Simple and Scalable Approach for Virtualized Network Function Placement in Wireless Multi-hop Networks

by

Zahra Jahedi, M.Sc.

A thesis submitted to the
Faculty of Graduate and Postdoctoral Affairs
in partial fulfillment of the requirements for the degree of

Doctor of Philosophy in Electrical and Computer Engineering

Ottawa-Carleton Institute for Electrical and Computer Engineering
Department of Systems and Computer Engineering
Carleton University
Ottawa, Ontario
August, 2021

Abstract

Network Function Virtualization (NFV) can lower the CAPEX and/or OPEX for service providers and allow for quick deployment of services. The main challenge in the use of Virtualized Network Functions (VNF) is the VNFs' placement in the network. This research provides mathematical models and heuristics for NF placement for wired and wireless networks. We use Integer Linear and Non-Linear Programming as a mathematical optimization program for NF placement. We start from a basic model for a wired network and extend it gradually to develop a traffic-aware mathematical model for NF placement in wireless multi-hop networks. For the first time, we model the interference which is a major difference between a wired and wireless network and included it in our optimization model. We identified the issue of scarcity of BW in wireless multi-hop networks and its role in the average cost of placement and acceptance rate of requests. The critical problem of mathematical models is that they are NP-hard, and consequently not applicable to larger networks. While there exist many efforts in designing a heuristic model that can provide solutions in a timely manner, the primary focus with such heuristics was almost always whether they provide near-optimal results. Consequently, the heuristics themselves become quite non-trivial, and solving the placement problem for larger networks still takes a significant amount of time. In our research, in contrast, we focus on designing a simple and scalable heuristic. We propose a set of heuristics, which are gradually becoming more complex. We start from the random placement heuristic as the simplest approach and at each step add a parameter such as choosing between shortest paths, sort NFs based on their nodal resources, and replacing previously placed NFs to our heuristic. We compare the performance of our heuristics with each other, related heuristics, and our mathematical model. Our results demonstrate that the simple approach of placing NFs along their shortest path can find near-optimal solutions much faster than the other more complicated heuristics while keeping the ratio of accepted requests close to the acceptance ratio of a NP-hard optimization model.

Acknowledgments

Firstly, I would like to express my sincere gratitude to my advisor, Professor Thomas Kunz, for the continuous support of my Ph.D. study and related research, for your patience, motivation, and immense knowledge. Your guidance helped me in all the time of research and writing of this thesis. Your constant willingness to share ideas with me regarding my research, in spite of your busy schedule, is sincerely appreciated.

I cannot express my unfailing gratitude and love to my caring, loving husband, Arash Karami-Mohammadi, who has supported me throughout this process. Your encouragement when the times got rough is much appreciated.

Last, but not least, I owe the greatest debt to my parents for not only their never-ending affection, love, and patience, but also their encouragement and unconditional support to me in all my decisions. Their guidance helped me to keep my motivations high throughout my studies. Without the continual support from my parents and siblings, there is no way I would be where I am today. It is my great pleasure to dedicate this thesis to my dear husband and parents.

Table of Contents

List of Tables

List of Figures

Abbreviations

AMPL	A Mathematical Programming Language
BW	Bandwidth
CI	Confidence Interval
DPI	Deep Packet Inspection
FAST	Fast and Cost-Efficient Placement Algorithm
FW	Firewall
InPs	Infrastructure Providers
ILP	Integer Linear Programming
INLP	Integer Non-Linear Programming
ISP	Internet Service Provider
IDS	Intrusion Detection System
LFGL	Least-First-Greatest-Last
LP	Linear Programming
LRP	Location Routing Problem
MANET	Mobile Ad-hoc Network
MILP	Mixed Integer Linear Programming
MME	Mobility Management Entity
NAT	Network Address Translation
NF	Network Function

NFEP	Network Function Embedding Problem
NFV	Network Function Virtualization
NLP	Non-Linear Programming
OFDMA	Orthogonal Frequency Division Multiple Access
PCS	Parental Control Service
P-GW	Packet Gateway
S-GW	Serving Gateway
SG	Service Graph
SLA	Service Layer Agreement
SP	Service Provider
SLFL	Simple Lazy Facility Location
SDN	Software Defined Network
STT	Stateless Transport Tunneling
TMS	Threat Mitigation Service
VANET	Vehicular Ad-hoc Network
vDPI	Virtual Deep Packet Inspection
vEPC	Virtual Evolved Packet Core
vEPDG	Virtualized Evolved Packet Data Gateway
vPG	Virtualized Packet Gateway
vTWAG	Virtualized Trusted Wireless Access Gateway
VO	Video Optimization
vIMS	Virtual IP Multimedia Subsystem
VNF	Virtual Network Functions
WAN	Wide Area Network

Symbols

$alter_i$	Traffic changing factor for NF i that can be a positive or a negative value. NFs which increase the traffic rate have positive traffic changing factor, and NFs which decrease the traffic rate have negative traffic changing factor.
$BW_{E_{uv}}$	Available BW over the physical link between node u and v.
bw_f	BW request of flow f.
$bw_{f,E_{uv}}$	A positive real variable representing BW usage of flow f over the physical link E_{uv}.
$bw_{f,e_{ij}}$	Requested BW for the link that is connecting NF i to NF j in flow f.
$bw_{f',e_{ij}}$	Requested BW for the link that is connecting NF, i to j in the joint flow f'.
C_u	Available processing units in physical node u.
$c_{f,i}$	Requested processing units for NF i of flow f.
$c_{f',i}$	Requested processing units for NF i of the joint flow f'.
$candid_i$	Defined for each NF of the SG i and is equal to the number of nodes along the shortest path that can be used for the placement of that specific NF.
d_f	The destination of flow f.
$Dist_u$	An array that represents the shortest distance in terms of the number of hops from the the source node.
d_{uv}	The distance between nodes u and v.

E_{uv}	The physical link connecting physical node u to v where $E_{uv} \in L_p$.
e_{ij}	The virtual link which connects NF i to j where $e_{ij} \in L_f$.
$e_{f',ij}$	The virtual link which connects NF i to j from the joint flow f' where $e_{f',ij} \in L_{f'}$.
f	Representing the current flow that consists of a set of requested NFs with required resources.
f'	A joint flow which includes the current flow and all of the placed and not expired flows.
$F_{f,E_{uv}}$	A binary variable where one means flow f passing physical link from node u to v.
$F_{f,e_{ij},E_{uv}}$	A binary variable which is equal to one when the virtual link between NF i and j is mapped to one or more physical links and physical link E_{uv} is one of them.
$F_{f',e_{ij},E_{uv}}$	A binary variable which is equal to one when the virtual link between NFs i and j of the joint flow f' is mapped to one or more physical links and physical link E_{uv} is one of them.
$intset_{E_{uv}}$	Interference set for the physical link E_{uv}, consists of all the links that are connected to the nodes in the transmission range of its sender or receiver.
L_f	Set of virtual links between NFs of flow f.
$L_{f'}$	Set of virtual links between NFs of the joint flow f'.
L_p	Set of physical links between nodes of the physical network.
M_u	Available memory units in physical node u.
$m_{f,i}$	Requested memory units for NF i of flow f.
$m_{f',i}$	Requested memory units for NF i of the joint flow f'.
N_p	Set of physical nodes where u is representing node $u \in N_p$.
N_f	Set of NFs where $i \in N_f$ represents NF i in flow f.

$N_{f'}$	Set of NFs where $i \in N_{f'}$ represents NF i in the joint flow f'.
$Nodes_u$	A matrix which records nodes involved in each different shortest paths found from source node to node u.
R	Transmission range of nodes in a multi-hop wireless network.
S_u	Available storage units in physical node u.
s_f	The source of flow f.
$s_{f,i}$	Requested storage units for NF i of flow f.
$s_{f',i}$	Requested storage units for NF i of the joint flow f'.
SG_f	A set of requested NFs with required resources of flow f.
t_{f,u^-}	A real variable representing the traffic rate factor before node u.
t_{f,u^+}	A real variable representing the traffic rate factor after node u based on the NFs that were placed in that node.
w_c	The cost of consuming each unit of processing resources.
w_m	The cost of consuming each unit of memory.
w_s	The cost of consuming each unit of storage.
$X_{f,u}$	A binary variable where one means flow f traversing physical node u.
$x_{f,i,u}$	A binary variable where one means that function i from flow f is placed in physical node u.
$x_{f',i,u}$	A binary variable where one means that NF i from the joint flow f' is placed in the physical node u.

Chapter 1

Introduction

1.1 Motivation

Network Function Virtualization (NFV), the practice of decoupling network hardware and software to allow network services to run on commodity servers, is a transformational vision that attracts a lot of attention in the telecommunication industry. The hope is that virtualization brings advantages such as enabling faster deployment of new services with less risk, allowing iterative improvement of existing services, broadening the developer ecosystem to include new entrants, and reducing network cost structure through infrastructure sharing and automation [4]. NFV covers a wide spectrum of Network Functions (NF) such as firewalls, Deep Packet Inspection (DPI), Intrusion Detection System (IDS), Network Address Translation (NAT), and Wide Area Network (WAN) accelerators. It also covers a variety of network nodes such as broadband remote access server, data network gateways (S-GW/P-GW), Mobility Management Entity (MME), Home Subscriber Server (HSS), and virtual IP Multimedia Subsystem (vIMS) for virtual Evolved Packet Core (vEPC). These are the critical devices in mobile broadband and cellular networks [5].

By leveraging NFV and Software Defined Network (SDN), Virtual Network Functions (VNF) can be installed, removed, or migrated dynamically to adapt to the dynamic network resource requirements due to changes in network topology or network traffic load. In this context, the VNFs are commonly placed in a chain of a specific order in the substrate network. The chained VNFs form a Service Graph (SG) and process traffic flows to deliver end-to-end network services [6]. Two simple SG models for mobile Internet upstream and downstream traffic is shown in Figure 1.1 [1]. As more and more of the traffic includes video, in both SGs we have a Video Optimizer

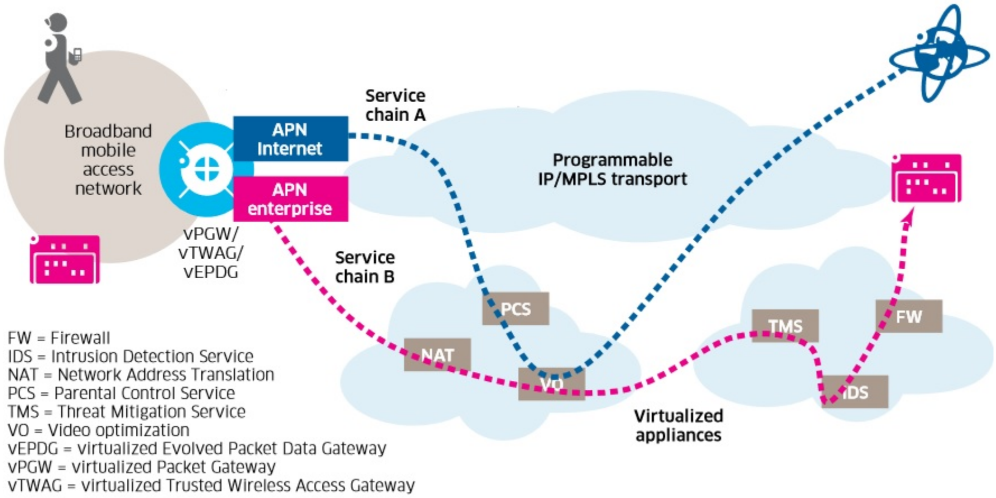
Broadband mobile access network
vPGW/ vTWAG/ vEPDG
APN Internet
APN enterprise
Service chain A
Service chain B
Programmable IP/MPLS transport
PCS
NAT
VO
TMS
IDS
FW
Virtualized appliances
FW = Firewall
IDS = Intrusion Detection Service
NAT = Network Address Translation
PCS = Parental Control Service
TMS = Threat Mitigation Service
VO = Video optimization
vEPDG = virtualized Evolved Packet Data Gateway
vPGW = virtualized Packet Gateway
vTWAG = virtualized Trusted Wireless Access Gateway

Figure 1.1: Example of two SGs [1]

(VO) that optimizes the video content from the internet for mobile networks and devices. In the first SG, the Parental Control Service (PCS) prevents some users from accessing specific content. The second SG uses Network Address Translation (NAT) that maps the private IP address space dedicated to user equipment to a public IP address and a set of security VNFs such as Firewall (FW), Intrusion Detection Service (IDS), and Threat Mitigation Service (TMS) that protects the carrier network from the outside.

The placement of NFs of an SG can be referred to as a Network Function Embedding Problem (NFEP). NFEP can be explained as a way to map VNFs and the links between them to the physical network so that the computing resources such as CPU, memory, and network resources such as link bandwidths are efficiently utilized and the service requirements are met [2]. NFEP can be optimized based on the characteristics and available resources of the network. The placement of NFs can affect the path traffic flows take and consequently, bandwidth usage in the network [7]. Figure 1.2 shows an example of an SG deployment problem. The upper graph is a SG composed of two VNFs between a source node and a destination node, and the bottom graph is a substrate network with six substrate nodes on which the SG can be deployed.

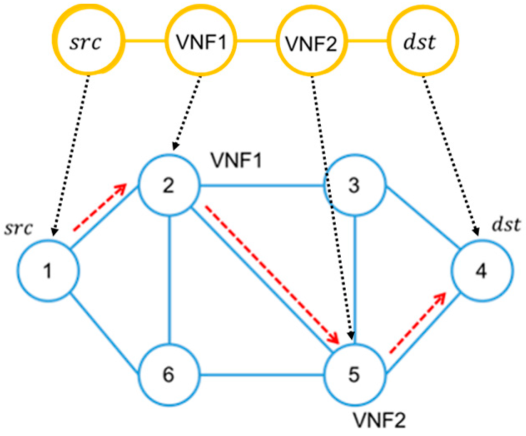

Figure 1.2: An Example of SG Deployment on Substrate Network [2]

There are several types of algorithms proposed to solve the NFEP. Previous studies are mostly focused on the placement of VNFs in wired networks, while the use of NFV can bring comparable advantages to wireless networks. NFV introduces new possibilities to wireless networks such as network virtualization, that for example, subscribers can customize their exclusive access networks while using the shared infrastructure. The amount of literature on wireless network virtualization shows the importance of NFV in wireless networks. However, there are only a few papers considering the problem of NFEP in wireless networks.

1.2 Thesis Contributions

Our first goal is to design a comprehensive NFEP model for multi-hop wireless networks. We identify the important characteristic of wireless networks, which is interference, and include it in our model. To our knowledge, none of the proposed methods for NFEP in wireless networks included the effect of interference in their optimization model. It is assumed that the interference is being handled by using orthogonal channels in the network. This assumption can be challenged from two perspectives. First, the use of orthogonal channels is only possible when we have a multi-radio multi-channel network which is not always the case in wireless networks. Second, even in the multi-radio multi-channel networks, there is still a possibility of interference and it is not possible to eliminate the effect of interference.

We use one of the comprehensive models provided for placing NFs in a wired network as a basic model that provide an optimal placement for one request at a time. The basic model formulates the NFEP in wired networks as an optimization problem that can be solved with Integer Linear Programming (ILP). In this method, the objective is to minimize the mapping cost based on the resource requirements of the NFs and available resources in the network. The cost of a mapping is based on the costs of the consumed resources by the NFs in the physical network which include:

- The cost of total units of CPU, memory, and storage used by NFs in physical nodes.
- The cost of total units of bandwidth used by virtual links in the physical network.

We improved the model in order to include the effect of interference, based on the protocol model. Our results show that interference increases the BW usage and consequently increases the placement cost and decreases the number of accepted requests (i.e., SGs that can successfully be placed in a physical network shared by many flows). We further expanded the ILP model for wireless networks in order to consider the following parameters:

- A source and destination for each request.
- Traffic changing factor, which considers the effect of NFs on BW consumption.

The resulting optimization problem becomes non-linear, which raises complexity issues. In studying the results, we observed, even in the case of the simpler linear models, solving the optimization problem for small networks demands high computational resources and the execution times are high as well. It is been shown that ILP models are NP-hard and not applicable to large scale networks [8].

In order to provide a simpler and time-efficient solution for the problem of embedding VNFs that is applicable to larger networks we focused on the design of heuristic algorithms. A number of heuristics have been proposed in the literature, the main purpose of such heuristics is to provide results approximately as good as the mathematical model. Consequently, the heuristics become time consuming and not applicable to larger networks. In contrast, our goal is to provide a heuristic that considers the parameters that contribute to improving results and avoid non-essential complexities. We propose five heuristics that are gradually becoming more complex and explore the effectiveness of each of the added parameters at each step.

Our heuristics start from our simplest placement algorithm, random placement, which places the NFs randomly. We then gradually add parameters that can potentially improve the performance of our heuristic in terms of the number of accepted requests, the average cost of placement, and the execution time. Our second heuristic, the shortest path placement, considers the scarcity of BW and places the NFs along the shortest path. The all shortest path heuristic, our third heuristic, searches for all shortest paths and chooses one for placement in a way to increase the possibility of accepting a request. The Fast and Cost-Efficient (FACE) heuristic is our fourth heuristic. FACE heuristic chooses between all shortest paths and prioritizes placement of the NFs that have higher nodal resource demand to increase the probability of accepting a request. Last but not least, the joint heuristic considers the previously placed NFs along with the current request in order to place the current SG.

In summary in this document, we provide a comprehensive mathematical model and use it as a benchmark to compare with the performance of our heuristics. By starting from the simplest approach of placing the NFs randomly and step by step adding one parameter to our heuristic we identify the effectiveness of added parameters. We compare the performance of our heuristics with each other, our mathematical model, and similar heuristics to show that a very simple heuristic can find near-optimal solutions much faster than the other more complicated heuristics while keeping the number of accepted requests close to the results achieved with an NP-hard optimization model.

1.3 Publications

The list of our publications are as follows:

1. Z. Jahedi and T. Kunz, "Virtual network function embedding in multi-hop wireless networks," in Proceedings of the 15th International Joint Conference on e-Business and Telecommunications, ICETE 2018 - Volume 1: DCNET, ICE-B,OPTICS, SIGMAP and WINSYS, Porto, Portugal, July 26-28, 2018., pp. 199–207, 2018. [7]

2. Z. Jahedi and T. Kunz, "Optimal VNF placement: Addressing multiple min-cost solutions," in e-Business and Telecommunications, pp. 1–23, Springer International Publishing, 2019 [9]

3. Z. Jahedi and T. Kunz, “Fast and cost-efficient virtualized network function placement algorithm in wireless multi-hop networks,” in Ad-Hoc, Mobile, and Wireless Networks(L. A. Grieco, G. Boggia, G. Piro, Y. Jararweh, and C. Campolo, eds.), pp. 23–36, Springer International Publishing, 2020. [10]

4. Z.Jahedi and T. Kunz, “The value of simple heuristics for virtualized network function placement,”Future Internet, vol. 12, no. 10, 2020. [11]

Our first publication, titled ”Virtual network function embedding in multi-hop wireless networks” [7] includes the extended model described in Section 3.4 of Chapter 3 and was presented in the 15th International Joint Conference on e-Business and Telecommunications. Our paper also has been selected to be included in the CCIS Series book published by Springer [9]. Our mathematical joint model described in Chapter 4, Section 4.5.1, was published in the CCIS series book published by Springer, titled ”Optimal VNF placement: addressing multiple min-cost solutions” [9]. In this paper [9], we introduce a joint optimization model that provides optimal placement of both previous and current requests at once. In the joint optimization model, a new request and previously placed requests will be placed in the network optimally. Our joint all shortest path heuristic described in Section 4.5.2 of Chapter 4 uses this idea in the design of our joint heuristic.

Our third publication describes the FACE heuristic mentioned in Chapter 4, Section 4.4, and was presented in the 19th International Conference on Ad Hoc Networks and Wireless (AdHoc-Now 2020), titled ”Fast and cost-efficient virtualized network function placement algorithm in wireless multi-hop networks” [10]. Our forth publication, ”The value of simple heuristics for virtualized network function placement” [11] describes four increasingly complex heuristics: random placement, shortest path placement, all shortest path placement, FACE heuristic, and their performance comparison that are described in Chapter 4. Our results in [11] demonstrate that more complex placement heuristics not only do not improve the performance of the algorithm in terms of the number of accepted placement requests, but also take longer to solve, limiting their applicability to larger networks.

1.4 Thesis Organization

The thesis document is organized as follows: Chapter 2 reviews the previously proposed mathematical models, and covers a review of a wide range of heuristics previously proposed for placing VNFs in wired and wireless networks. The basic, extended, and traffic-aware mathematical models that we design for NFEP in wireless multi-hop networks are described in Chapter 3. Chapter 4 describes our proposed heuristics, and Chapter 5 is dedicated to the results collected from deploying the mathematical model proposed in Chapter 3 and heuristics proposed in Chapter 4 and its analysis. Finally Chapter 6 is dedicate to our conclusion and plan for future works.

Chapter 2

A Review of NF Placement Models

A Service Chain (SC) is a chain of high-level services, where each service is composed of NFs [12]. A chain of NFs with predefined parameters is referred to as a Service Graph (SG). The placement of all NFs of an SG can be referred to as a Network Function Embedding Problem (NFEP) [12]. There are several types of methods to approach the problem of NF placement. This problem can be modeled by using mathematical methods or by designing a heuristic algorithm. In this chapter, we review mathematical models and heuristics formerly proposed for the NFEP in wired and wireless networks. By reviewing the proposed methods we can identify the parameters that are important in the design of the optimization model for wired and wireless networks that minimize the cost of used resources and maximize the number of accepted requests. The considered parameters for a wired and wireless network can differ due to differences in the network characteristics.

