Optimizing resource allocation in infrastructure networks based on network function virtualization

Thi Minh Nguyen

Thèse de Doctorat
Université Pierre et Marie Curie – Paris VI

Spécialité

Systèmes Informatiques

présentée par

Thi Minh Nguyen

pour obtenir le grade de

Docteur de l'université Pierre et Marie Curie

Optimisation de l'Allocation des Ressources dans les Réseaux d'Infrastructure basés sur la Virtualisation des Fonctions Réseau

Soutenue le 07 Dec 2017 devant le jury composé de

Kavé SALAMATIAN	**Rapporteur**	**Professeur, Université de Savoie**
Fabio MARTIGNON	**Rapporteur**	**Professeur, Université Paris-Sud**
Vania CONAN	**Examinateur**	**Directeur de Recherche, Thales Communications & Security**
Marinho P BARCELLOS	**Examinateur**	**Professeur Associé, INF/UFRGS**
Michel MINOUX	**Examinateur**	**Professeur, Université Pierre et Marie Curie**
Anne FLADENMULLER	**Examinateur**	**Maître de Conférences, Université Pierre et Marie Curie**
Tuan-Minh PHAM	**Examinateur**	**Docteur, Thuy Loi University, Vietnam**
Serge FDIDA	**Directeur de thèse**	**Professeur, Université Pierre et Marie Curie**

Numéro bibliothèque : _____

Doctor of Science Thesis
Université Pierre et Marie Curie – Paris VI

Specialization

Computer Science

presented by

Thi Minh Nguyen

Submitted in partial satisfaction of the requirements for the degree of

Doctor of Science of the Pierre et Marie Curie University

Optimization of Resource Allocation in Network Function Virtualization Infrastructure

Committee in charge:

Kavé SALAMATIAN	**Reviewer**	**Professor, Université de Savoie**
Fabio MARTIGNON	**Reviewer**	**Professor, Université Paris-Sud**
Vania CONAN	**Examiner**	**Research Director, Thales Communications & Security**
Marinho P BARCELLOS	**Examiner**	**Associate Professor, INF/UFRGS**
Michel MINOUX	**Examiner**	**Professor, Université Pierre et Marie Curie**
Anne FLADENMULLER	**Examiner**	**Associate Professor, Université Pierre et Marie Curie**
Tuan-Minh PHAM	**Examiner**	**Dr, Thuy Loi University, Vietnam**
Serge FDIDA	**Supervisor**	**Professor, Université Pierre et Marie Curie**

Acknowledgment

Undertaking PhD at University Pierre et Marie Curie (UPMC) has been one of the most special time in my life. I would like to thank all people who contributed in various ways to this dissertation.

First and foremost, I express my deep gratitude to my advisor Professor Serge Fdida who welcomed me in 2014 at the start of my PhD at LIP6. He raised my interest in the subject area of this thesis, guided and supported me in every step of this thesis along these years. I thank Serge for being such a perfect advisor and always encouraging me to be more confident. I would also like to thank him for being patient with my questions and for giving me the invaluable lessons that I will never forget. Without his help, it would be impossible for me to overcome many difficulties in my research. It is a great fortune to have pursued my PhD studies under his guidance at LIP6.

Next, I would also like to thank Professor Kavé Salamatian, Vania Conan, Fabio Martignon, Marinho P Barcellos and Anne Fladenmuller for having accepted to be part of my PhD defense committee.

I want to special thank Tuan-Minh Pham from Thuy Loi University, Vietnam for introducing me to pursue my studies at the University Pierre et Marie Curie. I really appreciate all the hard work he has done to help me. I have learned so much, and it is thanks to him. I am also truly grateful to Professor Catherine Rosenberg and Andre Girard from Canada, Professor Michel Minoux from LIP6 for guiding me a lot during my research. Their guidance and support have been amazing.

I would like to give my special thanks to the researchers, James Roberts, Ludovic Noirie, Marc-Olivier Buob, Spathis Prométhée, Marcelo Dias de Amorim, Thai Kim Loan, etc., whom I had the pleasure of meeting at LIP6, LINCS, and at various events over the past several years, for the open discussions and their valuable comments on part of this research. I am grateful to Stefano Secci and Ludovic Noirie for their precious advice on the midterm review of my thesis. Special thanks to Tran Ngoc Hoang Tien and Phuong Hoa Le for their corrections on my French writing.

During my doctoral research, I have been lucky to be a member of the NPA team where I have enjoyed discussing, chatting with my friends during memorable times. In particular, thanks to my PhD friends: Salah, Wafa, Bu, Benjamin, Fadwa, Alexandre, Quentin, Yu Ting Lin and all the administrative staffs who support great working facilities at LIP6.

My life as a PhD student would not have been enjoyed without my friends from Paris. I take this opportunity to special thank Hoang Co Thuy Thanh, Hoang Co Thuy Hoa Dung, and all their family, for giving me a new family here, spending a lot of time to take care of me and for all their kindness, continuous help, and love on these days in Paris. Many

thanks to Truong Giang Pham for his help on my initial days. I really do appreciate it. Thanks to my friends, Mai Nguyen, Luong Nguyen, Phuong Hoa Le, Quang Tran, Thu Thuy Nguyen, Nhu Nguyen, Thi Thu Trang Nguyen, Thanh Ha Nguyen, Xuan Tuan Dau, Ha Ngan Nguyen, Trong Hieu Nguyen, Le Thang, Ngu Duong, Duy Nguyen, Quang Hoa Tran, Viet Vu for their support and for being with me all wonderful moments during my thesis.

Finally, from the bottom of my heart, I am sincerely grateful to my whole family in Vietnam who has always supported me and put their trust in me. I dedicate this dissertation to them with my love and greatest respect.

Résumé

Les prestataires de service réseau doivent faire face à la demande croissante des besoins des utilisateurs, en particulier vers une plus grande fléxibilité et toujours plus de capacité. La "softwerisation" et la "cloudification" des composants du réseau offrent une solution prometteuse pour obtenir l'agilité nécessaire afin de répondre dynamiquement à l'exigence au niveau de la consommation des ressources. Cette vision se traduit par le déploiement de la Virtualisation des Fonctions Réseau (NFV) où les Fonctions de Réseau Virtuels (VNFs) peuvent être associées pour créer des services réseau.

Cette thèse étudie la problématique de l'allocation de ressources dans un système NFV afin de minimiser son coût sous contraintes sur l'interconnectivité entre les VNF, les ressources du système et les exigences de service. La principale considération est la réduction du coût global du déploiement en ressources informatiques. Nous étudions également d'autres objectifs à satisfaire tels que la migration des fonctions réseau et la gestion de la congestion.

Notre premier objectif est d'augmenter notre compréhension de la performance d'un système NFV en étudiant le placement et le routage des fonctions réseau. Nous formalisons le problème dans une approche globale en tenant compte d'un large ensemble de paramètres pertinents. Nous prendrons en compte les cas statiques (Hors Ligne) et dynamiques (En Ligne) du problème. Nous proposons et analysons trois algorithmes heuristiques : deux sont conçus pour traiter de grandes dimensions du scenario "Hors Ligne" et le dernier est conçu pour résoudre le scénario "En Ligne". Les résultats montrent que notre solution surpasse l'état de l'art par rapport à l'indicateur de performance critique. Nous évaluons également l'impact de la migration d'une série de demandes simultanées et proposons une technique de migration simple pour ce système dynamique. A la lumière de ces premiers résultats, nous étendons notre étude afin d'améliorer l'efficacité de notre solution en proposant un modèle plus simple.

La seconde partie de notre étude se concentre sur l'optimisation de l'utilisation des ressources d'un système NFV. La principale distinction est que nous pouvons appliquer le modèle à un système dynamique avec de grandes instances. De plus, nous fournissons également une méthode originale pour engendrer de fortes inégalités afin d'améliorer la résolution de la programmation linéaire (LP) dans un espace de dimansion supérieur. Les résultats obtenus n'améliorent pas seulement le modèle, mais promettent aussi de pouvoir être utiliser efficacement dans d'autres modèles.

Une troisième contribution de notre travail concerne le problème de routage dans NFV. En effet, une évolution importante des besoins des utilisateurs est représentée par la demande d'accès croissante aux ressources réseau, de stockage et de calcul afin de combiner dynamiquement le niveau de consommation de ressources avec leurs besoins de service. Par conséquent, nous nous intéressons au routage efficace d'une demande utilisateur à travers les nœuds qui traitent les fonctions impliquées dans une chaîne de services donnée. Nous proposons une formulation originale de ce problème basée sur la construction d'un réseau étendu. Nous formulons une solution mathématique exacte et proposons plusieurs algorithmes approximatifs tenant compte les principaux paramètres du système. Nous concluosn en soulignant les contributions principales de notre travail et proposons quelques pistes pour des travaux futurs.

Mots-clés :

NFV, Virtualisation des Fonctions Réseau, Allocation des Ressources, Optimisation, Migration, Problème de Routage, Performance, Placement et le Flux de Coûts Minimaux Entiers.

Abstract

Network service providers have to cope with the growing on-demand need from end-users as well as the diversity of usage. The "softwerization" and "cloudification" of the network components offer a promising solution to achieve the agility necessary to dynamically match the servcice requirements with the level of resource consumption. Cloud-based solutions promises an economy of scale and simpler management. Virtualizing the many network appliances offers the flexibility to adapt to the varying service demand. This materializes with the deployment of Network Functions Virtualization (NFV) where Virtual Network Functions (VNFs) may be chained together to create network services.

This dissertation studies the resource allocation problem in an NFV system for minimizing its cost under constraints on interconnectivity among VNFs, system resources, and service requirements. The main consideration is the reduction of the overall deployment cost while efficiently utilizing the available resources. In addition, a number of other important constraints are considered such as migration and congestion.

Our first goal is to increase our understanding of the performance of an NFV system with respect to network functions placement and routing. We formalize the problem in a comprehensive maner taking into account a broad set of relevant parameters. The static (OFFLINE) and dynamic (ONLINE) cases are considered. We propose and analyze three heuristic algorithms: two for handling large dimensions of the OFFLINE problem and one designed to address the ONLINE scenario. The results show that our solution outperforms the state of the art with respect to critical performance index. We also evaluate the impact of migrating a set of running demands, and propose a simple migration technique for the dynamic system. We extend this work by proposing a simpler model to improve the performance of our solution.

The second part of our work focuses on minimizing the resource utilization of an NFV system. The main distinctive point is that we can apply the model to a dynamic system with large instances. Moreover, we also provide an interesting method for generating some strong inequalities to improve the Linear Programming (LP) solving in a higher dimensional space. The obtained results are not only making the model easier but also can be used efficiently in other models.

A third contribution focuses specifically on the routing problem in NFV. An important evolution of the users' needs is represented by the dynamic on-demand access to network,

storage and compute resources. Therefore, routing efficiently a demand across nodes handling the functions involved in a given service chain constitutes the a novel problem that we address in this last section. We provide an original formulation of this problem based on the construction of an expanded network. We derive the exact mathematical formulation and propose several approximate algorithms taking into account the main system's parameters. We conclude by deriving some interesting insights both about the algorithms and the network performance.

We finally conclude with our main findings and highlight many avenues for future work.

Key Words:

NFV, Network Function Virtualization, Resource Allocation, Optimization, Migration, Routing Problem, Performance, Placement, and Integer Minimum-cost Flow.

Table of contents

Chapter 1

Introduction

Contents

This dissertation studies the performance of a Network Function Virtualization Infrastructure with an emphasis on solutions for satisfying multiple objectives. NFV is an emerging network paradigm based upon virtualization technology deployed onto commercial off the shelf (COTS) servers, in order to replace network appliances. NFV promises to bring agility and flexibility to Service Providers (SPs) when deploying services to Customer Premises Equipments (CPEs). In order to enable NFV, one of the first challenging problems is related to the mapping of VNFs on VNF Infrastructure. This problem also received a lot of attention from the community.

We start this chapter by presenting the research context of this dissertation. Next, we show that due to the challenging characteristics of its environment, efficiently modeling an NFV placement and routing with multiple objectives has become a key issue for deploying NFV systems (Section 1.2). We then outline our methodology and proposed solutions to the NFV resource allocation problem (Sections 1.3-1.4). Finally, we summarize our contributions and describe the structure of the dissertation (Sections 1.5-1.6).

1.1 Research Context

Nowadays, telecommunication networks are over populated with a large and increasing variety of proprietary hardware appliances built for specific purposes such as Routers, Firewalls, and Load Balancers. It raises a significant effort in the integration, operation and maintenance of a communication infrastructure. For example, to launch a new network service, it often requires another appliance as well as finding the space and power to accommodate these boxes. It makes it more difficult and expensive in power, capital investment and human resources. Moreover, hardware-based appliances rapidly becomes obsolete and requires to be updated. Virtualization Technology appears as a solution to overcome these concerns.

Emerging NFV technology promises significant simplification as current hardware appliances become replaced by software running on standard servers (eventually enhanced with specific hardware). NFV is a new way to design, deploy, and manage networking services by decoupling the *Network Functions* from physical network equipments, called *Dedicated Hardwares*, and move them to *Virtual Servers* that can run on general-purpose CPUs or virtual machines. The flexibility, agility, scalability, capital and operational cost savings that are made possible with NFV opens up new innovation, design paradigm and enables new network architectures.

In particular, the application of Network Functions Virtualization brings many benefits to network operators, contributing to significant changes in the telecommunications industry as follows:

- Cost efficiency is a main driver of NFV. It should cut operator Capital Expenditures (CAPEX) and Operational Expenditures (OPEX) through reduced equipment costs and reduced power consumption.

- Faster Service Life Cycle: New network services can be deployed more quickly, in an on-demand and on-need basis, providing benefits for end users as well as the network providers. Therefore, it reduces time-to-market to deploy new network services.

- Scalability and Elasticity: NFV allows capability changes by offering a mean to expand or shrink the resources used by the VNFs. It also can implement elasticity by offloading a VNF's workload and spinning off a new instance to implement the same network function and split the load with an existing VNF.

- Operational Efficiency and Agility: With common hardware hosting different VNFs, tasks associated with running the business, such as inventory management, procurement process, can be centralized. This reduces the operational overhead compared to segregated deployments of different network services using multiple hardware devices.

It is clear what NFV can offer in terms of benefits, but challenges also exist. In order to make NFV deployment a reality, research challenges and future directions should be considered including NFV performance, the relevant Resource Allocation, the Management and orchestration of NFV, and the Security issue.

- NFV performance: The effort of NFV is to run Network Functions on industry standard servers. However, it is challenging to offer guaranteed network performance for virtual appliances. This raises the question of whether the Network Functions run on industry standard server could achieve similar performance when compared to those running on specialized hardware [1], [2].

- Management and Orchestration of NFV: The deployment of NFV challenges the current management systems and profoundly change the way to address it. It requires significant changes not only to provide network and service solutions, but also to exploit the dynamism and flexibility made possible by NFV [3], [4].

  OpenMANO project of the European Telecommunications Standards Institute (ETSI) aimed to provide a MANO framework [5] for provisioning of VNFs and related operations, such as the configuration of the VNFs and the its target infrastructure. In a related effort, Cloud4NFV [6], [7] has proposed an end-to-end management platform for VNFs, which is based on the ETSI architectural specification. Several other projects are on their way.

- Security, Privacy and Trust: Security is always one of the most challenging problem in any network system, specially when the function to be virtualized and deployed in third party clouds. Because sensitive information may be transferred to the cloud, service providers need to provide a security mechanism in NFVI.

  ETSI established a security expert group to focus on this concern. They provided a security and trust guidance that is unique to NFV deployment, architecture and operation [8]. However, this does not consist in prescriptive requirements or specific implementation details.

- Resource Allocation: Ideally network operators should locate VNFs where they will be used most effectively and least expensively. Although the virtualization of certain network functions is straightforward, there are a number of network functions that have strict delay requirements. For instance, a NFV solution may take a longer path than the likely shortest path followed by a non-NFV system where network functions are placed on the direct path between two end points. Therefore, the placement of Virtual Machines (VMs) that carry VNFs is crucial to the performance of offered services. It becomes one of the important challenges in NFV system. The aim of this

dissertation is to better understand the right level of abstraction and improve the performance of such system with considering multiple objectives.

NFV is considered as an opportunity and a challenge by all stakeholders. The explosive growth of virtualization technology and the advantages that NFV offers have given rise to preliminary deployments based on NFV. Therefore, efficient resource allocation is a central problem in an NFV Infrastructure and one should better understand its complexity and efficiency parameters. The importance and challenges of the above mentioned issues will be discussed in the next two sections.

1.2 Challenges of Resource Allocation and Routing in NFV

Although there exist a lot of attention from standardization organizations, academic and industrial research projects, or vendors, NFV Resource Allocation and Routing problems are still at an early stage of development. Placing VNFs on NFV Infrastructure requires an efficient solution able to guarantee multiple constraints. This problem is challenging for the following main reasons:

- This problem is related to Virtual Network Embedding (VNE) [9], [10] and hence similar approaches may be applied. However, any formulation should be able to take the function chaining and/or precedence requirements into consideration to avoid network congestion. This characteristic of an NFV system makes the problem formulation more complex and difficult to handle.

- One of the selling points of NFV is the ability to scale resources dynamically. There must be capabilities to increase or reduce the amount of resources allocated to specific functions or VMs. While current virtualization or cloud platforms allow this, many of them require a manual trigger by the user or resource owner. Therefore, automation and self-allocation mechanisms that allow the network to dynamically manage resources are critical to the success of NFV. However, in order to adapt to the demand dynamics but also to changes in service requirements, one should depend on many variables such as the arrival time and the processing time of each demand, the limited storage and energy, the resource requirement of VNFs. However, we must make enough simplifying assumptions for the model to be tractable, where these assumptions should be acceptable in practice. As a conclusion, due to the complexity of the problem, its numerous parameters involved and the sensitivity of the performance criteria considered, the model as well as its resolution are complex challenges.

- The servers used to host VNFs have a finite amount of memory, compute and storage capacity. And since in practice these servers may be distributed across mul-

tiple domains, inter-domain link capacity will also be finite. Therefore, to achieve the economies of scale expected from NFV, physical resources should be efficiently managed. This problem might introduce additional complexity due to possible inter-dependencies between the network functions as in the case of function chaining and potential new constraints regarding additional aspects like security [11].

- In an NFV system, the wide range of available services and the sevice on-demand model create dynamic traffic conditions that necessitates a flexible and automatic network platform to redirect traffic according to network conditions. For a dynamic provisioning model, it is hard to develop an efficient algorithm to meet all requirements, especially visiting the function nodes in the prescribed order of package flows.

Understanding the performance of the resource allocation problems in NFV, as well as finding the best mapping of VNFs into NFVI nodes constitutes the core of this work. As far as we know, there exists no model to consider multiple objectives and constraints for NFV placement and routing together. We provide solutions to this problem in the sequel and evaluate their efficiency.

1.3 Goal of the Dissertation

The main goal of this dissertation is to model a NFV placement and routing problem while taking into account the execution order of VNFs in a service demand with regard to resource constraints for nodes and links. Specifically, our goals include the following:

- Get a better understanding of an NFV framework. The solution should help us to measure the impact of the location of VNFs, and derive how to improve the performance of an NFV system through the placement of VNFs.
- Develop a solution for the resource allocation in NFV with multiple objectives such as minimizing resources cost, maximizing acceptance ratio, minimizing penalty cost for VNFs migration. The solution should help us to decide the best locations to put VNFs that optimize the performance of the NFV system. Moreover, it provides guidelines for the percentage of traffic to be migrated (in the online case) in order to achieve the best gain and discuss some architectural implications.
- Develop a solution for routing problem in NFVI. The problem is to design an algorithm that will find an appropriate path very quickly for each request in a large network. The time limit is due to the fact that the requests arrival rate may be large and a decision has to be taken fast enough so that there is accumulation of requests waiting for a connection path.

To achieve these goals, we proceed with the following steps:

1. Develop a mathematical formulation capturing the resource management and placement problem while considering the order of VNFs in a chain with constrained resources for nodes and links (Chapter 3).

2. Propose an extended model to improve the performance for solving the LP in a dynamic scenario with large instances (large number of nodes and functions). The model takes into account the decision of Service Providers to serve a demand or to reject it in an NFV dynamic system (Chapter 4) .

3. Develop a mathematical formulation of the routing problem as an integer minimum-cost flow model with side constraints on the expanded network (Chapter 5).

4. Evaluate the analytical solutions under various scenarios, random datasets as well as real datasets (Chapter 3-5).

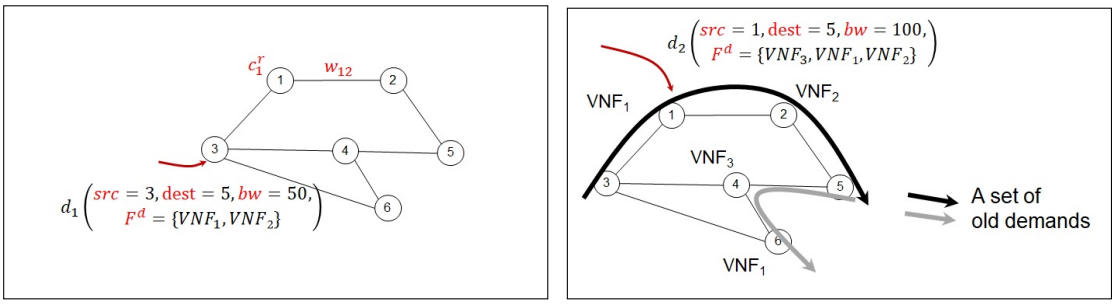
1.4 Overview of the Solutions

In this dissertation we develop analytical solutions for improving the performance of NFV systems by optimizing its global resource consumption according to some performance objectives. This section outlines our solutions to the above problem in the following steps.

1.4.1 Resource Management and Placement in NFV

We first consider the problem of resource management and placement in NFV. We study a NFV system where a network function is offered as a service. The system is composed of three components, including a set of VNFs, a set of customer demands and a virtual network that provides resources to the VNFs. VNF is a software implementation of a Network Function (NF) deployed on an NFVI. Each demand requests access to a network service (NS) that requires special functionalities. It can be composed of several VNFs in order to support advanced network connectivity, e.g. deep packet inspection, firewall, load balancer. In a NS, VNFs will need diverse resources such as CPU, memory, storage, bandwidth, etc. It is recognized that virtualized solutions are quite costly in resource consumption and therefore we assume that it is not accepatble to deploy all functions in all NFVI nodes. Resources are distributed anywhere on the NFVI. A customer demand materializes as a packet flow from a source to a destination visiting a set of nodes where the required VNFs are deployed.

How to efficiently share resources for all demands with delay constraints or limited capacity is an important challenge for NFV providers. It is likely that the solution achieved



(a) OFFLINE Case (b) ONLINE Case

Figure 1.1: Resource Management and Placement in NFV

will trade-off agility with efficiency. For instance, a NFV solution may take a longer path than the likely shortest path followed by a non-NFV system. However, it is still applicable if the delay lies within an acceptable range.

We develop a comprehensive solution addressing the static and dynamic scenario, when taking into account the execution order of VNFs in a service demand with regard to resource constraints for nodes and links.

Fig. 1.1 describes an overview about the resource management and placement problem in NFV. We address both the OFFLINE Case and the ONLINE Case. Fig. 1.1(a) illustrates an OFFLINE scenario with a network of 6 nodes and 7 links. We need to provide resources for a customer demand that requests to send packets from *node 3* (its source) to *node 5* (its destination) and goes through function VNF_1 and VNF_2 in the right order. The available resources including processing resources on nodes (c_1^r) and bandwidth resources on links (w_{12}) are given. Fig. 1.1(b) shows a ONLINE scenario where an operational network is serving a set of customer demands (called old demands, i.e., the black and gray curves in the figure). At this stage, a set of VNFs are already deployed (for instance, VNF_1 on *node 1*, VNF_2 on *node 2* of the black demand, and VNF_3 on *node 4*, VNF_1 on *node 6* of the gray demand) and the routing paths for the old demands are also provisioned. The problem is to handle new demands (demand d_2 in this case) that the network operator needs to provision and select their appropriate VNFs and routing paths. The problem objectives are to maximize the number of served demands and minimize the system resource costs.

We consider an interval τ (according to the system configuration) where we will update the status of all demands running in the system. Upon completion, the system releases the resources allocated to a demand. At time t, we collect new demands D_n that arrived during the period $(t - \tau)$ to t.

Another concern in the ONLINE problem is to decide if and how a set of old demands should be migrated. It is the fact that after a certain time, some demands will finish release resources and return them for the system. As a consequence, the distributed resources

are fragmented and not optimal. For example, in the Fig. 1.1(b), when the gray demands finishes, it will return resources for *node 4* and *node 6*. Therefore, it is better to re-optimize the black flow by moving to a new position ($\{node\ 3 \rightarrow node\ 4 \rightarrow node\ 5\}$, and put VNF_1 on *node 4*, VNF_2 on *node 5*).

We formulated the problem as a quadratic program for both optimal VNF location and optimal routing. Our model is generic, able to handle a large and diverse set of key parameters and can easily be customized. A major advantage with this model is that we take into account the interconnectivity among VNFs, service requirement, and NFV costs (including both resource cost and traffic cost). We investigate both the OFFLINE case and the ONLINE case and propose heuristic algorithms to solve the problem with large instances (large size system). We also compute the cost incurred by a network relying on NFV to capture the overhead introduced by a solution based on the virtualization of functions. In addition, by exploring the distribution of VNFs on a multi-tier network topology, we analyze the impact of VNFs locations and suggest that the network functions should be located on the network edge rather than the network core. Finally, we propose a simple migration technique that can efficiently manage the dynamic situation produced by a continuous flow of arriving demands.

In Chapter 3, we present a detailed description of the mathematic model for the resource management and placement problem. We also describe simulation results obtained by validating the algorithms and analyzing the impact of input parameters on both random and real data-sets.

1.4.2 Resource Utilization in an NFV Dynamic System

In the first step, we formulated the NFV placement and routing problem as a quadratic program and investigate both OFFLINE and ONLINE cases. However, the proposed model is quite complex and the optimal solution is only found in case of a static problem for small instances. This model is restrictive to apply in an NFV dynamic system. Therefore, this raises the question of how to find the optimal solution in an NFV dynamic system? It is necessary to develop an efficient model for the NFV placement and routing problem that can be solved exactly with large instances and robust to be applied in a dynamic scenario.

Therefore, we concentrate on proposing a mathematical model for a joint problem of VNFs placement and routing paths in order to minimize the resource utilization of an NFV dynamic system. The routing path will be steered through a number of VNFs, with the goal of executing the needed service function in the required order. Moreover, we take into account the possibility for Service Providers to serve dynamic demands.

In order to improve the performance for solving the LP problem, we also investigate an interesting method for generating some strong inequalities in a higher dimensional space.

The results show that we can get a significant improvement after adding some flow covers to the model. In addition, we develop two heuristics to find a feasible solution in an acceptable execution time. Via an event-driven simulation, we evaluate our solution on realistic network topology and compare the objective value of the problem obtained by thr proposed algorithms. As a consequence, our model is simple and straightforward to deploy in large dynamic systems. Moreover, the lifted flow covers added to LP are powerful and can be used efficiently in other models.

Chapter 4 provides a detailed mathematical formulation for the resource utilization problem in an NFV infrastructure. The chapter also presents our simulation to evaluate the performance of the solutions. In addition, we analyze the impact of input parameters in various scenarios on realistic data-sets.

1.4.3 Routing via Functions in NFV

The solution derived in the two first steps provides a quantitative analysis to efficiently share resources for all demands with delay constraints or limited capacity in an NFV system. It focuses on the placement of VNFs in the network rather than provide a quick solution for each request in a large network. It is due to the fact that in a large network, the requests arrival rate maybe high and a decision has to be taken fast enough to avoid build-up of request waiting for a connection path. The challenge here is to design a routing algorithm that will find such a path very quickly for each request.

We consider a network where connection requests arrive randomly with random holding time. Each connection must use a Service Chain, i.e., a given set of functions, specific to the connection, in a prescribed order. Multiple of these function are placed a priori in some subset of nodes, called the "*function nodes*".

For each connection request with given origin and destination, we want to find a path through the network that will meet all requirements and in particular, will visit the function nodes in the prescribed order. If this is not possible, the connection is blocked. The most important measure of the quality of a routing algorithm is the connection blocking probability. This is measured from an event-driven stochastic simulation in a set of typical networks. Fig. 1.2 illustrates a connection arrival using a Poisson process of parameter λ.

In order to provide a quick answer whenever a connection arrives, we propose to use an expanded network to meet the constraints on Service Chain and give a precise mathematical formulation of the routing problem as an integer minimum-cost flow model with side constraints on the expanded network. Based on the Lagrangian relaxation of the flow problem [12], we propose several fast approximate algorithms to compute solutions to this problem. The results show that introducing function requirements can degrade significantly

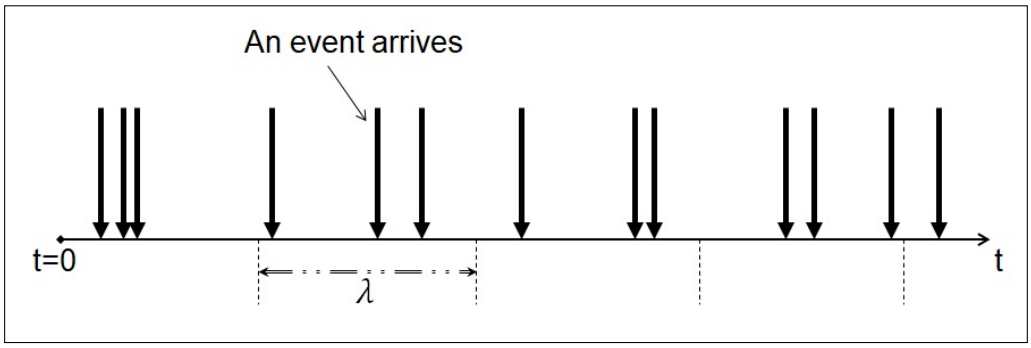


Figure 1.2: An Event-Driven Simulation

the network performance and that this is closely tied to the length of the paths produced by the shortest path algorithms.

In Chapter 5, we describe in detail a system model for routing via Functions in NFV Infrastructure and give a mathematical formulation of that problem. We also provide simulations that shows extensive results on their performance.

1.5 Contributions

Most proposed mathematical formulation for resource allocation in NFV Infrastructure have been carried out in the case where some constraints are not taken into account such as link constraints, delay constraints, or the order of VNFs in a service chain, etc. Recent works have considered the resource allocation with multiple types of constraints and multiple objectives. However, it is still necessary to better understand the right level of abstraction for a practical NFV deployment. In the joint problem of VNF placement and routing, the important questions about the locations of VNFs and the distribution of resources allocating to them still hold: 1) How and where to locate and chain VNFs? 2) How to distribute resources to run VNFs? 3) How to route effectively in the NFV system? In addition, the complexity of the solutions should be managed, especially with the rapid growth of the current network. The dissertation focuses on an optimal solution with multiple objectives including minimizing the resource cost on both nodes and links, maximizing the acceptance ratio and minimizing penalty cost for VNF migrating. The most important contributions of this dissertation are the efficient, generic and practical solutions for the resource allocation problem in an NFV Infrastructure. They consist of (*i*) a solution for minimizing the NFV deployment cost while considering multiple constraints and multiple objective in the resource allocation problem in NFV; (*ii*) an optimal solution for minimizing the resource utilization in an NFV dynamic system; and (*iii*) a solution to quickly find the best path in an NFV system that can scale to large networks. The analytical results are

validated in many scenarios, using both random and real datasets, provide an accurate and effective evaluation for the best locations to put VNFs and how to route requests in an NFV system. The main contributions of this dissertation are as follows:

- First is the mathematical formulation of a comprehensive resource management and placement problem that takes into account the order of VNFs in a service chaining and the resource constraints for both nodes and links. We address the problem with multiple objectives and combine both OFFLINE case and ONLINE case. The model is generic, able to handle a large and diverse set of key parameters and can easily be customized. We evaluate the performance of our solutions with a large set of parameters. The analysis results provide a useful suggestion for the improvement in the performance of VNFs deployment. Moreover, the solution provides a simple migration technique for the dynamic system and guidelines for the percentage of traffic to be migrated in order to achieve the best gain.

- Second is an extended model, Linear Programming, to provide exact solutions for the resource utilization problem in NFV. The model is simple and can be applied to a large dynamic system. The main idea is that we provide an interesting method for generating some strong inequalitiesin order to improve the LP solving in a higher dimensional space. In addition, we propose two heuristic algorithms to find the feasible solution in an acceptable execution time. The obtained results provide us the optimal allocation for each required VNF regarding the resource utilization of an NFV dynamic system.

- Third is the solution for the routing problem in large systems. It provides a quick answer whenever a customer request arrives. We proposed two classes of algorithms based on the construction of an expanded network to take into account the function constraints and the Lagrangian decomposition. In order to evaluate the performance of the solution, we use an event-driven stochastic simulation of a set of typical networks. The results confirms again that the placement of functions play a significant role since this will impact the path length needed to meet the requirements.

1.6 Structure of the Document

The remaining of the dissertation is structured as follows. Chapter 2 provides a state of the art of NFV systems in general and the resource allocation problem in NFV Infrastructure in particular. Chapter 3 describes analytical results for the resource management and placement problem in NFV and analyzes the impacts of input parameters on the performance of placing VNFs to the NFV Infrastructure. Chapter 4 presents an optimal solution

for the resource utilization problem of an NFV dynamic system and provides a method to generate some strong inequalities for LP solving in a higher dimension space. Chapter 5 proposes a solution for the routing problem in NFV system using an expanded network. The last chapter summarizes the dissertation and discusses directions for future research.

The dissertation are organized so that each chapter is relatively self-contained. Chapters 3, 4 and 5 give more details to the ideas outlined in Section 1.4. The combination of solutions presented in the three chapters provide a comprehensive study that develops from the performance evaluation to a solution for the resource allocation in NFVI with multiple objectives. The reader can read the Introduction chapter first to quickly get a general view and capture the key issues. The remaining chapters can be read subsequently to complete the picture.

