ where E_p is the set of