The exact mathematical models for solving the optimization problem can be different forms of Linear Programming (LP), Non-Linear Programming (NLP), etc. These methods are designed to achieve the best outcome in a mathematical model whose requirements are represented by linear or non-linear relationships. The constraints can be defined based on the limitations of the physical network and the NFs. The objectives are defined to optimize one or multiple parameters. The proposed mathematical models can be categorized into two groups based on their target network, which can be wired or wireless. For each category, we review mathematical methods and parameters considered in each model to be able to provide a comprehensive mathematical model for NFEP in wireless multi-hop networks.

Although mathematical models can provide an optimal solution they are complex and proven to be NP-hard [8]. They are not applicable to large networks and it is common to develop a heuristic algorithm. The heuristic algorithms are applicable to larger networks and they are mainly designed to achieve results close to the mathematical models with lower complexity and execution time. Following our review of mathematical models, we provide a review of a wide range of heuristics proposed for NFEP in wired and wireless networks. In our review we identify the potentially effective parameters that we will consider in the design of our own simple and time-efficient heuristics.

2.1 Mathematical Models of NFEP

The amount of work on NFEP in wired and wireless networks is considerable. The proposed optimizations can be categorized based on the parameters considered in the constraints and objectives of the optimization model. The parameters can be summarized in the following categories:

- Nodal resources: Including CPU, memory and storage.
- Bandwidth
- Delay: Including different kinds of delay such as propagation delay, transportation delay, etc.
- Energy Consumption: Including power consumption and the number of active nodes.
- Deployment and maintenance of NFs: Including the number of instances used to deploy services or the number of used licenses .

These parameters can be considered in the objective function or consider as a constraint. Table 2.1 summarizes some of the proposed methods based on the factors considered in their optimization model. The check mark means the related parameter is included in the model and an x means that the parameter is not considered. Nodal resources can be considered as a constraint, they usually are CPU, memory or the storage of the physical nodes which must be considered in the process of placement. Physical links' available BW is another parameter which can be considered in the constraints and the objective function. There is usually an effort in minimizing the BW usage in the placement of NFs. As we can see in Table 2.1, most of the proposed

Authors	Method	Nodal Resources	BW	Delay	Energy	Deployment Cost	Objective
[12]	ILP	✓	✓	✓	✗	✗	Minimizes the resources usage
[13]	MILP	✓	✓	✓	✗	✗	Minimizes maximum link and core utilization
[14]	ILP	✗	✓	✗	✗	✓	Minimizes the activation and maintenance cost
[15]	MILP	✓	✓	✗	✗	✓	Minimizes activation and maintenance cost
[16]	ILP	✓	✗	✗	✓	✗	Minimizes the resource-consumption and energy
[17]	ILP	✓	✓	✓	✗	✓	Minimizes the cost of NFEP
[18]	LRP	✓	✓	✗	✗	✓	Minimizes the installation and deployment cost
[19]	ILP	✓	✗	✓	✗	✓	Minimizes network provider's cost
[20]	MINLP	✓	✗	✓	✗	✗	Minimizes the maximum link's BW to load ratio
[21]	ILP	✓	✓	✗	✗	✗	Minimizes the resource usage
[22]	Knapsack Problem (MKP)	✗	✓	✗	✗	✗	Maximize revenue

Table 2.1: Mathematical Models for NFEP.

methods consider BW in their optimization. Some flows are delay sensitive and have a threshold for the maximum tolerable delay which can be defined as one of the constraints.

2.1.1 Mathematical Models of NFEP in Wired Networks

Sahhaf et al. in [12] consider the available resources of the nodes, the available bandwidth of the links and the requested QoS (Quality of Service) as constraints and minimize the resource usage. This is one of the mathematical models proposed that considers all of the nodal resources (CPU, memory, and storage) and links' BW in its model. In [13] the authors used Mixed Integer Linear Programming (MILP) to find an optimal solution. The proposed optimization is based on maximizing the number of services that can be supported in a switch. In this solution, the constraints are based on the number of free cores, tolerable delay of flows and links' bandwidth. The objectives are minimizing maximum link utilization and maximum core utilization, which leads to the distribution of load between available resources.

In the models where the users demand a service which should be deployed in the cloud environment, the model aims at minimizing the cost for the service provider or users. In these models, the main objective is to minimize the deployment and maintenance cost. Deployment cost can be translated into minimizing the number of instances used to deploy services or minimizing the number of licenses used. The physical network limitations such as available nodal resources and physical links' available BW can also be considered in these models. In [14], the authors used Integer Linear Programming (ILP) in order to find an optimum solution for placing Deep Packet Inspection (DPI) as a VNF in the network. In the proposed method, the objective function is to minimize the activation and maintenance cost of the virtual DPI (vDPI) and the considered constraint is the network's available bandwidth. In another mathematical model, Leivadeasa et al. proposed a MILP formulation with the nodal capacity and the bandwidth of the links as the constraints in [15]. It considers minimizing activation, maintenance cost, and load balancing among the resources as the objective function [15]. The model in [16] is based on an ILP which aims at minimizing the resource consumption and energy saving by turning off unused resources. In [17], the objective is minimizing the cost of VNF placement. VNF placement cost includes the cost of deploying VNF instances, using servers, and communication between servers. The constraints are defined based on the available resources of the physical nodes and also the delay threshold. The considered delay is the delay in delivering a service which consists of two components, the network communication delay and the VNF processing delay on the servers.

Ghaznavi et al. in [18] divided time into slots where the placement can be changed or modified in each slot. The static version of the proposed method generalizes to the NP-Hard Location Routing Problem (LRP). The objective is to reduce the cost of placing requested services at each time slot. The costs include the cost of VNF installation, BW usage, the penalty if the location of already placed request changed from one time slot to another, as well as the cost of migrating a set of VNF instances from one time slot to another. The constraints are defined based on the processing capacity of the physical nodes. The objective function of the model proposed by Luizelli et al. [19] aims at minimizing the number of virtual network function instances mapped on the infrastructure in order to minimize the network provider's cost. The constraints are processing capacity of the physical nodes and delay, which consists of an end-to-end delay and packet processing delay. Last but not least [20] considers the effect of each NF in changing the traffic volume which makes its proposed mathematical model non-linear. It uses MINLP, where some of the constraints and the objective function are non-linear. [20] Considers the effect of each NF in changing the traffic volume. As a result of processing a stream of packets, VNFs may change the bandwidth, some examples mentioned in the paper are: The Citrix CloudBridge WAN optimizer which may compress traffic to 20 percent, a Stateless

Transport Tunneling (STT) proxy adds 76 bytes to each processed packet due to the encapsulation overhead, and a firewall will keep the traffic rates of allowed flows unchanged and will reduce the rates of denied flows to zero. As the traffic changing factor can affect the BW usage we will consider it in our proposed model in the next chapter in order to have a comprehensive mathematical model.

In the next chapter, we describe how we used the mathematical model presented in [12] as a basis for our optimization model. Similar to our objective, the model presented in [12] specifically focuses on minimizing the deployment cost of general VNFs. Although some parameters in [12] are also considered in other reviewed models such as [13, 15–18], their focus is mainly on minimizing delay, load balancing, minimizing activation and maintenance cost, not minimizing the resource usage cost. Additionally, the proposed model in [12] is not specific to the type of VNFs such as the model presented in [14].

2.1.2 Mathematical Models of NFEP in Wireless Networks

The topic of NFV in wireless networks has received significant attention in the literature, where most of the focus is on wireless network virtualization. NFV introduced new possibilities to wireless networks such as decoupling of functionality in a networking environment by separating the role of the traditional Internet Service Providers (ISPs) into two: infrastructure providers (InPs), who manage the physical infrastructure, and Service Providers (SPs), who create virtual networks by aggregating resources from multiple infrastructure providers and offer end-to-end network services [23]. Specifically, network virtualization is a networking environment that allows multiple service providers to dynamically compose multiple heterogeneous virtual networks that coexist together in isolation from each other. Service providers can deploy and manage customized end-to-end services on those virtual networks for the end users by effectively sharing and utilizing underlying network resources leased from multiple infrastructure providers [24].

The amount of literature on wireless network virtualization shows the importance of NFV in wireless networks. However, there are only a few papers considering the problem of NFEP in wireless networks. Similar to wired networks, NFEP can be considered as an optimization problem that can be solved mathematically. The same parameters can be considered in wireless networks. However, the differences between wired and wireless networks must be considered. The main difference between wired and wireless networks is interference. Reviewing the works that have been done for NFEP in wireless networks we see that none of the proposed methods in wireless networks consider interference in their optimization model.

In [21] Riggio et al. discuss the virtual WiFi, where kernel-based virtual machines are used as a virtual wireless LAN device. Riggio et al. provide an integer linear programming model for placing VNFs in a hybrid wireless network where there are forwarding nodes, some with processing capacity, and some are access points. The objective of the model is to minimize the deployment cost of the NFs and considers nodal resources such as memory, CPU, storage, and BW usage as the constraints. In this paper, the optimization method is designed without considering the effect of interference. Its authors assumed that Orthogonal Frequency Division Multiple Access (OFDMA) is being used in order to handle the problem of interference. The principle of OFDMA is to divide the available subcarriers into several mutually exclusive groups (i.e., subbands) according to the subcarrier allocation strategies. Then each group of subcarriers is assigned to one user for simultaneous transmission. The orthogonality among subcarriers ensures that users are protected against interference [25]. However, the problem of interference cannot be solved completely by using OFDMA and there is ongoing research to tackle the problem of interference in an OFDMA wireless network such as [26], and [27].

In [22] Lv et al. consider the embedding of virtual wireless mesh gateways and the virtual links between them. The problem of interference between wireless links in this placement model has been solved by considering multi-radio multi-channel networks. The authors assign orthogonal channels to neighboring links. It is stated that the effect of zero-interference can be achieved at a lower physical distance by increasing the channel separation between two links. For instance, a channel distance of two (say Channels 1 and 3) is enough for both links to transmit without interference with a physical distance of about 50 m. [22] focuses on the problem of channel assignment to avoid assigning channels to the links that are in the interference range of each other.

To our knowledge, none of the papers considering NFEP in wireless networks included the effect of interference in their optimization model. In a wireless network, using a link will affect the adjacent links' available bandwidth and lowers their bandwidth. We mentioned in our review of the mathematical models for NFs' placement in wireless networks that there are methods for reducing the effect of interference but none of them can eliminate its effect on bandwidth consumption. This unique characteristic of wireless networks increases the bandwidth consumption. Scarcity of bandwidth makes it necessary to consider the effect of interference in an NFs' placement model.

2.2 NFV Placement Heuristics

Mathematical models can provide an optimal solution for the problem of NF placement, However, they are complex and proven to be NP-hard [8]. The alternative is to design a heuristic that can provide near-optimal solutions with less computational demand. Although there is (potentially) some performance loss between the heuristic algorithms and mathematical models, heuristic algorithms have an advantage in computational complexity when solving large-scale network optimization problems [28]. There exists a wide range of heuristics proposed for VNF placement. The proposed methods are designed based on one or more objectives. Here we review recently proposed heuristics that provide novel methods for mapping SGs' NFs to a physical network. We divide the reviewed heuristics based on their objective and compare them in terms of their objective, parameters they consider, and the performance of the proposed heuristic in terms of the cost of placing NFs and number of accepted requests. Our goal in the design of our heuristics is to provide algorithms with the lowest complexity possible that can place (ideally) as many requests as the mathematical model while reducing the resource consumption by SGs. The main parameters considered in heuristics can be categorized as follows:

- Resource usage: Including the Nodal and BW resource usage and their cost.
- Delay: Can be expressed in terms of different kinds of delays such as processing delay, propagation delay, end-to-end delay, etc.
- Load balancing: The avoidance of congestion, data overload, etc.
- Energy consumption: Translated to power consumption and the number of active nodes.
- Deployment and maintenance cost: Including the number of instances used to deploy services or the number of licenses used.

Authors	Method	Resource Usage	Delay	Load Balancing	Energy	Deployment Cost	Objective
[13]	Combines a heuristic with mathematical model	✓	✓	✓	✗	✗	Minimizes the maximum link and CPU core utilization and maximum delay of flows
[14]	Works based on centrality matrix	✓	✓	✗	✓	✓	Minimizes the number of activated nodes
[15]	Proposes 3 heuristic each place NFs based on one the objectives	✓	✗	✓	✓	✓	Minimizes the number of active nodes and balance the load between physical nodes
[17]	Works based on centrality	✓	✗	✗	✓	✓	Minimizes the overall cost of deployment
[18]	A combination of installation, migrations, and reassignments is applied to optimize the placement	✓	✗	✗	✓	✓	Minimizes the number of active nodes to reduce Energy and deployment cost
[20]	Places VNFs increasing BW close to destination and VNFs decreasing BW close to source	✓	✗	✓	✗	✗	Minimizes the maximum link load ratio on the flow path
[21]	Breaks the problem of placement into 3 parts	✓	✗	✗	✗	✗	Minimizes the resources usage
[29]	Dynamic Programming	✓	✗	✗	✗	✓	Minimizes the resources usage
[30]	Optimizes placement of each NF	✓	✗	✗	✗	✗	Minimizes the resources usage
[31]	Reduces the search space and then applies the mathematical model	✓	✗	✗	✗	✗	Minimizes the resources usage
[32]	Graph matching theory	✓	✗	✗	✗	✗	Minimizes the resources usage
[33]	Two phase algorithm that reuse already deployed VNF or a adds a new VNF	✗	✓	✗	✗	✓	Minimizes end-to-end delay
[34]	Stable Matching algorithm	✗	✓	✗	✗	✗	Minimizes delay between user and the deployed VNF
[35]	Breaks the problem of an SG placement into placing each NF and its link	✓	✓	✗	✓	✗	Minimizes cost, delay, Energy
[36]	Uses graph partitioning game	✓	✗	✓	✗	✗	Minimizes the nodal cost and balance the load between physical nodes.
[37]	Places each NF in nearest node	✓	✗	✗	✓	✓	Minimizes BW and number of active nodes
[38]	A sampling-based Markov approximation (MA) approach	✓	✗	✗	✓	✗	Minimizes both the BW consumption and operational cost
[39]	Calculates the minimum number of servers for placing all NFs	✗	✗	✗	✓	✓	Minimizes number of active nodes

Table 2.2: Heuristic Algorithms for Placement of SGs.

Table 2.2 summarizes the reviewed proposed heuristics based on the considered parameters. The checkmark means the related parameter is included in the heuristic and an x mark means that the parameter is not considered.

2.2.1 Minimizing Resource Usage

The first set of heuristics focus on minimizing the resource usage, consisting of both the use of nodal resources and BW usage. The proposed heuristic algorithm in [29] breaks the placement into smaller parts and optimizes placement of each NF. The objective of the proposed heuristic is to minimize the nodal resource consumption by VNFs and the BW consumed due to mapping virtual links between VNFs to one or more than one physical links. The authors use Dynamic Programming (DP) to organize the problem into smaller interdependent sub-problems of placing each VNF and the virtual link connected to it towards the next VNF. The solutions for the sub-problems are then aggregated to compose the overall chain placement. The results in [29] show the comparison of its method of dynamic programming with another similar approach that is using the multi-stage method in order to break the complex problem of service graph placement into simpler parts of placing each NF and the link connected to it. It is shown that both method's execution times are similar since both can find solutions in polynomial time. This heuristic only optimizes the placement of each NF, not the whole SG, which lowers the execution time but decreases the number of accepted requests. In the design of a heuristic algorithm it is important to achieve near-optimal acceptance rate. The near-optimal acceptance rate shows that although the heuristic algorithm solution is not optimal, it is assigning resources in a way that accommodates almost the same number of requests as the mathematical method and does not use resources wastefully.

The proposed heuristic in [21] is another example of a heuristic with the objective to minimize nodal resources and BW consumption that breaks the complex problem of NF placement into 3 simpler parts. First, it computes the list of physical node candidates for each VNF. Second, it sorts the NFs based on the number of candidates for placement in an increasing order i.e. VNFs with the smallest number of candidates are put at the top of the list. In the last step, the heuristic computes the placement cost of that VNF and its virtual link to the physical network and chooses the one with the lowest cost. Prioritizing the placement of NFs with lower options for placement and giving priority to the NFs that are harder to place can potentially improve the acceptance ratio. We use this method in one of our heuristics and show its effect on the number of accepted requests. While the proposed heuristic in [21] is designed for placement of VNFs in a wireless network, the effect of interference is not considered and instead, it is mentioned that a BW provisioning model must be used to indicate the available BW. Another disadvantage of this method is that the algorithm does not have a view of the whole SG in the process of placement. Instead, it focuses on placement of each NF and the link connected to it. The authors in [30] break the placement problem into placing NFs and connecting them. In the first step, the proposed heuristic places the NFs based on their resource demand. This heuristic gives priority to the NF with the highest demand and places it in the cheapest node of the network. The value of each node is obtained from a formula that considers the available resource capacity, the price for each resource unit, and the ability to connect to other nodes. The placed NFs then connect through the available shortest path. Although the proposed algorithm considers multiple factors in obtaining a node for placement of NFs, it does not consider the whole chain of NFs in its placement and may end up taking a path much longer than the shortest path from source to destination.

The proposed heuristic in [35] is one of the multi-objective algorithms whose main goal is to minimize the Operational Expenditure (OPEX). OPEX includes nodal and BW resource usage, penalties due to excessive propagation delay, and energy consumption that is related to the time a physical node is active and consuming power. The proposed heuristic in [35] breaks the problem of an SG placement into sub-problems of placing each NF of an SG and the link connected to the NF. The algorithm starts from the source node and considers all the nodes that are connected to it, including itself, for placing the first NF and chooses the one with the lowest cost. This process is repeated for all NFs of an SG and the links connected to them until they are all placed in the network. The authors showed that breaking the whole problem of placing a SG into sub-problems lowered the execution time in comparison to their mathematical model. However the number of accepted requests is much lower than the mathematical model specially as the number of VNFs per request increases.

The proposed heuristic in [31] can be combined with any mathematical model to reduce the search space and consequently the execution time of the mathematical model. The proposed heuristic focuses on reducing the BW consumption and narrows the target search space of VNF placement by introducing a smaller accessible scope to which the possible locations of VNFs are confined. The requests are categorized based on their source and destination. Nodes with the lowest sum of distance from source and destination are in the accessible scope of the request. The size of each accessible scope for each set of requests is proportional to the total traffic volume of those requests. It is shown that the size of the accessible scope will impact the time efficiency and performance of the NF placement. Considering all nodes to be in the accessible scope will not reduce the execution time but will provide the acceptance ratio of the optimization model. On the other hand, a very small accessible scope will decrease the execution time but also the acceptance ratio. This approach considers the whole SG and its source and destination. In the design of one of our heuristics we adopted this idea to narrow the search space. In our results, we compared the performance of our proposed heuristics with the accessible scope heuristic proposed in [31].

Following a completely different approach, [32] uses graph matching theory instead of modeling the problem by some form of linear programming (LP). The proposed heuristic in [32] considers links' BW and nodes' CPU, and works based on the similarity of two graphs by using adjacency matrices of the graphs. The authors solved the problem of VNF placement and chaining by use of an eigen decomposition of the adjacency matrices of the request and the hosting infrastructure graph. In this method, the host network and the requests are being considered as two weighted graphs, where the weight of each node in the physical network is its computational capacity and the weight of each link is based on its BW. The weight of each node and link in the SG is based on their requested resources. The goal here is to minimize the defined distance between the adjacency matrices of the two graphs. The authors showed that the algorithm can scale to thousands of nodes and links and to be insensitive to the number of requested NFs, and the connectivity in input graphs. However, the proposed method's performance has not been compared to the performance of a mathematical model to show the differences in terms of the acceptance ratio. Although the algorithm might be faster than the LP, the approximations used in different stages of the method would certainly lower the acceptance ratio. It is shown in [40] that the acceptance ratio of this method is lower than other similar heuristics. In addition, only one cost has been considered for the nodal resources, as in the graph matching theory we can only consider one weight for each node and one weight for each link.

2.2.2 Minimizing Delay

Delay is another important objective in design of a heuristic for placement of VNFs. For instance, virtualization in 5G will be very useful to resolve latency problems [41]. Some services such as ultra-Reliable Low-Latency Communication (uRLLC) services require low latency access between the remote server and the clients while running in a highly reliable environment to guarantee service continuity. The authors of [33] propose a two stage heuristic. The first phase considers reusing VNFs already placed in the cloud since hosting a new one incurs additional cost. The algorithm first checks VNFs that are already placed in the network and have the same functionality. Next, the algorithm selects only the VNFs with enough capacity. If there is no VNF with the same type already activated in the network, the algorithm passes to the second phase that consists of placing a new VNF instance in the hosts respecting capacity constraints. The second phase of the placement algorithm selects nodes within the shortest paths, tests all possibilities for placement, and chooses the one that satisfies QoS metrics (i.e., minimum end-to-end latency) in the selected nodes.

The proposed heuristic in [34] is another example of considering delay minimization in the placement of VNFs and focuses on minimization of delay between the user and the deployed VNF. The authors designed a heuristic based on the Stable Matching (SM) algorithm to solve the problem of VNF placement. The stable matching algorithm starts by creating two priority matrices for the two groups that we want to match. These matrices are created based on the latency. The lower latency is given more priority for both groups (the VNFs and the physical nodes). The matching is done according to the priority matrix, where the VNF wants to connect to the hosting device that is first on its priority list. The same case exists for physical nodes as they want to connect to the VNF that is first on their priority list [34]. The proposed algorithm then runs for all the VNFs and matches them to physical nodes until a stable matching is achieved. In the next stage, the local search is used to search locally for a solution with lower latency. The local search algorithm begins by picking two connected pairs of physical nodes and VNFs and checks whether the latency can be improved by changing the connections locally. The algorithm stops when no further improvement is possible. This algorithm only considers one part of the problem, which is the latency between the user and the first VNF. The provided results shows that the use of the local search algorithm improved the latency of the stable matching algorithm by less than one percent and is only adding complexity and increasing the execution time. On the other hand, it is stated in the results that the execution time of the proposed algorithm is less than the mathematical model but its performance is not compared with other similar heuristics.