Chapter 2

The State of the Art

Contents

2.1 Introduction

NFV emerges as an initiative from the industry in order to increase the deployment flexibility and integration of new network services and to obtain significant reductions in operating expenditures and capital expenditures. NFV promotes Virtualizing Network Functions such as Transcoders, Firewall, and Load Balancers, among others, which were carried out by specialized hardware devices and migrating them to software based appliances. With the emergence of the NFV concepts [13] provided by the European Telecommunications Standards Institute, there are a lot of attentions from several other standard organizations, academic and industrial research projects, and vendors in dealing with diverse objectives.

Since these evolutions have motivated this thesis and our investigations, this chapter will provide a comprehensive state of the art of Resource Allocation in NFV by introducing a novel classification of the main approaches that pose the solutions to solve it.

This chapter begins with some backgrounds related to the NFV architecture to provide basic concepts in an NFV system. We then present three topics related directly to Resource Allocation in NFV including: (i) Service Function Chaining; (ii) Service Orchestration and Management; (iii) Traffic Engineering. In each section, we provide an overview of each problem, provisioning challenges and surveys some existing models and algorithms. Finally, the crucial research directions for further exploration will be discussed at the end of this chapter.

2.2 NFV Architecture

Network Function Virtualization is a recent initiative from the industry [1], [14], [15]. The goal of NFV is transforming the way network operators and network providers design, manage and deploy their network infrastructure thanks to the evolution of virtualization technologies. This section will provide a short background on NFV, including relevant concepts, its architecture framework and use cases.

2.2.1 NFV Concepts

There are many concepts on NFV detailed in [13], [16]. Here, we summarize some concepts that are used in the dissertation.

1. *Network Function (NF)*: A functional building block within a network infrastructure, which has well-defined interfaces and functional behavior such as Firewall, Load Balancer, Deep Packet Inspection (DPI)...

2. *Virtualized Network Function (VNF)*: A software implementation of NF that can be deployed in a virtualized infrastructure.

3. *Network Service (NS)*: A composition of a chaining of VNFs in a given preference order and defined by its functional and behavioral specification. Fig. 2.1 is an example of a Network Service composed by four VNFs: Firewall, Network Address Translation (NAT), Dynamic Host Configuration Protocol (DHCP), and DPI.

4. *VNF Forwarding Graph*: A graph of logical links connecting NF nodes for the purpose of describing traffic flow between these NFs. In VNF Forwarding Graph, network connectivity order is an important characteristic.

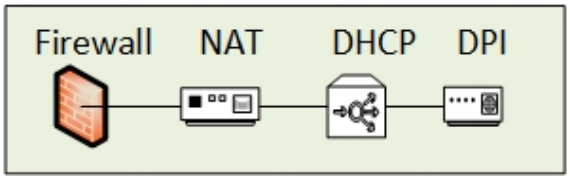


Figure 2.1: An example of a Network Service

5. *NFVI Point of Presence (PoP)*: A location where a NF is implemented as either a Physical Network Function (PNF) or a Virtual Network Function.

6. *NFV Infrastructure (NFVI)*: The set of all hardware and software components which build up the environment in which VNFs are deployed. The NFVI can span across several locations, i.e., multiple N-PoPs. The network providing connectivity between these location is regard to be part of the NFVI.

7. *NFV Orchestrator*: A key component of the NFV-MANO (Network Functions Virtualization Management and Orchestration) architectural framework, which operates, manages and automates the distributed NFV Infrastructure. It provides GUI and external NFV-Interfaces to the outside world to interact with the orchestration software.

2.2.2 NFV Architecture

VNFs can be deployed and reassigned to share different physical and virtual resources of the infrastructure, so as to guarantee scalability and performance requirements. In general, there are three main components in the NFV architecture: NFVI, Services and NFV-MANO, see in Fig. 2.2.

1. *NFVI*: NFV Infrastructure is composed of NFVI-PoPs which contains hardware and software components used to run all the VNFs needed for the network. NFVI networks interconnect the computing and storage resources contained in an NFVI-PoP.

2. *Services*: A set of VNFs, that can be implemented in one or multiple virtual machines. A VNF is usually administered by an Element Management System (EMS) that plays role for its creation, configuration, monitoring, performance and security. An EMS provides the essential information required by the Operations Support System (OSS) and Business Support System (BSS).

3. *NFV-MANO*: NFV Management and Orchestrator covers the orchestration and lifecycle management of physical and/or software resources that support the infrastructure virtualisation, and the lifecycle management of VNFs. VNF-MANO focuses on

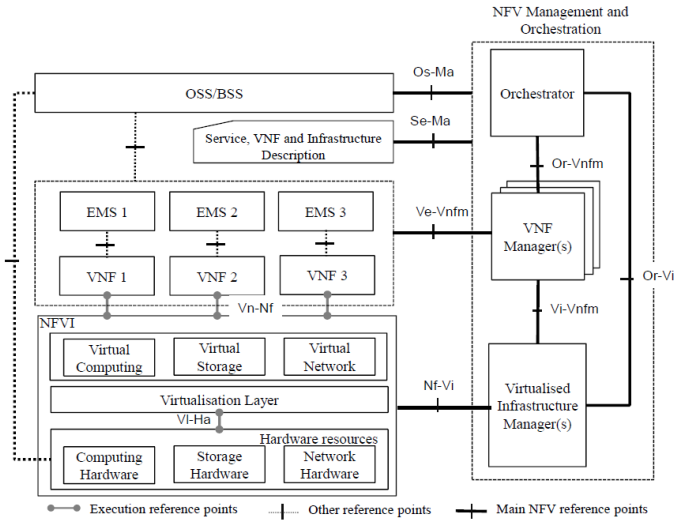


Figure 2.2: ETSI NFV Architecture Framework

the virtualization-specific management tasks necessary in the NFV framework. The NFV-MANO also interacts with the (NFV external) OSS/BSS landscape, which allows NFV to be integrated into an already existing network-wide management landscape. It is composed of: the orchestrator, VNFs managers and Virtualized Infrastructure Managers. Such blocks provide the functionality required for the management tasks applied to the VNFs, e.g., provisioning and configuration [14] [17].

In short, if a customer want to use an NS composed by two NFs: a Firewall and a Load Balancer, two VNFs are deployed. NFV-MANO then will be responsible to say where these VNFs should be located on the physical network. In turn, these VNFs are controlled by the EMS and the same MANO. Besides, the virtualization layer exposes the physical resources of chosen NFVI locations to the VNFs.

In order to clarify the roles and interactions of the various types of commercial entities acting in a marketplace for services delivered via these VNFs, potential use-cases are required to represent and develop specific technical and business opportunities. Some use-cases described in [18] by ETSI such as NFVI as a Service, VNF as a Service (VNFaaS), Virtual Network Platform as a Service, VNF Forwarding Graphs, Virtualisation of Mobile Core Network and IMS, etc. In this dissertation, we choose one use case to better understand how to make the VNF functionality available to the enterprise as a service and the advantages of the VNFaaS.

2.2.3 Use Case: Virtual Network Function as a Service

This section describes a Virtual Network Function as a Service (VNFaaS) use case that permits to access VNFs to perform some kind of service chaining with other generic network functions.

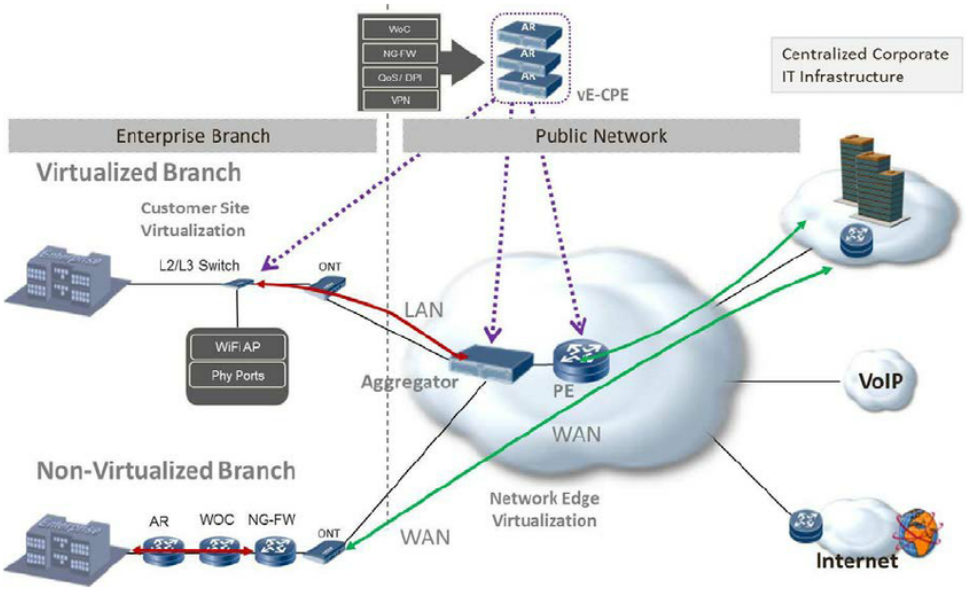


Figure 2.3: Non-Virtualised CPE and vE-CPE

Many enterprises are currently implementing more services at the edges of their networks. Despite this behavior, the use of dedicated equipment with high cost, technological inflexibility, delayed installation and difficult maintenance, makes this process difficult. As the enterprise continues to evolve, more services and applications migrate to data centers or public clouds [19], causing non-stop changes in the structure of today's networks. All of these changes require big investments, forcing enterprises to seek outsourcing alternatives for the provision of services. The possible virtualizaion targets may be enterprise access router/enterprise CPE, provider edge router, enterprise firewall, enterprise NG-FW, enterprise WAN optimization, deep packet inspection (appliance or a function), IPS and other security appliances, network performance monitoring. In this scenario, CPE virtualization has become an alternative, as the enterprise does not have to invest additional capital, rather it can benefit from the Service Providers (SPs) that efficiently leverage existing resources in their data centers, virtualizing with the NFV. In VNFaaS, the NFV service provided by the SPs is similar to Software as a Service (SaaS) running on a cloud infrastructure. The VNF is a service provider's application. The enterprise is a consumer of the service. The Service Provider can scale the NFVI resources allocated to the VNF instance

in response to increasing usage of the VNF. In this environment, operators can virtualize the edge-owner (PE), although there are greater benefits in virtualizing the CPE because there are a lot more CPEs and their performance is only local. The vE-CPE can replace a whole set of existing equipments on the customer side by NFV solutions located at either the enterprise cloud or the operator of the NFV framework. vE-CPE is able to provide routing services with QoS, 7-tier firewall, intrusion detection and prevention, application acceleration, and more, all without the need to include special hardware.

Fig. 2.3 provides the functionality re-distribution as a result of the virtualisation of the CPE being made up of two parts: virtualized and non-virtualized CPE. In the virtualized CPE, the local traffic is handled by a local L2 or L3 switch providing physical connectivity with the client. While the functionalities provided by the CPE are virtualized including routing, VPN termination, QoS support, DPI, NG-FW and a WOC (WAN Optimization Controller). The dotted purple lines indicate where this vE-CPE functionality may be located.

Both the VNFaaS Provider and the user share the responsibility for managing the vPE and vE-CPE. Enterprise users expect to manage and configure their CPE devices and manage SW versions when upgrades happen, even when they are virtualized and provided as a service. As a result, VNFaaS would need the appropriately management mechanism to be used as the basis of service billing arrangements.

We have given a simple illustration of an NFV Architecture. In next sections, we focus on three topics related directly to this dissertation and provide the survey of these topics.

2.3 Service Function Chaining for NFV

Service Function Chaining (SFC) is a common abstraction for the expression of network service requirements. A service chaining represents the exact ordered sequence of Network Functions traversed by one or multiple flows.

Fig. 2.4 illustrates how data travel from source to destination with and without the introduction of a dynamic SFC architecture. Today, each and every data package has to be processed by a predefined series of (often dedicated-hardware at the customer premises) “services” such as a security gateway service, a Firewall, a Load balancer, a Deep Packet Inspection service, and so on. However, this approach is costly and cumbersome to manage especially when a new function is added to a service. By using virtualization technology, NFV decouples functions from dedicated hardware and move them to virtual servers. So Service Providers can reduce the deployment cost when performing network functions. Fig. 2.4 is representative of what SFC principles can accomplish, resulting in a dynamic, software-configurable and upgradeable system. Network service can be implemented as part

of a dynamic chain where each flow is processed by various service functions thus avoiding the need for deploying different physical network elements.

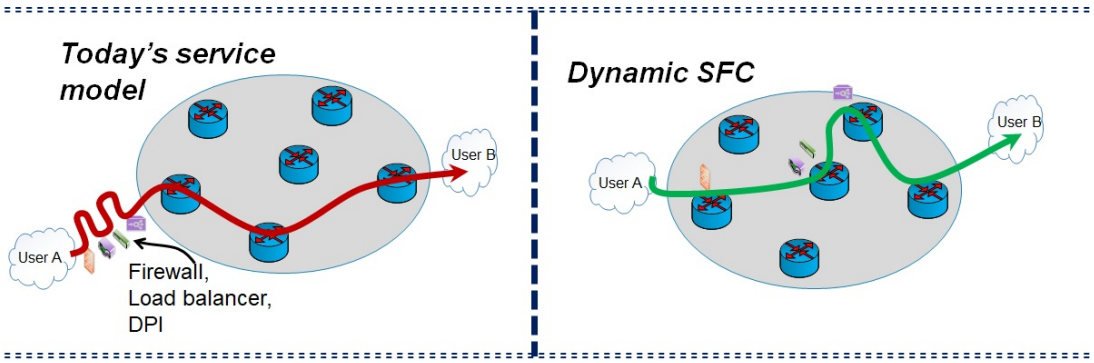


Figure 2.4: A dynamic service function chaining

The problem consists of making sure network flows go efficiently through end-to-end paths traversing several set of middleboxes that provide Network Functions. This is particularly challenging mainly for two reasons. First, depending on how virtual network function are provisioned and chained, end-to-end latencies may become intolerable. Second, resource allocation must be performed in a cost-effective manner, preventing over- or under-provisioning of resources. Therefore, placing network functions and deploying SFC in a cost-effective manner while ensuring controlled end-to-end delays represents an essential step toward enabling the use of NFV in production environment. Some works have already been done to efficiently perform the placement and chaining of virtual network functions. Among various issues, the placement of VNFs and Services Chains has called for several studies. It is directly related to the VNF placement and routing problem. The problem has been proven to be NP-complete [20]. The surveys [21], [22] provide valuable references to relevant previous works. In [6], Herrera and Botero present a comprehensive state of the art of NFV resource allocation (NFV-RA) problem. In their novel classification, the main approaches for solving NFV-RA are described as three stages: VNFs - Chain Composition (Section 2.3.1), VNF - Forwarding Graph Embedding (Section 2.3.2) and VNF - Scheduling (Section 2.3.3).

2.3.1 VNFs - Chain Composition

The task of the first stage of the problem VNFs - Chain Composition (VNF-CC) is to efficiently build a suitable VNF Forwarding Graph (VNF-FG) with regard to the operator's goals. Fig.2.5 [23] depicts the case of a nested VNF-FG constructed from three network functions VNF-1, VNF-2 and VNF-3. They are interconnected via logical links (the dash line). It is due to the fact that performance of Network Service will be affected by both

the different composing functions' behavior and the order in which functions are processed. Therefore, it is paramount to achieve an efficient service chain composition with regard to network operator's objectives.

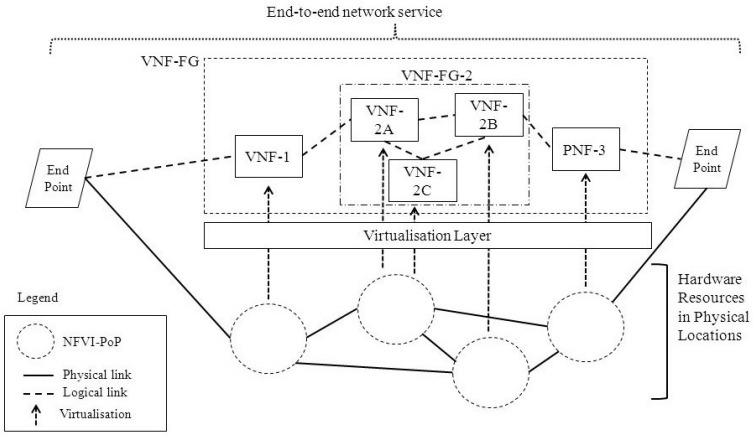


Figure 2.5: VNF Forwarding Graph

However, most NFV resource allocation proposals consider the VNF-FG as an input of the problem, i.e., the chain composition stage is neglected. Hence, several approaches have been proposed to solve VNF-CC such as [24], [25], [26].

In particularly, Beck [25] proposed a coordinated heuristic that generates and embeds VNF-FGs simultaneously. Their heuristic aims to minimize bandwidth utilization while computing results with reasonable runtime.

Recently, in [26], Mehraghdam *et al.* introduced a formal model for specifying VNF chaining requests, including a context-free language for denoting complex composition of VNF. They describe the mapping as a Mixed Integer Quadratically Constrained Program (MIQCP) for finding the placement of the network functions and chaining them together considering the limited network resources and requirements of the functions.

2.3.2 VNF - Forwarding Graph Embedding

The resulting graph in the first stage is given as the input of this embedding stage. Each VNF-FG is composed by the ordered set of VNFs, called *Network Service*. The task of the second stage VNF - Forwarding Graph Embedding (VNF-FGE) is to find allocation of the VNFs in the network infrastructure in a suitable way considering a set of requested network services. Fig.2.6 shows the deployment of an end-to-end network service composed by five VNFs (Firewall → Load Balancing → Encryption → PacketInspection → Decryption) [22]. Here, as in the Virtual Network Embedding (VNE) problem [9], [27], there are virtual node

and link mapping phases. The result of this stage includes the allocation of VNFs and the path from a source to a destination going through the ordered set of these VNFs for each request. For example, in Fig. 2.6, in the virtual node mapping phase, VNF1 hosted onto HVS1, VNF2 hosted onto HVS2, and so on. Where as in the virtual link mapping phase, each virtual link has to map on the dash line of the NFVI.

Obviously, a VNF-FGE algorithm depends on the objective to optimize. Many previous works focused on various objectives, e.g., maximization of the remaining network resources, minimization of the power consumption, minimization of the network utilization [28–33, 33–44]. In recent year, most of existing NFV Resource Allocation approaches solve just the VNF-FGE stage, they consider the VNF-FG as a given input of the problem.

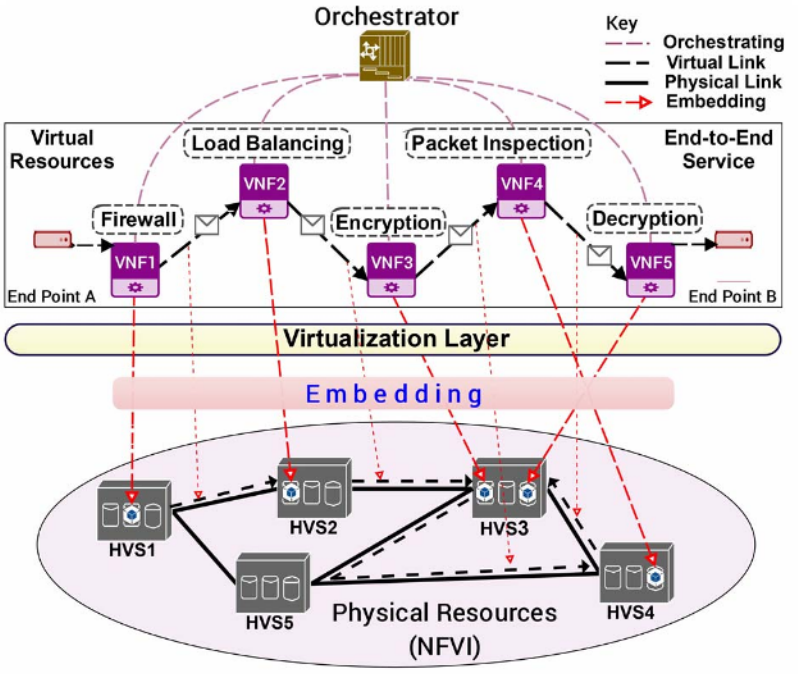


Figure 2.6: VNF Forwarding Graph Embedding

For example, [45] formalized the network function placement and chaining problem and propose an optimization model to solve it. They have proposed a binary search heuristic to jointly place VNFs and map service chains onto them without any ordering constraints. The objective is to minimize the number of functions while meeting end-to-end delay guarantees.

In additional, some mathematical models, including linear and non-linear programming, have been proposed in [46], [47], [48], [49] and [50]. In particularly, in [46], Lukovszki *et al.* formulated the offline (SCEP: Service Chain Embedding Problem) and online problems

(OSCEP: Online SCEP). They assumed that all potential paths that can be routed through VNFs chaining for each demand are known.

PACE [47] addressed the VNF placement for unordered service chains in cloud, with the objective to satisfy as many tenant's requests as possible. Also, Elias *et al.* [48] formulated the centralized version of VNF-FGE as a non-linear integer optimization model and assume that the placement of functions is fixed.

Recently, Sun *et al.* investigated the offline and online solutions for the VNF placement problem with the aim of minimizing the deployment cost [50]. They took into account the chaining of VNFs, but they only considered a pre-defined set of service chaining.

Ma *et al.* [51] proposed a network function virtualization scheme, depending on the dynamic requirements of the network. They developed an algorithm based on Min-Max routing algorithm. Liu *et al.* [52] also jointly optimized the deployment of new users' SFCs and the readjustment of in-service users' SFCs while considering the trade-off between resource consumption and operational overhead. They formulated a path-based ILP model to solve the problem exactly. They assumed that a set of pre-calculated paths between any pair of nodes are given. They also developed an approximate algorithm based on Column Generation (CG) model to further accelerate the problem-solving. In their simulation, they used the 5-shortest paths between two nodes to replace the total paths. Although they used a CG model to reduce the time complexity, their model only solved with small topologies such as 6-nodes topology and 14-nodes NSFNET topology.

In [53], Addis *et al.* defined the generic VNF chain routing optimization problem and devised a mixed integer linear programming formulation. Their model takes into consideration specific NFV forwarding models (standard and fastpath modes) as well as flow bit-rate variations that make the allocation of edge demands over VNF chains unique yet complex.

Recently, Kuo et al. [54] studied the joint problem of VNF placement and path selection to better utilize the network. They studied the relation between the path length and the virtual machine reuse factor. However, they did not formulate the problem as a mathematical programing model. They focused on proposing a systematic way to estimate a proper path length and reuse factor for each demand.

2.3.3 VNFs - Scheduling

A third and final stage of the NFV Resource Allocation problem is the scheduling process, that we call VNFs - Scheduling (VNF-SCH). This stage is to answer to the following question: which ideal VNFs are assign to resources at particular time considering the ordered chaining of VNFs?

For this stage, little research has been conducted on VNFs scheduling. For instance, Rieara *et al.* [55], [56] provided the first formalization of the scheduling problem in NFV

as a Resource Constrained Project Scheduling Problem. However, although the problem is formulated, no solution is proposed to deal with the problem.

Recently, Shifrin *et al.* [57] modeled the problem of virtual network function allocation and scheduling by queuing system with flexible number of queues and introduce a control model based on Markov Decision Process (MDP) formulation. They showed analytically and by simulations that the optimal policy possesses decision thresholds which depend on several parameters.

In this dissertation, we focus on the second stage VNF-FGE that considers the VNF-FG as an input of the problem. In contrast with past work, we overcome several issues in relaxing the problem model such as expressing the order of VNFs in a chaining, considering the resource constraints on both nodes and links, taking into account the arrival time and processing time of each customer demand in the dynamic scenario, as detailed in Chapter 3 and 4.

2.4 Service Orchestration and Management in NFV

In the previous section, VNF locations can be determined intelligently to ensure efficient traffic routing. NFV also opens up the opportunity to simultaneously optimize VNF locations and traffic routing paths, which can significantly reduce the network OPEX. SFC can be orchestrated by dynamically deploying a composition of VNFs either on a single server or on a cluster of servers. However, placing just enough VNFs to match traffic processing requirements may yield to lowest deployment and energy costs, but steering traffic through these VNFs will increase traffic forwarding cost and may eventually lead to Service Level Agreement (SLA) violations. As a consequence, Service Orchestration and Management is also one of the most challenging problem in NFV systems. An optimal VNF orchestration strategy must address these issues during the cost minimization process. Fig. 2.7 illustrates the high-level NFV framework [23]. In this figure, NFV Mangement and Orchestration covers the orchestration and lifecycle management of physical and/or software resources that support the infrastructure virtualisation, and the lifecycle management of VNFs.

In this section, we survey relevant researches in the literature related to Service Orchestration and Management issues in NFV. At first, in [58], Bari *et al.* identified the VNF orchestration problem and provide an Integer Linear Program (ILP) for this problem. Their model can be used to determine the optimal number of VNFs and to place them at the optimal locations to optimize network operational cost and resource utilization. But they relaxed the order constrain of VNFs in a chain.

Moreover, some appropriate ways to deal with NFV management and orchestration problems are adressed in the literature [59], [60], [20], [61–68]. In [59], the authors consider

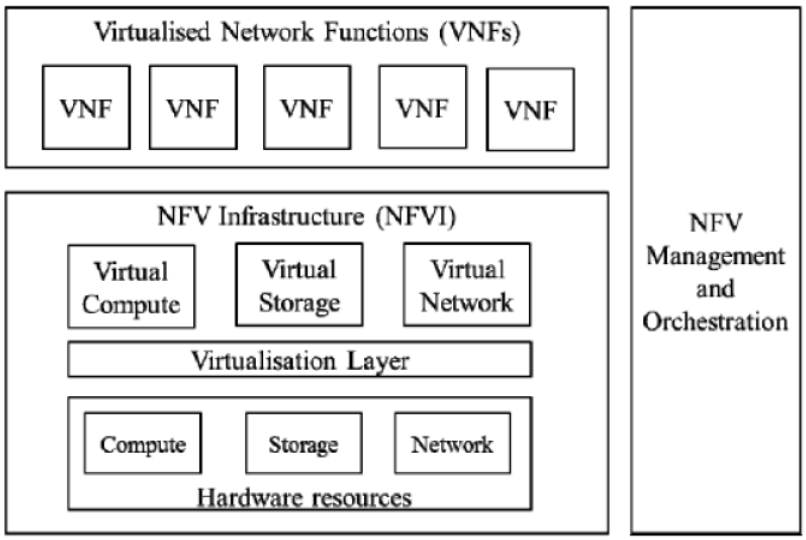


Figure 2.7: High Level NFV framwork

the online orchestration of VNFs and proposing heuristic to scale with the online nature of the framework. In [60], the authors consider a VNF-Placement and Routing problem for data-center environments with both optical and electronic elements, formulating the problem as a binary integer programming problem, and propose a heuristic algorithm to solve it. In their work, VNF chains are set as an input to the problem. In [20], the specific VNF (Deep Packet Inspection) node placement problem (with no chaining) is targeted, with a formal definition of the problem and a greedy heuristic algorithm to solve it.

In [69], L. Wang *et al.* proposed a coordinated approach JoraNFV to jointly optimization three steps in NFV resource allocation. They proposed a comprehensive cost model to take in account for capital cost, operating cost and link cost as a mixed-integer linear programming (MILP).

Dietrich*et al.* [70] also derived a linear programming formulation for service chain partitioning and introduced a new service model to simplify the specification of network service requests and deal with traffic scaling NFs. Their evaluation study is focused on multi-provider aspects and more specifically, on the impact of contrasting provider and client objectives on Network Service Embedding (NSE) efficiency in terms of request acceptance rate, generated revenue, service cost, and network load balancing.

In [71], Rankothge *et al.* presented a comprehensive analysis on the proposed Genetic Programming based resource allocation algorithms for new VNFs provisioning and scaling of existing VNFs with the traffic changes.

However, previously described work formulates the network function placement problem without considering the order of network function in the chain that is a sensitive characteristic in NFV system. Moreover, in this dissertation, we also investigate a migration technique to orchestrate resources efficiently for the dynamic system, see detail in Chapter 3 and 4.

2.5 Traffic Engineering in NFV

Traffic dynamics often trigger for reallocation and reconfiguration of network resources. In case of high demand, some resources end up being over-utilized, resulting in to higher latency and SLA degradation, while on other occasions, end up being underutilized. In such circumstances, in order to meet the performance and energy objectives, the NF instances (NFIs) need to be dynamically instantiated or decommissioned or even relocated/migrated. However, to make it happen, several key decisions need to be made in terms of knowing when to instantiate, decommission or migrate the instance, which network instances need to be scaled, where in the network to place the instances and how to redistribute the load among the available instances.

In several contributions, the network traffic cost has been considered specifically in the NFV context. For example, Qu *et al* [72] proposed a delay-aware solution for VNF scheduling and resource allocation for services. They consider VNF transmission and processing delays and formulate the joint problem of VNF scheduling and traffic steering as a mixed linear program.

In [37], the authors studied the joint problem of VNF placement and path selection to better utilize the network. They focus on the relation between the link and server usage.

In [73], Guo *et al.* investigated the problem of maximizing throughput in SDN-enabled networks with respect to service chaining specifications under both traditional and new constraints. They developed two types of routing schemes, including Offline and Online.

In [74], authors devised a traffic-aware and energy-efficient VNF placement for service chaining that optimizes both the operational and network traffic cost, considering the heterogeneous physical nodes and workload.

More recently, Addis *et al.* [75] proposed an ILP model that considers the traffic compression induced by multimedia gateways or firewalls for the cost efficient placement of VNF chains. They applied linearization techniques to increase the execution speed of an ILP solver (i.e. CPLEX).

In 2016, Leivadeas *et al.* [76] studied the problem of deploying service chains on a SDN enabled data center network. They proposed algorithms to recalculate the routing paths as to adjust the dynamic traffic. They focused on accommodating the service chains on servers and orchestrating network resources for the SDN controller, they do not consider

where the origin and destination of user/tenant requests are. We overcome this issue by providing some efficient approximate algorithms in Chapter 5.

2.6 Summary

This chapter describe 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. For example, many possible constraints that may be encountered in a real environment such as delay, congestion or multi-path considerations should be expressed in NFV system.

The next chapters describe the thesis contributions in terms of resource management and placement models, optimization algorithms as well as some effective routing algorithms in the NFV infrastructure.

Chapter 3

Resource Management and Placement for NFV

Contents

In the previous chapter, the emergence of advanced software virtualization techniques leads to a level of performance suitable for the future network. This evolution tends to get potential gains in operational expenditures, hence basic ISPs components such as routing switching, firewall, and inspection equipments are now considered as Virtual Network Functions nodes. Since VNFs are deployed on NFV system, it raises important questions about the resource management regarding the order of VNFs in a chaining of customer requests. Thus, we begin our study by developing a comprehensive resource management model in NFV system while taking into account the execution order of VNFs in a service

demand with regard to resource constraints for nodes and links. Specially, we address both static and dynamic scenarios.

The chapter is organized as follows. The next section defines the problem of managing resources for network system with NFV in a context of our work. Section 3.2 describes our NFV placement model and formalize as a MIP. Section 3.3 present some heuristics to solve the problem. Section 3.4 validates the algorithms and analyze the impact of input parameters on the performance metrics. Finally, Section 3.5 discusses the limitations and extensions of our results and summarizes the chapter.

3.1 Context

Softwerization of network components is a candidate to provide the agility requested by the increasing on-demand need of customers as well as the ability to gently match those needs with the resource consumption. This trend is achieved by using a cloud approach where virtualization techniques are intensively exploited. Although this trend is prevalent, it is still necessary to better understand the right level of abstraction and hence performance that will be made possible with this approach. NFV relying upon virtualization techniques should support network operators to meet the growing customer requirements while controlling capital and operational expenditures.

In order to enable such an agile solution, a network service can be decomposed into an ordered sequence of VNFs, which can run on several standard physical nodes. Therefore, the resources allocated to a VNF instance will impact the capacity and performance of the network service as a whole. This raises important issues related to NFV deployment such as: 1) How and where to locate and chain VNFs? 2) How to distribute resources to VNFs? and 3) What is the cost of NFV deployment in a network? Such problems are different from the VM placement optimization in cloud computing as VNFs are associated with a single network service. In this chapter, we consider a joint problem of VNFs allocation upon service requests from users and routing paths for chaining them. We develop a comprehensive solution addressing the static and dynamic scenario, when taking into account the execution order of VNFs in a service demand with regard to resource constraints for nodes and links.

The major contributions of our work are as follows:

- We formulate the optimization problem of VNF placement as a quadratic program (QP) solving multiple objectives including optimal service location and optimal routing among VNF instances of a network service, under both resource and traffic cost constraints, simultaneously optimizing the acceptance ratio of new demands. Both the static and dynamic problems are considered.

- We propose three efficient heuristics for large dimensions of the problem within an acceptable computation time.
- We evaluate our proposed solutions in various scenarios and compare it against the ProvisionTraffic (PT) algorithm [58] that illustrates the state of the art, as well as a Random Algorithm.
- We provide a cost comparison between a network relying on NFV and a network that does not use NFV and discuss some architectural implications.
- We also propose a simple migration technique for the dynamic problem. The result provides guidelines for the percentage of traffic to be migrated in order to achieve the best gain.

3.2 Problem Statement

We first describe a resource allocation problem in a system implementing NFV. Then, we formulate the problem as a quadratic program (QP) dealing with both optimal service location and optimal routing under constraints on service function chaining and system resources.

3.2.1 System Description

In this section, we introduce two scenarios: the static problem (OFFLINE case) and the dynamic problem (ONLINE case).

3.2.1.1 OFFLINE Case

We study a NFV system where a network function is offered as a service. The system is composed of three components, including a set of VNFs, a set of customer demands and a virtual network that provides resources to the VNFs. VNF is a software implementation of a Network Function (NF) deployed on an NFVI. Each demand requests access to a network service (NS) that requires special functionalities. It can be composed of several VNFs in order to support advanced network connectivity, e.g. deep packet inspection, firewall, load balancer. In a NS, VNFs will need diverse resources such as CPU, memory, storage, bandwidth, etc. It is recognized that virtualized solutions are quite costly in resource consumption. Resources are distributed anywhere on the NFVI. A customer demand materializes as a packet flow from a source to a destination visiting a set of nodes where the required VNFs are deployed.