[13] seeks to minimize the maximum link and CPU core utilization and the maximum delay of flows in the network. The authors of [13] propose different heuristics and compare their performance in terms of the maximum number of requests that can be placed in a network. Between the proposed heuristics only one, named heuristic-A, is not combined with a proposed mathematical model. Heuristic-A places requests one by one along the shortest path between source and destination. The other heuristics, heuristic-B, B+, B+COR, and C are all being combined with their mathematical model to reduce their execution time. As the proposed mathematical model in [13] aims at solving the placement problem for all flows at once, the flows are divided into the groups in their proposed heuristics. These heuristics start from the first group and solve the optimization problem for this group. Based on the solution, the problem is updated again and being solved for the next group. Heuristic B randomly groups the

requests, heuristic C is the same as B but also considers minimizing the number of used cores in the network. Heuristic B+COR sorts nodes in ascending order based on the number of flows passing through them. Less crowded nodes are selected first to distribute the load away from bottleneck nodes. It is stated that B+COR can place more requests in the network in comparison to other proposed heuristics. However, the authors did not compare the execution time of the models and only considered the maximum number of the flows that each heuristic can place in the same network. As we will show later in our results, the more complex heuristics combined with a mathematical model may bring better results in terms of the number of accepted requests but suffer from high execution time and therefore are not applicable to large scale networks.

2.2.3 Load Balancing

The main objective of the heuristic proposed in [20] is to minimize the maximum link load ratio on the flow path. [20] provides an algorithm that considers the NF placement, routing, and the traffic changing effect of NFs. The heuristic divides NFs of an SG into two categories: the ones that increase the BW usage (calling them expanding NFs) and the ones that decrease the BW usage (calling them the shrinking NFs). The Least-First-Greatest-Last (LFGL) algorithm proposed by the authors starts from the shrinking NFs and traverses the network from the flow source, and iteratively calculates the path with Minimum Maximum link load ratio (Min-Max path) to each node as in Dijkstra's algorithm. For each of the Min-Max paths found from the source node to the next node the LFGL places all of the shrinking VNFs until there is no more resource availability on that node, then searches for the next node with the Min-Max path. This process repeats until all shrinking VNFs are placed. The last node of each found Min-Max path is called a junction node. The same process applies to the VNFs that increase the BW usage but this time the process starts from the destination node and continues until all expanding VNFs are being placed. After the second stage finishes, the LFGL algorithm collects all the junction nodes, each corresponding to a different path. The proposed LFGL algorithm compares the maximum link load ratio of each path and selects a path with the minimum maximum link load ratio. LFGL heuristic assumes that we have a freedom to re-organize the NFs in the SG which is not realistic: the order of the NFs can not be changed in most cases.

The authors of [36] designed multiple multi-objective heuristics inspired by a graph partitioning game which breaks an SG into its VNFs. The proposed placement algorithm is broken into two stages. The main goal of each stage of the placement algorithm is available nodal resource, BW usage, or delay. The initial placement of every VNF of the SG can be random, based on the node that can host more types of VNFs, or the node with maximum nodal resource. Then at each iteration of the algorithm, a VNF is reallocated to a new node and the new cost is evaluated. The selection of a VNF to be relocated at each iteration can be random, or the VNF with highest nodal resource demand, or the VNF that depicts the highest latency communication with the next VNF in the SG. The algorithm is terminated after a maximum number of iterations is reached. The number of iterations can be tuned to provide a trade-off between solution quality and computation time. It is shown in the results that selecting the servers with the highest available capacity for the initial allocation leads to a lower deployment cost and faster convergence of the algorithm in comparison to the other parameters considered, such as latency [36]. Interestingly it is shown by the results that in the second stage of the algorithm a random permutation provides the best solution converging towards the optimal solution. However, even in that case, the execution time for a 50 nodes network with 500 or fewer iterations is in the order of 1000s of seconds, which is very high in comparison to the other reviewed heuristics and our provided heuristics.

2.2.4 Minimizing Energy Consumption

Another parameter to be considered as an objective in the design of a placement algorithm is the energy consumption. It can be minimized by using fewer nodes for placement or reducing the time the nodes are active. The proposed heuristic in [37] aims at minimizing the power consumption by taking a path with a minimum number of hops to power up less number of nodes. The heuristic algorithm places NFs one by one based on their order in the SG. The authors exploit the intuition of finding the nearest server which supports the first NF in the chain of NFs for each flow. Then the algorithm removes the VNF under consideration from the chain and finds the nearest server that supports the next VNF of the chain and so on. The proposed heuristic is fast and simple but only considers optimization for each NF, not the whole SG.

Another example of a heuristic that considers power consumption is the one proposed in [38]. The proposed heuristic in [38] uses a sampling-based Markov approximation (MA) approach to solve the NP-hard problem which requires a long convergence time. Its authors provided a heuristic that is considering both the BW consumption and operational cost, including the energy consumed by the activated physical nodes in the network and the wear and tear cost. The wear and tear cost indicates the effect of turning nodes into sleep/off mode and bringing them back to normal operation to the lifetime of physical devices. The proposed algorithm begins with a random feasible solution and iterates the process of transformation from the current solution to another feasible solution until the steady-state distribution of the Markov chain appears. To reduce the execution time, the solution space is reduced to a subset of randomly chosen nodes that satisfy the resource demands of a request. It is been stated that the problem can be solved in polynomial time but the execution time of the algorithm is not being reported or compared with other proposed heuristics with similar time complexity. Additionally, the subset of nodes could be chosen based on more sophisticated parameters to reach a near-optimal solution faster.

2.2.5 Minimizing Deployment and Maintenance Cost

The optimization of the number of active nodes can help considerably in cutting down the power consumption and deployment cost. The proposed heuristic by the authors of [39] finds the minimum number of nodes that should be used to serve all requests to reduce the OPEX. The authors assume that we have the information of all requests prior to their arrival and calculate the minimum number of servers for placing all NFs without considering the BW consumption. The algorithm starts from the first request and attempts to place all of its NFs in one node. If that fails, it adds another node that can host the NFs and so on to identify the minimum number of servers for placing all of the requests. The next step is selecting the minimum number of nodes for each request and placing the NFs in those servers. Finally, the algorithm deals with the chaining of NFs of the requests by assigning VNFs on the elected nodes and building a path between NFs of a request. The results gathered in [39] only compare the heuristic with the ILP model for a small network with a limited number of requests. They stated that, in a small network, the average cost of placement of both methods is close. BW is not being considered as a constraint and the only attempt to reduce BW consumption is to choose the shortest path to connect the servers containing the NFs of a request.

Three heuristics are being proposed in [15]. The first heuristic is a basic greedy VNF placement heuristic, where a random node is selected at each iteration and allocated as many VNFs of the SG as possible. The placed VNFs are then being connected by using the Dijkstra algorithm. The goal of the second heuristic is to minimize the number of servers that are used to place a SG. It iteratively checks the available servers and selects the server with the maximum available capacity that will allow the complete allocation of the SG in a single server. If this is not possible, it will use the second server with the maximum available computational capacity and so on. The third heuristic includes a load balancing policy during the placement of the VNFs. This load balancing algorithm selects not only the node with the highest available capacity but the node that also has a sufficient available bandwidth in its adjacent links to facilitate load balancing on the upcoming path selection between the VNFs of a SG.

The heuristic algorithm proposed in [14] works based on the centrality matrix and aims at minimizing the number of nodes which are activated to be used for placement of the VNFs. The centrality for each node in [14] is defined as the sum of the total size of flows which have their shortest path going through that node. The authors assume that the source and destination of all flows are known and determine the node that has the most total traffic volume passing through. The main goal of the algorithm is to activate fewer licenses of a certain type of NF. At each iteration, the algorithm compares the cost of activating a new VNF in the node with the highest centrality with the cost of the flow traversing through already placed VNFs and only activates a new one if it decreases the placement cost.

[17] is another example of a heuristic that works based on centrality. In this heuristic the server with the highest centrality is the potential node to host the VNF instances. As a result, it assigns the flows to the VNFs without deviating from their shortest path and without using additional network resources to lower the overall cost of deployment [17].

In another example, [18] proposes a solution called Simple Lazy Facility Location (SLFL) that optimizes the placement of VNF instances in response to on-demand workload. Upon new demand arrival, a combination of installation, migrations, and reassignments can be applied to optimize the placement. In each case, the cost of migrating the already deployed VNF, installing the new VNF instance, or adding to the already deployed VNFs are being compared with each other and the one with the lowest cost will be considered. Although it is mentioned that SLFL runs in polynomial time, its execution time is not reported. The performance of the SLFL is only being compared with an algorithm that places the NFs randomly and is not being compared with other similar heuristics that took the same approach or have the same time complexity to indicate the better performance of the proposed algorithm in terms of the number of accepted requests.

2.3 Summary

We reviewed a wide rage of mathematical models and show that the common approach is to use different forms of linear or non-linear programming to provide an optimal solution for embedding SGs. As can be seen from Table 2.1, common parameters to be considered are nodal resources and links' BW. In the design of our mathematical model we follow [12], defining constraints and objective functions that consider different types of nodal resources and links' BW for a wired network. However, as we are aiming to provide a mathematical model for wireless multi-hop networks, we define our model based on wireless network characteristics. In our model we give priority to minimizing the BW consumption as we are facing a scarcity of BW due to the presence of interference. In order to model BW usage, unlike the other embedding models provided for wireless networks, we consider all parameters that may affect BW consumption. For the first time, we model the effect of interference and consider it in both our constraints and the objective function. We consider the fact that each SG request is a flow that has a source and a destination and an optimal placement should consider these in order to minimize BW consumption. Another parameter that can affect BW usage is the effect of NFs on changing traffic volume. As mentioned in [20], there are NFs that can increase or decrease the traffic volume. We follow the approach proposed in [20] and consider the effect of NFs on changing traffic volume in our mathematical model's BW constraint and objective function.

We reviewed heuristics, as they can be a faster and less resource-demanding alternative to mathematical models. It can be seen from Table 2.2 that, similar to the mathematical models, different objectives and parameters are considered. There are a wide range of methods that can be used to reduce the complexity of the NF embedding problem. In the design of a heuristic model for wireless networks, the scarcity of bandwidth should be considered and given priority. One of the widely used heuristic methods, described in [21,30,35,37] breaks the problem of placing a SG into smaller sub-problems, placing each NF of a SG individually. Although this approach can provide fast solutions, it only provides near-optimal results for each sub-problem. This may not lead to a near-optimal solution for the whole SG. In order to minimize the BW consumption it is important to consider the whole SG at once, as the union of shortest paths between placed NFs may not result in a shortest path for the whole SG. Maintaining a balance between reducing the execution time and increasing the acceptance ratio is another factor that should be considered. We can not oversimplify our heuristic model and select nodes randomly without considering its impact on future requests and expect to reach a high acceptance ratio. One of the interesting methods we reviewed here is the one proposed in [31]. We will be using the idea behind their heuristic in our design to reduce the search space to nodes that are along the shortest paths between source and destination of a request. This reduction in the search space will decrease the execution time and also keep optimal placement options. We believe it would be beneficial to give placement priority to those NFs that have fewer candidate nodes for placement as it is mentioned in [21] and will consider this factor in our algorithm too. Our goal is to provide a simple, and low complexity, heuristic that can provide near-optimal results as fast as possible. We start from the simplest approach of placing NFs randomly and gradually add complexities to our heuristic algorithm and observe the effect of each added parameter on our results. We compare our heuristic algorithms against each other, our mathematical model and against [31] to show the improvement that we bring by our model. Our results of comparing our heuristics show that considering multiple parameters do not improve the performance of the algorithm in terms of the ratio of accepted requests, they take longer to solve and therefore are not applicable to larger networks. Interestingly, placing the NFs along their shortest path is the only parameter that is necessary to be considered and can find near-optimal solutions much faster than the other more complicated heuristics while keeping the number of accepted requests close to the results achieved with an NP-hard optimization model.

Chapter 3

Mathematical Models for NF Embedding Problem

In this chapter we describe our overall system, assumptions, and optimization models for placement of Virtual Network Functions (VNF). We start with describing our system and the problem we are aiming to solve. In the next section we describe the proposed model in [12] as a basic model for wired networks. Gradually we extend the basic model and provide a comprehensive placement model for VNFs in wireless multi-hop networks. To include the effect of interference in the BW consumption, we use an interference model widely used in the literature called the protocol model [42]. The final model also includes traffic related parameters. We use Integer Linear Programming (ILP) to solve the placement problem for the basic model and Integer Non-Linear Programming (INLP) to solve the placement problem for the traffic-aware model.

3.1 System Overview

Prior to introducing our placement model, we briefly review the VNF chaining and placement system and the interactions among the different components of the system. We base our approach on the system proposed in [3] that uses a centralized SDN controller. A centralized SDN controller allows viewing the network as a global entity and provides programmable forwarding rules that reduce the complexity of service chaining enforcement [3]. This system in the end places VNFs on the hypervisors and installs the flow rules on the SDN switch to steer flows to VNFs. Figure 3.1 depicts this approach. The admin specifies the SG request. The Service Layer Agreement

(SLA) Manager translates the logical policies into machine-readable form to be further processed by the Control Application. Monitoring the network and collecting the state information such as switch connection information and link utilization are duties of the Control Application. The Control Application manages the Orchestrator and Flow Manager.

The Orchestrator runs the placement model/algorithm that determines the location of VNFs. The topological information for the Orchestrator are provided by the Topology Manager. The Flow Manager obtains the locations for VNF placement from the Orchestrator and steers the traffic from the source to the destination. The flow rules are then installed on the switches such that the flow traverse all VNFs of its service chain (in the correct order).

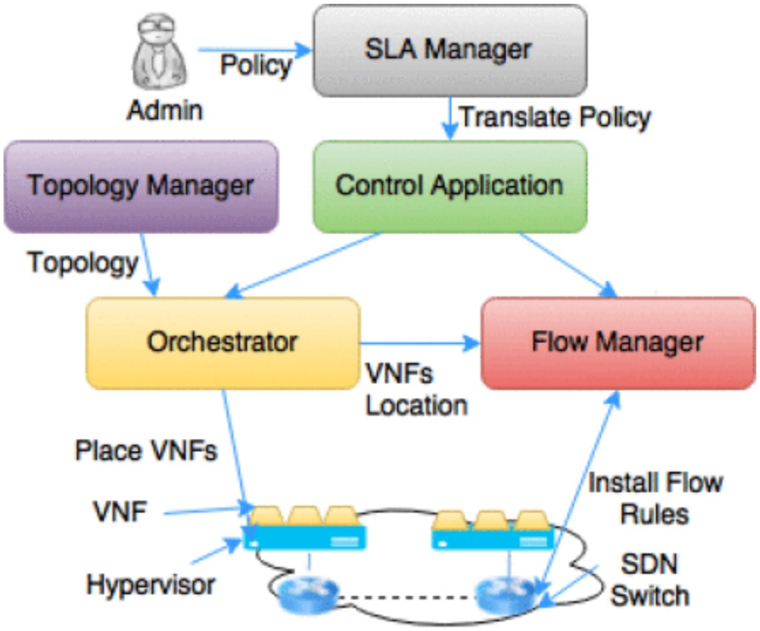

Figure 3.1: Overall System [3]

3.2 Problem Definition and Assumptions

Our work is focused on the problem of VNF placement. In all of our proposed mathematical models and heuristics, it is assumed that SG requests arrive one at a time and are placed separately. Each SG request has a specific nodal resource demand for each VNF and a BW demand for all virtual links. If a VNF is mapped to a node in the physical network it means the demanded nodal resources for that VNF are provided by that physical node and the same applies to the links in the SG. If a link of the SG is being mapped to one or more physical links, its BW demand is being provided by the mapped physical links. Each request has a duration, once an accepted request expires, it will be removed from the network and the associated used resources will be released. The physical network is being considered as a connected graph of nodes. The initial available nodal resources of each node, the initial available links' BW, and the topology of the network are known (or discoverable by the Topology Manager in Figure 3.1, for example). After placement of each SG, the available resources are updated before we place the next arriving SG.

The objective of our mathematical models and heuristics is to minimize the mapping cost based on the requirements of the VNFs and available resources in the network. The mapping cost is calculated based on the cost of consumed resources by the SG in the physical network which includes [12]:

- The costs associated with consuming nodal resources such as CPU, memory, and storage, used by VNFs in physical nodes.
- The cost associated with consuming bandwidth, used by virtual links that interconnect the VNFs, in the physical network.

Furthermore, we assume that the costs per unit of nodal resources or BW are independent of which link or node we use and are known.

3.3 Basic Model

The basic model is similar to the proposed model In [12], that formulated the NFEP as an optimization problem which can be solved with Integer Linear Programming (ILP). As service requests arrive over time, the embedding algorithm decides where to place the NFs in the physical network subject to various constraints. Each request has an associated duration. If the request is accepted, the required resources will be assigned and when the request expires the used resources will be released. We are using ILP as the optimization method. ILP consists of two parts, an objective function and constraints. ILP will choose the mapping that satisfies all of the constraints and minimizes/maximizes the objective function. In this section, we define the variables, constraints, and the objective function similar to the optimization method in [12] and then introduce the extension of the model and the added constraint for multi-hop wireless networks.

3.3.1 Input Parameters

- Sets
  - N_p, set of physical nodes where u is representing node $u \in N_p$.
  - L_p, set of physical links where $E_{uv} \in L_p$ is representing the physical link connecting node u to v.
  - N_f, set of NFs where $i \in N_f$ represents NF i in flow f.
  - L_f, set of virtual links between NFs of flow f, where $e_{ij} \in L_f$ represents the virtual link which connects NF i to j.
- Constants
  - f, representing the current flow that consists of a set of requested NFs with required resources.
  - C_u, available processing units in physical node u.
  - $c_{f,i}$, requested processing units for NF i of flow f.
  - w_c, the cost of consuming each unit of processing resources.
  - M_u, available memory units in physical node u.
  - $m_{f,i}$, requested memory units for NF i of flow f.
  - w_m, the cost of consuming each unit of memory.
  - S_u, available storage units in physical node u.
  - $s_{f,i}$, requested storage units for NF i of flow f.
  - w_s, the cost of consuming each unit of storage.
  - $BW_{E_{uv}}$, available BW over the physical link between node u and v.
  - $bw_{f,e_{ij}}$, requested BW for the link that is connecting NF i to NF j in flow f.
  - w_{bw}, the cost of consuming each unit of BW.

- Decision Variables
  - $x_{f,i,u}$, a binary variable where one means that function i from flow f is placed in physical node u.
  - $F_{f,e_{ij},E_{uv}}$, a binary variable which is equal to one when the virtual link between NFs i and j is mapped to one or more physical links and physical link E_{uv} is one of them. In the case of mapping a virtual link to multiple physical links all the related variables must be set to one.

3.3.2 Objective Function

As mentioned before, the objective is to minimize the placement cost. The cost consists of cost of resources that are used in the physical network, which include the cost of nodal resources (processing, memory, and storage) and the cost of consuming links' BW. The first part of the objective function in (1) considers the cost of consuming nodal resources and the second part the cost of consuming BW. (2) is a more detailed version of the objective function in (1), expressing the same objective function in terms of the notation introduced earlier.

$$min \sum_{u \in N_p} \sum_{i \in N_f} cost(i,u) + \sum_{E_{uv} \in L_p} cost(f, E_{uv}) \tag{1}$$

$$\begin{aligned} min \sum_{u \in N_p} \sum_{i \in N_f} (w_c * c_{f,i} + w_s * s_{f,i} + w_m * m_{f,i}) * x_{f,i,u} + \\ \sum_{E_{uv} \in L_p} \sum_{e_{ij} \in L_f} (w_{bw} * bw_{f,e_{ij}} * F_{f,e_{ij},E_{uv}}) \end{aligned} \tag{2}$$

3.3.3 Constraints

Constraints are sets of equalities and inequalities which are defined based on the conditions the optimization model must satisfy. Over-assignment of the physical resources will be prevented by the constraints. The first three constraints ensure that the summation of processing, memory and storage units of the placed NFs do not exceed each node's available resources.

$$\sum_{i \in N_f} c_{f,i} x_{f,i,u} \leq C_u, \forall u \in N_p \tag{3}$$

$$\sum_{i \in N_f} m_{f,i} x_{f,i,u} \leq M_u, \forall u \in N_p \tag{4}$$

$$\sum_{i \in N_f} s_{f,i} x_{f,i,u} \leq S_u, \forall u \in N_p \tag{5}$$

Constraint (6) prevents over-assignment of bandwidth in each physical link.

$$\sum_{e_{ij} \in L_f} bw_f F_{f,e_{ij},E_{uv}} \leq BW_{E_{uv}}, \forall E_{uv} \in L_p \tag{6}$$

Each virtual link between the NFs can be mapped to one or more than one of the physical links. In case a set of physical links connected to each other are chosen to connect two NFs, Constraint (7) assures all the related physical links are chosen.

$$\sum_{E_{uv} \in L_p, u=src} F_{f,e_{ij},E_{uv}} - \sum_{E_{uv} \in L_p, u=dst} F_{f,e_{ij},E_{uv}} = x_{f,i,u} - x_{f,j,u} \tag{7}$$

$$\forall e_{ij} \in L_f, \forall u \in N_p$$

Last but not least each NF should be placed in the physical network once.

$$\sum_{u \in N_p} x_{f,i,u} = 1, \forall i \in N_f \tag{8}$$

3.4 Extended Model for Wireless Networks

The basic model is designed for wired networks. In order to extend the model to be applicable to wireless networks, a couple of changes must be made in the constraints and objective function. The BW usage of wireless links is different from wired ones. In a wired network, it is sufficient to require that the summation of required bandwidth for the mapped virtual links should not exceed the physical link's bandwidth. In multi-hop wireless networks, where nodes share access to a common shared channel, using each link will affect the adjacent links' available bandwidth. In order to consider this effect, we have to model the interference between wireless links. We use the interference model in [42] and [43] and redefine the constraint for wireless links.

The interference in wireless networks can be modeled based on either the protocol or the physical model. Each of these models defines conditions for a successful transmission in the wireless network [43]. In our optimization model, we used the protocol model. We assume that in case of a single wireless channel, d_{uv} expresses the distance between nodes u and v, and all nodes have the same identical transmission range R. With these assumptions, the transmission from node u to node v is successful if the following two conditions are satisfied:

- $d_{uv} \leq R$
- Any node k, such that $d_{ku}, d_{kv} \leq R$, is not transmitting.

These two conditions imply that transmission in the link between node u and v will affect the BW usage of all the links whose transmitter is within transmission range of the sender or the receiver. To formulate this as one of the constraints in the optimization, an interference set has been defined for each link. It consists of all the links that are connected to the nodes in the transmission range of the sender or receiver.

$$intset_{E_{uv}} = \{E_{u'v'} | d_{u'u} \vee d_{v'v} \vee d_{v'u} \vee d_{u'v} \leq R\}, \forall E_{uv} \in L_p$$

Then for the bandwidth constraint, instead of Constraint (6), we have:

$$\sum_{e_{ij} \in L_f} bw_{f,e_{ij}} * F_{f,e_{ij},E_{uv}} +$$

$$\sum_{e_{ij} \in L_f} \sum_{E_{u'v'} \in intset_{E_{uv}}} bw_{f,e_{ij}} * F_{f,e_{ij},E_{u'v'}} \leq BW_{E_{uv}} \tag{9}$$

Also the objective function changes to the following term:

$$\sum_{u \in N_p} \sum_{i \in N_f} (w_c * c_{f,i} + w_s * s_{f,i} + w_m * m_{f,i}) * x_{f,i,u} +$$

$$\sum_{E_{uv} \in L_p} \sum_{e_{ij} \in L_f} w_{bw} * (bw_{f,e_{ij}} + \sum_{E_{u'v'} \in intset_{E_{uv}}} bw_{f,e_{ij}}) * F_{f,e_{ij},E_{uv}} \tag{10}$$

3.5 Traffic-Aware Model

The NFs are traffic processing devices and with the help of NFV, we can place the NFs in the path of traffic flows. Traffic patterns in the network can affect the optimum placement of the NFs. If we do not place NFs carefully, many flows get re-routed to find nodes with enough capacity to host the required NFs, which will lead to excessive BW usage.

3.5.1 Adding Source and Destination

The first step to expand the model to place functions based on traffic pattern is to consider source and destination for each request. The added constraints will place the NFs and find an optimum path from source to destination at the same time. The model assumes that flows follow physical links until they hit the SG.

Added input parameters

The following parameters will be added to the model.

- Constants
  - bw_f, instead of a BW request for each virtual link $bw_{f,e_{ij}}$ we assume each flow has a BW request which is the same for all of its virtual links.
  - s_f, the source of flow f.
  - d_f, the destination of flow f.
- Decision Variables
  - $X_{f,u}$, a binary variable where one means flow f traverses physical node u.
  - $F_{f,E_{uv}}$, a binary variable where one means flow f passes through the physical link from node u to v. $F_{f,E_{uv}}$ is related to $X_{f,u}$ which will be described in constraints.

Added Constraints

The following constraint, commonly referred to as flow-balance constraint, assures each flow starts from its source node, ends at its destination node, and only follows one path. To assure that the flow starts from the source and ends at the destination node there must be one extra outgoing physical link from the source node and one extra physical link toward the destination node. The number of outgoing and incoming physical links must be the same for intermediate nodes.

$$\sum_{v \in N_p} F_{f,E_{uv}} = \sum_{v \in N_p} F_{f,E_{vu}} + l$$

$$l = \begin{cases} 0 & \forall u \in N_p - \{s_f, d_f\} \\ 1 & u = s_f \\ -1 & u = d_f \end{cases} \tag{11}$$

The next set of constraints assure that the nodes the flow traverses are consistent with the physical links that were chosen for the path from source to destination.

$$\sum_{u' \in N_p} F_{f,E_{uu'}} + \sum_{v' \in N_p} F_{f,E_{v'v}} = X_{f,u} + X_{f,v}, \forall E_{uv} \in L_p \tag{12}$$

The following constraint selects the source node based on the source specified in the request.

$$X_{f,s_f} = 1 \tag{13}$$

Constraint (14) shows the relation between $x_{f,i,u}$ and $X_{f,u}$. It shows that a physical node can be in the path of a SG placement but it may or may not contain a NF.

$$x_{f,i,u} \leq X_{f,u}, \forall i \in N_f, \forall u \in N_p \tag{14}$$

The BW constraint will change to the following constraint:

$$bw_f F_{f,E_{uv}} + \sum_{E_{u'v'} \in intset_{E_{uv}}} bw_f F_{f,E_{u'v'}} \leq BW_{E_{uv}} \tag{15}$$

The changed terms in the BW constraint are only $F_{f,E_{uv}}$ which was $F_{e_{ij},E_{uv}}$ in Constraint (9). This is due to the fact that in this model we know which links have been used for the path from source to destination, therefore, the same variable can be used to calculate the used BW and there is no need to consider $F_{e_{ij},E_{uv}}$ as an extra decision variable.

3.5.2 Traffic Changing Factor

NFs may change the volume of processed traffic and may do it in different ways. As we mentioned earlier in Chapter 2, the Citrix CloudBridge WAN optimizer [44], and Stateless Transport Tunneling (STT) proxy [45] are examples of NFs that impact the traffic volume. Here we introduce the variables and constraints that should be considered in the mathematical model to capture this aspect. The added parameters and constraints change our linear model to a non-linear programming model.

Added Input Parameters

The following parameters must be added to the model in order to consider the traffic changing factor $alter_i$.

- $alter_i$, traffic changing factor for NF i. The factor can be a positive or a negative value. The positive value models NFs which increase the traffic rate and a negative one models NFs which decrease the traffic rate.
- t_{f,u^-}, a real variable representing the traffic rate factor before node u.
- t_{f,u^+}, a real variable representing the traffic rate factor after node u based on the NFs that were placed in that node. If v is the next node in the path toward destination node d_f we have: $t_{f,u^+} = t_{f,v^-}$
- $bw_{f,E_{uv}}$, a positive real variable representing BW usage of flow f over the physical link E_{uv}.

Added Constraints

Each NF will have the requested resources and a traffic changing factor $alter_i$ which changes the traffic rate after the flow passes the NF i. The value of $alter_i$ shows the percentage it increases or decreases the traffic rate. $alter_i$ can have a positive value if a NF increases the traffic rate and between -1 and 0 if it decreases the traffic rate. The BW after NF_i is equal to the multiplication of BW before $NF\ i$ and $(1+alter_i)$. With considering the traffic changing factor, the BW demand of a virtual link depends on the placement of the NFs and is equal to the multiplication of $(1 + alter_i)$ of all placed NFs before that virtual link in the flow.

To calculate the BW demand for each virtual link we added two variables. We assumed two physical nodes u and v are used consecutively for placing the NFs of flow f. The t_{f,v^-} is the traffic rate factor before entering node v which will be defined based on the traffic rate factor after the node u, t_{f,u^+}. Constraints (16) and (17) shows the mathematical form of the traffic rate factor before and after each node.

$$t_{f,v^-} = \sum_{u \in N_p} F_{f,E_{uv}} t_{f,u^+} \tag{16}$$

$$t_{f,v^+} = \sum_{u \in N_p} (F_{f,E_{uv}} t_{f,v^-} \prod_{i \in N_f} (1 + x_{f,i,v} alter_i)) \tag{17}$$

Constraint (18) uses Constraint (16) and (17) to determine the BW usage of flow f over each link in the physical network.

$$bw_{f,E_{uv}} = bw_f t_{f,u^+} \tag{18}$$

These three newly added constraints will change the Constraint (15) for link's BW to the following inequality.

$$bw_{f,E_{uv}} F_{f,E_{uv}} + \sum_{E_{u'v'} \in intset_{E_{uv}}} bw_{f,E_{u'v'}} F_{f,E_{u'v'}} \leq BW_{E_{uv}} \tag{19}$$

Chapter 4

Placement Heuristics

The main focus of the reviewed heuristic models in Chapter 2 for embedding VNFs into a physical network is on designing a heuristic that can achieve near-optimal results. However, the main reason for avoiding mathematical optimization models is their complexity and high execution time. Here we aim at exploring the level of complexity that is required and benefits we can achieve with simple heuristics. Additionally we consider the characteristics of wireless networks in our design. Compared to wired networks, multi-hop wireless networks such as MANETs, VANETs, or wireless sensor networks suffer from severe BW limitations. That is due to a number of reasons: typical wireless technologies operate at lower transmission rates, compared to wired technologies such as Ethernet, etc. Also, when multi-hop wireless networks are built up from devices using a single radio, flows interfere with themselves (a node that is a relay between source and destination can only either receive or transmit, but not both at the same time). Finally, wireless technologies typically experience significant interference (either from other flows or due to the above self-interference), lowering the available BW for each link. We therefore emphasize on minimizing the BW consumption while placing as many requests as possible.

In this chapter we study a sequence of five, increasingly complex, heuristics. Each heuristic is designed to show the effect of the added parameter on the acceptance rate, the average cost of the resources consumed by a SG, and the execution time of the algorithm. We start from the simplest heuristic and in each heuristic consider an additional parameter in the NFs' placement. In all heuristics, it is assumed that requests arrive one at a time and are placed separately. Each SG request has a specific source and destination, nodal resource demand for each NF, and a BW demand for all virtual links. The physical network consists of nodes that have nodal resource and links that have available BW. We only consider a single nodal resource and its consumption. The nodal resource could represent memory, storage, or CPU resources of a physical node. The model is designed for a wireless multi-hop network and considers the effect of interference in BW consumption. As we described in our extended model in Chapter 3, we use an interference model widely used in the literature called the protocol model [42] to include the effect of interference in the BW consumption.

4.1 Random Placement

The first heuristic is the simplest algorithm that can be used for embedding the NFs of an SG into a physical network. The random placement heuristic can be divided into two parts: placing the NFs and connecting the NFs. To place a NF, the algorithm randomly chooses a node that has sufficient nodal resources. In this stage, if there is no node with sufficient nodal resource for any of the NFs the request will be rejected. If all NFs are placed successfully, the algorithm moves to the connecting stage. The algorithm starts from the source and connects it to the node used for placement of the first NF of the SG via Dijkstra algorithm. The Dijkstra algorithm finds a shortest path in terms of the number of hops from source to destination. This process of connecting nodes continues until the node that contains the last NF of the SG connects to the destination. As the path from source to destination that passes all the NFs based on their order is identified, the availability of the BW will be checked. The BW consumption is based on the summation of the BW consumed by passing each link and due to interference.

To describe this heuristic and the next ones better, consider the following example of an SG placement that will be solved by all of the heuristics. Assume we have a wireless multi-hop network with six nodes. Nodal resources of the nodes in the wireless network are $C_n = \{12, 11, 7, 20, 14, 8\}$ units, and the available BW of the links in the physical network are all 20 units. The nodal resource demand of the SG of 3 NFs is $c_f = \{4, 1, 4\}$ and the BW demand of the request is 2 units. The SG's source is node 1 and its destination is node 5. Figure 4.1 shows the topology of the physical network and random placement of the SG. As can be seen, although all three NFs could have been placed in the source node and the shortest path between source and destination is only two hops, the length of the chosen path is six hops which consumes significantly more BW.

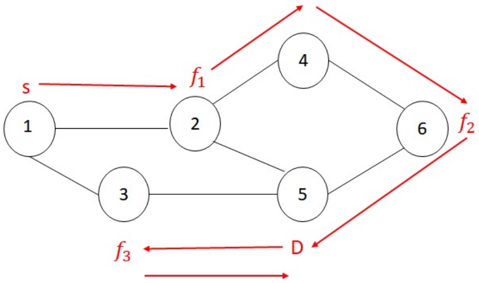

Figure 4.1: Random Placement Example

4.2 Shortest Path Placement

The presence of interference in wireless multi-hop networks causes a scarcity of BW. In order to reduce BW consumption, our second heuristic first finds a shortest path between source and destination of the SG request with the use of Dijkstra's shortest path algorithm. The shortest path then will be checked for availability of sufficient BW to accommodate this flow, considering both the actual links used and the impact on adjacent links due to interference. Should any link exceed their available BW the request gets rejected. NFs then will be placed along the shortest path. The shortest path placement places the first NF in the first node of the path that has sufficient nodal resources. The next NF of the SG will be placed in the same node as the previous NF if possible, otherwise it will be placed in the next node along the shortest path with sufficient available nodal resource. This process continues until all NFs are being placed. If any of the NFs can not be placed due to running out of nodes with sufficient nodal resources, the request gets rejected. Figure 4.2 shows the shortest path placement applied to the same example as the previous section. We can see from the example that this model reduces the BW consumption in comparison to the random placement.

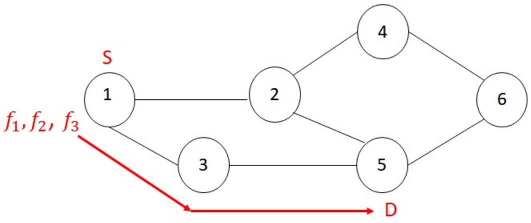

Figure 4.2: Example of Shortest Path Placement

4.3 All Shortest Path Placement

The problem with picking a single shortest path, as was done before, is that this path may not have sufficient resources, so exploring other shortest paths may allow us to still place a request. In the all shortest path heuristic we search for all shortest paths between the source and destination of the request in the network and choose the one that has maximum-minimum nodal resource to increase the probability of accepting a request. The search for all shortest paths is performed by using a search method similar to Breadth First Search (BFS). BFS explores the links of the graph to discover the node that is reachable from the source node. It computes the shortest distance (in terms of number of hops) from the source to each reachable node in the graph. We modified BFS to start from the source and end when it reaches the destination node. Also, in addition to the distance, we record the shortest paths themselves. In our search for shortest paths we define an array and a matrix for each node u in the physical network:

- $Dist_u$: An array that represents the shortest distance in terms of the number of hops from the source node.
- $Nodes_u$: A matrix which records nodes involved in each different shortest paths found from source node to node u.

The initial value of $Dist$ for all nodes is infinity, except for the source node which is equal to 0. The initial matrix of $Nodes$ for all nodes is empty. The search algorithm starts traversing the physical network graph and while visiting neighbor v of node u it compares the value of $Dist_v$ with $Dist_u + 1$. If $Dist_v$ is greater than $Dist_u + 1$ it means that $Dist_v$ describes a path longer than the shortest path. So we decrease $Dist_v$ to $Dist_u + 1$ and assign $Nodes_u$ to $Nodes_v$. If $Dist_v = Dist_u + 1$ then it means we found another shortest path to node v. In this case $Nodes_v$ is the union of $Nodes_u$ and $Nodes_v$. The pseudo-code of this search algorithm is presented in Algorithm 1. This algorithm can find all possible shortest paths for each pairs of nodes. The output of the algorithm is $Nodes_d$, which includes all shortest paths between the source node s, and destination node d.

Algorithm 1: Finding all shortest paths

```
Result: $Nodes_d$ that contains all shortest paths from source to destination,
        and $Dist_d$ set to shortest distance from source to destination
Graph $G(V, E)$, undirected, vertices $s$ and $d \in V$;
$s$ is the source node;
$d$ is the destination;
$Q$ is the queue data structure that has a list of the nodes that should be
  visited.;
$\forall u \in V$: $Dist_u = \infty$;
$\forall u \in V$: $flag_u = 0$;
$Dist_s = 0$;
$Q = [s]$;
while $Q \neq \emptyset$ do
    $u = Dequeue(Q)$;
    if $flag_u = 0$ then
        for $all\ edges\ (u, v) \in E$ do
            if $Dist_v > Dist_u + 1$ then
                $Dist_v \leftarrow Dist_u + 1$;
                $Nodes_v \leftarrow Nodes_u$;
            else if $Dist_v = Dist_u + 1$ then
                $Nodes_v \leftarrow [Nodes_v; Nodes_u]$;
            $Enqueue(Q, v)$;
        end
        $flag_u = 1$;
    if $u = d$ then
        Break
end
```

The following example demonstrates how we update the parameters of each node during the search for all shortest paths. As shown in Figure 4.3, we consider a network of 6 nodes and want to find all shortest paths from node 1 to 5. In the first step, we update the parameters of the source node's neighbors, which are nodes 2 and 3. Figure 4.3 shows the first step and updated parameters of the neighbors of node 1 and Figure 4.4 shows the second step, after we updated the parameters for neighbors of node 2. In the final step, when processing node 5, $Dist_5 = Dist_3 + 1$. So we update $Nodes_5$ to the union of $Nodes_3$ and $Nodes_5$. Figure 4.5 shows the final step and all

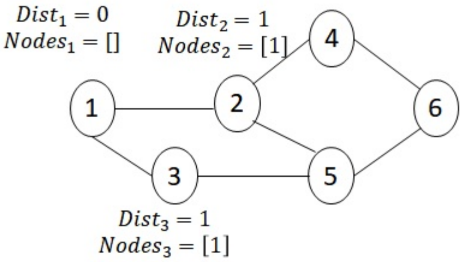

Figure 4.3: First Step

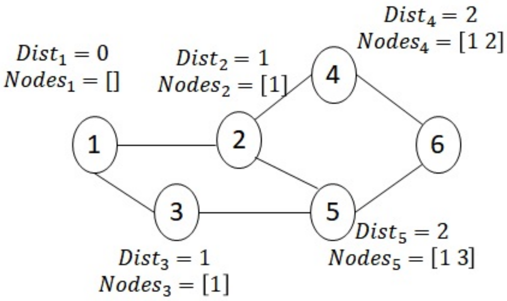

Figure 4.4: Second Step

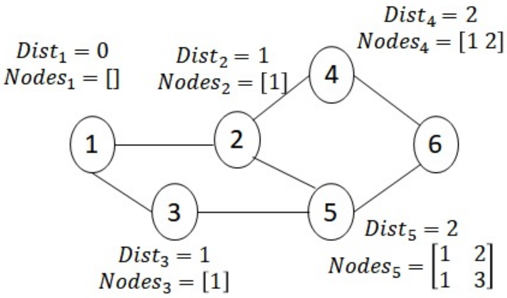

Figure 4.5: Final Step

shortest paths from 1 to 5 can be found in $Nodes_5$.

We applied this all shortest path heuristic to the same example in the previous sections and the resulting placement is shown in Figure 4.6. Between the source and destination of the SG there are two shortest paths $\{1, 3, 5\}$, and $\{1, 2, 5\}$. The maximum-minimum resource belongs to the second path. All NFs are placed in the source node as the policy here is to place them in the first node that has sufficient nodal resources.

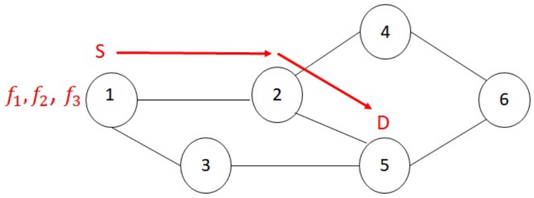

Figure 4.6: All Shortest Path Placement Example

4.4 Fast and Cost-Efficient Placement Algorithm (FACE)

Our next heuristic, called FACE (Fast and Cost-Efficient Placement Algorithm) heuristic builds on the previous heuristic in choosing a shortest path. Additionally, the FACE algorithm keeps a decreasing list of shortest paths based on their maximum-minimum nodal resource. In each stage of the placement process, if the placement was not possible for the chosen shortest path, it tries the next shortest path in the list. The following strategy will be deployed to place the NFs along the chosen shortest path.

In the FACE heuristic, the NFs are sorted based on the number of possible candidate nodes in an increasing order to give priority to the NFs that are harder to place and have fewer candidates for their placement. A candidate node parameter $candid_i$ is defined for each NF of the SG i and is equal to the number of nodes along the shortest path that can be used for the placement of that specific NF. In choosing the nodes along the shortest path we consider two parameters: a node has to provide sufficient nodal resources, and the NFs that previously were placed. The order of the NFs in the SG is fixed and we can not re-organize them. Furthermore, we do not want to have a placement that passes a physical link more than once. E.g. if the third NF of the SG is being placed in the second node of the shortest path, subsequent NFs in the SG can not be placed in the first node. The candidate nodes are being chosen based on the placement of previous NFs to avoid loops and backtracking in the placement. If there are no candidate nodes for any of the NFs at any stage of placement, the chosen path is infeasible and the placement process will choose the next shortest path with maximum-minimum nodal resource and repeat the process of NF placement.

To place the chosen NF in one of the nodes along the shortest path we sort its candidate nodes based on their index difference and choose the node with the lowest index difference. The index of the nodes along the shortest path is equal to their order in the shortest path e.g. the source node's index is one. The index of a NF is equal to its order in the SG, e.g. the index of the first NF of the SG is one and the index of second NF is two. We compare the index of the chosen NF with the index of the candidate nodes and choose the one with the minimum index difference with the chosen NF. In the end, the available resources of the nodes, BW of the links, and the list of candidate nodes for the remaining NFs will be updated.

We applied FACE heuristic to the same example as previous sections. Figure 4.7 shows the result of placement by FACE heuristic. As all of NFs had three options for placement the algorithm started from the first NF and placed it in the node with minimum index difference which is the source node. The second NF is placed in the second node of the chosen shortest path as it has minimum index difference and the third NF is placed in the third node of the chosen shortest path.