Table 3.1: Summary of Variables

Variable	**Meaning**
(n,m,l)	Number of virtual nodes, VNFs, and demands
(V,E)	Set of virtual nodes and links
(D_n,D_o)	Set of new demands and old demands
$D_m \subseteq D_o$	Set of old demands that will be migrated
F	Set of VNFs
R	Set of resources (CPU, STORAGE, MEMORY, etc.)
F^d	Set of ordered VNFs of demand d
ℓ_d	The number of VNFs in set F^d
ϕ_f^d	The order of function f in the chain of VNFs F^d
$w_{v_1v_2}$	Bandwidth capacity of link between $v_1, v_2 \in V$
c_v^r	Resource capacity $r \in R$ of node $v \in V$
λ^d	SLA coefficient of demand d
$p_v(\lambda^d)$	The price of resource charged per hour of node v, according to λ^d
μ_f^r	Required resource $r \in R$ of function $f \in F$
(s^d,t^d)	Source node and destination node of demand d
b^d	Required bandwidth of demand d
(a_s^d,Δ^d)	Starting time and processing delay of demand d
p_b	Price of bandwidth charged per hour
η	Migration percentage
x_{fv}^d	A binary variable that equals to 1 if and only if virtual node v hosts function f of demand d
y_{uv}^d	A binary variable that equals to 1 if and only if link (u, v) is used by demand d
χ_{uv}^d	A binary variable that equals to 1 if and only if node u is a ancestor node of v in the routing path of d
z^d	An integer variable that equals to 1 if new demand d is served, -1 if old demand d is migrated and 0 otherwise
$\hat{x}_{fv}^d$	A binary variable that equals to 1 if and only if virtual node v hosts function f of old demand d
$\hat{y}_{uv}^d$	A binary variable that equals to 1 if and only if link (u, v) is used by old demand d
(min_l,max_l)	Minimum and maximum number of demands

How to efficiently share resources for all demands with delay constraints or limited capacity is an important challenge for NFV providers. It is likely that the solution achieved will trade-off agility with efficiency. For instance, a NFV solution may take a longer path than the likely shortest path followed by a non-NFV system. However, it is still applicable if the delay lies within an acceptable range.

A virtual network is represented as an edge-weighted undirected graph denoted by $G = (V, E, w)$ where V is a set of virtual nodes, E is a set of links, and w is the bandwidth capacity assigned to each link. Each node $v \in V$ is associated with two components: 1) c_v^r that denotes a set of resource capacities with $r \in R$, where R denotes the set of resources (CPU, STORAGE, and MEMORY, etc.) and 2) $p_v(\lambda^d)$ that is the price of resource charged per hour ($/hour) according to a Service Level Agreement (SLA) coefficient λ^d of the customer d. We denote D_n as a set of l customer demands. A demand $d \in D_n$ is represented by a set of parameters $(s^d, t^d, b^d, F^d, \Delta^d)$ where s^d is a source, t^d is a destination, b^d is a required bandwidth, $F^d = \{f_{i_1}, ..., f_{i_k}\}$ is an ordered chaining of sub-set VNFs and Δ^d is an average total processing time of demand d. We assume that each VNF $f_i \in F^d$ only appears once in each demand $d \in D_n$. We define $F = \{f_i | i = \{1, ..., m\}\}$ as a set of m VNFs. Each function $f_i \in F$ has a vector $\mu_{f_i}^r$ representing its resource requirement ($r \in R$). In the OFFLINE model, we minimize both the computing-resource cost and the bandwidth cost under constraints on resource and bandwidth capacity with respect to VNFs chaining.

3.2.1.2 ONLINE Case

We consider a scenario where an operational network is serving a set of customer demands D_o (called old demands) . At this stage, a set of VNFs is already deployed and the routing paths for the demands in D_o are also provisioned.Let's consider that new demands D_n are incoming and that the network operator needs to provision the required VNFs and routing paths for them. Maximizing the number of served demands and minimizing the system resource cost are the research challenges addressed in the sequel.

We consider an interval τ (according to the system configuration) where we will update the status of all demands running in the system. Upon completion, the system releases the resources allocated to a demand. At time t, we collect new demands D_n that arrived during the period $(t - \tau)$ to t.

We propose a method to select the subset of old demands to be migrated in Section-3.3.2.2.

The ONLINE model addresses multiple objectives:

- Minimize resource cost on nodes to satisfy the demands
- Minimize bandwidth cost on links from source to destination passing through VNFs

- Maximize the number of accepted demands
- Minimize the penalty cost for VNF migration

3.2.2 Problem Formulation

We aggregate both cases and formulate the above system as a quadratic problem with the following inputs:

- Set of m functions $F = \{f_i | i = \{1, ..., m\}\}$.
  - μ_f^r: a resource requirement of function $f \in F$
- D_o being defined as set of old demands is characterized as $D_o = \{(s^d, t^d, b^d, F^d, \Delta^d, a_s^d, \hat{x}_{vf}^d, \hat{y}_{uv}^d)\}$ where a_s^d is the starting time, and $(\hat{x}_{vf}^d, \hat{y}_{uv}^d)$ is their solution deployed on NFVI.
- $D_m \subseteq D_o$ is the set of old demands selected to be migrated. A migration percentage is denoted by $\eta = \frac{|D_m|}{|D_o|}$
- Set of new demands $D_n = \{(s^d, t^d, b^d, F^d, \Delta^d)\}$
- An edge-weighted undirected graph $G = (V, E, w)$ is representing the virtual network.

The notations can be found in Table 3.1.
We define the outputs of our problem as a set of decision variables as follows:

- x_{fv}^d is a binary variable that equals to 1 if and only if node v hosts function f of demand d.
- y_{uv}^d is a binary variable that equals to 1 if and only if demand d uses link (u, v).
- χ_{uv}^d is a binary variable that equals to 1 if and only if node u is an ancestor node of v in the routing path of demand d.
- z^d is a binary variable that equals to 1 if a new demand d is served or an old demand d is migrated, and equals to 0 otherwise.

We aim to:

1. Minimize nodes resource cost allocated for demands

$$U_n(t) = \left(\Delta^d - t + a_s^d\right) \sum_{d \in D_m \cup D_n} z_d x_{fv}^d p_v(\lambda_d) + \left(\Delta^d - t + a_s^d\right) \sum_{d \in D_o \setminus D_m} \hat{x}_{fv}^d p_v(\lambda_d) \tag{3.1}$$

2. Minimize bandwidth cost on links from source to destination passing through VNFs

$$U_l(t) = \left(\Delta^d - t + a_s^d\right) \sum_{d \in D_m \cup D_n} p_b z_d \sum_{u,v \in V} b^d y_{uv}^d + \left(\Delta^d - t + a_s^d\right) \sum_{d \in D_o \setminus D_m} p_b \sum_{u,v \in V} b^d \hat{y}_{uv}^d \tag{3.2}$$

3. Maximize the number of accepted demands

$$U_{accept} = \sum\nolimits_{d \in D_n} z^d \tag{3.3}$$

4. Minimize the penalty cost of VNF migration

$$U_{penalty} = \sum\nolimits_{d \in D_m; u,v \in V; f \in F} x_{fu}^d \hat{x}_{fv}^d z^d M(.) \tag{3.4}$$

$M(.)$ is a function of $dist_{uv}$, where $dist_{uv}$ characterizes the length of the shortest path from u to v. It indicates the cost to migrate a function from node u to node v. In our simulation, we use two different cost functions to consider the migration penalty: a linear function and a square root function, discussed later in Section-3.4.3.

Constraints:

In order to guarantee that a node capacity is not over-subscribed by the allocation of VNFs to this node, we have:

$$\sum_{d \in \{D_m \cup D_n\}} z^d x_{fv}^d \mu_v^r + \sum_{d \in D_o} \hat{x}_{fv}^d \mu_v^r \leq c_v^r \quad \forall v \in V \tag{3.5}$$

Similarly for the bandwidth capacity of a link.

$$\sum_{d \in \{D_m \cup D_n\}} z^d y_{uv}^d b^d + \sum_{d \in D_o} \hat{y}_{uv}^d b^d \leq w_{uv} \quad \forall u, v \in V \tag{3.6}$$

Constraint (3.7) guarantees that each VNF in a demand has to be processed in a given preference order.

$$x_{f_1 u}^d x_{f_2 v}^d - \chi_{uv}^d \leq 0 \qquad \phi_{f_1}^d < \phi_{f_2}^d \tag{3.7}$$

Each VNF only appears exactly once in a demand. This constraint is expressed as follows:

$$\sum\nolimits_{v \in V} x_{fv}^d = \begin{cases} z^d & \text{if } f \in F^d \\ 0 & \text{otherwise} \end{cases} \tag{3.8}$$

Constraint (3.9) ensures that the variables y_{uv}^d are non-symmetric.

$$y_{uv}^d + y_{vu}^d \leq 1 \qquad \forall u, v \in V, d \in D_m \cup D_n \tag{3.9}$$

Equations (3.10) enforce that, for each computer path associated to a demand, it always begins at its source and terminates at its destination.

$$\sum\nolimits_{v \in V} y_{uv}^d - \sum\nolimits_{v \in V} y_{vu}^d = \begin{cases} z^d & \text{if } u = s^d \\ -z^d & \text{if } u = t^d \\ 0 & \text{otherwise} \end{cases} \tag{3.10}$$

In addition, we need constraints (3.11), (3.12) and (3.13) to guarantee the compatibility between variables as follows:

$$\chi_{uv}^d - \sum_{i \in V, i \neq v} \chi_{ui}^d y_{iv}^d \geq 0 \qquad \forall d \in D_m \cup D_n \tag{3.11}$$

$$x_{fv}^d \sum_{i \in V} y_{iv}^d = x_{fv}^d \quad \forall d \in D_m \cup D_n, v \in V \setminus \{s^d, t^d\} \tag{3.12}$$

$$(\chi_{uv}^d - 1) y_{uv}^d = 0 \qquad \forall d \in D_m \cup D_n \tag{3.13}$$

Now, for the OFFLINE case, we simply set $D_o = \emptyset$.

3.2.3 Problem Complexity

In this section, we indicate that the NFV placement problem is NP-hard. In fact, we consider each demand, and we assume that we know the location of required functions (as a sub-problem of the proposed problem), our problem becomes the integral multi-commodity flow problem whose decision version is NP-complete. Obviously, our problem is harder than multi-commodity flow in special cases, which means that it can not be solved by a pseudo-polynomial time algorithm unless $\mathcal{P} = \mathcal{NP}$. Therefore, we have: We provide a polynomial reduction of the NP-hard Steiner Tree problem (STP) [77] to the NFV placement problem (NFV-P).

Theorem 1. *NFV Placement is strongly $\mathcal{NP}$-hard*

Proof. Let's consider the following problem:

Steiner Tree Problem: Given an edge-weighted graph $G = (V, E, w)$ and a subset $S \subseteq V$ of required vertices. Find a minimum-weight Steiner tree that spans all vertices of S.

We will show that STP $\trianglelefteq_P$ NFV-P. Let $T = (V_T, E_T, S)$ be the given STP in G with a set of vertices $V_T \in V$, a set of edges $E_T \in E$ and a set $S \subseteq V_T$ of Steiner points. We are going to construct a solution of NFV-P from T.

We consider each demand, one by one, of the NFV-P problem. For each demand, we form the subset of vertices $S \in V$ that can host VNFs and satisfy constraints in (3.5). We define a path P from s to d passing through the subset $S \subseteq V$ as follows:

$$\begin{aligned} P = \{&\text{the shortest path}_{i \in S}(s, i)\} \cup \{T\} \cup \\ &\{\text{ the shortest path}_{j:j \neq i, j \in S}(j, d)\} \end{aligned} \tag{3.14}$$

Path P is a part of an instance of the NFV-P problem (corresponding to one customer demand). The instance of NFV-P is then a set of path for l demands (constructing from l Steiner Trees), which we can easily create in polynomial time.

It follows from (3.14) that Steiner has a solution for each subset S if and only if $G(V, E)$ contains a path from s to t passing through S. Thus, we have a polynomial-time reduction from Steiner Tree Problem to NFV Placement and conclude that the NFV Placement is NP-hard. □

3.3 Approximate Algorithm

As our problem is NP-hard, we only apply exact algorithms to small size dimension of our system that runs within an acceptable time. So, we use the GUROBI Optimizer [78] to solve the quadratic model in order to obtain the optimal solution. In addition, we designed three heuristic algorithms to cope with large size dimensions ((Max-Min and Min-Min for the OFFLINE case, RBP for the ONLINE case). In Section-3.4, we compare these algorithms with the one produced by a Random Algorithm (OFFLINE case) and the PT algorithm (ONLINE case). Hence, we discuss a migration technique aiming at efficiently managing resources and balancing flows in a dynamic environment.

The three heuristics are introduced below.

3.3.1 OFFLINE Case

Our heuristics (Max-Min and Min-Min algorithm) are composed of two phases: the node mapping phase and the link mapping phase. In the first phase, we rank nodes in the virtual network to order nodes able to host a VNF for a customer demand. The link mapping phase uses the result of the node mapping phase to find a path with minimum cost for each customer demand.

3.3.1.1 The Node Mapping Phase

We introduce a polynomial-time heuristic finding an allocation of VNFs available on the virtual network to match customer demands. The main idea is to assign a virtual node, based on its rank in the virtual network, to a VNF. We propose two algorithms, namely the Max-Min algorithm and the Min-Min algorithm. The Max-Min algorithm allocates a "costly" function (i.e., a function whose resource requirement is large) to a cheap node that is a node whose rank is low. The Min-Min algorithm allocates a "cheap" function (i.e., a function whose resource requirement is small) to a cheap node.

We use the equation (3.15) of the rank for node v to find the cheapest or the best virtual node. This value is always updated after deploying a VNF on a virtual node. It accounts for the available resource capacity, the price for each resource unit and the ability to connect to other nodes. It also considers the benefit obtained by the situation where a given function is already hosted on one node and could be mutualized with another demand. We propose a formula for ranking the nodes based on the components as below:

$$\begin{aligned} rank_v = \alpha & \left(\frac{p_v}{\sum\limits_{i \in V} p_i} + \frac{1}{\sum\limits_{f \in F} C(f,v) + 1} \right) \\ + \beta & \left(\frac{\sum\limits_{i,j \in V} w_{ij} - \bar{y}}{\sum\limits_{i \in V} w_{iv} - \overline{x_v}} + \frac{\sum\limits_{d \in D_n} T(s^d, v) + \sum\limits_{d \in D_n} T(t^d, v)}{\sum\limits_{d \in D, i \in V} T(s^d, i) + \sum\limits_{d \in D_n, i \in V} T(t^d, i)} \right) \end{aligned} \tag{3.15}$$

where $\alpha, \beta \in [0,1]$ $(\alpha + \beta = 1)$ characterize the influence of the resource cost on nodes and the communication cost on links, respectively. In particular, these two parameters will determine the priority of our system according to the resources associated with nodes and communication links. We will estimate thoroughly the impact of these parameters in Section-3.4. $\overline{x_v}, \overline{y}$ are the amount of bandwidth decreased for node v and other nodes after a function is selected to be hosted on node v. $T(u,v)$ is the minimum distance between two nodes $u, v \in V$. $C(f,v)$ specifies the number of occurrences of function $f \in F$ in all demands that were assigned to node $v \in V$.

The Pseudo-code of the Max-Min algorithm is provided in Algorithm 1. The Max-Min algorithm first selects a function $f \in F$ whose resource requirement is the highest, denoted by f_{max}. It then finds a virtual node with the minimum rank in the virtual network, denoted by v_{min}. Virtual node v_{min} is selected to host function f_{max} for a customer demand. The available resource capacity of virtual node v_{min} is updated. Finally, the function f_{max} is removed from F when f_{max} required in all customer demands is satisfied. We repeat the process until all functions are allocated.

Algorithm 1 Max-Min Resource Heuristic

function MAX-MIN($P(f)$)
▷ $P(f)$= number of occurrences of f in all demands
  $\overline{x_v} = 0\ \forall v$, $\overline{y} = 0$
  while $F \neq \emptyset$ **do**
    $f_{max} \leftarrow \arg\max_{f \in F}\{\mu_f^r\}$
    while $P(f_{max}) > 0$ **do**
      $v_{min} \leftarrow \arg\min_{v \in V \text{ and } v \text{ satisfies (3.5)}}\{rank_v\}$
      $g(f_{max}) \leftarrow g(f_{max}) \cup v_{min}$
      $max_{bw} = \max\{d \in D_n | f_{max} \in F^d\}$
      $xo_{v_{min}} \leftarrow \overline{x_{v_{min}}} + max_{bw}$
      $\overline{y} \leftarrow \overline{y} + max_{bw}$
      $P(f_{max}) \leftarrow P(f_{max}) - 1$
      Update $rank_v, c^r_{v_{min}}$
    end while
    Remove f_{max}
  end while
  Compute $U_n(t)$ using equation (3.1).
  return g.
end function

The Min-Min algorithm is similar to the Max-Min algorithm where, f_{max} in the Max-Min algorithm is replaced by f_{min} that is a function whose resource requirement is the lowest.

3.3.1.2 The Link Mapping Phase

The second phase of the heuristic is to find a path from a source to a destination with the minimum cost for each demand. Its Pseudo-code is presented in Algorithm 2. Specifically, using the result of phase 1, for each $f \in F$, we have a set of virtual nodes $g(f)$ that can support function f. We denote the ith function of the customer demand d by $F^d(i)$. For each customer demand, the algorithm first finds the shortest path between the source and a node providing $F^d(0)$. Then, for adjacent functions $(F^d(i), F^d(i+1))$, it finds the shortest path from a set of nodes $g(F^d(i))$ to a set of nodes $g(F^d(i+1))$. Finally, it finds the shortest path between the destination of the demand and a node that can provide the last function in the demand (i.e., a node in $g(F^d(\ell_d - 1))$ where ℓ_d is the number of functions for demand d).

3.3.2 ONLINE Case

3.3.2.1 Routing Before Placement Algorithm

We now present an algorithm to solve the ONLINE case.

Algorithm 2 Path Selection

function PATHSELECTION($g : F \mapsto V$)
  $\{p, p_{sol}\} \leftarrow \{\emptyset, \emptyset\}$
  for demand $d \in D_n$ **do**
    $F^d(i)$: the i^{th} VNF in demand d, $i = 0..\ell_d - 1$
    $F^d(-1) \leftarrow s^d, F^d(l_d) \leftarrow t^d$
    for a pair $F^d(i), F^d(i+1)$,$i = -1..\ell_d - 1$ **do**
      $U_1 \leftarrow g(F^d(i)), U_2 \leftarrow g(F^d(i+1))$
      $p_i \leftarrow$ shortest_path($u_1 \in U_1, u_2 \in U_2$), satisfying (3.6)
      $g(F^d(i+1)) = u_2$
      $p \leftarrow p \cup p_i$
    end for
    Add p to p_{sol}
  end for
  Compute $U_l(t)$ using equation (3.2).
  return p_{sol}
end function

In contrast to the OFFLINE case, the Routing Before Placement (RBP) algorithm finds the routing path prior to locating VNFs on virtual nodes. RBP will consider each demand sequentially and decide whether and how to serve it. For each demand d, we find in graph G, a shortest path p from s^d to t^d. Then we allocate one by one VNF in F^d on the path p, satisfying all resource constraints (3.5)→(3.13). If this path does not satisfy all resource constraints, we try another shortest path p' from s^d to t^d, and so on. If no such path exists, the demand d is rejected. The Pseudo-code of the RBP algorithm is provided in Algorithm 3.

3.3.2.2 Migration Technique

Another concern in the dynamic NFV problem is to decide if and how a set of old demands should be migrated. Selecting old demands to migrate will impact the current resources of our system as well as affect the system cost. Our goal is to move a flow of old demands including nodes and links to a new position to optimize the system resources. As mentioned earlier, there exist a penalty cost associated to the migration tasks (3.4). In this section, we propose a simple method to select a set of old demands to be migrated. The main idea of the method is to identify old demands that have the largest wasted bandwidth cost for passing through nodes that do not provide resources for any required VNFs. It results in the computation of the wasted bandwidth cost for each old demand, as follows:

$$\Gamma_d = \sum_{f_1 * f_2 = 0; v_1, v_2 \in V} \hat{x}^d_{f_1 v_1} \hat{x}^d_{f_2 v_2} \hat{y}^d_{v_1 v_2} w_{v_1 v_2} \tag{3.16}$$

Algorithm 3 Routing Before Placement(RBP) Algorithm

function RBP($F, D_m, D_o, D_n, G(V, E, w)$)
  $AcceptNo = 0$;
  for demand $d \in \{D_m \cup D_n\}$ **do**
    $\{v_{subsol}, f_{subsol}, v_{sol}, f_{sol}\} \leftarrow \{\emptyset, \emptyset, \emptyset, \emptyset\}$
    reject=**true**;
    while true do
      $p \leftarrow$ Shortest_path(G, s^d, t^d)
      for $f \in F^d, v \in p$ **do**
        Locate f to v
        if satisfy constraints (3.5) **then**
          $v_{subsol} \leftarrow v_{subsol} \cup v$
          $f_{subsol} \leftarrow f_{subsol} \cup f$
          Update c_v^r
          reject = **false**; **break**;
        else
          Mark p in G; **continue**;
        end if
      end for
    end while
    if reject== **false then**
      Add v_subsol to v_sol; f_subsol to f_sol
      AcceptNo $\leftarrow$ AcceptNo +1;
    end if
  end for
end function

Then we select η % number of old demands D_o depending on the wasted resource as in (3.16), called D_m and re-configure this set of demands. We release all resources provisioned in D_m and find the solution for D_m using Algorithm 3. Finally, we continue this algorithm until the set of all new demands D_n has been processed.

3.4 Evaluation

In this section, we evaluate the performance of our solutions according to a large set of parameters.

3.4.1 Parameter Setting

We define the following scenario in order to evaluate the performance of our algorithms:

- Network Topology: In this section, we use two types of topologies: random networks $(R1, R2, R3, R4)$ and data center networks $(B1, B2, F1, F2)$ as defined in Table 3.2. We focused on two types of data center topologies, namely, Fat-Tree [79] and BCube [80]. For Fat-Tree networks, we generated two topologies $(F1, F2)$ with parameters $k = 4$ and $k = 6$, where k represents the number of ports in a switch. For BCube networks, we generated two topologies $(B1, B2)$ with parameters $n = 2, k = 2$ and $n = 4, k = 2$, where n represents the number of ports in a switch and k stands for the link of hierarchy of switches.

- Customer demands: We evaluated each network topology, called a scenario, using eight different tests, varying the number of demands l in a test from min_l in the first test to max_l in the last one. Each customer demand including the source node, the destination node, a chain of VNFs, the resource requirement and the cost factor of SLA, is generated randomly for each individual test.

- The price of resources (i.e. CPU, storage, memory) are collected from the Microsoft [81] and Amazon websites [82]. For example, the price of a virtual machine with 4 CPUs, 285GB storage and 7GB memory on the Microsoft cloud is 0.36 $\$/hour$.

We have simulated our proposed algorithms in Java.

- In the OFFLINE case, we evaluate the performance (**the NFV cost**) of our heuristics (Max-Min and Min-Min) with the optimal solution, and a Random algorithm (to be described later).

- In the ONLINE case, we evaluate the performance (**the average routing path length and the acceptance ratio**) of our heuristic (RBP algorithm) against the PT

Table 3.2: Scenarios

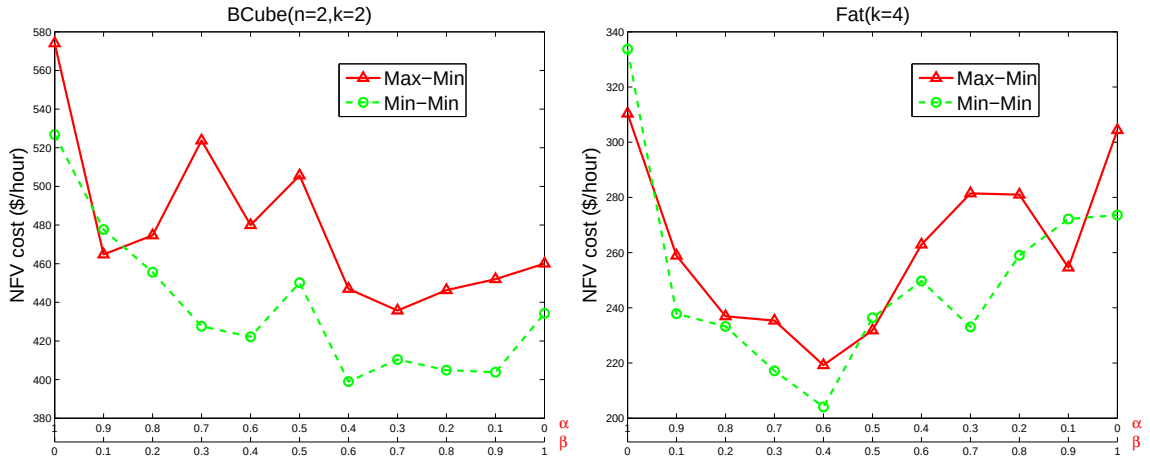
Scenario	Number of virtual nodes n	Number of VNFs m	Number of new demands size (D_n) $min_l \rightarrow max_l$	Number of old demands size (D_o)
R1	7	4	$1 \rightarrow 1$	1
R2	15	5	$3 \rightarrow 20$	5
R3	50	15	$10 \rightarrow 100$	20
R4	200	20	$50 \rightarrow 600$	30
B1	n=2,k=2	10	$2 \rightarrow 100$	10
B2	n=4,k=2	10	$2 \rightarrow 100$	10
F1	k=4	10	$2 \rightarrow 100$	10
F2	k=6	10	$2 \rightarrow 100$	10

algorithm. We also compare the gain obtained when applying our migration strategy as a function of the percentage of old demands considered for migration.

3.4.2 OFFLINE case

We first assess the influence of the parameters α, β in equation (3.15). We consider different values of α, β in $[0, 1]$ and then compare the NFV cost produced by our two heuristics. Fig.3.1 shows the influence of the NFV cost on data center networks when selecting different value of parameters (α, β). The NFV cost is computed by adding the resource cost $U_n(t)$ as defined in equation (3.1) to the bandwidth cost $U_l(t)$ as in equation (3.2). We observe that the best values for both algorithms (Max-Min, Min-Min) depend on the network topology. For instance, the best values for a BCube topology are $\alpha < \beta$, whereas for a Fat Tree topology they are $\alpha > \beta$. When $\alpha < \beta$, the priority of nodes will mainly depend on the communication cost on links whilst if $\alpha > \beta$, the priority of nodes will mostly depend on the resource cost on nodes. In fact, a BCube topology has more links connecting switches and servers than a Fat Tree topology. Therefore, in order to select a node that can host VNFs in a BCube topology, it is more important to consider the communication cost. In the reminder of the chapter, we use ($\alpha = 0.6$, $\beta = 0.4$) with Fat Tree topologies, and ($\alpha = 0.4$, $\beta = 0.6$) with BCube topologies for the two heuristics.

Then, we use the GUROBI Optimizer [78] to find the optimal solution for small instances where $(m = 4, l = 1)$, namely scenario 1 and introduce a simple Random algorithm to evaluate the efficiency of our solutions for larger dimensions. The Random algorithm is defined below. We first generate randomly locations for hosting all VNFs supporting all demands and satisfying constraints (3.5). We then find a path satisfying constraints (3.6) for each demand. Each path will be composed of nodes hosting the required VNFs in a given order. The obtained result constitutes a Random solution to our problem.



(a) B1 (BCube, $n = 2, k = 2$) (b) F1 (Fat Tree, $k = 4$)

Figure 3.1: (OFFLINE) The NFV cost with different parameters (α, β) on a data center network

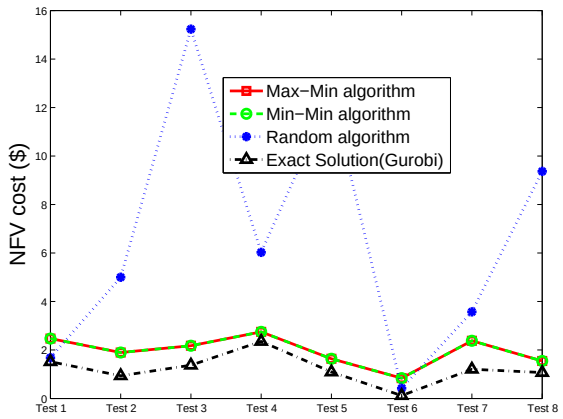


Figure 3.2: (OFFLINE) The NFV cost of small scale scenarios

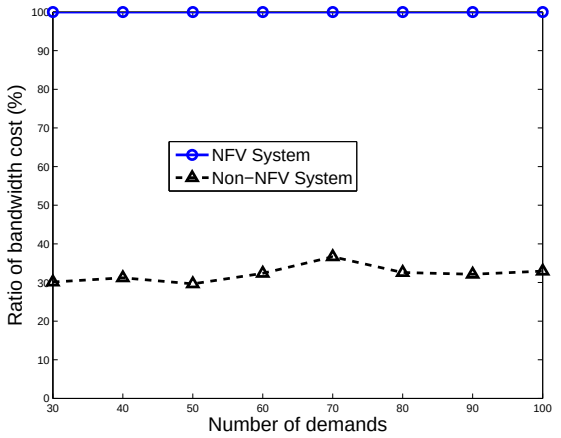
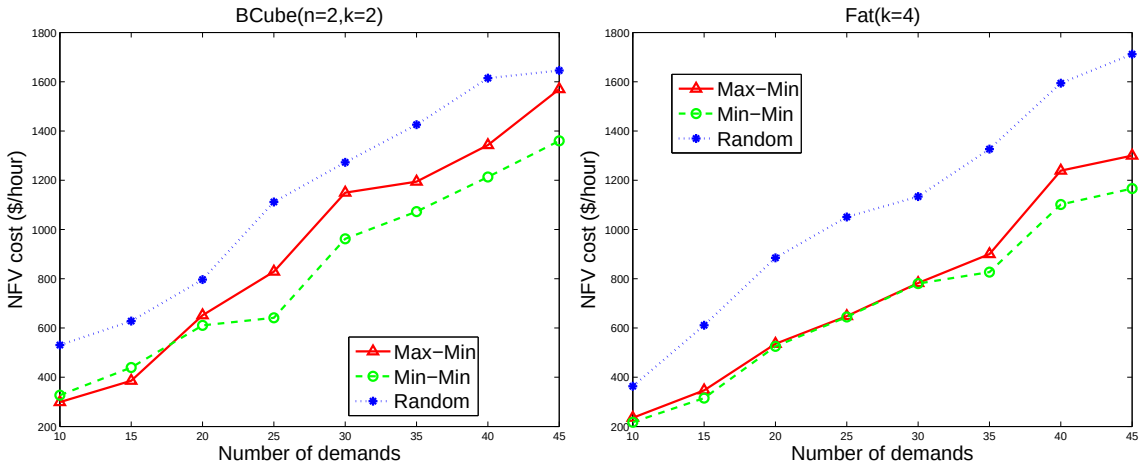


Figure 3.3: (OFFLINE) Ratio between the bandwidth cost of a non-NFV system and that of a NFV system

Fig. 3.2 shows the NFV cost obtained by Max-Min, Min-Min, Random algorithms and the optimal solution (GUROBI optimizer [78]) for a small size system. All tests (x-axis) in the small size scenario consider the same demand for different network topologies that are generated randomly with the number of vertices equals to 7. It is obvious that the solutions produced by the heuristic algorithms (Max-Min and Min-Min) are far better than the one provided by the Random algorithm and close to the optimal solution. For larger scenarios ($R2, R3, R4, B1, B2, F1, F2$) in Table 3.2 (i.e. hundreds of virtual nodes, hundreds of demands), we run our heuristic algorithms and Random algorithm where the



(a) B1 (BCube, $n = 2, k = 2$)

(b) F1 (Fat Tree, $k = 4$)

Figure 3.4: (OFFLINE) The NFV cost of large scale scenarios

customer demand is generated randomly. Regarding the NFV cost (Fig. 3.4) on data center topologies ($B1$ and $F1$), our heuristics (green line and red line) performs obviously better than Random algorithm (blue line) and become more effective when the number of demands grows. In addition, Min-Min algorithm (green line) is better than Max-Min algorithm (red line). As expected, allocating a "cheap" function (i.e. a function whose resource requirement is small) to a cheap node provides a more efficient resource management.

We now consider a non-NFV system where a source node is able to host the all set of required functions. We simply use Dijkstra algorithm to find a shortest path for all pairs of demand from source to destination. We then sum the costs for all links on all the shortest paths to obtain a bandwidth cost. We use this solution to compare it against our NFV-based one and assess the overhead due to virtualizing the network functions. In Fig. 3.3, we compare the bandwidth required to transfer packets from a source to a destination in a NFV system and that in a non-NFV system. We use the bandwidth cost computed by equation (3.2). The result shows that a trade-off has to be found between efficiency and the agility provided by the virtualization of functions. It means that, in our scenario, an operator needs to spend 60% more resources to be able to benefit from the agility and dynamicity of NFV. This confirms the importance for instrumenting an efficient deployment of service chaining.