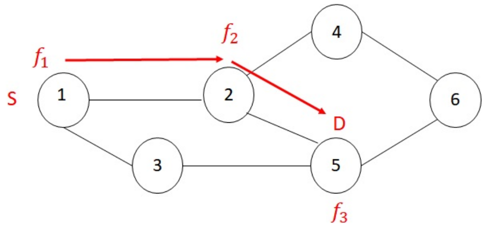

Figure 4.7: FACE Heuristic Placement

Our FACE heuristic is fast in comparison to our mathematical models and other related heuristics that we compare it to in the next chapter. Our results presented in Chapter 5 show that, in the exact same environment and for the exact same set of problems, FACE can provide a min-cost solution much faster than mathematical models and related heuristics. However, our results also show that simpler heuristics, such as the shortest path heuristic, is even faster than the FACE heuristic.

4.5 Joint Heuristics

Finally, our joint heuristic is another avenue that can increase the performance of any NF placement algorithm. The joint heuristic considers the possibility that reconfiguration of previously placed requests might enable us to place the current request. The joint heuristic can be combined with our mathematical models or our previously proposed heuristics. Here we describe the joint mathematical model and the joint heuristic combined with one of our heuristics.

4.5.1 Joint Mathematical Model

The joint heuristic can be combined with any mathematical model including our mathematical models described in Chapter 3. The joint mathematical model uses the same ILP described in Chapter 3 with some modifications. It considers not only the current SG but also already placed SGs in the network. Each time a new SG arrives, it will be added to the SGs currently deployed in the network (i.e., SGs that were previously admitted and have not yet expired). We then solve this joint placement problem to achieve a min-cost solution. The new solution may lead to a reconfiguration of already placed NFs.

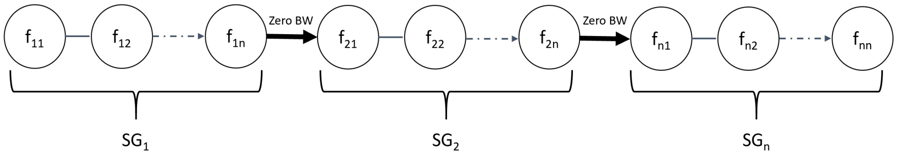

Figure 4.8: The Joint Request

The input of the joint optimization model is not the arriving request but the joint request f' which includes all of the placed and not expired requests and the current request connected to each other with virtual links with 0 BW demand. Figure 4.8 shows the combination of the SGs where the thick black arrows are the added virtual links that connect SGs to form a joint request. The constraints and objective functions are similar to the ones described in Chapter 3. Here we describe the input parameters but avoid repetition of the constraints and objective function as they are exactly the same as the extended model for the wireless networks.

- Input Parameters
  - Sets
    - $N_{f'}$, set of NFs where $i \in N_{f'}$ represents $NF\ i$ in the joint request f' which includes all previously placed flows and the current flow.
    - $L_{f'}$, set of virtual links between NFs of flow f', where $e_{f',ij} \in L_{f'}$ represents the virtual link which connects NF i to j from the joint flow f'.
  - Constants
    - $c_{f',i}$, requested processing units for NF i of the joint flow f'.
    - $m_{f',i}$, requested memory units for NF i of the joint flow f'.
    - $s_{f',i}$, requested storage units for NF i of the joint flow f'.
    - $bw_{f',e_{ij}}$, requested BW for the link that is connecting NF, i to j in the joint flow f'. The links that connect flows to each other have zero BW request.
  - Decision Variables
    - $x_{f',i,u}$, a binary variable where one means that function i from the joint flow f' is placed in the physical node u.
    - $F_{f',e_{ij},E_{uv}}$, a binary variable which is equal to one when the virtual link between NFs i and j of the joint flow f' is mapped to one or more physical links and physical link E_{uv} is one of them. In the case of mapping a virtual link to multiple physical links all the related variables must be set to one.

4.5.2 Joint All Shortest Path Heuristic

The joint heuristic combined with our mathematical model increases the computational complexity as it increases the number of NFs per joint request by considering all placed SGs along with the arriving SG. Here we provide an alternative version of the joint heuristic that has lower computational complexity and provides solutions much faster than the joint mathematical model. The following version of the joint heuristic can be combined with any heuristic that considers one request at a time. Here we describe the combination of the joint heuristic and our all shortest path heuristic as an example.

By the arrival of a request a SG will be treated the same way as in all shortest path heuristic. If the placement was successful the algorithm will move on to place the next arriving request. Otherwise, the joint heuristic will be applied. In the case of rejection of a request, a set of previously placed SGs that are not expired will be considered. The number of considered SGs can vary from two to more requests. In this step, instead of forming a joint request, the joint heuristic changes the arrival order of the considered SGs. It tries all permutations of the considered SGs' arrival order until one permutation results in the acceptance of all considered requests including the current request. For example if we consider the current SG and the last two already placed SGs, the number of permutations will be six. Each permutation of the order of arrival of requests will be tried until one of them results in accepting all three requests. In a case that none of the permutations were successful the current request gets rejected. It should be noted that the joint heuristic algorithm can not be applied to all VNFs. Some may provide real-time services and interrupting their service to change their placement is not possible. For other VNFs, the cost of interrupting services and their replacement should be compared to the gains we can achieve such as accepting more requests. As will be shown in Chapter 5, this joint heuristic does not meet our goal of placing SGs fast. We therefore did not investigate further to include the cost of VNF (and network routing) reconfigurations in our heuristic.

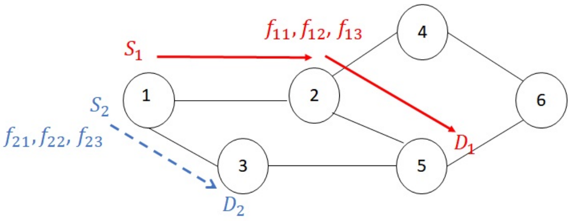

Figure 4.9: The Joint All Shortest Path Placement

Consider the same example as the one we mentioned for the all shortest path heuristic except here we have two SGs to place. The second SG's nodal resource demand is $c_{f_2} = \{4, 2, 3\}$. The source of the second SG is node 1 and its destination is node 3. Its BW demand is 3 units for all virtual links. If we first place the first

SG using all shortest paths heuristic then the second SG can not be placed as the nodal resources of nodes 1 and 3 are not enough to place the second SG and it gets rejected. However if we change the order of the SGs and first place the second SG as it is shown in Figure 4.9 then place the first SG, both SGs can be placed successfully.

Chapter 5

Results and Analysis

In this chapter, we evaluate the performance of our proposed mathematical models and heuristics under different circumstances. We identify the parameters that affect their performance and explore the effect of considering the identified parameters in our heuristics. We implement the basic model as a way to capture the NF placement under the assumptions of a wired network. The placement rate, placement costs, and execution time serve as a benchmark to compare our results against. We generate a number of multi-hop wireless topologies and place SGs based on our basic model, extended model, and traffic-aware model in the first part. To see the impact of our approach in bigger networks we increase the number of nodes and observe the results as a function of network size [7].

In the second part of this chapter, we apply similar scenarios to our increasingly complex set of heuristics and compared their performances with our mathematical models and similar heuristics. As heuristics can provide solutions much faster than mathematical methods we applied them to larger size networks to explore their performance. We measured the execution time, acceptance rate, and placement cost and compared it with mathematical methods and similar heuristics to identify the heuristic that can accept the highest ratio of the requests with lowest complexity and execution time. As mentioned earlier, the cost in our mathematical models and heuristics is calculated the same as the objective function mentioned in Equation (2), for wired networks and Equation (10) for wireless networks in Chapter 3. The cost is based on the cost of consumed resources by the VNFs in the physical network that includes the cost of total units of nodal resources used by VNFs, and the cost of total units of bandwidth used by virtual links in the physical network.

5.1 Modeling Environment

Two platforms are being used to solve the placement problems: MATLAB to implement our heuristic algorithms, and AMPL to solve the mathematical optimization model, the alternative heuristic proposed in [31], and a heuristic we call first feasible heuristic. In this heuristic, we run the optimization model only until we find a feasible solution, rather than the optimal solution. AMPL is a modeling language designed to be used for solving optimization problems such as linear and non-linear programming problems [46]. We used AMPL to solve the optimization model and the other two compared heuristics as it works with a wide range of solvers. We used BARON for solving our optimization model in AMPL. BARON is a general purpose solver that implements a branch-and-reduce algorithm for solving mixed-integer nonlinear and linear optimization problems. Purely continuous, purely integer, and mixed-integer nonlinear and linear problems can be solved with BARON. BARON is available under the AMPL, GAMS, and other modeling languages on a variety of platforms [47]. Unlike AMPL, which is designed for solving optimization models, MATLAB allows us to develop our heuristic algorithm for VNF placement. The wired and wireless topologies are generated with the use of the method proposed in [48].

The nodes are randomly deployed in a square area, based on a uniform distribution. We generate ten topologies for each network size of 20, 30, 40, 60, 80, and 100 nodes, the network area grows with the number of nodes. We keep the average node density constant, consequently, the network size ranges from $490 * 490$ m^2 for the 20 nodes network to $690 * 690$ m^2 for the 40 nodes network, and $1095 * 1095$ for 100 nodes network [48]. Nodes in the wireless network are directly connected if their distance is less than or equal to the transmission range of the nodes. This transmission range is constant for all nodes and is 150 meters. We verified that all of the generated topologies are connected. To be able to compare results between a wired and wireless network the wired networks' topologies are exactly the same as the ones used for wireless networks' scenarios. The properties of the generated topologies, including the average number of links for each generated topology, the average number of neighbours for each node, the average number of total shortest paths found for each source and destination pair, and the average number of shortest paths for all possible source and destination pairs are described in Table 5.1. The average is among the 10 generated topologies and the length of shortest paths is based on the number of hops.

Number of nodes	20	30	40	60	80	100
Average number of links	47.4	69.6	90.3	142.5	197.6	260.2
Average number of neighbours per node	4.7	4.6	4.5	4.7	5	5.2
Average number of shortest paths between each pair	2.3	3.2	3.5	11.2	10.4	16.3
Average length of shortest path	3.7	4.6	5.3	7.4	7.4	8
Average of maximum length of shortest path	7.1	10	11.3	6.7	17.4	17.5
Average of total number of shortest paths	443	1385	2753	19816	32942	80872

Table 5.1: Generated Topologies' Characteristics

Scenario	Topology	CPU	w_{cpu}	Memory	w_m	Storage	w_s	BW	w_{bw}
Mathematical model	Randomly placed	[100, 150]	1	[100, 150]	1	[100, 150]	1	[100, 150]	1
Heuristics	Randomly placed	[100, 150]	1	-	-	-	-	[100, 150]	1

Table 5.2: Physical Network Properties.

The parameters reflect the commonly chosen simulation setup in the literature such as [12, 49–51]. Table 5.2 shows the available processing, memory and storage capacity of the nodes and bandwidth of the links at the beginning of each simulation that are uniformly distributed between 100 and 150 units in all scenarios. In later scenarios, we increase the available BW and the nodal resource availabilities to observe their impact on the performance of the mathematical models and heuristics. We assumed the cost per unit of all nodal resources is one in all scenarios.

Flows arrive over time following a Poisson process with an average rate of four flows per 100 time units. Each flow has a lifetime, exponentially distributed with an average of $\mu = 1000$ time units for the scenarios comparing the performance of our mathematical models and $\mu = 500$ for the scenarios that involve heuristics and mathematical models. Each flow is accompanied by a Service Graph (SG), defining the required NFs and their interconnection to handle this flow. There are 6 NFs per request. The source and destination of each flow are being chosen randomly between the nodes. For the mathematical models' scenarios, the nodal resource demands of each NF follow a uniform distribution between 5 and 25 units. The bandwidth requirement of all links of a request is the same and chosen uniformly from between 1 and 50 units. The nodal resource demand and lifetime of the requests are slightly higher for mathematical model scenarios as it involves models such as the basic model that has fewer constraints than the other models and accepts more requests, all else

Scenario	Num of NFs	CPU	Memory	Storage	BW	lifetime(s)
Mathematical model	6	[5, 25]	[5, 25]	[5, 25]	[1, 50]	1000
Heuristics	6	[1, 20]	-	-	[1, 50]	500

Table 5.3: SGs Properties.

being equal. To avoid having 100% acceptance rates at all times and be able to capture the effect of changing parameters such as network size on the final results, we increased the nodal resource demand of the requests and their lifetime for mathematical model scenarios.

Table 5.3 shows the CPU, memory, and storage unit demands of each NF for mathematical models and heuristics scenarios. In order to make it easier to analyze and understand the performance of our heuristics, the heuristics consider only one nodal resource. That resource can represent CPU, memory, or storage unit demands of each NF, we have chosen to model CPU demands here.

5.2 Measurement Metrics

To measure the performance of our proposed mathematical models and heuristics, and to compare their performance with each other, we used the following metrics.

- Acceptance ratio: The total number of accepted requests during a simulated network lifetime of 20,000 seconds divided by the total number of requests.

- Shortest path ratio: The total number of accepted requests during a simulated network lifetime of 20,000 seconds that are placed along their shortest path, divided by the total number of requests.

- Average Cost: Average cost of the BW and nodal resource units used for the deployed requests that are not expired. Please note that this includes the bandwidth of links actually used by flows, as well as the bandwidth consumed on adjacent links due to interference. The cost for nodal resources equals to the total units of nodal resources used for deployed requests multiplied to their cost per unit of nodal resources.

- Average BW cost: Average cost of the BW units used for the deployed requests that are not expired multiplied to the cost per unit of BW. Please note that this includes the bandwidth of links actually used by flows, as well as the bandwidth consumed on adjacent links due to interference.
- Execution time: The total time that it takes to place all arriving requests over the simulated network lifetime of 20,000 seconds.

Each data point is the average of 10 runs, generating a new random topology and sequence of flow arrivals each time. In addition to the average, we also plot the 95% confidence intervals. We model/simulate 20,000 seconds, in order to reach a steady state where the curves flatten off after initial settling due to the initially unloaded network. The following figures represent the steady-state results over the simulated network lifetime of 20,000 seconds.

5.3 Mathematical Model Results

In our first scenario for mathematical models, we explore the performance of our basic and extended model for wired and wireless networks. We generate random topologies for 20, 30, and 40 nodes networks and solve them with our basic, extended, and traffic-aware model. Figure 5.1 shows the average acceptance ratios for wired and wireless networks and Figure 5.2 shows the average number of physical links used for placement of a SG. Figure 5.3 shows the average cost and average BW cost of the wired networks solved with basic model and wireless networks solved with the extended model that considers the effect of interference. The total of each bar represents the average cost, the yellow/blue part of bars represents the average BW cost and the red part of each bar is the average nodal resource cost for placing a SG. We will use the same presentation for all average cost and average BW cost figures in the remainder of this chapter.

Figure 5.1 and Figure 5.3 show that although the average acceptance ratio increases by increasing the network size, the average cost and especially average BW cost decreases significantly. This is mainly because the optimization method tends to minimize the resource usage for each SG; therefore, it chooses a placement that has fewer physical links involved. The decrease in the number of assigned physical

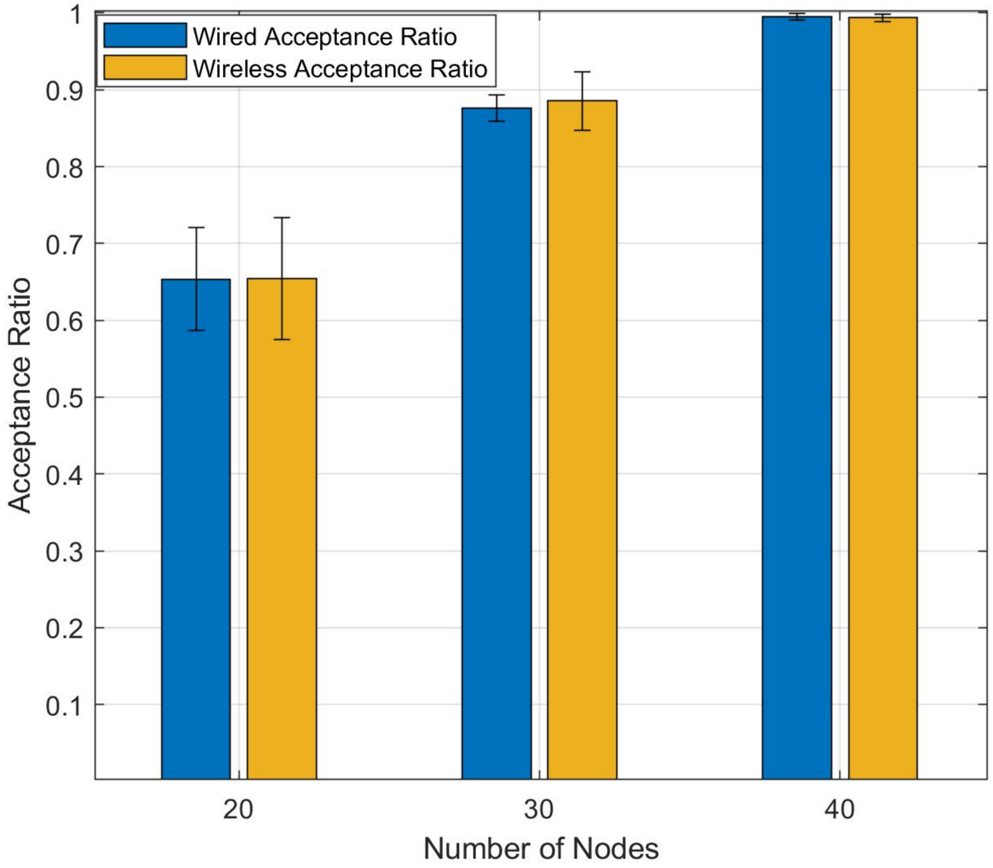

Figure 5.1: Acceptance Ratio in the Wired and Wireless Network

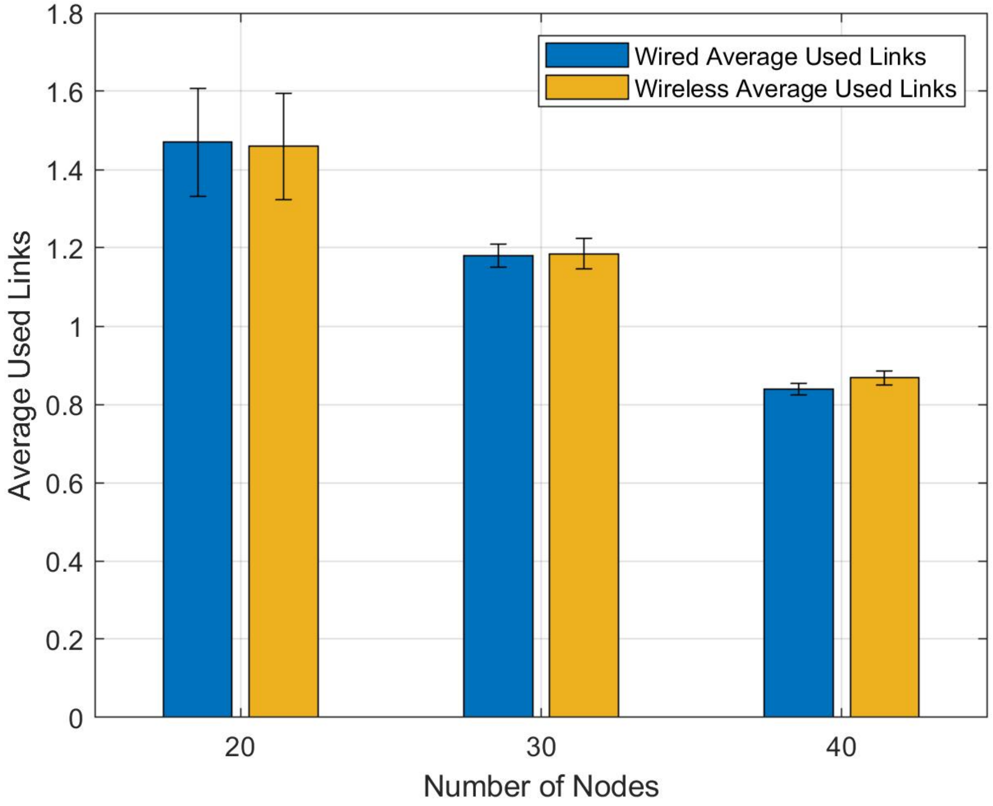

Figure 5.2: Average Physical Links Used for Placement of a SG in a Wired and Wireless Network

links can be seen in Figure 5.2, which shows the average number of links used for placement of each SG.