We would like now to evaluate the distribution of VNFs on a multi-tier network topology as illustrated in Fig. 3.5. We define three node categories: core nodes, aggregation nodes and access nodes. An access node is connected to an aggregation node, itself connected to core nodes, and core nodes are fully meshed. We assume that all nodes are NFVI nodes that can host VNFs. We compute the solution for the two heuristics and the PT algorithm in order

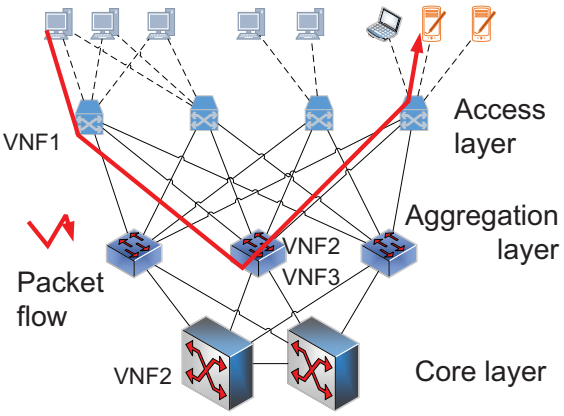


Figure 3.5: Multi-tier network topology

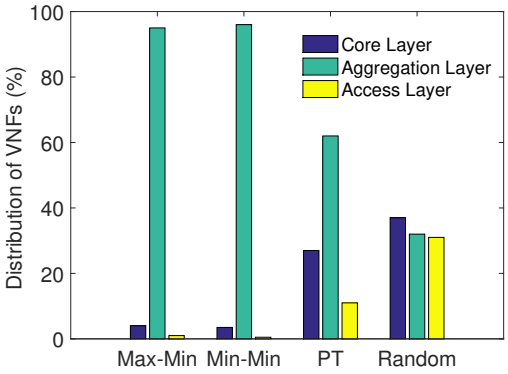


Figure 3.6: Distribution of VNFs on a multi-tier network

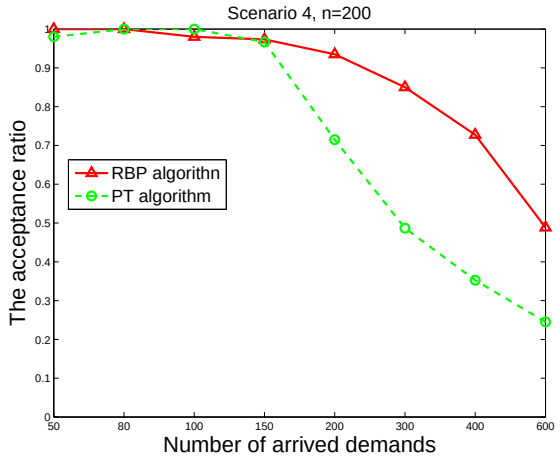


Figure 3.7: (ONLINE) Comparison between the acceptance ratio of RBP and that of PT

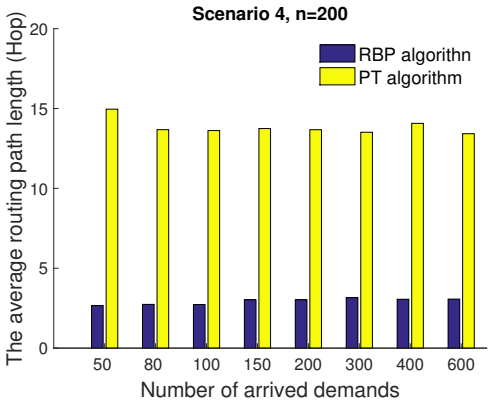


Figure 3.8: (ONLINE) Comparison between the average routing path length of RBP and that of PT

to compare the distribution of VNFs among those algorithms. Fig. 3.6 shows that for the Max-Min and Min-Min algorithms, VNFs are almost similarly distributed on aggregation nodes (95%), a small part on core nodes (4%), and rarely (1%) on access nodes, while the PT algorithm distributes more equally VNFs to the three node types including core nodes (21%), access nodes (17%), and aggregation nodes (62%). Finally and as expected, the Random algorithm distributes fairly VNFs to the three node types. It clearly suggests that the network functions should be located on the network edge rather than the network core. The best option will be to deploy all VNFs in every edge node but the cost might be excessive hence the importance to identify which VNFs can be mutualized and/or deployed only on some specific edge nodes whilst preserving performance constraints.

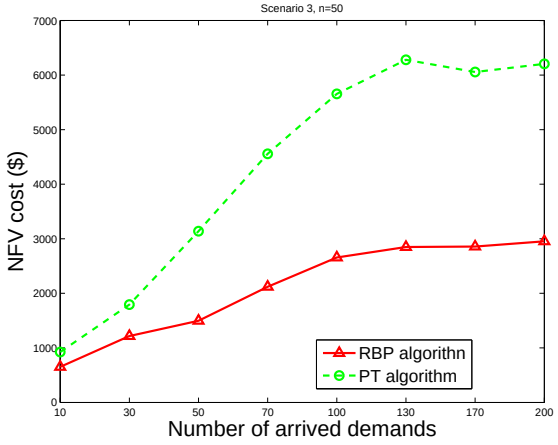


Figure 3.9: (ONLINE) Comparison between the NFV cost of RBP and that of PT when no migrating

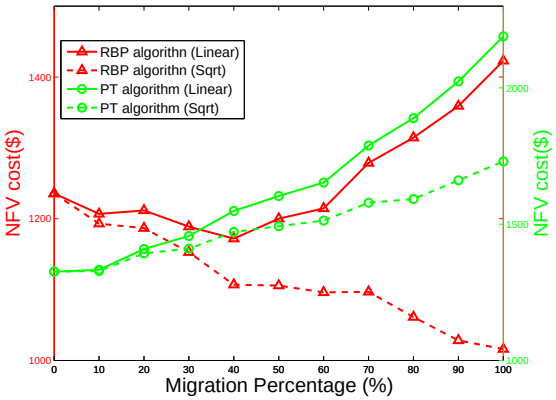


Figure 3.10: (ONLINE) Different percentages of migration with two penalty cost functions

3.4.3 ONLINE case

We now evaluate our solution suited to the ONLINE case.

We acknowledge the fact that there exists an abundant literature related to the NFV placement such as [28, 37, 45]. However, we claim that our (RBT) model is comprehensive and it is therefore difficult to compare it against the existing ones. For instance, as mentioned earlier in Chapter 2, the model in [45] does not consider the request or the customer demand as a flow from a source to a destination. Moreover, they evaluated their approaches with at most two chained network functions for each SFC request. Therefore, we decide to select the PT algorithm [58] as a comparison as we found that this is the one that share the largest set of characteristics with ours. Fig. 3.7 and Fig. 3.8 show the acceptance ratio and the average routing path length obtained by our heuristic (RBP) compared to the PT one. It shows that our solution achieve higher acceptance ratio and shorter routing paths than PT. Fig. 3.9 showing the NFV cost as defined by (3.1) and (3.2) provide evidence of the efficiency of our solution when no migration exists. The idea of RBP is to implement routing before placing the VNFs. This guarantees that demands always use good paths. Indeed, in Fig 3.3, 3.8, and 3.9, the solution obtained by RBP (red line) is obviously better than the one produced by the PT algorithm (green line). However, it has to pay the price for increasing the running time when trying all possible paths for each demand.

In addition, we also compare the acceptance ratio between RBP and PT applied on some data center network topologies ($B1, B2, F1, F2$). Fig.3.11 shows the acceptance ratio as a function of the number of arriving demands, from 2 to 100. Fig. 3.11(a) depicts the results with a network of 32 servers (respectively BCube $B1$, FatTree $F1$) and Fig. 3.11(b)

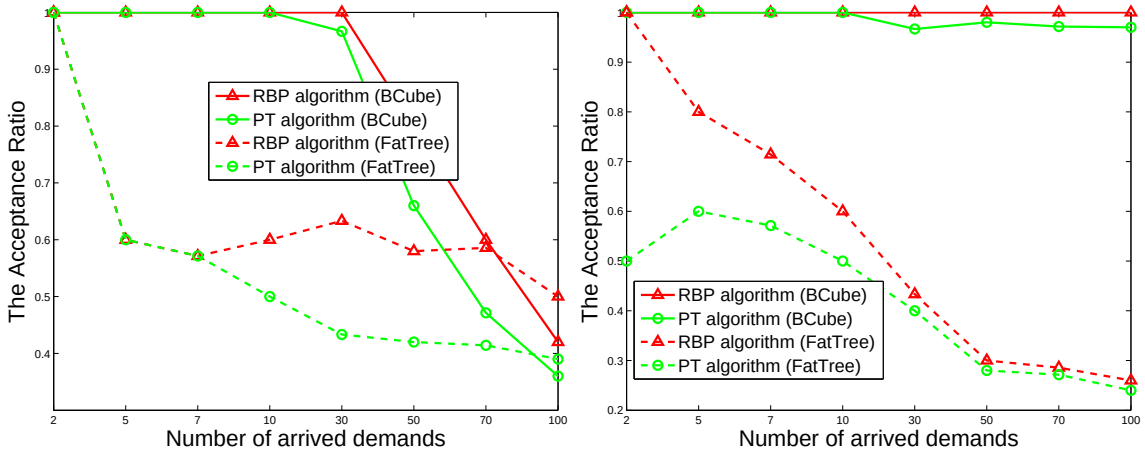
shows the results with a network of 128 servers (respectively BCube $B2$, FatTree $F2$). We observe that our heuristic performs better on a Bcube topology than on Fat-Tree topology. Indeed, a BCube topology provides much more links between server nodes and switch nodes than the Fat-Tree topology.

Finally, we evaluate the impact of migrating a set of old demands. We address the NFV cost using various migration percentages η from 0% (no migration) to 100% (all demands are migrated). As mentioned earlier, we select old demands eligible for migration according to the wasted bandwidth, defined in equation (3.16), of these demands. We also consider as a parameter, two cases for the migration penalty cost function $M(.)$ in equation (3.4): one linear function and one square root function.

Fig. 3.10 provides a comparison of the NFV cost, as a function of the migration percentage, obtained by the RBP and PT algorithms. The NFV cost is computed as the sum of the resource cost $U_n(l)$ defined by equation (3.1), the bandwidth cost $U_l(t)$ equation (3.2) and the migration penalty cost $U_{Penalty}$ (3.4). We observe that RBP will benefit from migrating demands whilst it is not the case for PT. The best cost is obtained for RBP with 40% migration of old demands when using a linear penalty cost. This cost will always decrease as a function of the percentage of flows being migrated for a square root function penalty cost. In summary, the migration will bring a certain benefit when we select a proper migration percentage (as 40% in our experiment with a linear penalty cost). However, if we only pay a smaller penalty cost for migrating (as square root function), we obtain more benefits when updating the system resource after an interval τ. It means we have to consider both the penalty cost to move a function to a new position and the distribution of system resources at present, when determining if and how a set of old demands should be migrated.

3.5 Summary

In this chapter, we addressed the resource allocation problem for NFV where a network service is composed of several VNF instances. Our model is generic, able to handle a large and diverse set of key parameters and can easily be customized. It takes into account the interconnectivity among VNFs, service requirements, and NFV costs (including both resource cost and traffic cost). We formulated the problem as a quadratic program for both optimal VNF location and optimal routing. We proposed two heuristic algorithms, called Max-Min and Min-Min, for the OFFLINE case and one heuristic, called RBP for the ONLINE. Our extensive evaluation shows that our heuristic algorithms always perform significantly better than the one produced by the PT solution in [58] considering various performance metrics such as the NFV cost and the acceptance ratio of arriving demands. In addition, we computed the cost incurred by a network relying on NFV to capture the



(a) Number of servers = 32

(b) Number of servers = 128

Figure 3.11: (ONLINE) Comparison between the acceptance ratio of RBP and that of PT on a data center network

overhead introduced by a solution based on the virtualization of functions. To complete our study, we investigated a simple migration technique that can efficiently manage the dynamic situation produced by a continuous flow of arriving demands.

Chapter 4

Resource Utilization in an NFV Dynamic System

Contents

In the first results of the previous work, we have developed an comprehensive resource management model in NFV system while taking into account the execution order of VNFs in a service demand. Due to the Quadratic Programming, the model is quite complex and hard to solve exactly. Moreover, we addressed the problem in both OFFLINE case and ONLINE case, but we still did not find an optimal solution for a dynamic problem.

Therefore, this chapter investigates an improvement to provide a new efficient MILP model for NFV placement and routing problem. The model not only can find an optimal solution with large instances of the problem but also be able to apply it to a dynamic scenario by taking into account the decision of Service Providers to serve a demand or reject it.

In this context, the work contributions are as follows:

- We define and formulate the resource utilization problem as a Mixed Integer Linear Programming (MILP) problem. The main distinctive feature contrasting with existing works is that the model is linear and can find the optimal solution for a large set of instances. The model can be also applied in the context of operating a dynamic system in order to minimize the resource utilization and maximize the number of accepted demands.
- We propose Flow covers to improve efficiency of the solution problem allowing us to report on experiments involving large instances of the problem.
- We propose two efficient heuristics to find good quality feasible solutions for larger instances both requiring reduced execution time.
- We evaluate the proposed model and solution method on realistic network topology using an event-driven simulation to run our model in a dynamic scenario.

This chapter is organized as follows. The next section describes the system model of optimizing resources utilization for an NFV dynamic system in a context of our work and presents the problem statement as a Mix Integer Linear Programming. The theoretical analysis for generating Flow covers is provided in Section 4.2. We design a heuristic algorithm that based on MILP relaxation method in Section 4.3, and another heuristic in Section 4.4. We then simulate and evaluate our work in the Section 4.5. Finally, conclusions and perspective are presented in Section 4.6.

4.1 Problem Formulation

4.1.1 System Description

In our problem, we map the VNFs- Forwarding Graph to the physical network. Each VNF must be mapped to a physical node that has the sufficient resources to host the VNF. In addition, the logical links between VNFs must be mapped to physical paths so that all physical links on the paths have sufficient bandwidth to accommodate the logical link.

We model a network as an edge-weighted vertex-weighted directed graph $G = (V, E)$, in which $w_{u,v}$ is the link capacity of each edge (link) $(u, v) \in E$ and c_v is the number of resources (i.e. virtual machines (VMs)) that can be hosted on each vertex (server) $v \in V$.

A virtual network function is defined through various characteristics such as the input data rate that it needs to handle, its output interfaces, based on the forwarding rules. In this paper, we assume that the data rate of the flows does not change when processing through network functions. Hence, a VNF is defined by the processing requirement for running (q_f) with each $f \in F$.

A demand (or request) d is defined by a source s^d, a destination t^d, the bandwidth requirement b^d, a chaining of ordered VNFs (F^d) through which the flow has to be processed in the right order.

In this paper, we focus on solving the dynamic problem. That means we consider the arrival time of demands. The demands can arrive any time. In a rush hour, lots of demands can send request to the system. Therefore, it is necessary to provide a model that decides which demand will be served and distributes efficiently resources for demands. After a time period δ (depending on the system configuration), we collect demands that arrive in the time slot $[t-\delta, t]$, where t is the current time. We also update the available system resources to take into account possible completion of demand processing in this time period. We call $D^{(t)}$ is the set of demands arriving from $t-\delta$ to t, $G^{(t)} = (V^{(t)}, E^{(t)})$ is the current state of network G at time t. The current state of network at time t includes available resources of the system after time $t-\delta$ and occupied resources of demands which have finished during the period from $t-\delta$ to t.

In short, the basic problem considered is to optimize function locations and demand routing in a typical period of time between $t-\delta$ to t. This is a subproblem of a complex dynamic system. The goal of the formulation is to obtain an efficient placement of VNFs and routing of the flows without violating the constraints of the available resources on nodes and links. The objective function aims to minimize the utilization of the links and nodes, thus maximizing the number of accepted demands.

4.1.2 Mixed Integer Linear Programming Formulation

4.1.2.1 Notation

We define the outputs of our problem as a set of decision variables.

- x^d_{fv}: a binary variable that equals to 1 if and only if node v hosts function f of demand d
- y^d_{uv} equals to 1 if and only if demand d uses link (u, v)
- $\phi^{f_i,d}_{u,v}, i = 0..l_d + 1$ equals to 1 if and only if (u, v) is used in a path between two successive functions f_{i-1}, f_i for demand d. Especially, $\phi^{f_0,d}_{u,v}$ is in case of a path between

the source and the first function f_0. And $\phi_{u,v}^{f_{l_{d+1}},d}$ is in case of a path between the last function f_{l_d} and the destination.

- z^d equals to 1 if and only if demand d is served.

We aim to minimize the maximum resource utilization rate and maximize the acceptance ratio. The resource utilization includes the utilization on nodes and links.

$$U^{(t)} = \alpha * \frac{\sum_{d \in D^{(t)}} z^d}{|D^{(t)}|} - \beta * (\mathcal{L} + \mathcal{N}) \tag{4.1}$$

where:

$$\mathcal{L} = \underset{(u,v) \in E^{(t)}}{Max} \left\{ \frac{\sum_{d \in D^{(t)}} b^d y_{uv}^d}{w_{uv}} \right\}$$

$$\mathcal{N} = \underset{v \in V^{(t)}}{Max} \left\{ \frac{\sum_{d \in D^{(t)}, f \in F^d} q_f x_{fv}^d}{c_v} \right\}$$

The choice of the two parameters α and β influences the way of our system to serve customers. That is we want to serve demands as much as possible (the first term) or we will refuse demands that serving them needs too much resources (the second term, including the resources on nodes and links). For example, in our simulation, with $|D^{(t)}| = 100$ we use $\alpha = 10$ and $\beta = 1$ to make sure that a demand is served if resources of its serving do not exceed 10% of the available resources (on both nodes and links) of the system.

4.1.2.2 Mathematical Model

($\mathcal{P}$): **Maximize** $U^{(t)}$

Subject to:

$$\sum_{d \in D^{(t)}, f \in F^d} x_{fv}^d q_f \leq \mathcal{N} c_v \qquad \forall v \in V^{(t)} \tag{4.2}$$

$$\sum_{d \in D^{(t)}} y_{uv}^d b^d \leq \mathcal{L} w_{uv} \qquad \forall u, v \in V^{(t)} \tag{4.3}$$

$$y_{uv}^d + y_{vu}^d \leq 1 \qquad \forall u, v \in V^{(t)}, d \in D^{(t)} \tag{4.4}$$

$$\sum_{v \in V^{(t)}} x^d_{fv} = \begin{cases} z^d & \text{if } f \in F^d \\ 0 & \text{otherwise} \end{cases} \tag{4.5}$$

$$\sum_{v \in V^{(t)}} y^d_{uv} - \sum_{v \in V^{(t)}} y^d_{vu} = \begin{cases} z^d & \text{if } u = s^d \\ -z^d & \text{if } u = t^d \\ 0 & \text{otherwise} \end{cases} \tag{4.6}$$

$$\varphi^{df}_{uv} \leq y^d_{uv} \qquad \forall u, v \in V^{(t)}, d \in D^{(t)}, f \in F^d \tag{4.7}$$

$$\mathcal{A} * \varphi^{df_i} = x^d_{f_{i-1}} - x^d_{f_i} \qquad \forall d \in D^{(t)} \tag{4.8}$$

$$y^d_{uv} \leq z^d \qquad \forall u, v \in V^{(t)}, d \in D^{(t)} \tag{4.9}$$

$$\varphi^{df}_{uv} \leq z^d \qquad \forall u, v \in V^{(t)}, d \in D^{(t)}, f \in F^d \tag{4.10}$$

$$x^d_{fv} - \sum_{u \in V} y^d_{uv} \leq 0 \qquad \forall u, v \in V^{(t)}, d \in D^{(t)}, f \in F^d \tag{4.11}$$

$$\sum_{v \in V} y^d_{s^d v} - \sum_{u \in V} y^d_{u t^d} = 0 \qquad \forall u, v \in V^{(t)}, d \in D^{(t)} \tag{4.12}$$

Constraint (4.2), (4.3) ensure that the node capacity and link capacity are not exceeded. Constraint (4.5) ensures that a required function for each demand is only processed by a node if the machine has capabilities to process the function. Constraint (4.4) states that for each demand we avoid to transfer back data on a path. While constraint (4.6) is a flow balance at each node. These constraints impose an upper bound value 1 to the total flow through each node in the network. Constraint (4.7) and (4.8) enforce the order of function in VNFs chaining.

In (4.8), $\mathcal{A}$ denotes the node-arc incidence matrix of the network, $x_{f_{i-1}}$ and x_{f_i} are integral vectors with a pair of successive functions (f_{i-1}, f_i). Especially, for each demand d, $x_{f_{-1}}$ is the n-dimensional vector with all components be 0 except component of its source which equals to 1. Similarly, $x_{f_{l_d+1}}$ is the n-dimension vector with all components be 0 except component of its destination which equals to 1.

The joint effect of constraints (4.6), (4.7) and (4.8) is to guarantee that for each demand d, the corresponding functions are indeed located on the elementary path from its source to its destination in the solution. As a result, the ordering constraints will be correctly expressed.

Constraints eqs. (4.9) to (4.12) take into account the decision of Service Providers to serve a demand or reject it in a dynamic system. For instance, if demand d is blocked (that

means $z^d = 0$), we do not provide any resources for this demand ($y^d_{uv} = 0$ and $\varphi^{df}_{uv} = 0$ with $\forall u, v \in V, f \in F, d \in D$).

4.2 Flow Cover Inequalities

The mathematical model above is a Mixed Integer Linear Programming which can be solved by MILP based branch-and-bound algorithms. Here, we use GUROBI 6.52 optimization software to get solutions. However, with the academic version of Gurobi, the efficient algorithms for finding strong inequalities (lifted cover inequalities) that cut off fractional solutions to MILP relaxations are limited. Moreover, without addition of valid inequalities, the relaxations are not strong enough to efficiently prune the nodes. Therefore, the problem maybe still difficult in a higher dimensional space. The purpose of this section is to present strong inequalities for the above model that can be used successfully in reducing the size of the branch-and-bound tree. Our inequalities are derived from the flow cover inequalities that were introduced by Padberg et al. [83] and Van Roy et al. [84–86].

Consider the set of feasible points for a linear integer programming given by:

$$Y = \{y^d_{uv} \in \{0,1\} : \sum_{d \in D^{(t)}} y^d_{uv} b^d \leq w_{uv} \quad \forall u, v \in V^{(t)}\}$$

Here $\{h^d_{uv} = b^d * y^d_{uv}\}$ is the used bandwidth on link (u, v) of demand d. We have:

$$\begin{cases} \sum\limits_{d \in D^{(t)}} h^d_{uv} \leq w_{uv} \quad \forall u, v \in V^{(t)} \\ h^d_{uv} \leq b^d * y^d_{uv} \quad y^d_{uv} \in \{0,1\}, h^d_{uv} \geq 0 \end{cases} \tag{4.13}$$

We consider (4.13) as a single-node flow model with exogenous supply w_{uv} and $|D^{(t)}|$ outflow arc (see Figure. 4.1)

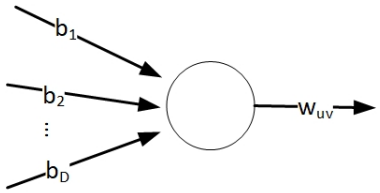


Figure 4.1: An example for a *single-node flow* model

For each demand $d \in D^{(t)}$, the flow h^d_{uv} on the d^{th} arc is bounded by the capacity $b^d > 0$ if the arc is open. Note that y^d_{uv} equals 1 if (u, v) is used by demand d, and equals 0 if otherwise. Hence, we have: $b^d \leq w_{uv}, \quad \forall (u, v) \in E^{(t)}$.

Definition 1. *A set S is called a flow cover, respectively link $(u, v) \in E^{(t)}$, if* $\sum\limits_{d \in S} b^d > w_{uv}$

Initially, we consider the subset of Y with $y_{uv}^d = \alpha^d$,i.e, α^d is fixed at one of its bounds, for $d \in D^{(t)} \backslash D_0$ given by:

$$Y^0 = \{y_{uv}^d \in \{0,1\} : \sum_{d \in D_0} y_{uv}^d b^d \leq \beta \quad \forall u, v \in V^{(t)}\}$$

where $\beta = w_{uv} - \sum_{d \in D^{(t)} \backslash D_0} \alpha^d b^d$

The flow cover inequalities in the context of our model are defined as follows:

$$0 \leq w_{uv} - \sum_{d \in S} y_{uv}^d b^d - \sum_{d \in S^+} (b^d - \lambda)(1 - y_{uv}^d) \tag{4.14}$$

where $\lambda = \sum_{d \in S} b^d - w_{uv}$, $S^+ = \{d \in S, b^d > \lambda\}$

Now the question that arises is how to find the cover sets S leading to strong inequalities of the form (4.14). We propose below a cover set selection strategy that uses knowledge from previous nodes on the branch-and-bound tree. Whenever we found a new node on the branch-and-bound tree, we know fractional solutions of MILP relaxation. That means we know the values of y_{uv}^d, denoted by $\overline{y}_{uv}^d$.

If we consider one by one link (u, v), we know a set of values $\overline{y}_{uv}^d$ with $d \in D^{(t)}$. We sort demands of $D^{(t)}$ in ascending order of the values $|\overline{y}_{uv}^d - \frac{1}{2}|$. We then select a set $S_0 \subset D^{(t)}$ which has $\sum_{d \in S_0} b^d > w_{uv}$. Stop here, we can obtain a valid cover set S_0. We then have to check the violation of (4.14) with the current solution before deciding to add this cover. A cover is added only if it is violated by (4.14). After this step, lots of non-violated covers are ignored.

Therefore, in order to increase the opportunity for finding the flow cover cuts that are not satisfied by the current fractional solution (called violated cuts), we use partial enumeration of an extended cover set. In particular, we add more τ demands from the sorted list and find valid covers from the extended cover set. Note that a cover S is valid if $\sum_{d \in S} b^d > w_{uv}$. As a result, we get at most $C^{|D^{(t)}|}_{|D^{(t)}|+\tau}$ cover sets.

In the following section, we propose two efficient heuristics to find a feasible solution in a small amount of time.

4.3 Progressive Rounding Heuristic Algorithm for MILP

The idea of the Progressive Rounding Heuristic Algorithm (PRH) is to repeatedly solve the continuous relaxation of the basic mathematical program and permanently fix the value of a few variables according to their fractional value.

Firstly, we relax the MILP $\mathcal{P}$ by solving the problem with continuous variables, called $\mathcal{P}_1$. The fractional component of $\overline{x_i}$ are sorted in descending order of value $|\overline{x_i} - \frac{1}{2}|$. A set of p first fractional variables $\overline{x_i}$ is chosen and set its value to 1 or 0 according to whether $\overline{x_i} > \frac{1}{2}$ or not. In each iteration, the variables in $\mathcal{P}_1$ are then updated by considering that some variables are set to 1 or 0. Hence, $\mathcal{P}_1$ is reduced to $\mathcal{P}_1'$ and so on. We stop doing the iteration until the number of variables decreases a mount of α percent of total variables, where p, α are parameters for the algorithm. Finally, we solve the optimal MILP formulation $\mathcal{P}_1'$.

PRH require solving MILP on the nodes in one branch of the branch-and-price tree. As the algorithms proceeds on each node of the investigated branch, the maximum number of the nodes on the branch is $n = |D^{(t)}|(|F|(|V^{(t)}| + |E^{(t)}|) + 2|E^{(t)}|)$. After the iterations, the number of variables of $\mathcal{P}_1'$ is $\alpha * n$. Therefore, the worst case complexity of PRH is $\mathcal{O}(\alpha|D^{(t)}|(|F|(|V^{(t)}| + |E^{(t)}|) + 2|E^{(t)}|))$.

4.4 Minimum Spanning Tree based Heuristic Algorithm

The idea of Minimum Spanning Tree based Heuristic Algorithm (MSTH) is to avoid using much resources on some nodes and links, especially in case of small capacity. The aim is to limit the possibility of using unnecessary bottleneck links and nodes, thus improving network utilization according to our problem objective.

4.4.1 Route Selection

In this section, we consider the problem of selecting the route that carries the traffic for each demand. Suppose we need to select a route and provision resources for a new demand $d \in D^{(t)}$ between source node s and destination node t. The main intuition behind route selection algorithm is to find a route that (a) has enough capacity to host VNFs (Section 4.4.2) and (b) keeps the more critical links available for future demands.

The problem now can be stated as finding a shortest path from s to t that $\underline{\sigma}(\mu) = Min\{w_{s,i_1}, w_{i_1,i_2}, .., w_{i_k,t}\}$ is maximized for each demand. This problem is recognized as the Maximum Capacity Path Problem [87] and is efficiently solved using a variant of Dijkstra's algorithm.

However, in order to guarantee two conditions (a) and (b), we should take into account the processing capacities. The idea is to choose these costs to be directly proportional to the volume the capacity of their incoming node is small, thus leaving some resources available for later demands and minimizing the network utilization. We propose the following costs:

$$a_{uv} = w_{uv} * \frac{c_v}{\mathcal{C}} \tag{4.15}$$

where $\mathcal{C} = \max_{u \in V^{(t)}} c_u$

In order to find a set of maximum capacity paths, we use a modified Dijkstra algorithm, see detail in [87].

4.4.2 VNFs Distribution

After obtaining a set of paths $\Pi = \{P_1, P_2, ...P_{|D^{(t)}|}\}$ to route for demands, the last important step is how to distribute VNFs on those to minimize the node utilization. In this section, we will present a procedure to distribute efficiently VNFs on given paths. The idea is to try put VNFs one by one on a path and consider the residual capacity of nodes on this path. The ideal distribution is that all needed resources for VNFs are shared equally for nodes on all paths. That means the ideal node utilization is computed by:

$$\mathcal{N}_{ideal} = \frac{\sum\limits_{d \in D^{(t)}} \sum\limits_{f \in F^d} q_f}{\sum\limits_{i=1..|D^{(t)}|} \sum\limits_{v \in P_i} c_v} \tag{4.16}$$

We consider the paths one by one. In particular, given demand d, we start from the first node on the path Π, and then put function $f_i : f_i \in F^d$ on node $v \in \Pi$ if

$$q_{f_i}/c_v \leq \mathcal{N}_{ideal} \tag{4.17}$$

and next to the succeeding node if otherwise. Finally we put all rest of functions on the destination of path even if it is violated by equation (4.17).

However, it also raises two issues when implementing as follows:

- The first one is node v can appear at r paths, i.e. $P_{i_1}, ..P_{i_r}$. We call those nodes as "common nodes". In that case, we set the available capacity of v on each path to c_v/r. This is aimed to ensure the resources of "common nodes" are shared equally among paths.
- The second one is the residual capacity of "common nodes" should be left for later demands to improve the node utilization. That means after finishing a demand, we check if the capacity of any "common nodes" has not used and then add the rest of resources to itself for later demand.

In summary, the MSTH algorithm first finds the maximum capacity path with large node capacity for each demand. It then distributes efficiently VNFs on obtained paths.

4.5 Performance Evaluation

In this section, we evaluate the performance of our proposed model via simulation. The experiments are executed in a Java based environment that allows running the simulation and optimization tool (Gurobi optimization [78]).

Table 4.1: Network parameter values for the case of unlimited resources and the case of limited resources.

Topology	Unlimited Resources		Limited Resources	
	W	C	W	C
Abilene	200	300	100	150
Nobel-eu	150	180	70	80
Nobel-germany	200	350	100	200
Test50	200	150	90	70
Test100	180	100	90	50
Test200	150	90	75	40

4.5.1 Simulation

The network used in the simulations was various real networks in SNDlib database [88]. Topologies (Test50, Test100, Test200) are generated by BRITE topology generation tool [89] using Waxman model with parameters $\alpha = 0.15, \beta = 0.2, m = 2$. The VNFs can be hosted at any node in the network. All nodes have the same capacity C. The capacity of the links was set to W. For each topology, we have tried two values of pair (C, W), which we refer to as unlimited and limited resources, as listed in Table 4.1.

There are a set of $F = 10$ VNFs available, each VNF requires a computation resource $q_f = \{1, 2, .., 10\}$ (resource unit). Demands require to run service chains composed by 5 VNFs from F. For each demand, its source and destination are generated uniformly at random from all the possible (s^d, t^d) pair. Demands were assumed to have the traffic handling capacity selected from a set $b^d = \{1, 2, .., 10\}$ (bandwidth unit).

We implement the first experiment to evaluate the efficiency of our proposed model and how big of instances we can solve the sub-problem. That means we solve our problem with set of given demands $D^{(t)} = 100$ and different topologies. We also evaluate the performance of generating flow cover inequalities for MILP solving by comparing to without adding flow covers.

In the second experiment, we run the simulation to assess some realistic metrics in a dynamic system such as the blocking rate, the resource utilization and the average length of demand's path. In the simulation, demand arrivals are generated using a Poisson process of parameter λ chosen such that the overall blocking is below 0.2. The holding time of each demand is generated from an exponential process with average $\tau = 3$.

The simulation save the current state of the network as a list of ongoing demands including the location of functions that they use and their routing path. In a dynamic system, the demand can arrive anytime. Therefore, we update the system after a certain time depending on the system configuration. In our simulation, we update the system after $t = 3ms$. Every t, we first check if any demands had finished during the previous

Table 4.2: The Solution Efficiency with Flow Covers for various values of τ in case of unlimited resources

Instances	$\|F\|$	$\|D^{(t)}\|$	n	τ	Objective Value	Link load ($\mathcal{L}$)	Node Load ($\mathcal{N}$)	$\|S0\|$	Number of generated flow covers	GAP	AcceptNo	Time
Abilene	5	100	12		8.8836	0.3364	0.7800	Without adding flow covers		0.0000	100	16.4741
				1	8.8836	0.3364	0.7800	46	1953	0.0000	100	12.8553
				2	8.8836	0.3364	0.7800	46	31032	0.0000	100	13.3232
				3	8.8836	0.3364	0.7800	46	424358	0.0000	100	24.0128
				4	8.8836	0.3364	0.7800	46	3831782	0.0000	100	187.3822
Nobel-eu	5	100	28		9.0933	0.3400	0.5667	Without adding flow covers		0.0012	100	4000.0000
				1	9.0933	0.3400	0.5667	30	1675436	0.0012	100	4000.0000
				2	9.0933	0.3400	0.5667	30	6030415	0.0012	100	4000.0000
				3	9.0878	0.3400	0.5722	30	24725614	0.0018	100	4000.0000
				4	9.0644	0.3467	0.5889	30	20373253	0.0044	100	4000.0000
Nobel-germany	5	100	17		9.3829	0.2800	0.3371	Without adding flow covers		0.0000	100	181.8498
				1	9.3829	0.2800	0.3371	37	156	0.0000	100	70.3442
				2	9.3829	0.2800	0.3371	38	795	0.0000	100	76.4480
				3	9.3829	0.2800	0.3371	38	6285	0.0000	100	95.2017
				4	9.3829	0.2800	0.3371	38	39099	0.0000	100	147.0585
Test50	5	100	50		9.1567	0.5700	0.2733	Without adding flow covers		0.0000	100	9.4003
				1	9.1567	0.5700	0.2733	32	509	0.0000	100	8.6511
				2	9.1567	0.5700	0.2733	32	7379	0.0000	100	10.9185
				3	9.1567	0.5700	0.2733	32	62335	0.0000	100	16.3051
				4	9.1567	0.5700	0.2733	32	669444	0.0000	100	51.7951
Test100	5	100	100		8.7589	0.9611	0.2800	Without adding flow covers		0.0000	100	745.3348
				1	8.7589	0.9611	0.2800	25	1619	0.0000	100	1133.2853
				2	8.7589	0.9611	0.2800	25	12376	0.0000	100	1958.5025
				3	8.7489	0.9611	0.2900	25	352334	0.0023	100	4000.0000
				4	8.7489	0.9611	0.2900	25	2869554	0.0023	100	4000.0000
Test200	5	100	200		9.1111	0.5667	0.3222	Without adding flow covers		0.0000	100	47.1607
				1	9.1111	0.5667	0.3222	21	2709	0.0000	100	42.7482
				2	9.1111	0.5667	0.3222	21	37373	0.0000	100	65.5487
				3	9.1111	0.5667	0.3222	21	462335	0.0000	100	71.9577
				4	9.1111	0.5667	0.3222	21	5669438	0.0000	100	48.9247

inter-arrival time, remove them from the list and then release the resources that they were holding. After that, we run our algorithm including MILP algorithm and heuristic under consideration the resource utilization of total system. The algorithm will find a set of served demands and then added to the list of ongoing demands and the available resources are updated simultaneously.

4.5.2 The Solution Efficiency with Flow Covers for Various Values of τ

Table 4.2 and 4.3 show the efficiency of generating flow covers for solving MILP with different values of $\tau = \{1, 2, .., 5\}$. We run MILP using default values of parameters in GUROBI. We then run MILP with generating flow covers as described in Section 4.2. Regarding the objective value and the gap, generating flow covers with small value of τ makes the problem easier to solve and get the optimal solution more quickly. For instance, with Abilene topology, we can get the optimal solution after 12.8553 seconds when generating some flow covers, where as we need 16.4741 seconds without adding flow covers.

As defined above, τ is number of demands to add cover set in order to generate more cover sets to the problem. When we increase the value of τ, we take more time to find $C^{D^{(t)}}_{D^{(t)}+\tau}$ cover sets for each link $(u, v) \in E^{(t)}$. The Table 4.2 shows that increasing the value of τ, or increasing the number of flow covers, will not get more benefit for solving the

problem. As a consequence, these extended flow covers are not strong for the problem. In addition, it takes a few computation time to find cover sets with the big value of τ.

Therefore, generating flow covers requires the proper number of strong flow covers to get the best gain for solving MILP problem. We need to implement various experiments to find the best value of τ. It is obvious to see that the best value of τ in our simulation is 1.

4.5.3 Comparison the Objective Value of Proposed Algorithm

Table 4.4 and 4.5 show the comparison the objective value of proposed algorithm. As a results, our model can find the optimal solution for a large instance, up to 200 nodes and 100 demands. The objective value obtained by PRH is close to the optimal, except for the biggest instance (Test200). Because, with large instances, the number of variables grows up quickly. Hence, the number of rounded variables increases. As a result, the objective value will decreases. Regarding MSTH, the objective value is quite close to the optimal solution in case of normal resources, see Table 4.4. However this objective values detonate quickly in case of limited resources, see Table 4.5. That means our heuristic MSTH only implements well in case of the normal resources system.

Table 4.3: The Solution Efficiency with Flow Covers for various values of τ in case of limited resources

Instances	$\|F\|$	$\|D^{(t)}\|$	n	τ	Objective Value	Link load ($\mathcal{L}$)	Node Load ($\mathcal{N}$)	$\|S0\|$	Number of generated flow covers	GAP	AcceptNo	Time
Abilene	5	100	12		8.1867	0.9200	0.8933	Without adding flow covers		0.0000	100	12.1562
				1	8.1867	0.9200	0.8933	14	6	0.0000	100	14.1240
				2	8.1867	0.9200	0.8933	14	13	0.0000	100	12.9792
				3	8.1867	0.9200	0.8933	14	43	0.0000	100	15.1745
				4	8.1867	0.9200	0.8933	14	113	0.0000	100	13.6983
Nobel-eu	5	100	28		8.3000	0.8000	0.9000	Without adding flow covers		0.0000	100	627.2564
				1	8.3000	0.8000	0.9000	12	2343	0.0000	100	245.7532
				2	8.3000	0.8000	0.9000	12	10340	0.0000	100	350.3655
				3	8.3000	0.8000	0.9000	12	29849	0.0000	100	337.3407
				4	8.3000	0.8000	0.9000	12	69312	0.0000	100	375.4856
Nobel-germany	5	100	17		8.4850	0.7200	0.7950	Without adding flow covers		0.0000	100	140.5745
				1	8.4850	0.7200	0.7950	19	310	0.0000	100	119.2832
				2	8.4850	0.7200	0.7950	20	1658	0.0000	100	126.8998
				3	8.4850	0.7200	0.7950	20	7341	0.0000	100	89.8468
				4	8.4850	0.7200	0.7950	20	34128	0.0000	100	69.7836
Test50	5	100	50		6.3778	0.5222	1.0000	Without adding flow covers		0.0270	79	4000.0000
				1	6.3778	0.5222	1.0000	14	974137	0.0281	79	4000.0000
				2	6.3778	0.5222	1.0000	14	6968816	0.0277	79	4000.0000
				3	6.3556	0.5444	1.0000	14	28845999	0.0318	79	4000.0000
				4	6.3778	0.5222	1.0000	14	70807063	0.0277	79	4000.0000
Test100	5	100	100		7.2711	0.9889	0.5400	Without adding flow covers		0.0028	88	4000.0000
				1	7.2711	0.9889	0.5400	11	879	0.0028	88	4000.0000
				2	7.2711	0.9889	0.5400	11	4964	0.0028	88	4000.0000
				3	7.2711	0.9889	0.5400	11	123489	0.0028	88	4000.0000
				4	7.2311	0.9889	0.5800	11	868324	0.0099	88	4000.0000
Test200	5	100	200		7.8750	1.0000	0.6250	Without adding flow covers		0.0063	95	4000.0000
				1	7.8750	1.0000	0.6250	12	1257	0.0063	95	4000.0000
				2	7.8750	1.0000	0.6250	12	3806	0.0095	95	4000.0000
				3	7.8750	1.0000	0.6250	12	147960	0.0095	95	4000.0000
				4	7.8750	1.0000	0.6250	12	3580647	0.0095	95	4000.0000

Table 4.4: Comparison the objective value of proposed algorithms in case of unlimited resources

Instances	F	D	n	Optimal Solution				PRH				MSTH			
				Obj Value	Link load ($\mathcal{L}$)	Node Load ($\mathcal{N}$)	AcceptNo	Obj Value	Link load ($\mathcal{L}$)	Node Load ($\mathcal{N}$)	AcceptNo	Obj Value	Link load ($\mathcal{L}$)	Node Load ($\mathcal{N}$)	AcceptNo
Abilene	5	100	12	8.8836	0.3364	0.7800	100	8.8836	0.3364	0.7800	100	7.7109	0.9791	0.7100	94
Nobel-eu	5	100	28	9.0933	0.3400	0.5667	100	9.0933	0.3400	0.5667	100	7.4300	0.9367	0.7333	91
Nobel-germany	5	100	17	9.3829	0.2800	0.3371	100	9.3829	0.2800	0.3371	100	9.1943	0.3800	0.4257	100
Test50	5	100	50	9.1567	0.5700	0.2733	100	9.1567	0.5700	0.2733	100	8.9350	0.6050	0.4600	100
Test100	5	100	100	8.7589	0.9611	0.2800	100	8.7589	0.9611	0.2800	100	8.5689	0.9611	0.4700	100
Test200	5	100	200	9.1111	0.5667	0.3222	100	8.8126	0.5937	0.5269	100	8.7533	0.6133	0.5333	99

Table 4.5: Comparison the objective value of proposed algorithms in case of limited resources

Instances	F	D	n	Optimal Solution				PRH				MSTH			
				Obj Value	Link load ($\mathcal{L}$)	Node Load ($\mathcal{N}$)	AcceptNo	Obj Value	Link load ($\mathcal{L}$)	Node Load ($\mathcal{N}$)	AcceptNo	Obj Value	Link load ($\mathcal{L}$)	Node Load ($\mathcal{N}$)	AcceptNo
Abilene	5	100	12	8.1867	0.9200	0.8933	100	8.1867	0.9200	0.8933	100	4.2300	0.7500	0.7200	57
Nobel-eu	5	100	28	8.3000	0.8000	0.9000	100	8.3000	0.8000	0.9000	100	4.0071	0.9429	0.7500	57
Nobel-germany	5	100	17	8.4850	0.7200	0.7950	100	8.4850	0.7200	0.7950	100	4.8900	0.9000	0.7100	65
Test50	5	100	50	6.3778	0.5222	1.0000	79	6.3778	0.5222	1.0000	79	3.2287	0.8999	0.7714	49
Test100	5	100	100	7.2711	0.9889	0.5400	88	7.2711	0.9889	0.5400	88	4.9000	0.9800	0.8200	67
Test200	5	100	200	7.8750	1.0000	0.6250	95	7.3987	1.0000	0.6013	90	4.8256	0.9994	0.7750	66

4.5.4 Comparison the Computation Time of Proposed Algorithm

Fig. 4.2 depicts the average computation time per demand of three our algorithms. It is worth mentioning, that in terms of execution time the MILP algorithm requires about 800 mili-seconds per demand to run, RPH requires about 8000 mili-seconds, where as MSTH only 0.02 mili-seconds. Obviously, the longer time means the better solutions we get.

Especially, RPH (the green line) takes a long time to obtain the solutions. It is because of using multiple loops for relaxing MILP before solving the optimal MILP problem to make the problem easier. It is influenced strongly by the values of α, p (In our simulation, we use $\alpha = 0.8, p = 100$). As analyzed in Sec.4.5.3, the objective value obtained by RPH is close to the optimal, see in Table 4.4 and Table 4.5. Hence, it is worth to use RPH in case of a hard instance of the problem.

4.5.5 The Impact of The Arrival Rate of Demands

Fig. 4.3 shows the blocking rate of the Nobel-Eu topology ($W = 100, C = 150$) for both heuristic and optimal solution obtained by MILP algorithm. As was expected, the MILP algorithm provides the optimal solution with the small blocking rate. Moreover, our heuristic is quite good with the value of λ is not too big (see the blue line).

Regarding the maximum link utilization (Fig.4.4) and the average length of path (Fig. 4.5), our heuristic is good comparing to the optimal solution by MILP algorithm.

4.5.6 The Impact of Resource Capacity

In the section, we would like to investigate the effect of resource capacity to both algorithms. We use the Abilene topology.

Firstly, we keep the value of link capacity $W = 20$ and we then increase the value of node capacity C from 30 to 100. Fig.4.6(a) shows the blocking rate of both heuristic and

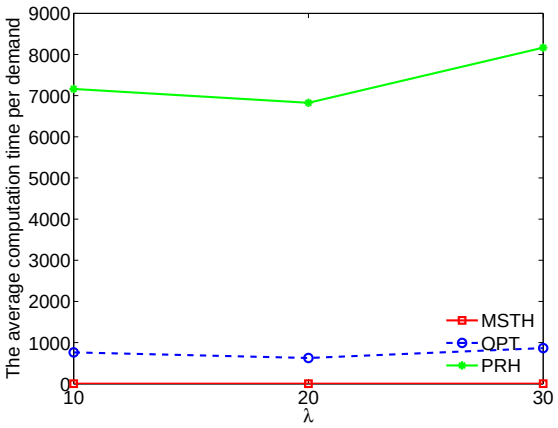


Figure 4.2: The average running time per demand vs λ

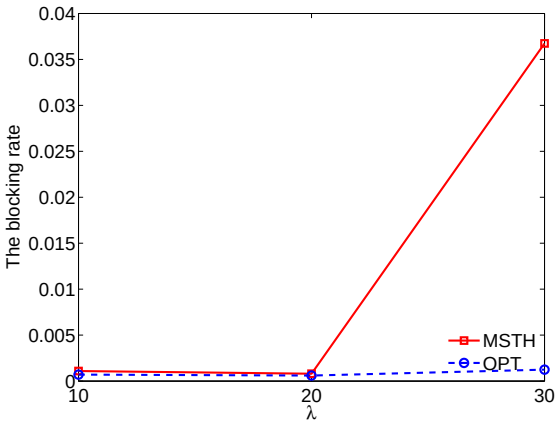


Figure 4.3: The blocking rate vs λ

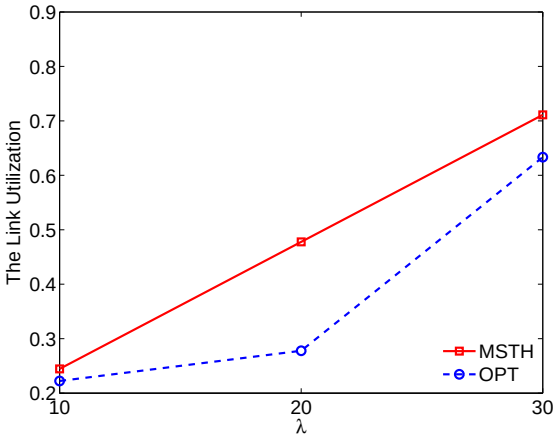


Figure 4.4: The maximum link load vs λ

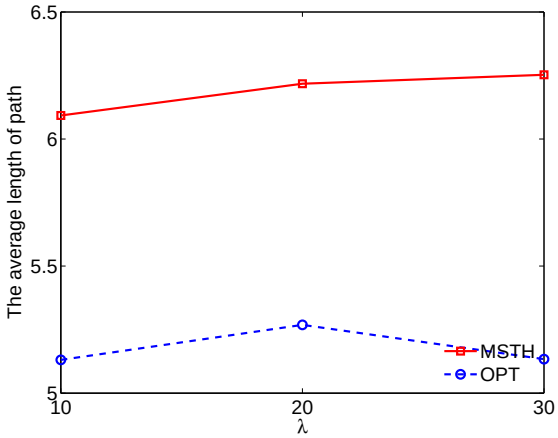
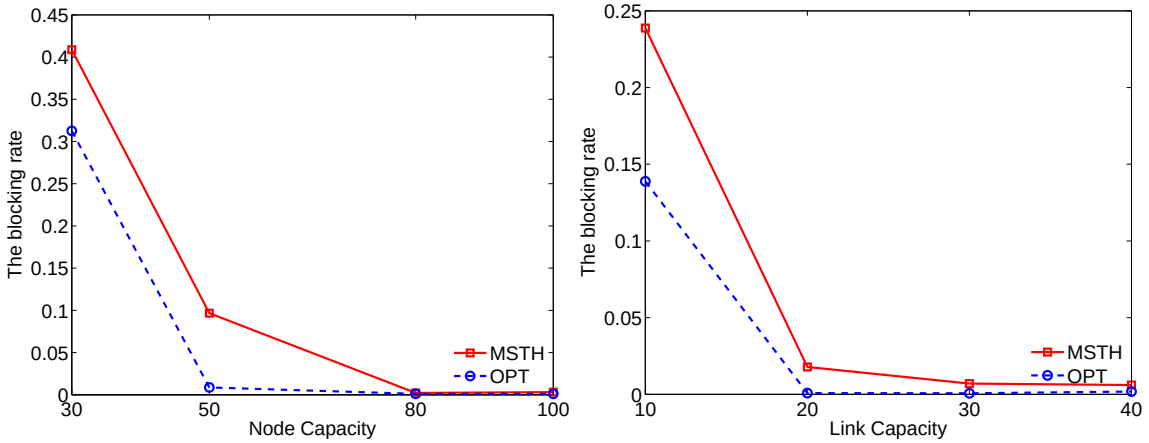


Figure 4.5: The average length vs λ

optimal. We can see that, if node capacity increases to a value, the blocking rate obtained by heuristic algorithm can be small and converge into the optimal solution.

We secondly implement an experiment to show the effect of link capacity W by keeping the value of node capacity $C = 50$, and then increase the value of link capacity W from 10 to 40, see Fig.4.6(a). The figure show that we get the same result as Fig. 4.6(b). That means if link capacity increases to a value, the blocking rate will converge. It is the fact that we can obtain the same profit when investing resource in nodes and links.



(a) Node Capacity (b) Link Capacity

Figure 4.6: The blocking rate vs Resource capacity

4.6 Conclusions

In this chapter, we studied the problem of optimizing the resource utilization in the context of operating an NFV dynamic system. We proposed a new efficient model as a Mix Integer Linear Programming while taking into account the decision of Service Providers to serve a demand or reject it. The overall solution is enhanced by generating some strong flow covers for MILP solver. The main point in contrast with existing works is that the model is linear and can find the optimal solution for a large set of instances. The results show that we can obtain the optimal solution for the problem up to 200 nodes and 100 demands. Furthermore, we proposed two efficient heuristics to find good quality feasible solutions for larger instances within an acceptable execution time. To complete our study, we evaluated our solution in various scenarios and used an event-driven simulation to assess some realistic metrics in an NFV dynamic system.

Chapter 5

Routing Via Functions in NFV

Contents

The two previous chapters concentrate on the resource management and migration techniques for NFV system. It focuses on the VNFs placement rather than provides a quick solution for each request in a large network. In this chapter, we study a resource distribution for large NFV system while considering that we know possible places of VNFs on network nodes, called the *function nodes*. The purpose is to find a path very quickly through network that will meet all requirements and visit the *function nodes* in the prescribed order. It is due to the fact that in a large network, the requests arrival rate maybe large and a decision has to be taken fast enough that there is no build-up of request waiting for a connection path.

We develop the solutions to satisfy above requirements. In the next section, we introduce the research context for this problem and describes the system model in Section 5.2. An expanded network and Minimum-cost Flow Model are presented detail in Section 5.3 - 5.4. We propose in Section 5.5 some approximate algorithms based on the Lagrangian relaxation of the flow problem. Finally, we evaluate the performance of these approximate algorithms in Section 5.6 and compare them to a straightforward heuristic called GreedyForward.

5.1 Context

The softwarization of network functions, relying upon virtualization techniques, aims to support network operators to meet the evolving customer requirements while controlling capital and operational expenditures. In addition, the agility of this approach should be able to efficiently match the resource consumption with these requirements. As a consequence, network functions that were usually embedded in hardware devices are now virtualized and hosted in nodes of the network virtual infrastructure. A network service can therefore be decomposed into a Service Chain, namely an ordered sequence of VNFs, which can run on several standard physical nodes. These functions will exist in more than one node and could migrate from one node to another over time, depending on operational criteria. The demand from a user will result in a flow from a source to a destination across nodes hosting the functions involved in the Service Chain, respecting their precedence constraints. This raises important new issues related to efficient routing in NFV. The problem is formalized according to the description below.

We consider a network where connection[1] requests arrive randomly with random holding time. Each connection must use an SC, i.e., a given set of functions, specific to the

[1]We use the terms connection and demand interchangeably.

connection, in a prescribed order. Multiple of these functions are placed a priori in some subset of nodes, called the *function* nodes.

A connection needs a certain amount of link bandwidth and whenever it goes through a node, it also needs a certain processing time as well as an additional processing time whenever a function is computed in the node. For each connection request with given origin and destination, we want to find a path through the network that will meet all requirements and in particular, will visit the function nodes in the prescribed order. If this is not possible, the connection is blocked.

The problem is to design an algorithm that will find such a path very quickly for each request in a large network. The time limit is due to the fact that the request's arrival rate may be large and a decision has to be taken fast enough that there is no build-up of requests waiting for a connection path.

If we consider a single request, we could argue that any path that meets the requirements will do. If we consider the broader network context, we want to find paths that will produce as low a blocking probability as possible. The quality of an algorithm will be evaluated first by the blocking probability that it produces in the network, second by the average path length it produces and third, by the average computation time per request.

This work offers a number of significant contributions for the solution of this problem:

1. The first and most important, is our proposal to use an **expanded network** to meet the constraints on Service Chain. This is discussed in Section 5.3.

2. Next, in Section 5.4, we give a precise mathematical formulation of the routing problem as an integer minimum-cost flow model with side constraints on the expanded network

3. Because it is unlikely that we can compute exact solutions to that problem in the required time, we propose, in Section 5.5, several fast approximate algorithms based on the Lagrangian relaxation of the flow problem. These algorithms differ by the number of iterations that they solve and the cost functions that are used.

4. Finally, we evaluate the performance of these approximate algorithms in Section 5.6 and compare them to a straightforward heuristic called GreedyForward.

The main messages are two fold: 1) routing via functions that are present in more than one node is hard and requires the use of an expanded network to take the precedence constraint into account; 2) a solution based on one iteration of the Lagrangian relaxation performs very well if the cost functions are selected well.

Table 5.1: Notation

$\mathcal{N}$ (N)	set of nodes (# of nodes)
$\mathcal{L}$ (L)	set of directed links (# of links)
$C_{i,j}$	(integer) transmission capacity of the directed link (i,j)
P_i	(integer) processing capacity of node i
c	a connection
$\mathcal{F}_c$ (K_c)	sequence of functions $f_1, \ldots f_{K_c}$ required by c (# of functions)
$\mathcal{A}_{k,c}$ $(n_{k,c})$	set of nodes containing f_k when c arrives (corresponding # of nodes)
b_c (p_c)	bandwidth (processing in each traversed node) requirement of c
$q_{k,c}$	processing requirement of function f_k in a node for c

5.2 The System Model

We consider a network characterized by its set of nodes $\mathcal{N}$ with $|\mathcal{N}| = N$, and its set of links $\mathcal{L}$ with $|\mathcal{L}| = L$ where $C_{i,j}$ is the (integer) transmission capacity of the directed link (i,j) and P_i is the (integer) processing capacity of node i. There is also a given set of functions $\mathcal{F} = \{f_1, \ldots, f_F\}$ and a number of copies of each function f_k are placed a priori in some subset $\mathcal{A}_k$ of nodes, called *function* nodes, with $|\mathcal{A}_k| = n_k$. A function f_k can only be computed once in $\mathcal{A}_k$.

Connection requests arrive randomly following a Poisson process with rate λ and a exponential random holding time of mean τ. Each connection c has a number of requirements, either on the links or the nodes. First, it needs a certain amount of link bandwidth b_c on each link it uses. Also, whenever connection c goes through a node, it also needs a processing time p_c. Finally, connection c must use a given sequence of functions $\mathcal{F}_c = \{f_1(c), \ldots, f_{K_c}(c)\}$, in that prescribed order, called the *Service Chain*, yielding *precedence* constraints. This entails an an additional processing time $q_k(c)$ whenever a function $f_k \in \mathcal{F}_c$ is computed in some node. The notation is summarized in Table 5.1.

For each connection c with given origin and destination $o(c)$ and $d(c)$, we want to find a path through the network that will meet all requirements and visit the function nodes in the prescribed order.

We assume that when a connection request c arrives, we have complete knowledge on the network state, i.e., the remaining capacity of all the links and nodes noted $C_{i,j}(c)$ and $P_i(c)$. From this, we can exclude from the network the links where the available transmission capacity is smaller than b_c or the remaining processing capacity is smaller than p_c, since they cannot carry the connection. We call this *pruning* the network. Here, the pruning is based on the values of b_c and p_c and the current state of the network. Let $\mathcal{R}(c)$ be the pruned network at the time the connection request for c arrives where $\mathcal{N}_{\mathcal{R}}(c)$ is the set of nodes and $\mathcal{L}_{\mathcal{R}}(c)$, the set of links in $\mathcal{R}$. We describe next the notion of expanded network which will be used to find a path for a new connection request c.

5.3 The Expanded Network

5.3.1 Introduction

The main difficulty with the problem of finding a path for connection c in $\mathcal{R}(c)$ is how to model the precedence constraints. The technique we have used is based on the following argument. Consider connection c that arrives to the network at some node $o(c)$. One could route it through a number of intermediate nodes before arriving at one of the nodes that contain $f_1(c)$. The connection can then be routed through a second set of intermediate nodes before arriving at one of the nodes that contain $f_2(c)$. The procedure is repeated until the connection finally reaches $d(c)$.

This suggests that the overall path from $o(c)$ to $d(c)$ can be viewed as a set of sub-paths, or segments, whose collection makes up the complete path. These segments are connected by decision points where we choose which nodes will compute the functions in $\mathcal{F}_c$. Since these segments can be chosen any way we want from the real graph, we construct an *expanded network* to model both the segment construction and the function selection at the nodes. We will then try to route the connection in this expanded network, which is built in such a way that all the precedence constraints will be met. The expanded network $\mathcal{R}_e(c)$ build out of $\mathcal{R}(c)$ is defined by its set of nodes $\mathcal{N}_e(c)$ and its set of links $\mathcal{L}_e(c)$. Note that we need to build a new expanded network for each new connection request.

First, we construct $K_c + 1$ *copies* of the real network. Copy $\mathcal{R}_k(c)$ is simply the set of nodes and links of the original network $\mathcal{R}(c)$. These are denoted $\mathcal{N}_k(c)$ and $\mathcal{L}_k(c)$ when we want to be specific as to which copy we are referring to. We use each copy to find one segment of the total path. For each $\mathcal{R}_k(c)$, we also define a set $\hat{\mathcal{L}}_k(c)$ of artificial links to connect $\mathcal{R}_k(c)$ and $\mathcal{R}_{k+1}(c)$. These artificial links can be defined for each node $i \in \mathcal{R}_k(c)$ that carries function f_k. The artificial link connects this node $i \in \mathcal{R}_k(c)$ to node $i \in \mathcal{R}_{k+1}(c)$. The interpretation is that a path goes through this artificial link if and only if it uses function f_k from node i. Altogether,

$$\begin{aligned}
\mathcal{N}_e(c) &= \bigcup_{k=1}^{K_c+1} \mathcal{N}_k(c) \\
|\mathcal{N}_e(c)| &= (K_c + 1)N_R(c) \\
\mathcal{L}_e(c) &= \bigcup_{k=1}^{K_c+1} \mathcal{L}_k(c) \bigcup_{k=1}^{K_c} \hat{\mathcal{L}}_k(c) \\
|\mathcal{L}_e(c)| &= (K_c + 1)L_R + \sum_{k=1}^{K_c} n_k(R, c)
\end{aligned}$$

where $n_k(R, c)$ is the number of nodes containing function f_k in the pruned network.

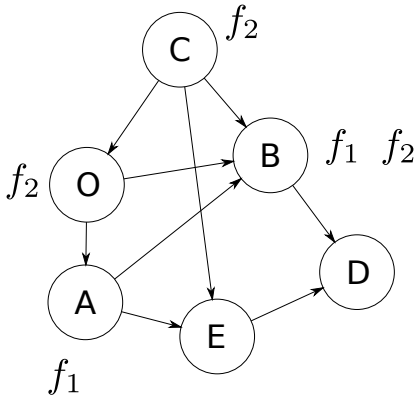


Figure 5.1: The pruned network when connection request c arrives

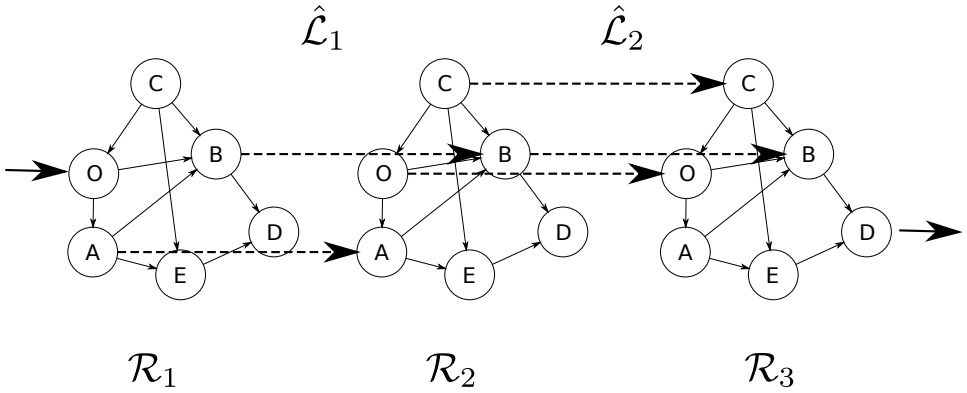


Figure 5.2: Expanded network for c from O to D where the index c is omitted for simplicity

As an example, consider the network of Figure 5.1 where we have two functions that are available in the nodes shown in the figure. The corresponding expanded network for c from $o(c) = O$ to $d(c) = D$, that has a SC $\{f_1, f_2\}$ is shown in Figure 5.2, where the artificial links are shown as dotted lines. We can see from this figure that a given node or link in the real network can appear in many places in the expanded network. Also, by construction, any path from node O in $\mathcal{R}_1$ to node D in $\mathcal{R}_3$ will pick up each function only once and in the right order since the only way to go from $\mathcal{R}_k$ to $\mathcal{R}_{k+1}$ is through one of the artificial links.

Remark 1. *The first step of any solution that we propose will be the creation of the expanded network for each new connection request.*

Remark 2. *The expanded network for connection c is $(K_c + 1)$ times larger than the original network. The algorithms that we will design would have to be "fast" on the expanded network.*

5.3.2 Notation

For ease of notation, we will omit the index c in the following. In general, there will be more than one copy of a given node or link of the original pruned network in the expanded network. For this reason, we need some indexing convention whenever we want to specify precisely which copy of a real node or link we are talking about. This is also useful to simplify the notation for some constraints in the mathematical model.

If we want to single out a node i in some $\mathcal{R}_k$, we use the notation (i, k). Similarly, a link in $\mathcal{R}_k$ is denoted by the pair $((i, k), (j, k))$ which we shorten to (i, j, k). A link in $\hat{\mathcal{L}}_k$ is denoted by (i, i, k). This is the artificial link from the function node i in $\mathcal{R}_k$ to the same node i in $\mathcal{R}_{k+1}$. It only exists if $i \in \mathcal{A}_k$ and $i \in \mathcal{N}_e$ since i would have been pruned if its remaining processing capacity is less than p.

We also define the following node sets to make the expression of the model easier. For each node $(i, k) \in \mathcal{N}_e$, we define the sets of incoming $\mathcal{I}$ and outgoing $\mathcal{O}$ links. Precisely, $\mathcal{I}(i, k)$ (resp. $\mathcal{O}(i, k)$) is the set of nodes j such that there is an incoming (resp. outgoing) link (j, i, k) (resp. (i, j, k)). Note that in these definitions, we can have $i = j$ whenever a link is an artificial link to or from node i in some set $\mathcal{R}_{k-1}$ or $\mathcal{R}_{k+1}$, as the case may be.

5.3.3 Capacity Constraints and Loops

We have defined $C_{i,j}$ as the leftover capacity on link (i, j) and P_i, the leftover capacity on node i when the connection request for c arrives. Recall that we pruned the real network by removing a priori the links and nodes that do not have enough resources to carry the connection. One might think that this would be enough to guarantee that a connection can

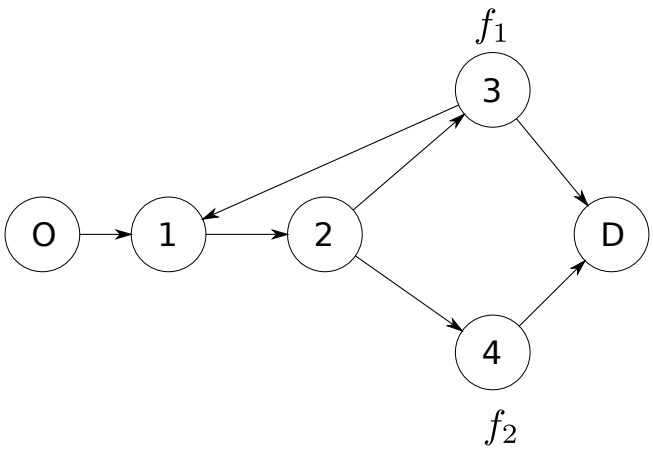


Figure 5.3: Capacity constraints and the possible need for loops

be routed. In fact, this is not enough, as we can see from Figure 5.3, where the functions to visit are f_1 followed by f_2. Assume that when the connection arrives, the available bandwidth is 1.5 units on all links and the connection needs 1 unit per used link. After the pruning, all the links would be seen as feasible and would show up in the expanded network. If we simply chose a path without taking into account the capacity, we could find a path $(O, 1)$, $(1, 2)$, $(2, 3)$, $(3, 1)$, $(1, 2)$, $(2, 4)$, $(4, D)$ where the connection needs to go twice through link $(1, 2)$. For this, we need 2 units on link $(1, 2)$ which is not possible. In other words, we need to define a model with explicit capacity constraints. Note that we do not know ahead of time how many times a path would have to go via a link or a node so that we cannot prune more than we do. Thus we cannot avoid explicit capacity constraints on the links and the nodes.

Remark 3. *This example also shows that we* cannot avoid loops *in this kind of problems since if the link capacities were equal to 2 units, the only solution would be to use link* $(1, 2)$ *twice.*

5.4 Minimum-Cost Flow Model

Once we have created the expanded network for a new connection request, we need to find a path for that connection on that network. We model this problem as an integer program subject to capacity constraints in the expanded network.

5.4.1 Finding a Feasible Path

The first part of the model has to do with finding a path that will meet the capacity constraints. For this, we define $x_{i,j,k}$, the binary decision variables indicating that link

$(i, j, k) \in \mathcal{L}_k$ is in the path. The problem can be stated as finding a set of $(x_{i,j,k})$ that meet the constraints

$$\sum_{j \in \mathcal{I}(i,k)} x_{j,i,k} = \sum_{j \in \mathcal{O}(i,k)} x_{i,j,k} + \delta_{i,k} \quad \forall (i,k) \in \mathcal{N}_e \tag{5.1}$$

$$\delta_{i,k} = \begin{cases} 1 & \text{if } i = k = 1 \\ -1 & \text{if } k = K, i = KN \\ 0 & \text{otherwise} \end{cases}$$

$$b \sum_k x_{i,j,k} \leq C_{i,j} \quad \forall (i,j) \in \mathcal{L}_k \tag{5.2}$$

$$p \sum_k \sum_{j \in \mathcal{I}(i,k)} x_{j,i,k} + \sum_k q_k \sum_{(i,i) \in \hat{\mathcal{L}}_k} x_{i,i,k} \leq P_i \quad \forall i \in \mathcal{N} \tag{5.3}$$

$$x_{i,j,k} \in \{0, 1\} \quad \forall (i,j,k) \in \mathcal{L}_e. \tag{5.4}$$

Equation (5.1) is the standard flow conservation equation at node (i, k). The term $\delta_{i,k}$ represents one unit of flow entering at o in copy 1 and leaving at node d in copy $K + 1$. These equations insure that there will be one and only one path between node $(o, 1)$ and $(d, K + 1)$. The total capacity constraint on each link $(i, j) \in \mathcal{L}_k$ is given by (5.2). The processing constraint (5.3) is made up of two terms. The first one is the total processing required whenever a connection enters a node and the second is the processing required to execute f_k. In this model, we only need to find a set of feasible $x_{i,j,k}$ and we call this the *feasibility* problem.