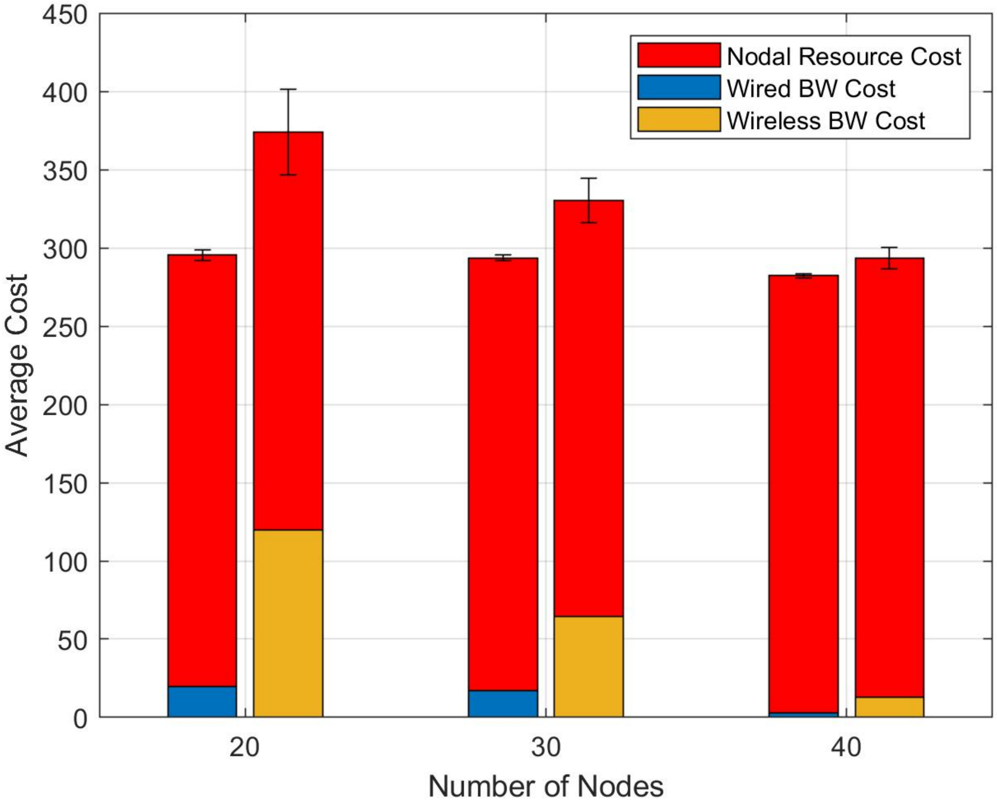

Figure 5.3: Average Cost and BW cost in the Wired and Wireless Network

The difference between the average cost of wired and wireless networks as seen in Figure 5.3 is mainly due to BW cost, which is affected by the presence of interference in wireless networks. However, as we can see from Figure 5.1 the difference in the BW cost is not high enough to affect the acceptance ratio and the recorded average acceptance ratio is similar for both wired and wireless networks. This is due to the fact that, for the chosen arrival rate of the requests and requested resources, most of the NFs can be placed in one or two nodes. This limits the impact of interference on the acceptance ratio. Figure 5.2 and Figure 5.3 demonstrate that increasing the number of nodes (and hence available nodal resources) will increase the possibility to place a SG such that it uses fewer physical links. This then consequently lowers the average cost and increases the acceptance ratio.

The traffic-aware model considers a source and destination for each SG request and, unlike the extended model, SGs can not be placed anywhere in the network. Flows start from their source and end at their destination. The source and destination of the flows are selected uniformly random. The bandwidth requirement of each flow, their nodal resources demand and the properties of the physical network are all the same as our previous scenario. For each network size, we repeated the experiment 10 times with different network topologies, averaged the acceptance ratios, shortest path ratios, average costs, and average BW cost, and calculated the corresponding 95% CI.

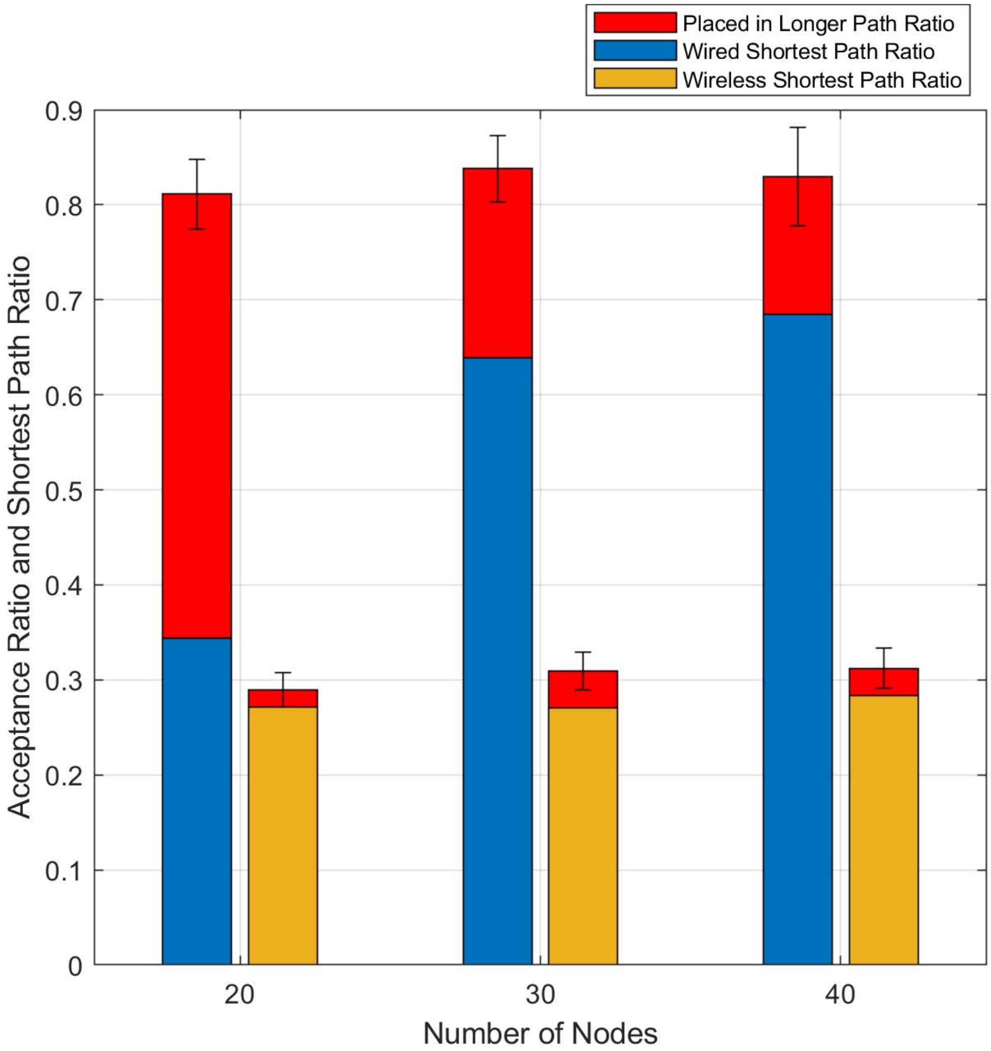

Figure 5.4: Acceptance Ratio of Wireless Networks of 20, 30, and 40 Nodes for the Extended Model with Source and Destination

Figure 5.4 shows the average acceptance ratio and average shortest path ratio at the same time. Each bar in Figure 5.4 represents the acceptance ratio, the blue/yellow part of each bar represents the average shortest path ratio, and the red part of each bar represents the ratio of the requests that are being placed along a path longer than their shortest path. We will use the same presentation in all of our graphs that show the average acceptance ratio and the average shortest path ratio at the same time. Figure 5.5 shows the average cost and average BW cost. Unlike our previous scenario, we can see a considerable difference in the average cost and average acceptance ratios recorded for wired and wireless networks for the traffic-aware model.

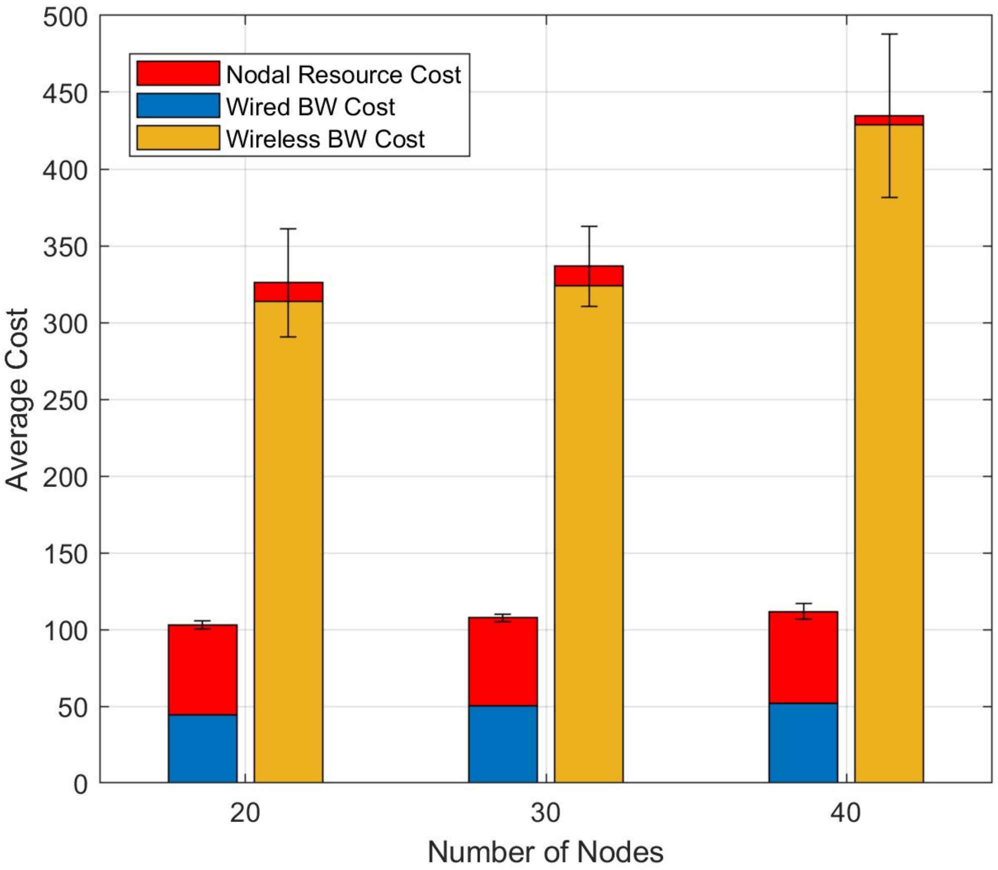

Figure 5.5: Average Cost of Wireless Networks of 20, 30, and 40 Nodes for the Extended Model with Source and Destination

Figure 5.5 clearly demonstrates that most of the average cost in wireless networks belongs to the BW cost and increases with increasing the number of nodes. This is mainly due to an increase in the length of shortest path in larger networks. The lower acceptance ratio recorded for wireless networks in Figure 5.4 is another consequence of higher BW consumption in wireless networks. Figure 5.4 shows that in wireless networks more than 90% of the accepted requests are being placed along their shortest path. This opens an avenue for the methods that are focused on reducing the execution time. As more than 90% of the requests are being placed along their shortest path, reducing the search for placement of NFs to these nodes would reduce the execution time while not eliminating the min-cost solutions. This is a factor we considered in the design of our heuristics and also is being considered in the accessible scope heuristic proposed in [31]. Due to the lower BW cost in wired networks, we accept a higher ratio of the requests in such networks. In addition, a relatively larger number of these accepted requests are placed along non-shortest paths. However, we can see from Figure 5.4 that by increasing the network size a higher ratio of requests are being placed along their shortest path.

In another attempt to explore the performance of our mathematical model we keep the number of nodes constant and increase the nodal resources and available BW of physical links. We choose the network size of 40 nodes to better observe the effect of increasing resources on the recorded acceptance ratio and average cost. In the first scenario, we increased the available BW of the physical link. To model increased bandwidth availability, we increase the uniform distribution that the BW link is chosen from. The intervals are [300, 400], [500, 600], [700, 800], [900, 1000], [1000, 1100].

Figure 5.6 shows the average acceptance ratio, its 95% CI, and the average ratio of the requests placed along their shortest path. Figure 5.7 shows the average cost, its 95% CI, and the average BW cost. The x-axis for both figures are labeled by the beginning of each BW interval. Figure 5.6 clearly shows that BW is a bottleneck in wireless networks. Increasing the available BW will mitigate the impact of this bottleneck, and increase the ratio of accepted requests. By increasing the available BW, the mathematical model can place requests with higher BW demand that increase the average BW cost and consequently the average cost, as can be seen from Figure 5.7.

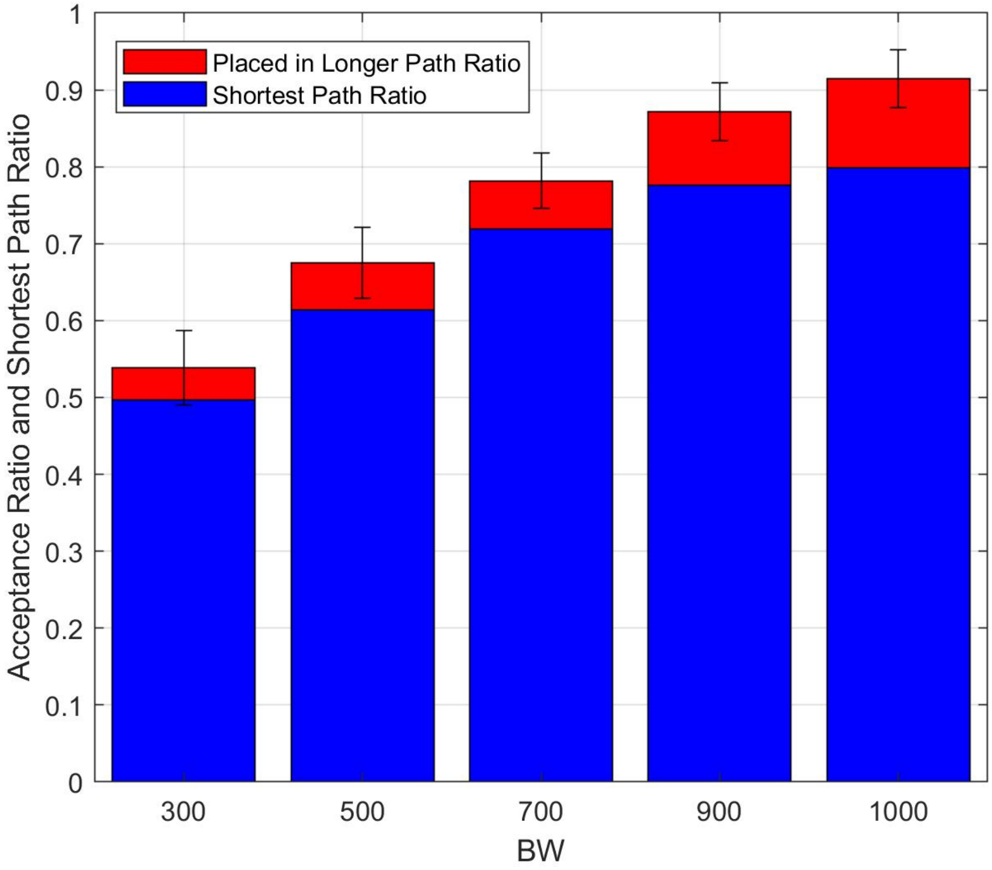

Figure 5.6: Acceptance Ratio and Shortest Path Ratio of 40 Nodes Wireless Network while Increasing Link BW

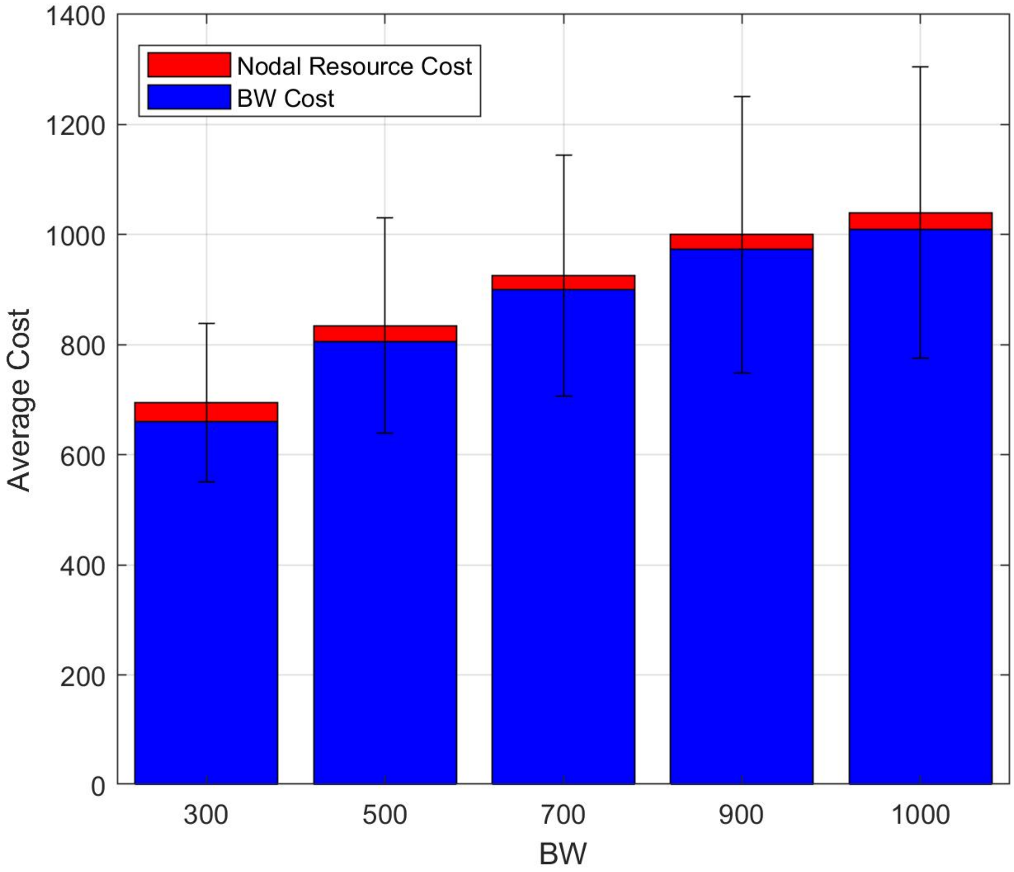

Figure 5.7: Average Cost and Average BW Cost of 40 Nodes Wireless Network while Increasing Link BW

In the last scenario for our mathematical model, for a network of 40 nodes, we increase the nodal resource by increasing the uniform distribution that the nodal resource is chosen from. The intervals are [1000, 1500], [2000, 2500], and [3000, 3500]. Interestingly we can see from Figure 5.8 and Figure 5.9 that increasing nodal resources does not impact the acceptance ratio, and consequently the average cost even for the highest available nodal resources. We conclude that bandwidth is a more significant factor in wireless networks and in the design of our heuristic we gave more priority to BW usage.

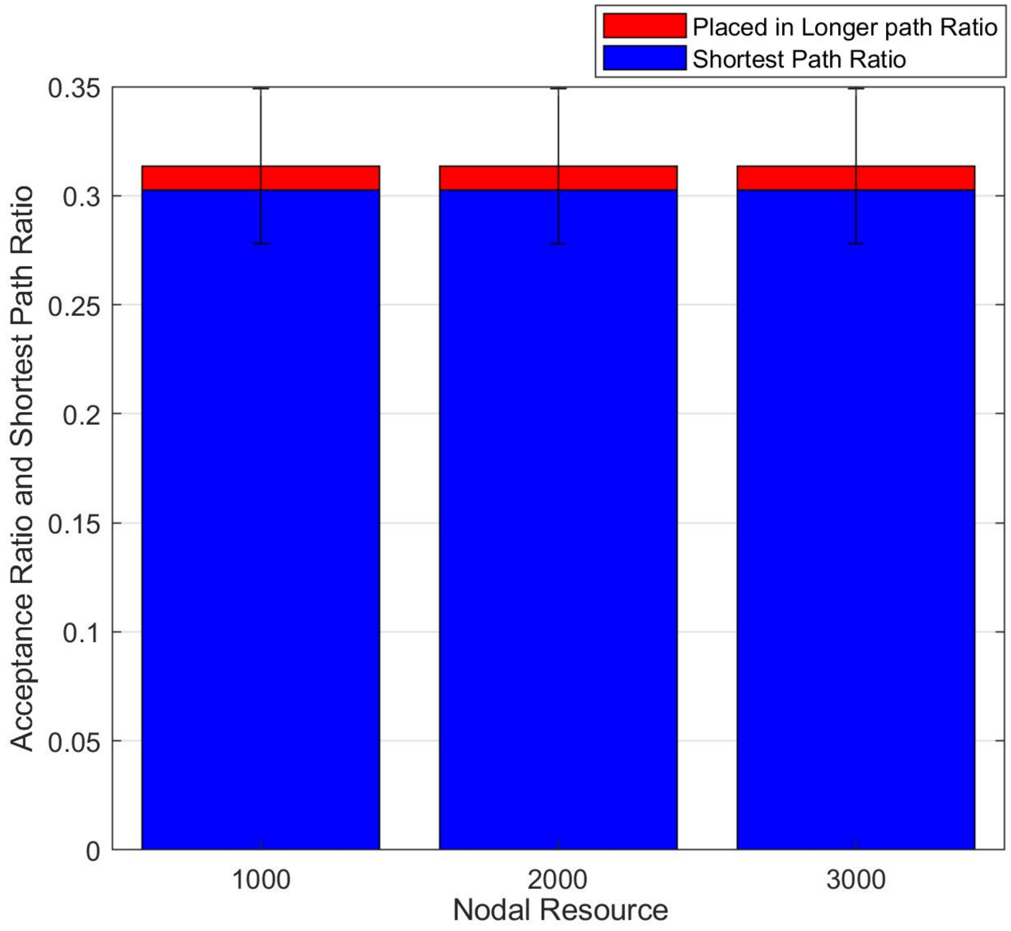

Figure 5.8: Acceptance Ratio and Shortest Path Ratio of 40 Nodes Wireless Network while Increasing Nodal Resources

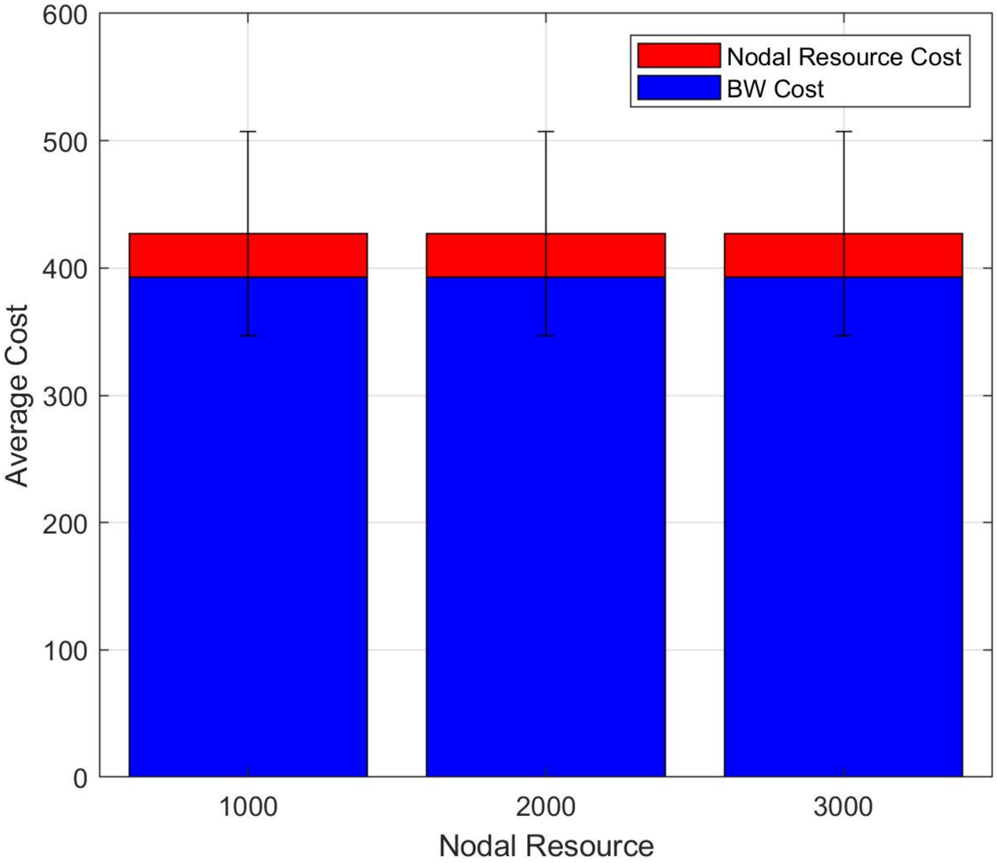

Figure 5.9: Average Cost and Average BW Cost of 40 Nodes Wireless Network while Increasing Nodal Resources

5.4 Heuristics Results

To evaluate the performance of our increasingly complex heuristics, we applied them to different size networks and compare their performance with our traffic-aware mathematical model, the accessible scope heuristic proposed in [31], and the first feasible heuristic. We used two versions of the joint all shortest path heuristic to explore the effect of increasing the number of considered requests on the execution time and acceptance ratio. The first version of the joint all shortest path heuristic considers the current request and the last two requests that have been placed and the second version considers the current request along with the last 4 placed requests. The first feasible heuristic uses the same mathematical model as our traffic-aware model with source and destination to find a placement for a SG. However, and different from our optimization models, it does not have an objective function and consequently accepts the first solution that satisfies all constraints. The first feasible heuristic shows the performance of a mathematical model that does not search for an optimal solution and accepts any solution that is feasible. We used AMPL and BARON as the solver for this heuristic..