Note that as far as the current connection is concerned, *any* set of $x_{i,j,k}$ that meet the constraints (5.1–5.4) is good enough in the sense that it allows us to connect the request. However, as we mentioned in Section 5.1, there is another requirement for the path selection algorithm: It should produce paths such that the network blocking is as low as possible. This means that we should use paths that use as few links and nodes as possible.

5.4.2 Finding Good Paths

We can take into account the requirement for short paths by adding an artificial length, or transportation cost, to the links, and converting the feasibility problem of Section 5.4.1 into a minimum-cost flow problem. First, we define $a_{i,j,k} \geq 0$ as the cost of transporting one unit of flow on link (i, j, k). Note that these costs *do not* represent an actual transportation cost. Rather, we will use them in Section 5.5.3 as a tool to lower the blocking. We will choose them differently for different approximate solution algorithms. We then add an objective function Z to the feasibility problem so that the path finding model becomes the following

minimum-cost flow problem.

$$\min_{\mathbf{x}} Z = \sum_{k} \sum_{(i,j)\in\mathcal{L}_k} a_{i,j,k} x_{i,j,k} + \sum_{k} \sum_{(i,i)\in\hat{\mathcal{L}}_k} a_{i,i,k} x_{i,i,k} \tag{5.5}$$
$$\text{subject to } (5.1 - 5.4).$$

Note that constraints (5.2) and (5.3) couple a number of links and nodes in the expanded network. This means that we cannot use the standard argument for capacitated minimum-cost flows that automatically have integer solutions for integer inputs. For this to be valid, the capacity constraints must apply to *each* link separately. This is why we need to state explicitly that the $x_{i,j,k}$'s are integer and why we must solve it with an integer programming algorithm. This makes the problem difficult and it is unlikely that we will be able to find optimal solutions for any network of realistic size within a time short enough to use in a virtual network environment. For this reason, we now focus on fast approximate algorithms. Most of these are based on the Lagrangian Relaxation of (5.5).

5.4.3 Lagrangian Relaxation

Problem (5.5) is known as the minimum-cost flow with side constraints. It is particularly well suited for a solution by Lagrangian relaxation since the subproblems are simple. We construct the partial Lagrangian function by relaxing constraints (5.2) and (5.3) with multipliers $\gamma_{i,j}, \mu_i \geq 0$. We get the Lagrange function

$$\begin{aligned}
\mathcal{L}(\mathbf{x}, \boldsymbol{\gamma}, \boldsymbol{\mu}) = & \sum_{k} \sum_{(i,j)\in\mathcal{L}_k} \left(a_{i,j,k} + b\gamma_{i,j}\right) x_{i,j,k} \\
& + \sum_{k} \sum_{(i,i)\in\hat{\mathcal{L}}_k} a_{i,i,k} x_{i,i,k} \\
& + p \sum_{i\in\mathcal{N}} \mu_i \sum_{k} \sum_{j\in\mathcal{I}(i,k)} x_{j,i,k} \\
& + \sum_{k} \sum_{(i,i)\in\hat{\mathcal{L}}_k} \mu_i q_k x_{i,i,k} \\
& - \left(b \sum_{(i,j)\in\mathcal{L}} \gamma_{i,j} C_{i,j} + \sum_{i\in\mathcal{N}} \mu_i P_i \right).
\end{aligned}$$

The dual function Φ is defined at some feasible point $(\boldsymbol{\gamma}, \boldsymbol{\mu})$ as

$$\Phi(\boldsymbol{\gamma}, \boldsymbol{\mu}) = \min_{\mathbf{x}} \mathcal{L}(\mathbf{x}, \boldsymbol{\gamma}, \boldsymbol{\mu}) \tag{5.6}$$
$$\text{subject to } (5.1) \text{ and } (5.4). \tag{5.7}$$

The minimization over $\mathbf{x}$ is called the *subproblem* for some value of $\boldsymbol{\lambda}$ and $\boldsymbol{\gamma}$. The capacity constraints (5.2) and (5.3) are removed from the subproblem and so it becomes an uncapacitated minimum-cost flow with the link lengths $w_{i,j,k}$ as

$$w_{i,j,k} = \begin{cases} a_{i,j,k} + b\gamma_{i,j} + p\mu_j & \text{if } i \neq j \\ a_{i,i,k} + q_k\mu_i & \text{if } i = j. \end{cases} \tag{5.8}$$

This problem can be computed very efficiently by any shortest-path algorithm. The first line of 5.8 is for the real links and the other, for the artificial links. Subproblem (5.6)–(5.7) can be solved very quickly, even for large expanded networks. This is the reason why the Lagrangian decomposition approach is so attractive.

Because of weak duality, we know that $\Phi(\boldsymbol{\gamma}, \boldsymbol{\mu}) \leq Z(\mathbf{x})$ for any feasible set of vectors $\boldsymbol{\gamma}, \boldsymbol{\mu}$ and $\mathbf{x}$. We then try to maximize the value of Φ over the dual variables $\boldsymbol{\gamma}$ and $\boldsymbol{\mu}$ to improve that bound and hopefully get a primal feasible solution. The standard technique is the subgradient method where the $\boldsymbol{\gamma}, \boldsymbol{\mu}$ are modified iteratively at iteration s using:

$$\gamma_{i,j}^{(s+1)} = \max\left\{0, \gamma_{i,j}^{(s)} + \alpha^{(s)} g_{i,j}^{(s)}\right\} \tag{5.9}$$

$$\mu_i^{(s+1)} = \max\left\{0, \mu_i^{(s)} + \alpha^{(s)} h_i^{(s)}\right\} \tag{5.10}$$

$$g_{i,j}^{(s)} = b\sum_k x_{i,j,k}^{(s)} - C_{i,j}$$

$$h_i^{(s)} = p\sum_k \sum_{j \in \mathcal{I}(i,k)} x_{j,i,k}^{(s)} + \sum_k \sum_{(i,i) \in \hat{\mathcal{L}}_k} q_k x_{i,i,k}^{(s)} - P_i$$

where $\alpha^{(s)}$ is called the *step size* at iteration s. Simply stated, (5.9) says that one should increase the $\gamma_{i,j}$'s for links where the flow is larger than the capacity and decrease it if it is smaller. The same rule applies for (5.10) when the processing requirement exceeds the node capacity.

To summarize, we have reduced the initial problem to a sequence of shortest path problems on the expanded network. Given enough time, the sequence of subgradient iterations will yield an optimal solution to the dual that will be feasible for the primal. As we have said before, however, we need to compute solutions quickly so that iterative procedures such as this are not possible. For this reason, we now look at approximate algorithms based on a few iterations.

5.5 Approximate Algorithms

We are now proposing different algorithms to find a feasible solution in a small amount of time. The first one is a greedy algorithm, called *Greedy Forward* in the following, that

works on the original network. The other algorithms that we propose belong to two classes. The first class is based on a single iteration of the subgradient method and the algorithms in that class differ from each other in terms of the cost functions that are used. The second class is based on two iterations and the algorithms in that class also differ from each other in terms of the cost functions that are used. The algorithms in these two classes work on the expanded network.

5.5.1 Greedy Forward

The basic idea of the algorithm is to find a sequence of shortest paths linking the functions. We start by finding the shortest path from the origin o to set of nodes $\mathcal{A}_1$ containing the first function f_1 required by the connection. Let i_1 be the selected function node. Then we find the shortest path from i_1 to set of nodes $\mathcal{A}_2$ containing the second function f_2 required by the connection. Let i_2 be the selected function node. Then, for adjacent functions $(f_j, f_{j+1}) \in \mathcal{F}_c$, it finds the shortest path from i_j to set of nodes $\mathcal{A}_{j+1}$ containing function f_{j+1}. Finally, it finds the shortest path from i_K containing function f_K to the destination d of the connection.

Remark 4. *Despite its simplicity, the complexity of this algorithm is quite high since a shortest path has to be computed from i_j to the n_k function nodes in $\mathcal{A}_{k+1}$ for each function set k.*

The pseudo-code of the algorithm is provided in Algorithm 4.

Algorithm 4

function GREEDYFORWARD($c \in C$)
  $p \leftarrow \emptyset$
  $\mathcal{A}_0 \leftarrow o(c), \mathcal{A}_{K+1} \leftarrow d(c)$
  for a pair (f_i, f_{i+1}),$i = 0..K+1$ **do**
    $Z_1 \leftarrow \mathcal{A}_i, Z_2 \leftarrow \mathcal{A}_{i+1}$
    $p_i \leftarrow$ shortest_path$(n_1 \in Z_1, n_2 \in Z_2)$, satisfying resource constraints.
    $\mathcal{A}_{i+1} = n_2$
    $p \leftarrow p \cup p_i$
  end for
  return p
end function

5.5.2 The Subgradient Method: Choosing the Costs

Recall from Section 5.4.2 that the link costs can be chosen as we want to reduce network blocking. Because we know that paths that use a larger number of links and nodes often

produce large blocking, we may simply choose all the $a_{i,j,k} = 1$. This is equivalent to computing a *min-hop* path in the expanded network.

One problem with this choice is that it depends only on the expanded network topology and does not consider the link and node utilization at the time the connection arrives. Thus, a second option is to choose values for the link costs $a_{i,j,k}$'s to take into account the bandwidth and processing capacities. We want to choose these costs to be high when they are close to their limit, thus leaving some resources available for later connections. We propose the following costs:

$$a_{i,j,k} = \begin{cases} \dfrac{b}{C_{i,j}} & \text{if } i \neq j \\ \dfrac{q_k}{P_i} & \text{if } i = j. \end{cases} \tag{5.11}$$

Remark 5. *The costs defined here correspond to two families of algorithms. We will use the term* min-hop *when we use* $a_{i,j,k} = 1$*, and* shortest path with unequal costs *when we use* (5.11)*. If we want to talk about both methods as one group, we use the plural form* shortest paths*.*

5.5.3 Subgradient Method: A Single Iteration

The first step of this method is to build the expanded network which might not be connected depending on the remaining capacities and on the pruning. This method operates on that expanded network. First, set $\boldsymbol{\gamma} = \boldsymbol{\mu} = 0$ which is equivalent to ignoring the capacity constraints. The link length is thus

$$w_{i,j,k} = a_{i,j,k} \quad \forall (i,j,k) \in \mathcal{L}_e.$$

The objective (5.5) is now simply the sum of the flow costs on all links. We first compute the min-cost path(s) from o to d. It may turn out that there is no such path because of the pruning and we say, in that case, that the connection has experienced a *blocking of type 1*. Note that, in that case, nothing can be done, there simply does not exist a path for the connection.

If we find some solutions for the unconstrained problem, we check if the capacity constraints (5.2–5.3) are met, in which case we say that the solution is feasible.

1. If there are feasible solutions, we consider two cases.
   (a) If there is only one, this is the optimal path.
   (b) If there are more than one, we choose one at random.
2. If there is no feasible solution, we say that the connection has experienced a *blocking of type 2*.

At this point, we may simply stop and declare that the connection is lost. The other option, if time permits, is to make one more iteration and try to find a feasible solution.

Remark 6. *This method will deliver solutions that are different depending on the way we choose the costs. We have tried different costs and found that the costs given in* (5.11) *gave the best results, i.e., the lowest blocking, as we shall see in Section 5.6.*

5.5.4 Subgradient Method: Two Iterations

The subgradient Method of Section 5.5.3 ignores the capacity constraints when we compute the paths. This makes faster calculation but does not guarantee feasible solutions. When we have a type 2 blocking, we then try one iteration of the *subgradient* algorithm to try to find a feasible path.

The Pseudo-code is provided in Algorithm 5.

As explained in Section 5.4.3, the subgradient algorithm increases the link weights that exceed their bounds and decreases the ones that are below. For a small step size α, the new link lengths may not be very different from the current values and it may turn out that the new shortest path is the same as before.

In a complete subgradient procedure, we would make many iterations and the lengths would eventually change enough that the current path will no longer be the shortest one and the algorithm will produce a new solution. Because of the time limit, however, we are limited to only one iteration so that we cannot go through this gradual adjustment process. This is why we want to choose a step size that makes sure that the new shortest path will be different from the current one after changing the multipliers. More precisely, we want the step size to be large enough that the link lengths of the current shortest path will increase by an amount large enough that it will no longer be the optimal solution.

First consider some paths l that do not satisfy some capacity constraints after the first iteration. Define J_l and $\hat{J}_l$ as the length of path l before and after changing the multipliers. We can compute the following:

$$\begin{aligned} J_l &= \sum_{(i,j,k)\in l} a_{i,j,k} x_{i,j,k} \\ \hat{J}_l &= \sum_{(i,j,k)\in l} a_{i,j,k} x_{i,j,k} \\ &\quad + \sum_{(i,j,k)\in l} (b\gamma'_{i,j} + p\mu'_j) x_{i,j,k} + \sum_{(i,i,k)\in l} q_k \mu'_i x_{i,i,k} \end{aligned}$$

Algorithm 5

function A-2($\mathcal{R}, c \in C$)
  $\{P_e, p_{min}\} \leftarrow \{\emptyset, \emptyset\}$
  Construct all feasible expanded graphs $\{\mathcal{R}_e\}$ based on the expanded graph describing in Section.5.3 while considering the node capacity constraints.
  P_e = a set of the shortest paths from $o(c)$ to $d(c)$ on $r_e \in \mathcal{R}_e$
  if $P_e == \emptyset$ **then**
    Reject connection c
    return Blocking of type 1.
  end if
  for p_e in P_e **do**
    Convert p_e of $\mathcal{R}_e$ to p of $\mathcal{R}$
    Check the resource constrains on the pruned graph $\mathcal{R}$
    if p is a feasible solution **then**
      p_{min} = shortest_path $\{p, p_{min}\}$
    end if
  end for
  if $p_{min}! = \emptyset$ **then**
    Update available resources
    return p_{min}
  else
    Recording the Blocking of type 2.
    Compute α from equation (5.14)
    Compute the new multipliers from equation (5.10), then update the new link costs from equation (5.4), we obtain the new expanded graph $\hat{\mathcal{R}}_e$
    Re-compute the shortest paths with the new link costs, called $\hat{P}_e$.
    if $\hat{P}_e == \emptyset$ **then**
      Reject connection c
      return Blocking of type 3.
    end if
    for $\hat{p}_e$ in $\hat{P}_e$ **do**
      Convert $\hat{p}_e$ of $\hat{\mathcal{R}}_e$ to $\hat{p}$ of $\mathcal{R}$
      Check the resource constrains on the pruned graph $\mathcal{R}$
      if $\hat{p}$ is a feasible solution **then**
        p_{min} = shortest_path $\{\hat{p}, p_{min}\}$
      end if
    end for
  end if
  return p_{min}
end function

where $\boldsymbol{\gamma}'$ and $\boldsymbol{\mu}'$ are the updated values from (5.9–5.10). The difference of path length when changing the multipliers is thus described as follows:

$$\hat{J}_l - J_l = \sum_{(i,j,k)\in l} (b\gamma'_{i,j} + p\mu'_j)x_{i,j,k} + \sum_{(i,i,k)\in l} q_k\mu'_i x_{i,i,k}. \tag{5.12}$$

From (5.9–5.10), we get:

$$\begin{aligned}\hat{J}_l - J_l &= \alpha \left(\sum_{(i,j,k)\in l} (bg_{i,j} + ph_j)x_{i,j,k} \right. \\ &\quad \left. + \sum_{(i,i,k)\in l} q_k h_i x_{i,i,k} \right) \\ &= \alpha S.\end{aligned}$$

Note that in general, some, but not all, g and h can be negative so that the slope S could be negative. In this case, we cannot increase the length of path l. We can avoid this problem by fixing the length of the links with negative h or g to their current value and computing S only for those links where $g \geq 0$ and $h \geq 0$. In order to insure that a new path will be found after the update, we impose the condition that the difference of the lengths of the paths before and after changing the multipliers with step size α has to be greater than some value δ.

$$\hat{J}_l - J_l \geq \delta. \tag{5.13}$$

From this, we get the condition on α:

$$\alpha \geq \frac{\delta}{S}. \tag{5.14}$$

Obviously, the bigger the value of δ, the more chances we have of making the current path non-optimal. We have tried several values of δ, i.e., $\{1, 2, 3, 5, 7, 10, 20, 25, 30\}$ to find the best parameter. We found out that $\delta = 10$ gives the best results.

5.6 Evaluation of the Approximate Algorithms

All the numerical results were obtained on Intel Xeon X5690 processors running at 3.47GHz with 16 cores and 150GB RAM. All calculations were made on a single processor.

5.6.1 Network Parameters

The networks that we used are made up of 200 nodes, each with the same processing capacity P. We have tried the two values of P listed in Table 5.2. We generate the links from

	P_i	$C_{i,j}$
Bandwidth-limited	4.00	1.14
Node-limited	1.71	2.66
Both	1.71	1.14

Table 5.2: Network limit parameters

a uniform random distribution with parameter π, the probability that there is a directed link between any given pair of nodes. The higher π the more connected the network. All links have the same capacity C. We have tried two values of C as listed in Table 5.2.

When we create the network, we define the set of functions $\mathcal{F}$ that all connections may use, say f_1, f_2, ... f_F. To simplify, we put a copy of each one of these F functions at random on n nodes. In the following, we will use $F = 10$ for all networks and consider two cases $n = 5$ and $n = 20$.

For a given network, when a connection arrives, it requires $K \leq F$ functions. The actual functions needed by the connection are chosen randomly from the F available functions. Each connection requires $b = 0.1$ unit of bandwidth on each link it uses, $p = 0.05$ processing units when it goes through a node and $q = 0.1$ processing units in a node whenever a function it needs is executed on that node,

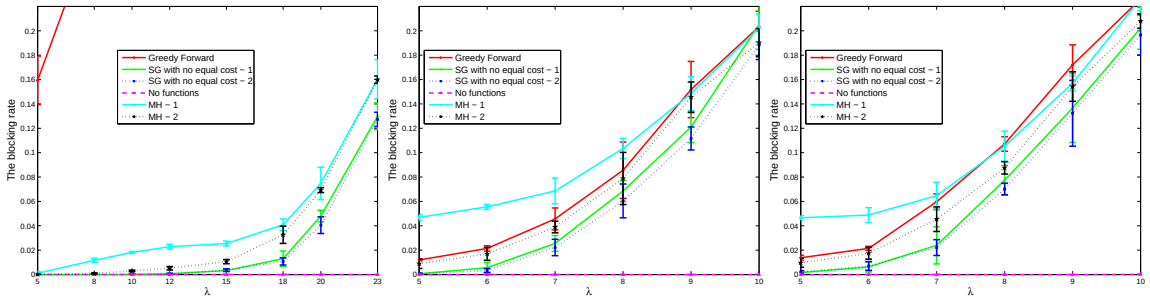
We adjust the link and node parameters to generate three kinds of networks. In *bandwidth-limited* networks, the node capacities are made relatively large so that the link capacities are the more constraining. Conversely, we define *node-limited* networks as networks where the node constraints are dominant. The final case is where both constraints are equally important. The values are summarized in Table 5.2.

5.6.2 Simulator

The most important measure of the quality of a routing algorithm is the connection blocking probability. This is measured from an event-driven stochastic simulation of a set of typical networks. Our simulation is programmed in Java. In the following, all time units are in seconds unless otherwise noted.

We first generate a network for a given value of $\pi = 0.032$. We have used this network in almost all the figures since the results and trends for networks with other values of π were similar to those presented here. The only exceptions are in Figures 5.11, where we have simulated networks for different values of π to see the impact of connectivity on our results, and Figures 5.9 and 5.10, where we use a lower value of π in order to get at least a small amount of blocking on these figures.

Connection arrivals are generated using a Poisson process of parameter λ chosen such that the overall blocking probability is below 0.2. The connection holding time is generated



(a) Bandwidth limited (b) Node limited (c) Both limited

Figure 5.4: Blocking vs λ, $n = 5$, $\pi = 0.032$, $K = 5$

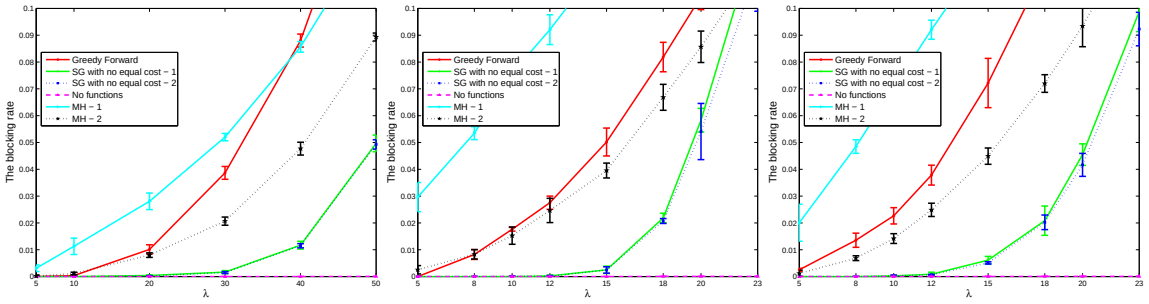
from an exponential process with average $\tau = 10$. Once a connection is generated, its source and destination are generated uniformly at random from all the possible (o, d) pairs. The set of K functions that the connection uses is selected randomly from $\mathcal{F}$. We generate 10 runs of length $T = 2000$ seconds each. The results are averaged out over the runs and we compute the 95% confidence interval for the results.

The simulation maintains the current state of the network as a list of ongoing connections including the location of functions that they use and their routing path. When a new connection arrives, we first check if any connection(s) had finished during the previous inter-arrival time, remove them from the list and then release the resources that they were holding. After that, we run the routing algorithm under consideration to try to find a path with enough resources. If possible, this connection is added to the list of ongoing connections and the available resources are updated. Different types of blocking conditions are updated whenever they occur, depending on the algorithm we are evaluating.

5.6.3 Effect of Load

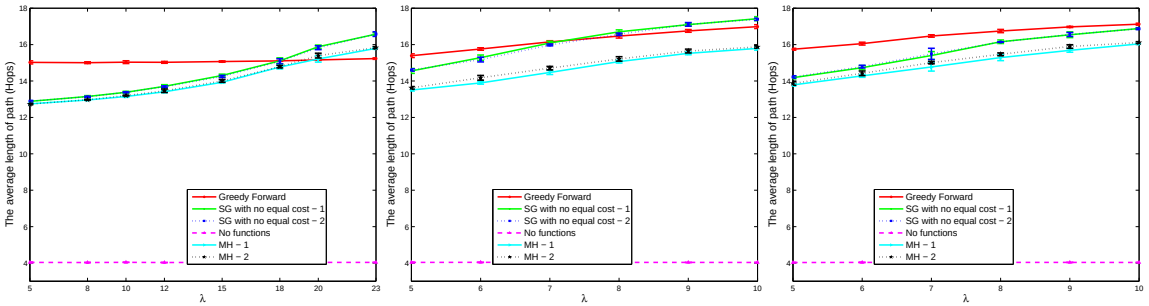
The first set of results shows the effect of λ on the blocking probability for the approximate algorithms described in Section 5.5 as well as the case where there is no function, where we compute a simple capacitated shortest path between o and d. The labels for the curves are SG, for Sub-Gradient, when we use the unequal costs of (5.11) and MH, for Min-Hop, when we use unit costs. In both cases, the label 1 or 2 indicates whether we did one or two iterations of the subgradient procedure. We present in Figure 5.4 the case for $n = 5$ and in Figure 5.5, for $n = 20$ for the three kinds of networks.

First, we see that the Greedy Forward algorithm produces significantly higher blocking than the subgradient-based algorithms. It is also remarkable that running the second iteration of the Lagrangian relaxation does not improve the performance of any one of the cases if we use the unequal costs to start with. There is, however, a significant improvement if we do the first iteration with unit costs. This finds a min-hop path, which is not very good,



(a) Bandwidth limited (b) Node limited (c) Both limited

Figure 5.5: Blocking vs λ, $n = 20$, $\pi = 0.032$, $K = 5$

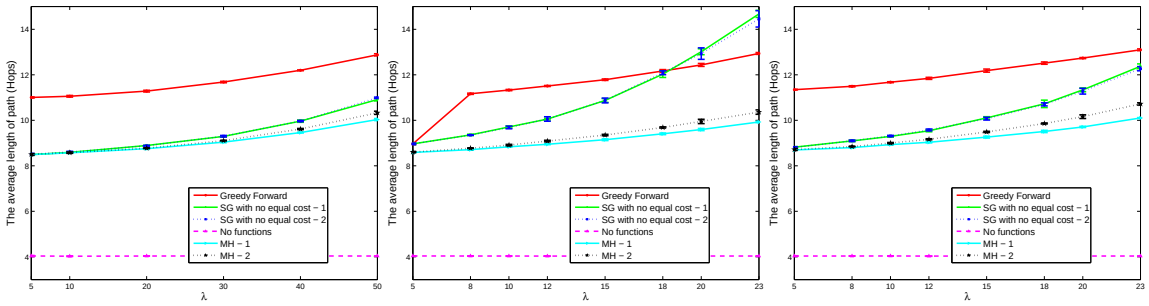


(a) Bandwidth limited (b) Node limited (c) Both limited

Figure 5.6: Average path length vs λ, $n = 5$, $\pi = 0.032$, $K = 5$

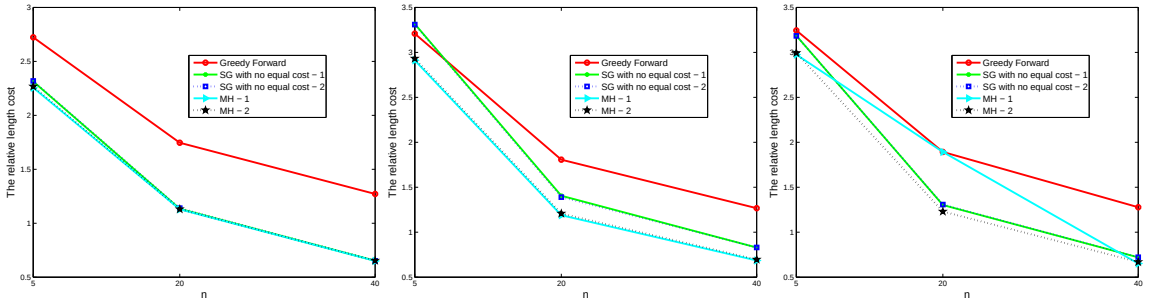
but the second iteration reduces the blocking because the updated link lengths computed from (5.9–5.10) now take into account the link and node loads.

We can see from these figures that the number n of copies of a function has a significant effect on the network blocking. If we choose a 5% level of blocking, we see that having 20 copies instead of 5 can more than double the amount of traffic that can be carried in the three kinds of networks using the two shortest path algorithms. This is directly tied to the length of paths produced by the algorithms as can be seen from Figures 5.6 and 5.7, where we have plotted the relative increment ρ in path length as a function of traffic. This is defined as $\rho = (L_i - L_0)/L_0$ where L_i is the average path length from algorithm i and L_0 is for the case where there is no function requirement. Note that here, the path length is the actual number of hops of the path, not the length as defined in the objective function. We can see that the algorithms finds shorter paths in the case $n = 20$. The reason is that when n is larger, we generate the networks with more function nodes so that the segments between two consecutive functions would tend to be shorter.



(a) Bandwidth limited (b) Node limited (c) Both limited

Figure 5.7: Average path length vs λ, $n = 20$, $\pi = 0.032$, $K = 5$



(a) Bandwidth limited (b) Node limited (c) Both limited

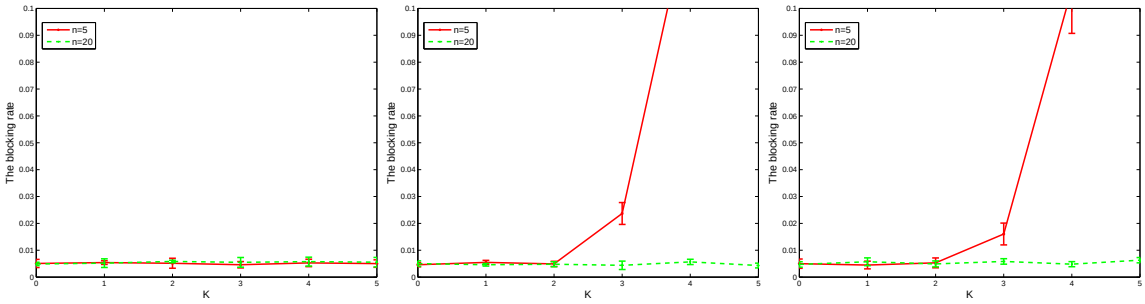
Figure 5.8: Relative path length vs n, $\lambda = 10$, $\pi = 0.032$, $K = 5$

5.6.4 Effect of Costs

We can also see on Figures 5.4 and 5.5 the effect of the costs $a_{i,j,k}$ on the blocking. Using $a_{i,j,k} = 1$, i.e., a min-hop solution, produces a higher blocking than that produces when using the values of (5.11). The results of Figures 5.6 and 5.7 show that even though the paths produced using the non uniform costs have more hops than the minimum possible, and sometimes by a significant amount, avoiding congested links is profitable overall in reducing the blocking.

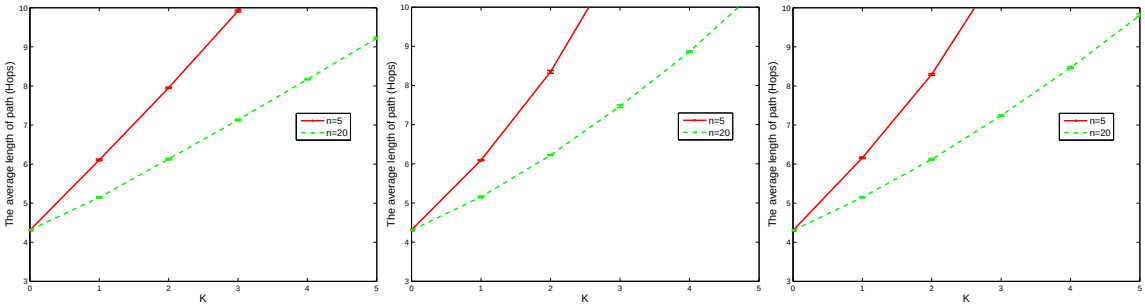
5.6.5 Effect of the Number of Copies

We investigate next the effect of n, the number of copies of a function, by plotting on Figure 5.8 the average relative path length ρ as a function of n. It is quite clear that a large value of n will significantly reduce the path length, typically from a factor larger than 300% for $n = 5$ down to about 50% for $n = 40$.



(a) Bandwidth limited (b) Node limited (c) Both limited

Figure 5.9: Blocking of "SG with unequal cost - 1" vs K, $\lambda = 10$, $\pi = 0.027$



(a) Bandwidth limited (b) Node limited (c) Both limited

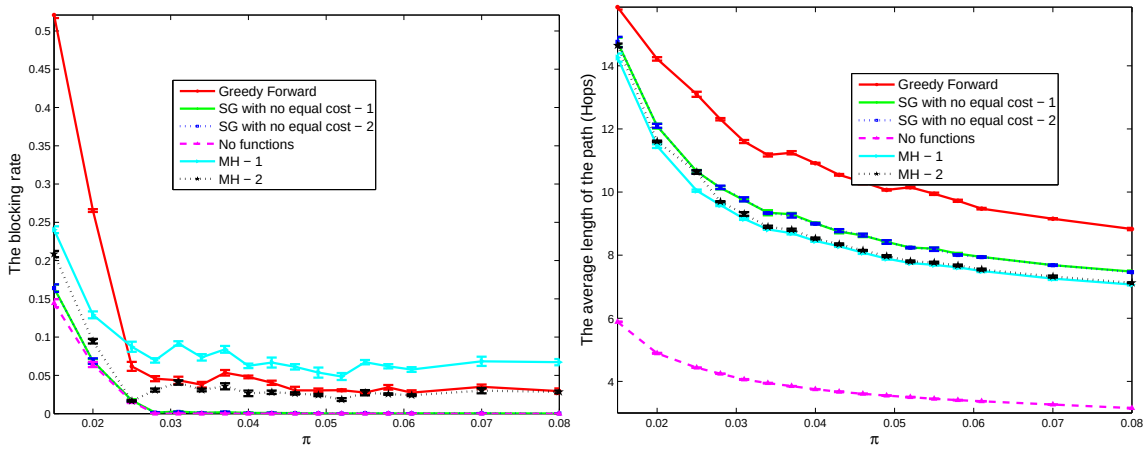
Figure 5.10: Average path length of "SG with unequal cost - 1" vs K, $\lambda = 10$, $\pi = 0.027$

5.6.6 Effect of K

We can also see from Figures 5.9 and 5.10 the effect that the number of functions K needed by each connection has on the blocking and the path length of the best algorithm, namely, "SG with unequal cost - 1". This is shown for the costs given by (5.11). We can see that in some cases, increasing the number of functions can have a large impact on blocking, as in 5.9(b) and (c). This impact can be reduced significantly by increasing the number of copies, as shown in the figures. In all cases, however, there is a significant increase in the number of hops, as expected.

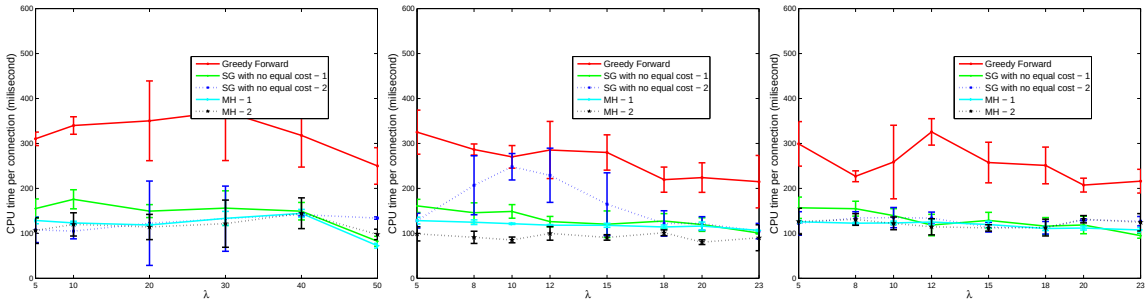
5.6.7 Effect of Connectivity

We have seen that increasing the number of function copies improves the blocking by reducing the path length needed by the connections. We can also do this also by increasing the connectivity of the network. We look at this effect, as measured by π, on the performance. First, we see on Figure 5.11.(a) that increasing the number of links has a very significant effect on the blocking. If we double the number of links, the subgradient algorithm yields



(a) Blocking rate vs π (b) Path length vs π

Figure 5.11: Impact of π, $n = 20$, $C = 2$, $P = 3$, $\lambda = 25$, $K = 5$



(a) Bandwidth limited (b) Node limited (c) Both limited

Figure 5.12: Computation time vs λ, $n = 20$, $\pi = 0.032$, $K = 5$

the same blocking as if there were no function requirements. This is due to a corresponding decrease in the average path length, as seen on Figure 5.11.(b).

5.6.8 Computation Time

As discussed above, a path needs to be computed quite fast if we would like to be able to serve the expected traffic in a large network. We can see in Figure 5.12 that this requirement is well met by the subgradient algorithms, which have computation times under 300 ms. It is also interesting that the Greedy Forward has the largest time, another reason not to use it.

5.7 Conclusions

In this chapter, we studied the routing problem via Functions in NFV Infrastructure. We have proposed two classes of algorithms based on the construction of an expanded network to take into account the function constraints and the Lagrangian decomposition [12] of the resulting problem to compute approximate solutions. We have then used simulation to provide extensive results on their performance. These results provide us with some interesting insights both about the algorithms and the network performance. First, is is quite clear that the Greedy Forward algorithm is not good: it provides higher blocking and takes more time to compute than the subgradient algorithms. The choice of the cost functions is important to obtain a good blocking as shown in our results. Also, doing a second sub-gradient iteration does not improve the results in any significant way. Whether some other algorithms would do better remains an open question.

As far as network performance is concerned, the results of Figures 5.6 and 5.7 clearly show that introducing the function requirements can degrade significantly the network performance and that this is closely tied to the length of the paths produced by the shortest path algorithms. We have also seen that we can improve network performance either by having more copies of the functions or by increasing the number of links, up to a point where the shortest path algorithms behave as if there were no function requirements.

Chapter 6

Conclusion

Contents

The evolution of virtualization technology has a tremendous impact in the design of future networks. In particular, software-defined networking and network function virtualization offer us new ways to design, build and operate telecommunication systems. They aim to advance a software-based approach to networking for more scalable, agile, and innovative networks that can better align and support the overall needs of the economic environment. Hence, it is important to understand and improve the performance of NFV deployment in practice. This dissertation has contributed to both the understanding of the right level of abstraction about the resource allocation problem in NFV and the development of VNFs placement with multiple objectives. In this chapter, we conclude the dissertation by summarizing the key contributions and discuss potential directions for future work.

6.1 Summary of Contributions

The main goal of our work is to model and analyze important design issues related to the NFV deployment. This dissertation makes two major contributions for the resource allocation problem in an NFV Infrastructure.

Our first contribution is to propose a formal model for two important problems in NFV Infrastructures. In the one hand, we have modeled the resource management and placement for NFV while considering the order of VNFs in a service demand and the

resource constraints for nodes and links. On the other hand, we have modeled and solved the routing problem.

In our first contribution, we investigate two models for a joint problem of NFV placement and routing. Firstly, we focus on the price model for deploying VNFs on an NFV Infrastructure. We aggregate both the OFFLINE and ONLINE cases and formulate the system as a quadratic problem with multiple objectives. This model considers many possible constraints that may be encountered in a real environment, such as resource constraints, arrival and processing time considerations. Then, we develop another efficient model to solve this problem exactly and find optimal allocations for required VNFs. The proposed model is straightforward and able to apply in a large NFV dynamic system. In the second problem, we use an expanded network to propose a mathematical formulation as an integer minimum-cost flow model. The problem is to design an algorithm that will find such a path very quickly for each request in an NFV-based large network. This model took into account the precedence constraint of VNFs by using the expanded network.

An important consequence of the models is that they are generic, able to handle a large and diverse set of key parameters and can easily be customized to be applied to other optimization problems in NFV.

The second major contribution of the dissertation is the performance evaluation for NFV placement and routing problem, where we consider many constraints and objectives. Our models can handle many variables, constraints and multiple objectives. Therefore, it is hard to compute the exact solutions to that problem in a reasonable time. We then provide several fast approximate algorithms to solve large instances. We first simplify the model by dividing it into two stages and implement them one by one. Therefore, the result of the first stage will become the input for the last stage. Based on the optimal model and approximate solutions, we offered evaluations to answer some important questions in NFV system, such as the distribution of VNFs or the overhead introduced by a solution based on the virtualization of functions. Then, we investigate an interesting method to generate some strong flow cover inequalities to improve the performance of LP solving. Thanks to the addition of these inequalities, the model becomes easier and can be solved for large instances. Finally, we construct heuristic algorithms based on the Lagrangian relaxation of the flow problem. The most important quality of a routing algorithm is the connection blocking probability. This is measured from an event-driven stochastic simulation of a set of typical networks. We have then used simulation to provide extensive results on their performance.

In summary, the key innovations of solution are:

- In the resource management and placement problem, we consider many possible constraints that may be encountered in a real environment, such as the resource con-

straints on nodes and links, delay considerations, and the order of VNFs in a chaining. The key idea underlying our solution is to model the price problem in NFV Infrastructure as a quadratic problem with multiple objectives. We investigate both OFFLINE case and ONLINE case. In addition, we also consider to re-optimize the existing flows in the dynamic system. We proposed three heuristic algorithms and evaluated their performance in many scenarios. We showed the distribution of VNF on the multi-tier network topology and suggested the best locations that should host VNFs. Furthermore, we also provide a guideline for the percentage of traffic to be migrated in order to achieve the best gain in an NFV system.

- Through these first results and their limitations, we develop an extended model to provide optimal solutions for the same problem. In this model, we focus on minimizing the resource utilization of an NFV dynamic system and hence maximizing the number of accepted demands. We also propose a method to generate some strong inequalities to improve the performance of Linear Programing solving. Via implementing simulation, we show the significant improvement when adding flow covers to Linear Programing. Moreover, the results also illustrate that our model is efficient and able to handle large dynamic systems.

- Unlike the resource management and placement problem, in the routing problem, we do not focus on placing VNF to virtual nodes. We assume that we know a set of nodes that can run each VNF (called *function nodes*). The purpose is to find quickly a path whenever a request arrives. This is an important and difficult routing problem in any large network system. We propose an expanded network to meet the constraints on Service Chain and proposed a mathematical formulation of the routing problem as an integer minimum-cost flow model. We then provided two classes of algorithms based on the construction of an expanded network to take into account the function constraints and the Lagrangian decomposition to compute approximate solutions. In order to evaluate their performance, we implement an event-driven stochastic simulation of a set of typical networks. These results provide us with some interesting insights both about the algorithms and the network performance. As a result, it is important to point out that the placement of VNFs plays a significant role since this will impact the path length needed to meet the service requirements.

6.2 Future Research Directions

Our work can be extended in various directions. In this dissertation, we have demonstrated the promising results and effectiveness of our solutions. However, extensions are needed

towards the goal of achieving a comprehensive performance model for NFV placement and routing problem in practical scenarios.

The relative impact of customer requests with different requirements should be considered and taken into account to adjust effectively our routing algorithms.

Another interesting point may concern the congestion issue that can arise in such an environment. Especially in a large network, the need of customers is continuously changing and the amount of traffic crossing the network will be quite bursty. Hence, the congestion control should also been considered carefuly in an NFV deployment.

Moreover, the work can extend to consider some possible constrains that may be encountered in a real environment, such as server/VNF affinity, delay considerations, new characters in some defined user-cases, the attribute of demands distribution in the network. In order to make an NFV deployment practical, we need to integrate many constraints to an NFV model but we should also verify that the model is simple enough to be deployed.

Finally, it is important to develop efficient tools to support the design of such novel networks. This is a new area that will still require in-depth research on optimization algorithms. These tools will help us to evaluate the performance of solutions before experimenting these solutions for instance in real testbeds.

Annexe A

Résumé de la Thèse

The Université Pierre et Marie Curie (UPMC) requires all PhD theses written in English to include a summary in French. The next pages correspond to this summary.

A.1 Introduction

La virtualisation des réseaux représente le concept émergent le plus prometteur de ces dernières années car il permet de prendre en considération l'évolution de la demande des utilisateurs qui souhaitent une flexibilité toujours accrue. En particulier, SDN (Software Defined Networks) et NFV (Network Function Virtualization) proposent des approches de virtualisation complémentaires visant à réaliser cette vision.

Nous nous intéressons à la solution NFV afin d'en mesurer les forces et faiblesses en terme de consommation de ressources et de coût de déploiement. Nous étudions plus particulièrement à l'optimisation de l'allocation de ressources (processeur, mémoire, réseau) dans une infrastructure de virtualisation de la fonction réseau. En effet, un premier défi dans la réalisation de NFV est l'attribution des VNFs dans l'infrastructure virtualisée.

Nous commençons ce chapitre en présentant le contexte et la motivation de ce travail. Nous montrons ensuite que la modélisation d'un placement NFV ainsi que le routage suivant de multiples objectifs constitue un problème complexe (Section 1.2). Nous introduisons ensuite nos solutions au problème de l'allocation de ressources dans NFV (Sections 1.3-1.4). Enfin, nous résumons les contributions et présentons la structure de la thèse (Sections 1.5-1.6).

A.1.1 Contexte de Recherche

De nos jours, les réseau de télécommunications sont constitués d'une variété croissante d'équipements matériels propriétaires conçus à des fins spécifiques tel que des routeurs, des pare-feux ou des équilibreurs de charge. Cela entraîne un effort considérable en matière d'intégration, d'exploitation et de maintenance d'une infrastructure de communication. Par exemple, pour lancer un nouveau service réseau, cela requiert souvent un nouvel équipement et la recherche de l'espace et de la puissance pour l'installer. Cette approche accroît également les coûts en énergie, les investissements en capital, et requiert une importante main-d'oeuvre qualifiée. En outre, les appareils deviennent rapidement obsolètes et ne répondent souvent plus aux contraintes auxquelles ils sont confrontés en raison des évolutions technologiques fréquentes. De fait, l'utilisation de la virtualisation promet d'apporter une solution à ces problèmes.

La technologie NFV permet une flexibilité significative en remplaçant le matériel actuel par des logiciels fonctionnant sur des serveurs standards (ou presque standards). NFV est une nouvelle approche visant à concevoir, déployer et gérer les services réseau en migrant les fonctions réseau à partir d'équipements de réseau physique, appelés appareils Dédiés, en les déplaçant vers des Serveurs Virtuels pouvant fonctionner sur des processeurs à usage général ou des machines virtuelles. La flexibilité, l'agilité voire les économies de coûts opérationnels

qui pourraient être rendus possible avec NFV, ouvre la voie à de nouvelles innovations, paradigmes de conception et permet d'envisager de nouvelles architectures de réseau.

En particulier, l'application de NFV pourrait apporter plusieurs avantages aux opérateurs de réseau, contribuant à des changements importants dans l'industrie des télécommunications comme suit :

- La réduction des coûts constitue le principal moteur de NFV. NVF vise à réduire les dépenses en immobilisations de l'opérateur (CAPEX) et les dépenses opérationnelles (OPEX) grâce à des coûts d'équipement réduits et une consommation d'énergie contenue.

- Le cycle de vie des services est plus court : de nouveaux services de réseau peuvent être déployés plus rapidement, en fonction de la demande et des besoins, offrant des avantages aux utilisateurs finaux ainsi qu'aux fournisseurs de services. Par conséquent, il réduit potentiellement le délai de mise sur le marché pour le déploiement de nouveaux services réseau.

- Échelle et élasticité : NFV permet des changements de capacité en offrant un moyen d'adapter dynamiquement les ressources utilisées par les fonctions réseaux virtualisées à l'évolution de la demande.

- Efficacité opérationnelle et agilité : avec un matériel courant hébergeant différents VNFs, les tâches associées à l'exploitation de l'entreprise peuvent être centralisées.

De fait, NFV présente de nombreux avantages mais fait également apparaître plusieurs défis. Nous pouvons par exemple citer la performance de NFV, l'allocation des ressources, la gestion et l'orchestration de NFV et la question de la sécurité.

- Performance de NFV : NFV consiste à exécuter des fonctions réseau sur des serveurs standards de l'industrie. Cependant, il est difficile d'offrir des performances de réseau garanties pour des appareils virtuels. Cela soulève la question de savoir si les fonctions réseau exécutées sur un serveur standard de l'industrie atteindrait une performance comparable à celles fonctionnant sur du matériel spécialisé [1], [2].

- Gestion et orchestration de NFV : le déploiement de NFV stresse les systèmes de gestion actuels. Cela nécessite des changements importants non seulement afin de fournir des solutions de réseau et de services, mais aussi afin d'exploiter le dynamisme et la flexibilité rendu possible par NFV [3], [4]. Le projet OpenMANO de l'Institut européen des normes de télécommunications (ETSI) a fourni un cadre MANO [5] pour la fourniture des VNFs, et les opérations connexes telles que la configuration des VNFs et l'infrastructure ou ces fonctions sont exécutées. Dans un effort similaire,

Cloud4NFV [6], [7] a proposé une gestion de plate-forme pour les VNFs, basée sur la spécification architecturale ETSI.

- Sécurité, confidentialité et confiance : la sécurité est toujours l'un des problèmes les plus difficiles dans tous les systèmes de réseau, spécialement lorsque la fonction doit être virtualisée et déployée dans des nuages tierce-partie. En raison d'informations personnellement identifiables qui peuvent être transférées au nuage, les fournisseurs de services doivent fournir un mécanisme de sécurité dans NFVI. ETSI a créé un groupe d'experts en sécurité pour se concentrer sur cette préoccupation. Ils ont fourni un cadre pour la sécurité et la confiance qui est unique au déploiement, à l'architecture et à l'opération [8]. Toutefois, elle ne se compose pas d'exigences prescriptives ou de détails spécifiques d'implémentation.

- Affectation des ressources : Idéalement, les opérateurs de réseau devraient placer les VNFs où elles vont être utilisées le plus efficacement et au moindre coût. Bien que la virtualisation des fonctions réseau sont simples, il existe un certain nombre de fonctions de réseau qui ont des exigences strictes de délai. Par exemple, une solution NFV peut prendre un chemin plus long que le chemin probable le plus court suivi par un système non NFV où les fonctions réseau sont placées sur le chemin direct entre deux points. Par conséquent, le placement des machines virtuelles (VM) qui portent des VNFs est cruciale pour la performance des prestations de service offertes. Ce sujet devient un des défis importants dans le système NFV. Cette dissertation se concentre principalement sur ce problème. L'objectif de cette dissertation est de mieux comprendre le juste niveau d'abstraction et d'améliorer les performances d'un tel système en considérant des objectifs multiples.

NFV est un sujet d'actualité. La croissance rapide de la technologie de virtualisation et les avantages qu'offrent NFV ont donné lieu au déploiement d'applications basées sur NFV. Par conséquent, l'allocation efficace des ressources est un problème central dans l'infrastructure NFV. L'importance et les défis des problèmes mentionnés ci-dessus seront discutés dans les deux prochaines sections.

A.1.2 Les Challenges de L'allocation de Ressource et du Routage dans une NFV

Bien qu'il y ait beaucoup d'attention des organisations de normalisation, des projets de recherche universitaires et industriels et des fournisseurs d'équipements, l'allocation de ressources et le problème de routage demeure un sujet ouvert. La mise en place de VNFs sur les infrastructures NFV nécessite un algorithme efficace qui garantit de multiples contraintes dans un système complexe. Ce problème est un défi pour plusieurs raisons fondamentales :

- Ce problème est lié à l'intégration d'un réseau virtuel (VNE) [9], [10] et donc des approches s'en inspirant peuvent être appliquées. Cependant, toute formulation devrait être en mesure de prendre les exigences en matière de chaînage de fonction et / ou des prérequis de préséance en vue d'éviter l'encombrement du réseau. Cette caractéristique du système NFV rend la formulation du problème plus complexe et difficile à traiter.

- L'un des avantages de NFV est sa capacité à échelonner les ressources dynamiquement. En effet, l'automatisation des mécanismes d'auto-allocation qui permettent au réseau de gérer dynamiquement les ressources sont essentielles au succès de NFV. Cependant, mieux réagir aux demandes dynamiques mais aussi aux changements des exigences de service, dépend de nombreuses variables telles que l'heure d'arrivée et le temps de traitement de chaque demande, le stockage et l'énergie limités, les besoins en ressources des VNFs. D'autre part, nous devons simplifier les hypothèses afin de nus permettre de résoudre un modèle mathématique, où ces hypothèses doivent être réalistes dans la pratique. Pour ces raisons, la modélisation de ce problème est difficile.

- Les serveurs utilisés pour héberger des VNFs ont une quantité finie de mémoire, de calcul et de capacité de stockage. Et, dans la réalité, ces serveurs peuvent être répartis sur plusieurs domaines, la capacité de liaison inter-domaine est également finie. Par conséquent, pour atteindre les économies d'échelle attendues de NFV, les ressources physiques devraient être efficacement gérées. Ce problème pourrait introduire une complexité supplémentaire en raison d'une inter-dépendance entre les fonctions du réseau comme dans le cas du chaînage des fonctions et de nouvelles contraintes potentielles concernant des aspects complémentaires comme la sécurité [11].

- Dans un système NFV, la vaste gamme de services disponibles et le modèle de service à la demande créent des conditions de trafic dynamiques qui nécessitent une flexibilité et une automatisation de la plate-forme réseau pour rediriger le trafic en fonction des conditions du réseau. L'un des défis, que ce modèle de provisionnement dynamique soulève, est la difficulté à développer un algorithme efficace pour répondre à toutes ces exigences et en particulier, le passage par les nœuds de fonction dans l'ordre prescrit.

Or, au début de cette thèse, il n'existait aucun modèle qui considère les objectifs multiples et les contraintes pour le placement et routage de NFV. C'est l'ambition de ce travail d'y apporter des solutions crédibles et efficaces.

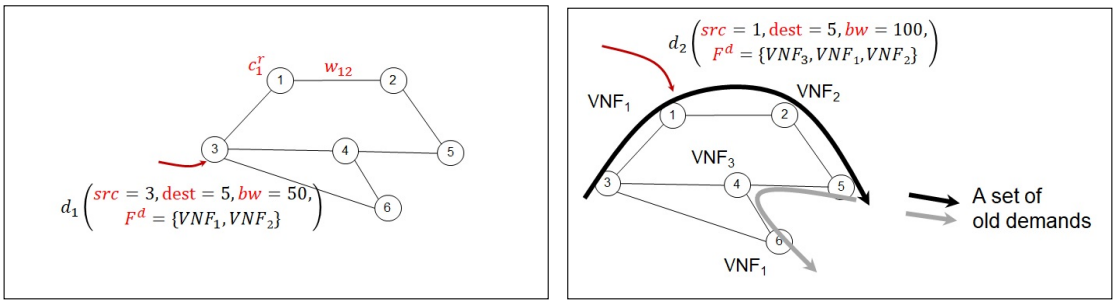
A.1.3 Objectif de la Thèse

L'objectif principal de cette thèse est de modéliser un problème de placement et de routage NFV tout en tenant compte de l'ordre d'exécution des VNFs dans une demande de service à l'égard de contraintes de ressources pour les nœuds et les liens. Plus précisément, nos objectifs sont les suivants :

- Apporter une meilleure compréhension du haut niveau d'abstraction du système NFV. La solution nous permet de mesurer l'impact de l'emplacement des VNFs et à déterminer comment améliorer les performances du système NFV en positionnant à propos les VNFs.
- Développer une solution pour l'allocation des ressources dans NFV avec des objectifs multiples tels que minimiser le coût des ressources, maximiser le taux d'acceptation, minimiser les coûts de pénalité pour migrer les VNFs. La solution nous assistera dans le choix des meilleurs emplacements pour poser des VNFs qui optimiseront les performances du système. De plus, elle fournit des lignes directrices pour le pourcentage de trafic à migrer afin d'atteindre le meilleur gain et pour analyser certaines implications architecturales.
- Développer une solution pour le problème de routage dans NFVI. Le problème est de concevoir un algorithme qui trouvera un chemin très rapidement pour chaque demande dans un grand réseau. Le problème du délai est dû au fait que le taux d'arrivée des demandes peut être important et la décision doit être prise assez rapidement pour qu'il n'y ait pas d'accumulation de demandes en attente d'un chemin de connexion.

Pour atteindre ces objectifs, nous procéderons comme suit :

1. Développer une formulation mathématique pour la gestion des ressources et les probabilités de placement, en considérant l'ordre des VNFs dans une chaîne ainsi que les contraintes de ressources pour les nœuds et les liens (Chapitre 3).
2. Proposer un modèle étendu pour améliorer les performances afin de proposer une solution dans un scénario dynamique avec de grandes instances. Le modèle prend en compte la décision des fournisseurs de services de répondre à une demande ou pas dans un système NFV dynamique (Chapitre 4).
3. Développer une formulation mathématique du problème de routage en tant que nombre entier minimum - modèle des coûts de flux avec des contraintes latérales sur le réseau étendu (Chapitre 5).
4. Évaluer les solutions analytiques sous divers scénarios, avec un ensemble de données aléatoires mais aussi un ensemble de données réelles (Chapitre 3-5).



(a) OFFLINE Case (b) ONLINE Case

FIGURE A.1 : Gestion des Ressources et Placement dans une NFV

A.1.4 Aperçu des Solutions

Nous développons des solutions analytiques efficaces visant à comprendre et améliorer la performance d'une solution au problème de répartition des ressources dans NFV. Cette section décrit nos solutions au problème ci-dessus avec les étapes suivantes.

A.1.4.1 Gestion des Ressources et Placement dans une NFV

Nous commencons par examiner le problème de la gestion des ressources et du placement dans NFV. Nous étudions un système NFV où une fonction réseau est offerte en tant que service. Le système est constitué de trois composants, un ensemble de VNFs, un ensemble de demandes de clients et un réseau virtuel qui fournit des ressources aux VNFs. La VNF est une application logicielle d'une fonction de réseau (NF) déployée sur un NFVI. Chaque demande d'accès à un service réseau (NS) nécessite des fonctionnalités spéciales. Elle peut être composée de VNFs générales afin de supporter une connectivité réseau avancée, par ex. l'inspection profonde des paquets, pare-feux, et équilibreurs de charge. Dans une NS, les VNF nécessiteront des ressources diverses telles que le CPU, la mémoire, le stockage, la bande passante, etc. Il est reconnu que les solutions virtualisées sont très coûteuses en consommation de ressources. Les ressources sont réparties partout sur le NFVI. Une demande du client se matérialise en tant que flux de paquets d'une source vers une destination visitant un ensemble de nœuds où les VNFs requises sont déployées.

Comment partager efficacement des ressources pour toutes les demandes avec des contraintes de retard ou de capacité est un défi important pour les fournisseurs de NFV. Il est probable que la solution sera un compromis entre agilité et efficacité. Par exemple, une solution NFV peut prendre un chemin plus long que le chemin le plus court probable suivi par un système non-NFV. Toutefois, il est toujours acceptable si le délai se situe dans une plage acceptable.

Nous développons une solution complète traitant des scénarios statique et dynamique, lorsque nous prenons en compte de l'ordre d'exécution des VNFs dans une demande de service en matière de contraintes de ressources pour les nœuds et les liens.

La figure A.1 décrit un aperçu de la gestion des ressources et du problème de placement dans NFV. Nous aborderons le cas hors ligne et le cas en ligne. La figure A.1 (a) illustre un scénario hors ligne ou nous avons un réseau avec 6 nœuds et 7 liens. Nous devons fournir les ressources pour une demande client en demandant d'envoyer des paquets du nœud 3 (sa source) au nœud 5 (sa destination) et passant par la fonction VNF_1 et VNF_2 dans le bon ordre. Les ressources disponibles, y compris le traitement des ressources sur les nœuds (c_1^r) et les ressources de bande passante sur les liens (w_{12}) sont données. La figure 1.1 (b) montre un scénario en ligne où un réseau opérationnel répond à un ensemble de demandes des clients (appelées anciennes exigences, à savoir les courbes noire et grise sur la figure). À ce stade, un ensemble de VNFs est déjà déployé (par exemple, VNF_1 sur le nœud 1, VNF_2 sur le nœud 2 de la demande noire et VNF_3 sur le nœud 4, VNF_1 sur le nœud 6 de la demande grise) et les chemins de routage pour les anciennes exigences sont également provisionnés. Le problème est d'envisager de nouvelles exigences (demande d_2 dans l'exemple) et l'opérateur de réseau doit fournir les VNFs requises et les chemins de routage pour ceux-ci. Les objectifs du problème sont de maximiser le nombre de demandes servies et de minimiser les coût des ressources système.

Nous considérons un intervalle τ (selon la configuration du système) où nous mettons à jour le statut de toutes les demandes s'exécutant dans le système. Une fois l'acheminement terminé, le système libère les ressources allouées à une demande. À l'instant t, nous recueillons de nouvelles demandes D_n qui seront arrivées pendant la période $(t - \tau)$ à t.

Une autre préoccupation dans le problème en ligne est de décider si et comment une série d'anciennes exigences devrait être migré. En effet, après un certain temps, certaines demandes seront terminées et des ressources libérées pour le système. Cela rend les ressources distribuées fragmentées et non optimales. Par exemple, dans la figure 1.1 (b), lorsque la demande grise finit, elle retournera des ressources pour le nœud 4 et le nœud 6. Par conséquent, il est préférable de ré-optimiser les flux noirs pour passer à une nouvelle position (nœud 3 $\rightarrow$ nœud 4 $\rightarrow$ nœud 5, et mettre VNF_1 sur le nœud 4, VNF_2 sur le nœud 5).

Nous avons formulé le problème en tant que programme quadratique pour l'emplacement optimal du VNF et un routage optimal. Notre modèle est générique, capable de gérer une grande variété de paramètres clés et peut facilement être personnalisé. Une avancée majeure de notre évaluation est qu'avec notre modèle, nous prenons en compte l'interconnectivité entre les VNFs, les exigences de services et les coûts de NFV (y compris le coût des ressources et le coût de la circulation). Nous enquêterons sur le cas hors ligne et le cas en ligne et proposons des algorithmes heuristiques pour résoudre le problème avec de grandes instances. Nous calculons également les coûts encourus par un réseau s'appuyant sur NFV pour calculer les coûts généraux introduits par une solution basée sur la virtualisation des fonctions. En outre, en utilisant la distribution des VNFs sur une topologie de

réseau multi-niveaux, nous analyserons l'impact des emplacements des VNFs et suggérons que les fonctions du réseau soient situées sur la bordure du réseau plutôt que dans le coeur. Enfin, nous fournissons une technique de migration simple qui pourra gérer efficacement la situation dynamique produite par un flux continu d'arrivée de demandes.

Dans le Chapitre 3, nous présentons une description détaillée du modèle mathématique pour la gestion des ressources et le problème du placement. Nous décrivons également les résultats de simulation obtenus en validant les algorithmes et en analysant l'impact des paramètres d'entrée sur les deux ensembles de données aléatoires et réelles.

A.1.4.2 Utilisation des Ressources dans un Système Dynamique NFV

Dans la première étape, nous avons formulé le problème de placement et de routage de NFV en tant que programme quadratique et avons enquêté sur les cas hors ligne et en ligne. Toutefois, le modèle proposé est assez complexe et la solution optimale est limlitée au cas de problème statique pour de petites instances. Ce modèle est restrictif à appliquer dans un système dynamique NFV. Donc, cela soulève la question de savoir comment trouver la solution optimale dans un système dynamique NFV ? Il est nécessaire de développer un modèle efficace pour le problème du placement et du routage NFV qui pourra être exactement résolu avec de grandes instances et qui sera capable de considérer un scénario dynamique.

Dans cette étape, nous nous concentrons sur la proposition du modèle mathématique pour un problème commun des chemins de placement et de routage des VNFs pour les chaîner afin de minimiser la ressource utilisation d'un système dynamique NFV. Le chemin de routage sera guidé par un nombre de VNFs, dans le but d'exécuter la fonction de service nécessaire dans l'ordre requis. En outre, nous prenons en compte la décision des fournisseurs de services de répondre à une demande ou non ce qui nous permet de pouvoir l'appliquer dans un système dynamique.

Afin d'améliorer les performances pour résoudre le problème LP, nous étudions également une méthode prométeuse pour générer de fortes inégalités dans un espace dimensionnel plus élevé. Les résultats montrent que nous pouvons obtenir une amélioration significative après avoir ajouté des couvertures d'écoulement au modèle. En outre, nous développerons deux algorithmes heuristiques pour trouver une solution réalisable dans un cadre acceptable de temps d'exécution. Via une simulation événementielle, nous évaluerons notre solution sur un réseau à topologie réaliste et comparerons la valeur objective du problème obtenu par les algorithmes proposés. En conséquence, notre modèle est simple et aisé à déployer dans un grand système dynamique. En outre, les contraintes de flux soulevées ajoutés à la LP sont puissants et utilisables efficacement dans d'autres modèles.

Le Chapitre 4 fournit une formulation mathématique détaillée pour le problème de l'utilisation des ressources dans l'infrastructure NFV. Le chapitre propose également des algorithmes et notre simulation pour évaluer la performance des solutions. En outre, nous analysons l'impact des paramètres d'entrée dans différents scénarios sur des ensembles de données réelles.

A.1.4.3 Le Routage par Fonctions dans une NFV

La solution de deux premières étapes fourni une analyse quantitative pour partager efficacement des ressources pour toutes les demandes avec des contraintes de retard ou une capacité limitée dans un système NFV. Elle s'est concentrée sur le placement des VNFs dans le réseau plutôt que de fournir une solution rapide pour chaque demande dans un grand réseau. C'est dû au fait que dans un grand réseau, le taux d'arrivée des demandes peut-être grand et il faut prendre une décision assez rapidement pour qu'il n'y ait pas de demande en attente d'un chemin de connexion. Le problème est de concevoir un algorithme de routage qui trouvera un tel chemin très rapidement pour chaque demande.

Nous considérons un réseau où les demandes de connexion arrivent aléatoirement. Chaque connexion doit utiliser une chaîne de service, c'est-à-dire un ensemble de fonctions donné, spécifique à la connexion, dans un ordre prescrit. Plusieurs de ces fonctions sont placées a priori dans certains sous-ensemble de nœuds, appelés "nœuds de fonction".

Pour chaque demande de connexion avec origine et destination, nous voulons trouver un chemin à travers le réseau qui répondra à toutes les exigences et en particulier, suivra la fonction dans l'ordre prescrit. Si cela n'est pas possible, la connexion est bloquée. La mesure la plus importante de la qualité d'un algorithme de routage est la probabilité de blocage de la connexion. Ceci est mesuré à partir d'une simulation stochastique événementielle d'un ensemble typique de réseaux. La figure A.2 illustre les arrivées de connexion à l'aide d'un processus de Poisson du paramètre λ.

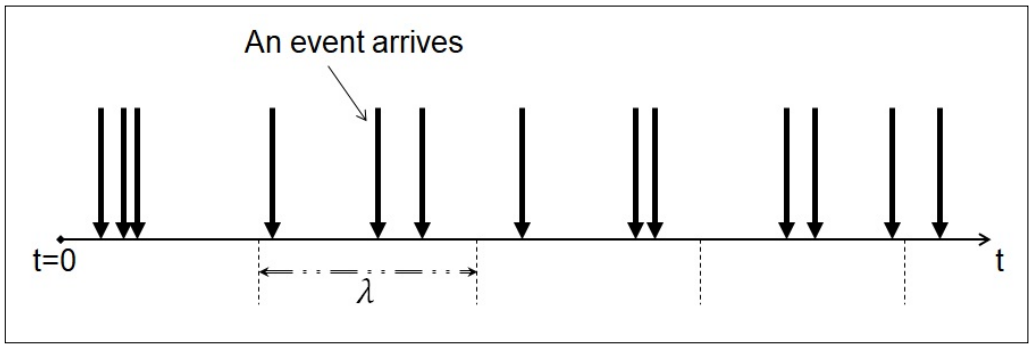


FIGURE A.2 : D'une Simulation Stochastique Événementielle

Afin de répondre rapidement chaque fois qu'une connexion arrive, nous proposons d'utiliser un réseau étendu pour répondre aux contraintes de la chaîne de service et donner une formulation mathématique précise du problème de routage en tant que modèle de flux à coût minimal entier avec des contraintes latérales sur le réseau étendu. Basé sur la relaxation lagrangienne du problème de flux [12], nous proposons plusieurs algorithmes approximatifs rapides pour calculer les solutions au problème. Les résultats montrent que l'introduction des contraintes sur les fonction peut dégrader significativement la performance du réseau et que cela est étroitement lié à la longueur des chemins produit par les algorithmes des chemins les plus courts.

Dans le Chapitre 5, nous décrivons un modèle de système pour le routage via les fonctions dans l'infrastructure NFV et formulons une représentation mathématique de ce problème. Nous fournissons également une simulation dont nous montrons les résultats étendus et l'impact sur leur performance.

A.1.5 Contributions

La formulation mathématique la plus proposée pour l'allocation des ressources dans l'infrastructure NFV a été effectué dans le cas où certaines contraintes sont relachées, telles que les contraintes des liens les contraintes de delai ou l'ordre des VNF dans un chaînage, etc. Des travaux récents ont considéré l'allocation de ressources avec plusieurs types de contraintes et de multiples objectifs. Cependant, il est encore nécessaire de mieux comprendre le bon niveau d'abstraction pour le déploiement réel de NFV. En particulier, nous considérons : 1) Comment et où localiser et chaîner les VNFs ? 2) Comment distribuer des ressources pour exécuter des VNFs ? 3) Comment acheminer efficacement dans le système NFV ? En outre, la complexité des solutions devrait être maîtrisée, en particulier avec la croissance rapide du réseau actuel. Le travail se concentre sur une solution optimale pour le modèle de coût avec des objectifs multiples, y compris la minimisation du coût des ressources sur les nœuds et les liens, en maximisant le taux d'acceptation et en minimisant le coût de pénalité pour la migration VNF. De plus, les contributions importantes de notre travail sont des solutions pour les problèmes particuliers dans le défi d'allocation des ressources dans l'infrastructure NFV. Elles consistent en (i) une solution pour minimiser les coûts de déploiement de NFV tout en considérant des contraintes multiples dans le problème d'allocation de ressources dans NFV ; (ii) une solution optimale pour minimiser l'utilisation des ressources dans un système dynamique NFV ; et (iii) une solution pour trouver rapidement le chemin de routage dans le système NFV dans un grand réseau. Les résultats analytiques qui ont été validés dans de nombreux scénarios, les ensembles de données aléatoires et réelles fournissent une information précise et une évaluation efficace à ce problème. Les principales contributions sont les suivantes :

- Une première contribution de la thèse est la formulation mathématique d'une gestion préventive des ressources et le problème de placement qui tiennent compte de l'ordre des VNFs dans une chaîne et les contraintes de ressources pour les nœuds et les liens. Nous aborderons le problème avec plusieurs objectifs et associons les deux cas hors ligne et en ligne. Le modèle est générique, capable de gérer une grande variété de paramètres clés et peut facilement être personnalisé. Nous évaluons la performance de nos solutions avec un large ensemble de paramètres. Les résultats de l'analyse permettent d'améliorer la performance du déploiement VNF. De plus, la solution fournit une technique de migration simple pour le système dynamique et les principes concernant pourcentage de trafic à migrer pour obtenir le meilleur gain.