We applied our heuristics, the accessible scope heuristic, the first feasible heuristic, and the ILP model to the same physical networks and the same sequence of requests. We increased the number of nodes from 20, to 30, and 40 nodes for the ILP and to 60, 80, and 100 nodes for the heuristics. We did not solve the placement problem with the mathematical model for the larger size networks as its execution time grows exponentially. The results are divided into two parts. The first part includes the results from smaller size networks of 20, 30, and 40 nodes that includes the performance of our mathematical model and the second part belongs to the results recorded for larger size networks of 60, 80, and 100.

Figure 5.10 and Figure 5.11 show the average acceptance ratio and their 95% CI for a simulated network lifetime of 20,000 s. The average execution time of each algorithm to provide a solution for all arriving requests for a simulated network lifetime of 20,000 s and for different size networks is shown in Table 5.4. We can see from Figure 5.10

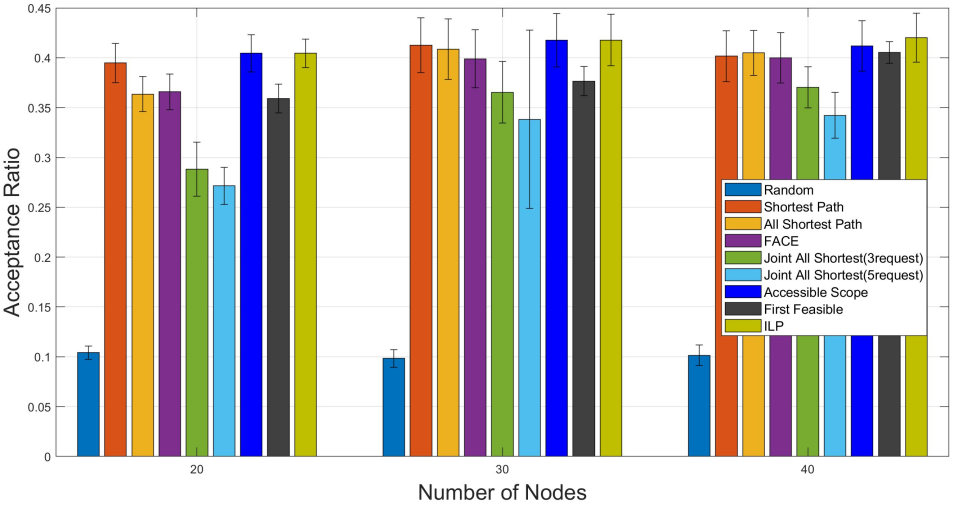

Figure 5.10: Comparison of Acceptance Ratio's of Heuristics and Mathematical Model in Smaller Networks

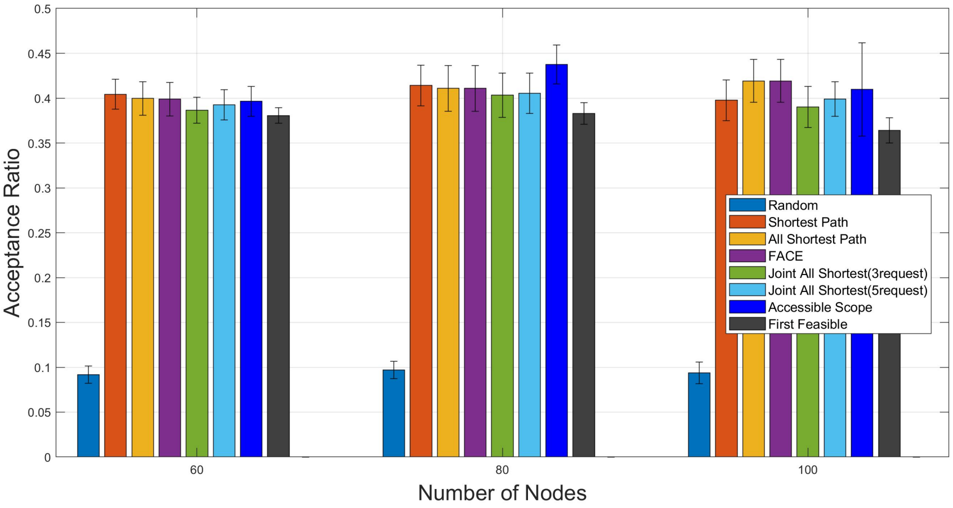

Figure 5.11: Comparison of Acceptance Ratio's of Heuristics in Larger Size Networks

Network size	**20**	**30**	**40**	**60**	**80**	**100**
Random	18.6	19.4	19.5	65.5	24.8	322.9
Shortest path	4.1	4	5.2	4.6	4.9	16.6
All shortest path	10.3	12.1	12.7	13.2	18.7	42.4
FACE	12.8	13.3	15.5	16.5	18.5	45.8
Joint all shortest path(3requests)	9.7	10.1	12.9	16.6	36.1	37.3
Joint all shortest path(5requests)	25.4	28.2	40.9	51.5	276.3	223.4
Accessible scope	139.9	268.1	429	1648.3	5866.7	10985
First feasible heuristic	128.6	285.2	661.5	2002.5	6537.8	13125
ILP model	146.1	276	681.1	-	-	-

Table 5.4: Execution Times in Seconds for Networks of Different Size

and Figure 5.11 that the acceptance ratio does not grow with an increase in the size of our network and it is fluctuating between 35% to 45%. That is mainly due to the fact that the length of shortest path grows with network size and increases the BW cost. Placing NFs randomly results in the highest average cost and lowest acceptance ratio and shows that any effort in reducing a placement cost of NFs would improve the acceptance ratio.

The accessible scope heuristic performs similar to our mathematical model and has the best performance among all heuristics. However, its average execution time. as shown in Table 5.4, is not much different from the mathematical model. A high execution time for the accessible scope heuristic means that our first goal in the design of a heuristic, providing an algorithm that can provide near-optimal solutions with low execution time, is not being satisfied.

Our three heuristics, shortest path, all shortest path, and FACE, are successful in providing statistically similar average acceptance ratio to the ILP model for smaller networks and statistically similar acceptance ratios to the accessible scope heuristic for larger networks. The advantage of our heuristics over the ILP model and accessible scope heuristic is their execution times. As we can see in Table 5.4, for a network of 40 nodes, the execution time of our heuristics are in the order of 10s of seconds while the ILP and accessible scope heuristic execution times are in the order of 600 seconds. These differences become even larger for larger size networks. In terms of the execution time, we can see even the performance of the first feasible heuristics is not much better than the ILP and accessible scope heuristic although it eliminates the process of searching for an optimal solution and accepts any feasible solution. The execution times of both the accessible scope heuristic and the first feasible heuristic indicate that using a solver will always be a lengthy process and lowering the search space or eliminating the process of a search for an optimal solution will not reduce the execution time considerably.

As we described in detail in Chapter 4, progressing from the shortest path heuristic to FACE and then the joint heuristics we added different parameters. For example, in the shortest path heuristic we only search for one shortest path and place the first NF in the first node of the shortest path that has sufficient nodal resources. However, in FACE we search for all shortest paths, give priority to NFs based on their demand, give priority to shortest paths with higher minimum nodal resources, and in case of failure try other shortest paths. While we are including all these parameters to increase the ratio of accepted requests, we can see from Figure 5.10 and Figure 5.11 that all three of our heuristics, namely shortest path, all shortest path, and FACE, have similar performance in terms of average acceptance ratio. For the more complex joint heuristics the situation gets worst and their average acceptance ratio is lower than our heuristics specially for smaller size networks. The recorded average acceptance ratios of our heuristics show that, other than placing a SG along its shortest path, the other parameters do not contribute to a higher acceptance ratio. Rather, they make the heuristic more complex and increase the execution time. Table 5.4 shows that the shortest path heuristic can provide the same acceptance ratio in 4.9 seconds for 80 nodes network while the execution time for the all shortest path heuristic is 18.7 s and 18.5 s for FACE. The difference between the execution time of our shortest path heuristic and the other two grows even more for 100 node networks.

Figure 5.12 and Figure 5.13 show both the average cost and average BW cost with their 95% CI for a simulated network lifetime of 20,000s. Figure 5.12 and Figure 5.13 show an increasing BW cost trend for all included heuristics and mathematical models that is mainly due to an increase in the length of the shortest path in larger networks. It can be seen that the average cost for the random heuristic increases the most as there are more physical links involved in connecting the randomly placed NFs in

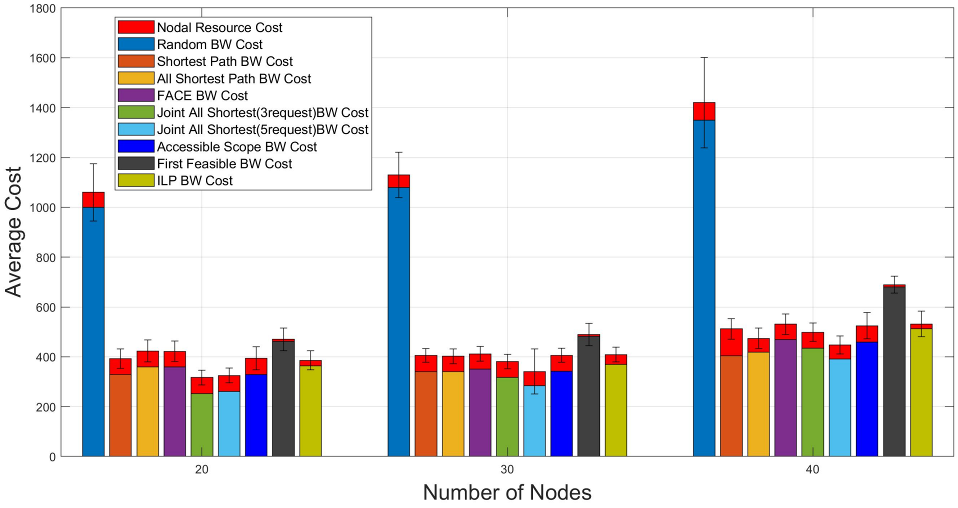

Figure 5.12: Average Cost and Average BW Cost of Heuristics and Mathematical Model in Smaller Networks

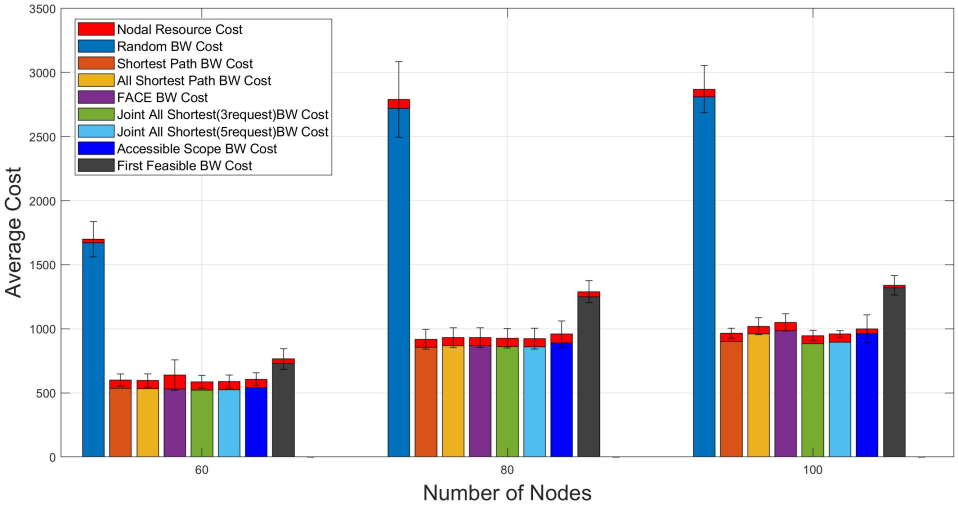

Figure 5.13: Average Cost and Average BW Cost of Heuristics in Larger Size Networks

larger size networks. The average cost of the accessible scope heuristic is lower than the ILP model as it only accepts the requests that can be placed along their shortest path. In constrast, the ILP model chooses a feasible solution with the lowest cost that may include taking a path longer than the shortest path. The first feasible heuristic recorded higher average cost in comparison to the ILP and the accessible scope heuristic which is mainly due to accepting the first solution it finds that is not necessarily a min-cost solution.

At last, we explored the performance of our heuristics for the scenario that the number of nodes is constant and the nodal resources or links' BW are increasing. We chose a network with 100 nodes in order to have more shortest paths available for each pair of nodes in comparison to smaller networks and be able to compare the performance of our increasingly complex heuristics. We vary the uniform distribution that the physical links' BW is chosen from to model increased bandwidth availability. The chosen BW intervals are [300, 400], [500, 600], [700, 800], [900, 1000], [1000, 1100]. As can be seen from Figure 5.14, the average ratio of accepted requests increases as we increase the initially available links' BW. The x-axis is labeled by the beginning of each BW interval. We can clearly see that for the last two intervals [900, 1000], and [1000, 1100] the average acceptance ratio is one and all requests can be placed by all heuristics except the random heuristic. High consumption of BW due to the presence of interference makes BW a bottleneck for NF placement in wireless networks. Increasing the available BW will mitigate the impact of this bottleneck, and increase the ratio of accepted requests uniformly for all heuristics. Figure 5.15 shows the average cost and average BW cost for each heuristic. Figure 5.15 shows that a high proportion of average cost belongs to average BW cost. If we increase the bandwidth sufficiently (to at least 900 units per link), essentially all flows can be accommodated, independent of the heuristic, as long as NFs are placed on a shortest path between source and destination.

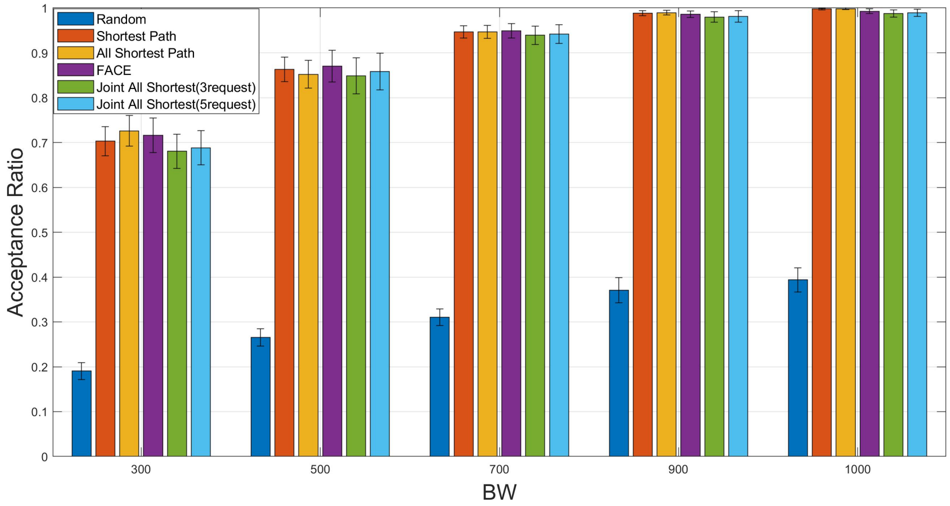

Figure 5.14: Acceptance Ratio of 100 Nodes Wireless Network while Increasing BW Resources

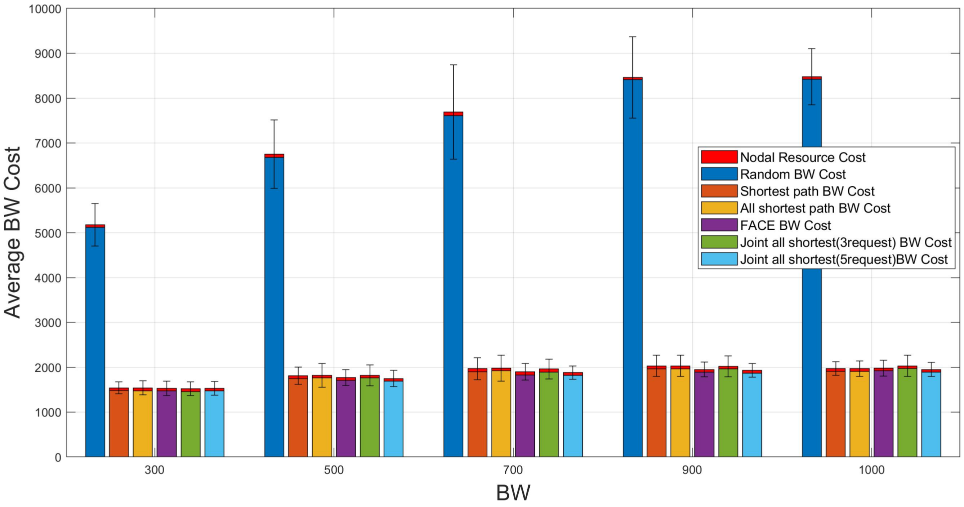

Figure 5.15: Average Cost and BW Cost of 100 Nodes Wireless Network while Increasing BW Resources

Finally, in the last scenario for a network of 100 nodes, we increase the nodal resource by increasing the uniform distribution that the nodal resource is chosen from. The nodal resource intervals are [1000, 1100], [2000, 2500], and [3000, 3500]. Figure 5.16 shows the acceptance ratio, and the x-axis is labeled by the beginning of each nodal resource interval. Unlike the results shown in Figure 5.14, increasing the nodal resource does not increase the acceptance ratio. As the acceptance rate stays the same, the average cost and BW cost do not change as it is shown in Figure 5.17. In comparison to our previous scenario, we can conclude that bandwidth is a more significant factor in multi-hop wireless networks. This is similar to the results we captured from the mathematical model as it is shown in Figure 5.6 and Figure 5.8. We can see that the performance of all of our heuristics are similar and for a 100 nodes network with high availability of BW or nodal resources none has any advantage over the other methods in terms of acceptance ratio. However their execution times vary based on their complexity and the shortest path heuristic can provide the same performance in less time than the others.

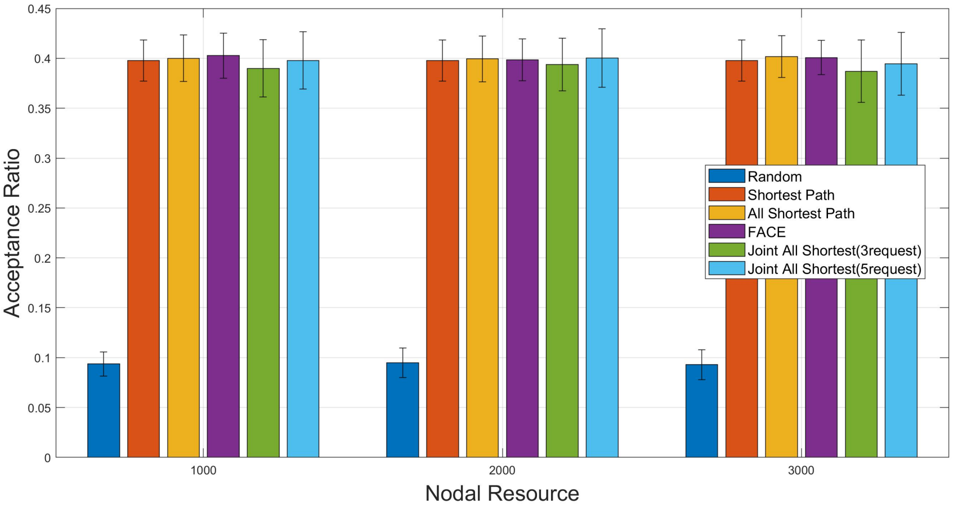

Figure 5.16: Acceptance Ratio of 100 Nodes Wireless Network while Increasing Nodal Resources

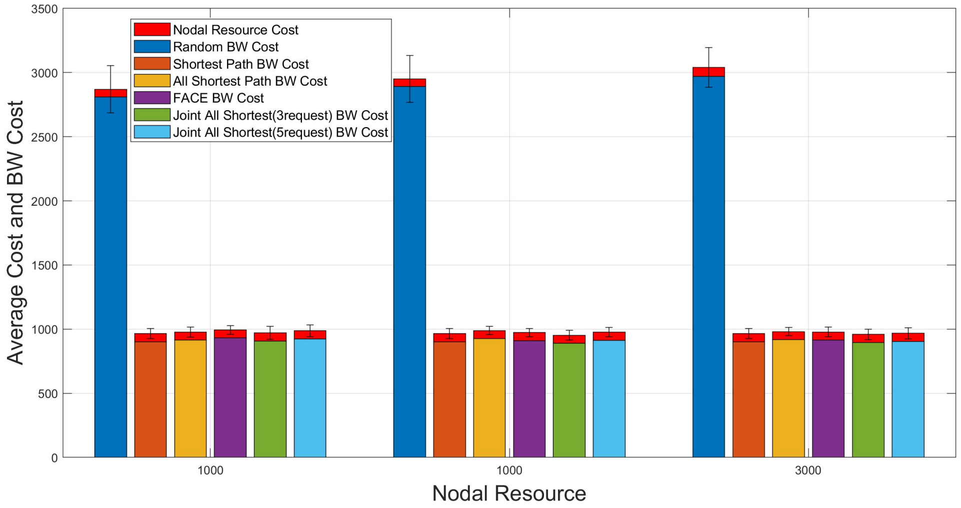

Figure 5.17: Average Cost and BW Cost of 100 Nodes Wireless Network while Increasing Nodal Resources

5.5 Summary

We have optimization models that give us the best possible answer, subject to a criteria, which in our case is the placement cost. We then use these models to explore how successful we can be in placing SGs subject to network size/topology and available resources and what factors have an impact on the placement success (the acceptance ratio and the overall costs). We identified the issue of scarcity of BW in wireless multi-hop networks and its role in the average cost of placement and acceptance ratio. We showed that by increasing the available BW of the network we can increase the ratio of the accepted requests and also identified that more than 90% of requests are being placed on their shortest path. We learned from our mathematical models and designed a set of increasingly complex heuristics. We compared their acceptance ratio and the overall costs with each other, our mathematical model, and two other heuristics. Our goal was to identify the effectiveness of each considered parameter in each of our heuristics. Our results show that the simple approach of placing SGs along their shortest path is the only parameter that we need to consider to provide the best results in terms of acceptance ratio at the lowest execution time. Our results indicate that the shortest path heuristic can provide results competitive with the accessible scope heuristic in a matter of 10 seconds while the accessible scope takes around 10,000 seconds.

Chapter 6

Conclusion and Future Work

6.1 Summary and Conclusion

The main challenge in the use of VNFs is to optimally map the SG requests to the physical network. We provided a mathematical model that considers a wide range of constraints. We started with our basic model dedicated to wired networks and improved it to our extended model that includes the effect of interference by using the protocol model. In the next step, provided the traffic-aware model that considers a source and destination for each arriving request. We added a set of necessary constraints to start flows from their source and end at their destination. The acceptance rate and the average cost of our mathematical models are evaluated under different scenarios. For each scenario, we compared the results from wired and wireless networks to show the effect of interference on the BW usage, average cost, and acceptance ratio. Our results clearly indicate that interference causes an increase in BW usage that increases the average BW cost and lowers the acceptance ratio in comparison to wired networks. We increased nodal resources and links BW in last scenarios for our mathematical model. The comparison between the increasing BW and nodal resources scenarios shows the important role of BW on the acceptance ratio. The results indicate that BW is the bottleneck and increasing it increases the acceptance ratio while increasing the nodal resources did not change the acceptance rate. The results also indicate that more than 90% of the requests in the traffic-aware model for wireless networks are being placed on their shortest path.

As optimal placement methods are NP-hard and cannot be applied to large networks, we have to design a heuristic with lower complexity that is scalable and can reach near-optimal results. Although there exists a wide range of heuristics proposed for VNF placement, none has focused on designing a simple heuristic that is time-efficient and hence scalable. By learning from the results of our traffic-aware mathematical model we gave priority to BW in design of our heuristics and placed SGs on their shortest path. We explored a set of heuristics, ranging from very simple to more complex, involving backtracking, in order to identify the simplest possible method that can place a high number of requests in the network in a timely manner.

Our five heuristics start with the random placement that does not put any effort into reducing the costs of placement. The shortest path heuristic searches for a shortest path and place NFs on their shortest path to reduce BW usage. We made it more complex in all shortest path heuristic by searching for all possible shortest paths and sort them based on their minimum nodal resource in decreasing order. The FACE heuristic follows the all shortest path heuristic in sorting shortest paths and adds more complexity by sorting NFs based on their nodal resource demand and the number of their candidates for placement. Finally, we proposed the joint heuristic that adds another level of complexity by considering previously placed SGs along the current SG. We compared the performance of our heuristics with our traffic-aware mathematical model and a popular heuristic proposed in the literature.

Our results show that randomly placing NFs, as expected, produces poor results (low acceptance rate, high costs). So some effort is warranted in placing NFs. However, and somewhat unexpected, the simple shortest path heuristic can reach similar results as more complex heuristics. Additional steps, added to the all shortest paths heuristic, the FACE heuristic, and the joint heuristic do not increase the number of accepted requests. Even more complex heuristics such as the accessible scope heuristic do not improve the acceptance rate. In fact, as shown in results, except the joint heuristics all these approaches provide, statistically speaking, the same performance as our mathematical model. However, as the recorded average execution time shows, the simpler the heuristic, the faster its execution time will be. While the accessible scope heuristic takes more than 10 seconds to process a newly arriving request, the shortest path heuristic can process a newly arriving flow in a few milliseconds in a network of 100 nodes, arguing for its scalability and suitability for real-time admission control.

We explored the effect of increasing BW and nodal resources on the performance of each heuristic. However, we kept the request arrival rate and network density constant. We showed that similar to the observation from the mathematical model increasing links' BW can increase the acceptance ratio and in the case of having at least 1000 units as the links' initial BW all requests can be placed by all heuristics.

6.2 Future Work

We explored our models and heuristics under different scenarios. However, there are additional scenarios that we did not explore and plan to explore in the future. First, we plan to consider the impact of traffic patterns. In the provided scenarios we assumed that the traffic is evenly distributed in the network and did not consider having traffic hotspots. In case of having traffic hotspots, choosing between shortest paths might not be the best solution as they might pass through the traffic hotspots. In future work, we plan to consider the impact of traffic patterns on our model and modify our heuristics to be effective in case of having traffic hotspots. This could be done, for example, by considering not only the shortest path but also the shortest possible path that does not pass the traffic hot spot. The second scenario that we plan to apply to our models and heuristics is to increase the density of the network topologies and explore its impact on the acceptance ratio and average cost. We are expecting to find more shortest paths between each source and destination, however the presence of interference can impact the shortest paths that are in vicinity of each other.

In the third scenario, we plan to observe the acceptance rate and execution times of our heuristic in the scenarios that parameters such as mobility are involved. As our simple and time-efficient heuristic can provide solutions in real-time we believe it can perform better than the more complex placement algorithms in the environment that some of its parameters such as topology change quickly. We will more thoroughly evaluate and compare the performance of the various heuristics as we vary other parameters as well, resulting in networks that are more lightly or highly loaded.

In our scenarios we focused on identifying a bottleneck in the process of placement of SGs and considered all resources of the same importance. We also assumed relatively equal amounts of resources. Also we did not evaluate the trade off between nodal costs and BW costs. In future work we can introduce weight factors for each type of resource and vary them to study the impact of shifting from more of an emphasis on nodal resources and their consumption to bandwidth resources. Also we can change the initial available resources of the nodes and links based on an existing wireless network to evaluate the performance of our heuristics in more realistic environment.

In our mathematical and heuristics, we avoided unrealistic assumptions such as knowing all requests' demands prior to the placement process, being able to change the order of NFs in a SG, etc. However, we did not consider constraints related to energy consumption, deployment, and maintenance cost of NFs such as the number of active nodes, or the number of availabe licenses. Going forward we want to extend our mathematical model and heuristics to consider these constraints on the nodal resources and energy consumption and explore their effects on NF placement, acceptance rate, and average cost.

In the process of collecting results due to the high complexity of our non-linear mathematical model, we could not collect results for the traffic-aware model that considers the traffic-changing factor. In the future, we plan to dedicate more computational resources to at least collect results for smaller networks and identify the effect of traffic changing factor on BW usage and NF placement. We would then be able to use this parameter in our heuristics and place the NFs based on their traffic changing factor to reduce the BW usage of the whole SG.

In heuristics that involve finding all shortest paths between the source and destination of the request, the number of shortest paths can grow exponentially in some topologies. As there are nodes and links in common between some shortest paths our approach is to consider shortest paths that are different enough (in terms of nodes and links in common) and have sufficient resources such as the ones that have enough BW resources and enough nodal resources to place a NF with highest nodal resource demand in the SG request. Also, we will identify the number of shortest paths that should be considered and limit our search to find a specific (limited) number of shortest paths based on the number of nodes in the network.

Our joint heuristic that considers reconfiguration of already placed SGs to accommodate more requests can be modified to be applicable to all kinds of VNFs. We mentioned in Chapter 4 that the reconfiguration of some type of VNFs is not possible as they are providing real-time services. An approach that can solve this problem is to defer admitting requests and place multiple service requests (by bundling them over time) instead of reconfiguring already admitted requests. However, as we showed in our results, even when we allowed the reconfiguration of already placed requests at essentially zero costs, this did not improve the results in terms of acceptance ratio.

Our ILP model is NP-hard and can not be applied to large scale networks. One avenue to reduce the complexity of the problem is to relax the ILP to a LP model that allows for faster solution and then use known approximation approaches to solve such a relaxed LP model to get results with known approximation ratios.

List of References

[1] Alcatel Lucent, "Virtualized EPC delivering on the promise of NFV and SDN." https://www.scribd.com/document/387432044/10743-alcatel-lucent-virtualized-epc-delivering-the-promise-nfv-pdf, accessed: 2021.07.22.

[2] D. Qi, S. Shen, and G. Wang, "Towards an efficient VNF placement in network function virtualization," *Computer Communications*, vol. 138, pp. 81–89, 2019.

[3] A. Hirwe and K. Kataoka, "Lightchain: A lightweight optimisation of VNF placement for service chaining in NFV," in *2016 IEEE NetSoft Conference and Workshops (NetSoft)*, pp. 33–37, 2016.

[4] K. Joshi and T. Benson, "Network function virtualization," *IEEE Internet Computing*, vol. 20, no. 6, pp. 7–9, 2016.

[5] Y. Zhang and O. for Higher Education (Firm), *Network Function Virtualization.* Wiley-IEEE Press, 1st ed., 2018.

[6] J. Halpern and C. Pignataro, "Service function chaining (SFC) architecture," tech. rep., 2015.

[7] Z. Jahedi and T. Kunz, "Virtual network function embedding in multi-hop wireless networks," in *Proceedings of the 15th International Joint Conference on e-Business and Telecommunications, ICETE 2018 - Volume 1: DCNET, ICE-B, OPTICS, SIGMAP and WINSYS, Porto, Portugal, July 26-28, 2018.*, pp. 199–207, 2018.

[8] A. Laghrissi and T. Taleb, "A survey on the placement of virtual resources and virtual network functions," *IEEE Communications Surveys Tutorials*, vol. 21, no. 2, pp. 1409–1434, 2019.

[9] Z. Jahedi and T. Kunz, "Optimal VNF placement: Addressing multiple min-cost solutions," in *E-Business and Telecommunications*, pp. 1–23, Springer International Publishing, 2019.

[10] Z. Jahedi and T. Kunz, "Fast and cost-efficient virtualized network function placement algorithm in wireless multi-hop networks," in *Ad-Hoc, Mobile, and Wireless Networks* (L. A. Grieco, G. Boggia, G. Piro, Y. Jararweh, and C. Campolo, eds.), pp. 23–36, Springer International Publishing, 2020.

[11] Z. Jahedi and T. Kunz, "The value of simple heuristics for virtualized network function placement," *Future Internet*, vol. 12, no. 10, 2020.

[12] S. Sahhaf, W. Tavernier, M. Rost, S. Schmid, D. Colle, M. Pickavet, and P. Demeester, "Network service chaining with optimized network function embedding supporting service decompositions," *The 21st IEEE International Workshop on Local and Metropolitan Area Networks*, vol. 93, pp. 492–505, 2015.

[13] 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 *The 21st IEEE International Workshop on Local and Metropolitan Area Networks*, pp. 1–6, IEEE, 2015.

[14] M. Bouet, J. Leguay, T. Combe, and V. Conan, "Cost based placement of vDPI functions in NFV infrastructures," *International Journal of Network Management*, vol. 25, no. 6, pp. 490–506, 2015.

[15] A. Leivadeas, M. Falkner, I. Lambadaris, and G. Kesidis, "Optimal virtualized network function allocation for an SDN enabled cloud," *Computer Standards & Interfaces*, vol. 54, pp. 266–278, 2017.

[16] J. f. Botero, X. Hesselbach, M. Duelli, D. Schlosser, A. Fischer, and H. de Meer, "Energy efficient virtual network embedding," *IEEE Communications Letters*, vol. 16, no. 5, pp. 756–759, 2012.

[17] S. Ahvar, H. P. Phyu, S. M. Buddhacharya, E. Ahvar, N. Crespi, and R. Glitho, "CCVP: Cost-efficient centrality-based VNF placement and chaining algorithm for network service provisioning," pp. 1–9, IEEE, 2017.

[18] M. Ghaznavi, A. Khan, N. Shahriar, K. Alsubhi, R. Ahmed, and R. Boutaba, "Elastic virtual network function placement," pp. 255–260, IEEE, 2015.

[19] M. C. Luizelli, L. R. Bays, L. S. Buriol, M. P. Barcellos, and L. P. Gaspary, "Piecing together the NFV provisioning puzzle: Efficient placement and chaining of virtual network functions," pp. 98–106, IEEE, 2015.

[20] W. Ma, J. Beltran, Z. Pan, D. Pan, and N. Pissinou, "SDN-based traffic aware placement of NFV middleboxes," *IEEE Transactions on Network and Service Management*, vol. 14, no. 3, pp. 528–542, 2017.

[21] R. Riggio, A. Bradai, T. Rasheed, J. Schulz-Zander, S. Kuklinski, and T. Ahmed, "Virtual network functions orchestration in wireless networks," in *11th International Conference on Network and Service Management (CNSM)*, pp. 108–116, IFIP, 2015.

[22] P. Lv, X. Wang, and M. Xu, "Virtual access network embedding in wireless mesh networks," *Ad Hoc Networks*, vol. 10, no. 7, pp. 1362–1378, 2012.

[23] N. M. K. Chowdhury and R. Boutaba, "A survey of network virtualization," *Computer Networks*, vol. 54, no. 5, pp. 862 – 876, 2010.

[24] N. M. K. Chowdhury and R. Boutaba, “Network virtualization: state of the art and research challenges,” *IEEE Communications Magazine*, vol. 47, no. 7, pp. 20–26, 2009.

[25] T. Jiang, L. Song, and Y. Zhang, *Introduction to OFDMA*, pp. 385–405. London, United Kingdom: Auerbach Publishers, Incorporated, 2010.

[26] N. Saquib, E. Hossain, L. B. Le, and D. I. Kim, “Interference management in OFDMA femtocell networks: issues and approaches,” *IEEE Wireless Communications*, vol. 19, no. 3, pp. 86–95, 2012.

[27] Y. Sun, R. P. Jover, and X. Wang, “Uplink interference mitigation for OFDMA femtocell networks,” *IEEE Transactions on Wireless Communications*, vol. 11, no. 2, pp. 614–625, 2012.

[28] W. Chen, X. Yin, Z. Wang, and X. Shi, “Placement and routing optimization problem for service function chain: State of art and future opportunities,” *CoRR*, vol. abs/1910.02613, 2019.

[29] C. Ghribi, M. Mechtri, and D. Zeghlache, “A dynamic programming algorithm for joint VNF placement and chaining,” pp. 19–24, ACM, 2016.

[30] T. Nguyen, S. Fdida, and T. Pham, “A comprehensive resource management and placement for network function virtualization,” in *2017 IEEE Conference on Network Softwarization (NetSoft)*, pp. 1–9, 2017.

[31] D. Qi, S. Shen, and G. Wang, “Towards an efficient VNF placement in network function virtualization,” *Computer Communications*, vol. 138, pp. 81–89, 2019.

[32] M. Mechtri, C. Ghribi, and D. Zeghlache, “A scalable algorithm for the placement of service function chains,” *IEEE Transactions on Network and Service Management*, vol. 13, p. 533–546, 2016.

[33] W. Attaoui, E. Sabir, H. Elbiaze, and M. Sadik, “Combined latency-aware and resource-effective virtual network function placement,” in *2020 IEEE Canadian Conference on Electrical and Computer Engineering (CCECE)*, pp. 1–7, 2020.

[34] K. S. Ghai, S. Choudhury, and A. Yassine, “Efficient algorithms to minimize the end-to-end latency of edge network function virtualization,” *Journal of Ambient Intelligence and Humanized Computing*, vol. 11, no. 10, p. 3963–3974, 2020.

[35] F. Bari, S. Rahman Chowdhury, R. Ahmed, R. Boutaba, M. Bandeira, and O. C. Duarte, “Orchestrating virtualized network functions,” *IEEE Transactions on Network and Service Management*, vol. 13, p. 725–739, 2016.

[36] A. Leivadeas, G. Kesidis, M. Falkner, and I. Lambadaris, “A graph partitioning game theoretical approach for the VNF service chaining problem,” *IEEE Transactions on Network and Service Management*, vol. 14, pp. 890–903, Dec 2017.

[37] M. M. Tajiki, S. Salsano, L. Chiaraviglio, M. Shojafar, and B. Akbari, "Joint energy efficient and QoS-aware path allocation and VNF placement for service function chaining," *IEEE Transactions on Network and Service Management*, vol. 16, no. 1, pp. 374–388, 2019.

[38] C. Pham, N. H. Tran, S. Ren, W. Saad, and C. S. Hong, "Traffic-aware and energy-efficient VNF placement for service chaining: Joint sampling and matching approach," *IEEE Transactions on Services Computing*, pp. 1–13, 2017.

[39] N. Tastevin, M. Obadia, and M. Bouet, "A graph approach to placement of service functions chains," in *2017 IFIP/IEEE Symposium on Integrated Network and Service Management (IM)*, pp. 134–141, May 2017.

[40] W. Wang, P. Hong, D. Lee, J. Pei, and L. Bo, "Virtual network forwarding graph embedding based on tabu search," in *2017 9th International Conference on Wireless Communications and Signal Processing (WCSP)*, pp. 1–6, Oct 2017.

[41] K. Gillani and J.-H. Lee, "Comparison of linux virtual machines and containers for a service migration in 5G multi-access edge computing," *ICT Express*, vol. 6, no. 1, pp. 1 – 2, 2020.

[42] P. Gupta and P. R. Kumar, "The capacity of wireless networks," *IEEE Transactions on Information Theory*, vol. 46, no. 2, pp. 388–404, 2000.

[43] K. Jain, J. Padhye, V. N. Padmanabhan, and L. Qiu, "Impact of interference on multi-hop wireless network performance," *Wireless Networks*, vol. 11, no. 4, pp. 471–487, 2005.

[44] "Roundtower helps SAS elevate training experience with citrix solutions." https://www.citrix.com/content/dam/citrix/en_us/documents/case-study/roundtower-helps-sas-elevate-training-experience-with-citrix-solutions.pdf, accessed: 2019.07.29.

[45] "Nexus 9000 series switches." http://www.cisco.com/c/en/us/products/collateral/switches/nexus-9000-series-switches/white-paper-c11-730116.html, accessed: 2019.07.29.

[46] R. Fourer, D. M. Gay, and B. W. Kernighan, *AMPL: a modeling language for mathematical programming.* 2003.

[47] "Baron." https://www.minlp.com/baron, accessed: 2019.07.29.

[48] T. Kunz, K. Mahmood, and L. Li, "Broadcasting in multihop wireless networks: The case for multi-source network coding," pp. 5157–5162, IEEE, 2012.

[49] M. Chowdhury, M. R. Rahman, and R. Boutaba, "Vineyard: Virtual network embedding algorithms with coordinated node and link mapping," *IEEE/ACM Trans. Netw.*, vol. 20, p. 206–219, Feb. 2012.

[50] Y. Zhu and M. Ammar, "Algorithms for assigning substrate network resources to virtual network components," in *Proceedings IEEE INFOCOM 2006. 25TH IEEE International Conference on Computer Communications*, pp. 1–12, 2006.

[51] M. Yu, Y. Yi, J. Rexford, and M. Chiang, "Rethinking virtual network embedding: Substrate support for path splitting and migration," *SIGCOMM Comput. Commun. Rev.*, vol. 38, p. 17–29, Mar. 2008.