- Une deuxième contribution de la dissertation est un modèle étendu, la programmation linéaire, pour fournir des solutions exactes pour le problème de l'utilisation des ressources dans NFV. Le modèle est simple et capable de traiter un grand système dynamique. Le principal point distinctif est que nous fournissons une méthode originale pour engendrer de fortes inégalités améliorant la résolution LP dans un espace dimensionnel plus élevé. En outre, nous proposons deux heuristiques pour calculer la solution dans un temps d'exécution acceptable. Les résultats obtenus nous fournissent l'allocation optimale pour chaque VNF requis concernant l'utilisation des ressources d'un système dynamique NFV.

- Une autre contribution importante de la thèse est la solution pour le problème du routage dans un grand système. Il fournit une réponse rapide chaque fois qu'une demande de client arrive. Nous avons proposé deux classes d'algorithmes basées sur la construction d'un exemple de réseau étendu pour prendre en compte les contraintes de fonction et la décomposition de Lagrange. Afin d'évaluer la performance de la solution, nous utilisons une simulation axée sur un ensemble de réseaux typiques. Les résultats confirment à nouveau que le placement des fonctions joue un rôle important avec un impact sur la longueur du chemin nécessaire afin de répondre aux exigences requises.

A.1.6 Structure du Document

Le reste de la thèse est structuré comme suit. Le Chapitre 2 fournit un état de l'art du cadre NFV en général et l'allocation des ressources dans l'infrastructure NFV en particulier. Le Chapitre 3 décrit les résultats analytiques pour la gestion des ressources et le problème de localisation de NFV et analyse les impacts des paramètres d'entrée sur la performance du placement. Le Chapitre 4 présente une solution optimale pour le problème d'utilisation des ressources d'un système dynamique NFV et fournit une méthode pour générer certaines

inégalités fortes pour la résolution de la LP dans un espace de dimension supérieure. Le Chapitre 5 propose une solution pour le problème de routage dans le système NFV en utilisant un réseau étendu. Le dernier chapitre résume la thèse et discute des orientations pour les recherches futures. La dissertation est organisée pour que chaque chapitre soit relativement autonome. Les Chapitres 3, 4 et 5 fournissent plus de détails sur les idées décrites dans la Section 1.4.

Le lecteur peut lire le chapitre Introduction comprenant les idées principales d'abord pour avoir rapidement une vue générale. Les chapitres restants peuvent être considérés ultérieurement pour compléter la vision globale du travail.

A.2 Conclusion

La vitualisation a un impact important sur l'évolution des architectures et des systèmes de communication. En particulier, la mise en réseau définie par logiciel et la virtualisation des fonctions réseau nous proposent de nouvelles façons de concevoir, construire et exploiter un réseau. Elle vise à proposer une approche logicielle pour des réseaux plus évolutifs, agiles et innovants plus adaptés aux défis actuels. Par conséquent, il est important de comprendre et d'améliorer les performances du déploiement de NFV dans la pratique. Cette thèse a contribué à la compréhension du bon niveau d'abstraction sur le problème de l'allocation des ressources dans NFV ainsi que le développement du placement des VNFs avec de multiples objectifs dans un système NFV. Dans ce chapitre, nous concluons la dissertation en résumant les contributions clés et en discutant des orientations pour les futures recherches.

A.2.1 Sommaire des Contributions

L'objectif principal de notre étude est de modéliser et d'analyser la gestion des ressources dans le déploiement d'une NFV. Cette thèse apporte deux importantes contributions :

La première concerne la gestion des ressources et le placement NFV tout en considérant l'ordre des VNFs lors d'une demande de service et les contraintes de ressources pour les nœuds et les liens. La seconde concerne le problème du routage.

Dans le premier problème, nous étudions deux modèles pour le problème joint du placement et du routage. Tout d'abord, nous nous concentrons sur le modèle de coût pour le déploiement des VNFs sur l'infrastructure NFV. Nous considérons le cas hors ligne et le cas en ligne et formulons les systèmes comme un problème quadratique avec de multiples objectifs. Ce modèle considère l'intégration de nombreuses contraintes possibles qui peuvent être rencontrées dans un environnement réel, tel que les contraintes de ressources, les délais d'arrivée et de traitement. Deuxièmement, nous développons un modèle efficace pour résoudre ce problème précisément et trouver des allocations optimales pour les VNFs requis. Le modèle proposé est simple et peut s'appliquer dans un grand système NFV dynamique.

Dans le second problème, nous utilisons un réseau élargi pour proposer une formulation mathématique en tant que modèle de flux à coût minimal entier. Le problème est de concevoir un algorithme qui trouvera un chemin très rapidement pour chaque requête dans un grand réseau basé sur NFV. Ce modèle prend en compte la contrainte de priorité des VNFs en utilisant l'extension réseau. Une conséquence importante des modèles est qu'ils sont génériques, capables de gérer un ensemble large et diversifié de paramètres clés et peut

facilement être personnalisé comme candidat pour d'autres problèmes d'optimisation dans la NFV.

La deuxième contribution majeure de la thèse est l'évaluation de la performance pour le problème de placement et de routage de NFV, où nous considérons de nombreuses contraintes et objectifs. En raison des modèles théoriques proposés, nos modèles ont beaucoup de variables, de contraintes et des objectifs multiples. Par conséquent, il est difficile de calculer les solutions exactes à ce problème dans un temps réduit. Nous fournissons ensuite plusieurs algorithmes approximatifs rapides afin de résoudre des scenarios de grande taille. Dans le premier problème, nous simplifions le modèle en le divisant en deux étapes et les mettant en œuvre successivement. Cela signifie que le résultat de la première étape produira une entrée pour la dernière étape. Sur la base du modèle optimal et des solutions approximatives, nous avons analysé les résultats pour répondre à certaines questions importantes dans le système NFV, telles que la distribution des VNFs, les coûts généraux introduits par une solution basée sur la virtualisation des fonctions. Dans le deuxième problème, nous étudions une méthode originale pour générer une couverture de flux forte des inégalités pour améliorer la performance de la résolution de la LP. En raison de l'ajout de ces inégalités, le modèle devient plus simple et peut résoudre des scenarios avec de grandes instances. Pour le troisième problème, nous construisons des heuristiques basées sur la relaxation lagrangienne du problème d'écoulement. La plus importante qualité d'un algorithme de routage est la probabilité de blocage de la connexion. Ceci est mesuré à partir d'une simulation d'un événement stochastique axée sur un ensemble de réseaux typiques. Nous avons ensuite utilisé la simulation pour fournir des résultats étendus sur leur performance.

En résumé, les innovations clés de la solution sont les suivantes :

- Dans le problème de la gestion des ressources et du placement, nous considérons des contraintes qui peuvent être rencontrées dans un environnement réel, telles que les contraintes ressources sur les nœuds et les liens, les délais et l'ordre des VNFs dans un chaînage. L'idée majeure qui sous-tend notre solution est de modéliser le problème des coûts dans l'infrastructure NFV comme un problème quadratique avec de multiples objectifs. Nous considérons le cas hors ligne et le cas en ligne. En outre, nous envisageons également une optimisation continue des flux existants dans le système dynamique. Nous avons proposé trois heuristiques et évalué leur performances dans de nombreux scénarios. Nous avons montré la distribution de VNF sur de multiples niveaux de la topologie du réseau et avons suggéré les meilleurs emplacements ou devraient être localiser les VNFs. Ensuite, nous fournissons également une ligne directrice pour le pourcentage de trafic à migrer pour obtenir le meilleur gain dans un système NFV.

- Sur la base de ces premiers résultats et de leurs limites, nous développons un modèle étendu pour fournir des solutions optimales pour un même problème. Dans ce modèle, nous nous concentrons sur l'utilisation des ressources d'un système dynamique NFV et, par conséquent, on maximise ainsi le nombre de demandes acceptées. Nous proposons également une méthode pour générer de fortes inégalités afin d'améliorer les performances de résolution de la programmation linéaire. Via la mise en œuvre de simulation, nous montrons une amélioration significative lors de l'ajout de couvertures de flux à la programmation linéaire. En outre, les résultats illustrent également que notre modèle est efficace et capable de gérer de grands systèmes dynamiques.

- Contrairement au problème de gestion des ressources et de placement, le problème de routage ne mets pas l'accent sur le placement VNF sur les nœuds virtuels. Nous supposons que nous connaissons un ensemble de nœuds qui peuvent exécuter chaque VNF (appelé nœuds de fonction). L'objectif est de trouver un chemin rapidement lors de l'arrivée des demandes. Il s'agit d'un problème de routage important dans les grands systèmes de réseau. Nous avons proposé une extension réseau pour répondre aux contraintes de la chaîne de service et avons conçu une approche mathématique pour la multiplication du problème de routage en tant que modèle de flux à coût minimal entier. Nous avons ensuite fourni deux classes d'algorithmes basées sur la construction d'un réseau étendu prenant en compte les contraintes de fonction et la relaxation de Lagrange pour calculer des solutions approximatives. Afin d'évaluer leur performance, nous mettons en oeuvre une simulation stochastique événementielle d'un ensemble de réseaux typiques. Ces résultats nous fournissent des idées intéressantes sur les algorithmes et la performance réseau. En conséquence, il est important de souligner que le placement des VNF joue un rôle essentiel car cela aura une incidence sur la longueur de chemin nécessaire.

A.2.2 Future Direction de Recherche

Notre travail peut être étendu dans diverses directions. Dans cette thèse, nous avons démontré les résultats prometteurs et l'efficacité de nos solutions. Cependant, un travail substantiel reste à réaliser avec l'objectif d'obtenir un modèle de performance complet pour le placement NFV et le problème de routage dans des scénarios réalistes.

Nos résultats ont montré que le placement des VNFs joue un rôle important puisque cela affecte la longueur de chemin nécessaire pour répondre aux exigences requises. De plus, les demandes détaillées et les demandes de clients avec de grandes exigences sont difficiles à adapter dans un réseau. De fait, l'impact relatif des demandes des clients avec des exigences différentes devrait être considéré pour modifier efficacement les algorithmes de routage.

Un autre point intéressant peut concerner le problème de la congestion qui peut résulter d'un tel environnement. Surtout dans un grand réseau, le besoin des clients change continuellement et la quantité de trafic traversant le réseau est élevée pendant les heures de pointe. Ainsi le contrôle de la congestion est également l'un des problèmes importants dans le déploiement de NFV.

En outre, le travail peut être également étendu pour considérer certaines contraintes possibles qui peuvent survenir dans un environnement réel, tel que l'affinité serveur/VNF, les considérations de délai, de nouvelles caractéristiques des utlisateurs, l'attribution de la répartition des demandes dans le réseau. Afin de rendre un déploiement NFV réalisable, nous devons intégrer de nombreuses contraintes mais nous devons également garantir que le modèle n'est pas trop compliqué lors de la mise en œuvre.

Enfin, il est nécessaire de développer des outils pour concevoir des réseaux NFV. Ces outils nous aideront à évaluer les performances des solutions et à expérimenter l'efficacité des solutions proposées en utilisant des plates-formes de test.

Appendix B

Thesis Publications

Published and submitted papers

- **Thi-Minh Nguyen**, Andre Girard, Catherine Rosenberg, Serge FDIDA. Routing Via Functions in Virtual Networks: The Curse of Choices. *(submitted for publication)*
- **Thi-Minh Nguyen**, Serge FDIDA, Tuan-Minh Pham. A Comprehensive Resource Management and Placement for Network Function Virtualization, *The 3rd IEEE Conference on Network Softwarization (NetSoft 2017)*, 03-07 July, Italy.
- **Thi-Minh Nguyen**, Serge FDIDA, Tuan-Minh Pham. Resource Management and Placement in NFV, poster session, LINCS workshop, 2016, Paris, France.

Talk and presentations

- Resource Management and Placement in NFV, presentation in the Network Functions Virtualization reading group, LINCS, 2016, Paris, France.

List of Figures

List of Tables

References

[1] ETSI ISG NFV (Operator Group), "Network Functions Virtualisation (NFV) Network Operator Perspectives on Industry Progress," *White paper*, Oct. 2013.

[2] L. Nobach and D. Hausheer, "Open, elastic provisioning of hardware acceleration in NFV environments," in *2015 International Conference and Workshops on Networked Systems, NetSys 2015, Cottbus, Germany, March 9-12, 2015*, pp. 1–5, 2015.

[3] J. Keeney, S. van der Meer, and L. Fallon, "Towards real-time management of virtualized telecommunication networks," in *10th International Conference on Network and Service Management, CNSM 2014 and Workshop, Rio de Janeiro, Brazil, November 17-21, 2014*, pp. 388–393, 2014.

[4] L. Bondan, C. R. P. dos Santos, and L. Z. Granville, "Management requirements for clickos-based network function virtualization," in *10th International Conference on Network and Service Management, CNSM 2014 and Workshop, Rio de Janeiro, Brazil, November 17-21, 2014*, pp. 447–450, 2014.

[5] ETSI GS NFV-MAN 001 V1.1.1:, "Network Functions Virtualisation (NFV); Management and orchestration," *ESTI Ind. Spec. Group (ISG) Network Functions Virtualisation (NFV), Sophia-Atipolis Cedex, France [online]*, 2014.

[6] J. Soares, M. S. Dias, J. Carapinha, B. Parreira, and S. Sargento, "Cloud4nfv: A platform for virtual network functions," in *3rd IEEE International Conference on Cloud Networking, CloudNet 2014, Luxembourg, Luxembourg, October 8-10, 2014*, pp. 288–293, 2014.

[7] J. Soares, C. Goncalves, B. Parreira, P. Tavares, J. Carapinha, J. P. Barraca, R. L. Aguiar, and S. Sargento, "Toward a telco cloud environment for service functions," *IEEE Communications Magazine*, vol. 53, no. 2, no. 2, pp. 98–106, 2015.

[8] ETSI GS NFV-SEC 003 V1.1.1, "Network functions virtualisation (nfv); nfv security; security and trust guidance," *ESTI Ind. Spec. Group (ISG) Network Functions Virtualisation (NFV), Sophia-Atipolis Cedex, France [online]*, 2014.

[9] A. Fischer, J. F. Botero, M. T. Beck, H. de Meer, and X. Hesselbach, "Virtual network embedding: A survey," *IEEE Communications Surveys and Tutorials*, vol. 15, no. 4, no. 4, pp. 1888–1906, 2013.

[10] M. G. Rabbani, R. P. Esteves, M. Podlesny, G. Simon, L. Z. Granville, and R. Boutaba, "On tackling virtual data center embedding problem," in *2013 IFIP/IEEE Int. Symp. Integrated Netw. Manag.*, pp. 177–184.

[11] S. Lange, S. Gebert, T. Zinner, P. Tran-Gia, D. Hock, M. Jarschel, and M. Hoffmann, "Heuristic approaches to the controller placement problem in large scale SDN networks," *IEEE Trans. Network and Service Management*, vol. 12, no. 1, no. 1, pp. 4–17, 2015.

[12] D. G. Luenberger, *Linear and Nonlinear Programming.* Addison-Wesley, 1984.

[13] ETSI, "White paper on network functions virtualization,"

[14] R. Mijumbi, J. Serrat, J. Gorricho, N. Bouten, F. D. Turck, and R. Boutaba, "Network function virtualization: State-of-the-art and research challenges," *IEEE Communications Surveys and Tutorials*, vol. 18, no. 1, no. 1, pp. 236–262, 2016.

[15] J. de Jesus Gil Herrera and J. F. B. Vega, "Network Functions Virtualization: A Survey," *IEEE Latin America Transactions*, pp. 983 – 997, 2016.

[16] ETSI, "Network function virtualization: An introduction,benefits,enablers,challenges, call for action," 2012.

[17] ETSI Ind. Specification Group (ISG), "Network Functions Virtualization (NFV) Management and Orchestration," *Valbonne, France*, Dec. 2014.

[18] ETSI GS NFV 001, "Network functions virtualisation (NFV) use cases," Oct. 2013.

[19] NIST SP 800-146, Badger et al., "Draft cloud computing synopsis and recommendations," pp. 7–2, May 2011.

[20] M. Bouet, J. Leguay, T. Combe, and V. Conan, "Cost-based placement of vdpi functions in NFV infrastructures," *Int. Journal of Network Management*, pp. 490–506, 2015.

[21] X. Li and C. Qian, "A survey of network function placement," in *13th IEEE Annu. Consumer Commu. & Netw. Conf., CCNC 2016, January 9-12*, pp. 948–953.

[22] J. G. Herrera and J. F. Botero, "Resource allocation in nfv: A comprehensive survey," *IEEE Trans. Netw. Service Manag.*, vol. 13, no. 3, pp. 518–532, Sep. 2016.

[23] ETSI, "Network functions virtualization (NFV); architectural framework," ETSI TR GS NFV 002, v1.2.1, Dec. 2014.

[24] S. Clayman, E. Maini, A. Galis, A. Manzalini, and N. Mazzocca, "The dynamic placement of virtual network functions," in *2014 IEEE Network Operations and Management Symposium, NOMS 2014, Krakow, Poland, May 5-9, 2014*, pp. 1–9, 2014.

[25] M. T. Beck and J. F. Botero, "Coordinated allocation of service function chains," in *2015 IEEE Global Communications Conference, GLOBECOM 2015, San Diego, CA, USA, December 6-10, 2015*, pp. 1–6, 2015.

[26] S. Mehraghdam and H. Karl, "Specification of complex structures in distributed service function chaining using a YANG data model," *CoRR*, vol. abs/1503.02442, 2015.

[27] A. Belbekkouche, M. M. Hasan, and A. Karmouch, "Resource discovery and allocation in network virtualization," *IEEE Communications Surveys and Tutorials*, pp. 1114–1128, 2012.

[28] R. Cohen, L. Lewin-Eytan, J. Naor, and D. Raz, "Near optimal placement of virtual network functions," in *2015 IEEE Conf. Computer Commu., INFOCOM, Apr. 26 - May 1*, pp. 1346–1354.

[29] R. Riggio, T. Rasheed, and R. Narayanan, "Virtual network functions orchestration in enterprise wlans," in *IFIP/IEEE International Symposium on Integrated Network Management, IM 2015, Ottawa, ON, Canada, 11-15 May, 2015*, pp. 1220–1225, 2015.

[30] M. Ghaznavi, A. Khan, N. Shahriar, K. Alsubhi, R. Ahmed, and R. Boutaba, "Elastic virtual network function placement," in *4th IEEE International Conference on Cloud Networking, CloudNet 2015, Niagara Falls, ON, Canada, October 5-7, 2015*, pp. 255–260, 2015.

[31] R. Bruschi, A. Carrega, and F. Davoli, "A game for energy-aware allocation of virtualized network functions," *J. Electrical and Computer Engineering*, vol. 2016, pp. 4067186:1–4067186:10, 2016.

[32] B. Martini, F. Paganelli, P. Cappanera, S. Turchi, and P. Castoldi, "Latency-aware composition of virtual functions in 5g," in *Proceedings of the 1st IEEE Conference on Network Softwarization, NetSoft 2015, London, United Kingdom, April 13-17, 2015*, pp. 1–6, 2015.

[33] A. Baumgartner, V. S. Reddy, and T. Bauschert, "Mobile core network virtualization: A model for combined virtual core network function placement and topology optimization," in *Proceedings of the 1st IEEE Conference on Network Softwarization, NetSoft 2015, London, United Kingdom, April 13-17, 2015*, pp. 1–9, 2015.

[34] A. Mohammadkhan, S. Ghapani, G. Liu, W. Zhang, K. K. Ramakrishnan, and T. Wood, "Virtual function placement and traffic steering in flexible and dynamic software defined networks," in *2015 IEEE International Workshop on Local and Metropolitan Area Networks, LANMAN 2015, Beijing, China, April 22-24, 2015*, pp. 1–6, 2015.

[35] T. Lin, Z. Zhou, M. Tornatore, and B. Mukherjee, "Optimal network function virtualization realizing end-to-end requests," in *2015 IEEE Global Communications Conference, GLOBECOM 2015, San Diego, CA, USA, December 6-10, 2015*, pp. 1–6, 2015.

[36] I. Jang, S. Choo, M. Kim, S. Pack, and M. Shin, "Optimal network resource utilization in service function chaining," in *IEEE NetSoft Conference and Workshops, NetSoft 2016, Seoul, South Korea, June 6-10, 2016*, pp. 11–14, 2016.

[37] T. Kuo, B. Liou, K. C. Lin, and M. Tsai, "Deploying chains of virtual network functions: On the relation between link and server usage," in *35th Annual IEEE International Conference on Computer Communications, INFOCOM 2016, San Francisco, CA, USA, April 10-14, 2016*, pp. 1–9, 2016.

[38] T. Taleb, M. Bagaa, and A. Ksentini, "User mobility-aware virtual network function placement for virtual 5g network infrastructure," in *2015 IEEE International Conference on Communications, ICC 2015, London, United Kingdom, June 8-12, 2015*, pp. 3879–3884, 2015.

[39] S. Lal, T. Taleb, and A. Dutta, "NFV: security threats and best practices," *IEEE Communications Magazine*, vol. 55, no. 8, no. 8, pp. 211–217, 2017.

[40] Y. Xu, J. Liu, Y. Shen, X. Jiang, and T. Taleb, "Security/qos-aware route selection in multi-hop wireless ad hoc networks," in *2016 IEEE International Conference on Communications, ICC 2016, Kuala Lumpur, Malaysia, May 22-27, 2016*, pp. 1–6, 2016.

[41] H. Ko, G. Lee, I. Jang, and S. Pack, "Optimal middlebox function placement in virtualized evolved packet core systems," in *17th Asia-Pacific Network Operations and Management Symposium, APNOMS 2015, Busan, South Korea, August 19-21, 2015*, pp. 511–514, 2015.

[42] J. Lee, H. Ko, D. Suh, S. Jang, and S. Pack, "Overload and failure management in service function chaining," in *2017 IEEE Conference on Network Softwarization, NetSoft 2017, Bologna, Italy, July 3-7, 2017*, pp. 1–5, 2017.

[43] J. Liu, Y. Li, Y. Zhang, L. Su, and D. Jin, "Improve service chaining performance with optimized middlebox placement," *IEEE Trans. Services Computing*, vol. 10, no. 4, no. 4, pp. 560–573, 2017.

[44] J. Liu, Y. Li, H. Wang, D. Jin, L. Su, L. Zeng, and T. Vasilakos, "Leveraging software-defined networking for security policy enforcement," *Inf. Sci.*, vol. 327, pp. 288–299, 2016.

[45] 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 *2015 IFIP/IEEE Int. Symp. Integrated Netw. Manag. (IM)*, pp. 98–106, May.

[46] T. Lukovszki and S. Schmid, "Online admission control and embedding of service chains," *CoRR*, vol. abs/1506.04330, 2015.

[47] L. E. Li, V. Liaghat, H. Zhao, M. Hajiaghayi, D. Li, G. T. Wilfong, Y. R. Yang, and C. Guo, "PACE: policy-aware application cloud embedding," in *Proc. IEEE INFOCOM 2013, Apr. 14-19*, pp. 638–646.

[48] J. Elias, F. Martignon, S. Paris, and J. Wang, "Efficient orchestration mechanisms for congestion mitigation in nfv: Models and algorithms," *IEEE Trans. Services Comput.*, vol. PP, no. 99, no. 99, pp. 1–1, 2015.

[49] M. Obadia, J. L. Rougier, L. Iannone, V. Conan, and M. Bouet, "Revisiting NFV orchestration with routing games," in *2016 IEEE Conference on Network Function Virtualization and Software Defined Networks (NFV-SDN), Palo Alto, CA, USA, November 7-10, 2016*, pp. 107–113, 2016.

[50] S. Quanying, L. Ping, L. Wei, and Z. Zuqing, "Forecast assisted nfv service chain deployment based on affiliation aware vnf placement," in *Proc. IEEE GLOBECOM*, 2016.

[51] W. Ma, C. Medina, and D. Pan, "Traffic-aware placement of NFV middleboxes," in *2015 IEEE Global Communications Conference, GLOBECOM 2015, San Diego, CA, USA, December 6-10, 2015*, pp. 1–6, 2015.

[52] J. Liu, W. Lu, F. Zhou, P. Lu, and Z. Zhu, "On dynamic service function chain deployment and readjustment," *IEEE Trans. Network and Service Management*, 2017.

[53] B. Addis, D. Belabed, M. Bouet, and S. Secci, "Virtual network functions placement and routing optimization," in *4th IEEE International Conference on Cloud Networking, CloudNet 2015, Niagara Falls, ON, Canada, October 5-7, 2015*, pp. 171–177, 2015.

[54] T. Kuo, B. Liou, K. C. Lin, and M. Tsai, "Deploying chains of virtual network functions: On the relation between link and server usage," in *35th Annual IEEE International Conference on Computer Communications, INFOCOM 2016, San Francisco, CA, USA, April 10-14, 2016*, pp. 1–9, 2016.

[55] J. F. Riera, E. Escalona, J. Batalle, S. Pack, E. Grasa, and J. A. Garcia-Espin, "Virtual network function scheduling: Concept and challenges," in *2014 International Conference on Smart Communications in Network Technologies (SaCoNeT), Vilanova i la Geltru, Spain*, 2014.

[56] J. F. Riera, X. Hesselbach, E. Escalona, J. A. Garcia-Espin, and E. Grasa, "On the complex scheduling formulation of virtual network functions over optical networks," in *2014 16th International Conference on Transparent Optical Networks (ICTON), Graz, Austria*, 2014.

[57] M. Shifrin, E. Biton, and O. Gurewitz, "Optimal control of VNF deployment and scheduling," *CoRR*, vol. abs/1611.06956, 2016.

[58] M. F. Bari, S. Chowdhury, R. Ahmed, and R. Boutaba, "On orchestrating virtual network functions," in *11th Int. Conf. Netw. Service Manag. (CNSM)*, pp. 50–56, Nov. 2015.

[59] R. Mijumbi, J. Serrat, J.-L. Gorricho, N. Bouten, F. De Turck, and S. Davy, "Design and evaluation of algorithms for mapping and scheduling of virtual network functions," in *IEEE Conf. Netw. Softwarization (NetSoft), 2015*.

[60] M. Xia, M. Shirazipour, Y. Zhang, H. Green, and A. Takacs, "Network function placement for NFV chaining in packet/optical datacenters," *Journal of Lightwave Technology*, April 2015.

[61] A. Hmaity, M. Savi, F. Musumeci, M. Tornatore, and A. Pattavina, "Virtual network function placement for resilient service chain provisioning," in *2016 8th International Workshop on Resilient Networks Design and Modeling (RNDM), Halmstad, Sweden, September 13-15, 2016*, pp. 245–252, 2016.

[62] M. Savi, M. Tornatore, and G. Verticale, "Impact of processing costs on service chain placement in network functions virtualization," in *IEEE Conference on Network Function Virtualization and Software Defined Networks, NFV-SDN 2015, San Francisco, CA, USA, November 18-21, 2015*, pp. 191–197, 2015.

[63] D. Krishnaswamy, R. Krishnan, D. Lopez, P. Willis, and A. Qamar, "An open NFV and cloud architectural framework for managing application virality behaviour," in *12th Annual IEEE Consumer Communications and Networking Conference, CCNC 2015, Las Vegas, NV, USA, January 9-12, 2015*, pp. 746–754, 2015.

[64] C. Lee, Y. Nakagawa, K. Hyoudou, S. Kobayashi, O. Shiraki, and T. Shimizu, "Flow-aware congestion control to improve throughput under TCP incast in datacenter networks," in *39th Annual Computer Software and Applications Conference, COMPSAC Workshops 2015, Taichung, Taiwan, July 1-5, 2015*, pp. 155–162, 2015.

[65] Y. Nakagawa, C. Lee, K. Hyoudou, S. Kobayashi, O. Shiraki, J. Tanaka, and T. Ishihara, "Dynamic virtual network configuration between containers using physical switch functions for NFV infrastructure," in *IEEE Conference on Network Function Virtualization and Software Defined Networks, NFV-SDN 2015, San Francisco, CA, USA, November 18-21, 2015*, pp. 156–162, 2015.

[66] S. Herker, X. An, W. Kiess, and A. Kirstädter, "Evaluation of data-center architectures for virtualized network functions," in *IEEE International Conference on Communication, ICC 2015, London, United Kingdom, June 8-12, 2015, Workshop Proceedings*, pp. 1852–1858, 2015.

[67] G. Agapiou, I. Papafili, and S. Agapiou, "The role of SDN and NFV for dynamic bandwidth allocation and qoe adaptation of video applications in home networks," in *Euro Med Telco Conference, EMTC 2014, Naples, Italy, November 12-15, 2014*, pp. 1–4, 2014.

[68] Y. Liu, Y. Li, M. Canini, Y. Wang, and J. Yuan, "Scheduling multi-flow network updates in software-defined NFV systems," in *IEEE Conference on Computer Communications Workshops, INFOCOM Workshops 2016, San Francisco, CA, USA, April 10-14, 2016*, pp. 548–553, 2016.

[69] L. Wang, Z. Lu, X. Wen, R. Knopp, and R. Gupta, "Joint optimization of service function chaining and resource allocation in network function virtualization," in *IEEE Access*, 2016.

[70] D. Dietrich, A. Abujoda, A. Rizk, and P. Papadimitriou, "Multi-provider service chain embedding with nestor," *IEEE Trans. Network and Service Management*, pp. 91–105, 2017.

[71] W. Rankothge, F. Le, A. Russo, and J. Lobo, "Optimizing resource allocation for virtualized network functions in a cloud center using genetic algorithms," *IEEE Trans. Network and Service Management*, pp. 343–356, 2017.

[72] L. Qu, C. Assi, and K. B. Shaban, "Delay-aware scheduling and resource optimization with network function virtualization," *IEEE Trans. Communications*, pp. 3746–3758, 2016.

[73] L. Guo, J. Pang, and A. Walid, "Dynamic service function chaining in sdn-enabled networks with middleboxes," in *24th IEEE International Conference on Network Protocols, ICNP 2016, Singapore, November 8-11, 2016*, pp. 1–10, 2016.

[74] C. Pham, N. H. Tran, S. Reny, W. Saad, and C. S. Hong, "Traffic-aware and energy-efficient vnf placement for service chaining: Joint sampling and matching approach," *IEEE TRANSACTIONS ON SERVICES COMPUTING*, 2017.

[75] B. Addis, D. Belabed, M. Bouet, and S. Secci, "Virtual network functions placement and routing optimization," in *2015 IEEE 4th Int. Conf. Cloud Netw. (CloudNet)*, pp. 171–177, Oct.

[76] A. Leivadeas, M. Falkner, I. Lambadaris, and G. Kesidis, "Dynamic traffic steering of multi-tenant virtualized network functions in SDN enabled data centers," in *21st IEEE International Workshop on Computer Aided Modelling and Design of Communication Links and Networks, CAMAD 2016, Toronto, ON, Canada, October 23-25, 2016*, pp. 65–70, 2016.

[77] H. J. Prömel and A. Steger, "The steiner tree problem. a tour through graphs, algorithms and complexity," Feb. 2002, ISBN 3528067624.

[78] `http://www.gurobi.com/`.

[79] M. Al-Fares, A. Loukissas, and A. Vahdat, "A scalable, commodity data center network architecture," in *Proc. ACM SIGCOMM 2008 Conf. Applications, Technol., Architectures, Protocols for Computer Commu., Aug. 17-22*, pp. 63–74.

[80] C. Guo, G. Lu, D. Li, H. Wu, X. Zhang, Y. Shi, C. Tian, Y. Zhang, and S. Lu, "Bcube: a high performance, server-centric network architecture for modular data centers," in *Proc. ACM SIGCOMM 2009 Conf. Applications, Technol., Architectures, Protocols for Computer Commu., Aug. 16-21*, pp. 63–74.

[81] `https://azure.microsoft.com/en-us/pricing/calculator/`.

[82] `https://aws.amazon.com/ec2/pricing/`.

[83] M. W. Padberg, T. J. V. Roy, and L. A. Wolsey, "Valid linear inequalities for fixed charge problems," *Operations Research*, vol. 33, no. 4, no. 4, pp. 842–861, 1985.

[84] T. J. V. Roy and L. A. Wolsey, "Valid inequalities for mixed 0-1 programs," *Discrete Applied Mathematics*, vol. 14, no. 2, no. 2, pp. 199–213, 1986.

[85] T. J. Roy and L. A. Wolsey, "Solving mixed integer programming problems using automatic reformulation," *Operations Research*, vol. 35, no. 1, no. 1, pp. 45–57, 1987.

[86] Z. Gu, G. L. Nemhauser, and M. W. P. Savelsbergh, "Lifted cover inequalities for 0-1 integer programs: Computation," *INFORMS Journal on Computing*, vol. 10, no. 4, no. 4, pp. 427–437, 1998.

[87] M. Gondran and M. Minoux, *Graphs, Dioids and Semirings*.

[88] S. Orlowski, R. Wessäly, M. Pióro, and A. Tomaszewski, "Sndlib 1.0 - survivable network design library," *Networks*, vol. 55, no. 3, no. 3, pp. 276–286, 2010.

[89] A. Medina, A. Lakhina, I. Matta, and J. W. Byers, "BRITE: an approach to universal topology generation," in *9th International Workshop on Modeling, Analysis, and Simulation of Computer and Telecommunication Systems (MASCOTS 2001), 15-18 August 2001, Cincinnati, OH, USA*.