Quality of Service Ensurance for Service Function Chaining supporting Network Slicing

Ph.D. Dissertation
Doctoral School of Informatics

Author
Khalil Mebarkia

Scientific Supervisor
Dr. Zoltán Zsóka
Associate Professor

Budapest, Hungary
September 2022
Declaration

I, the undersigned Mebarkia Khalil, hereby declare, that this Ph.D. dissertation was made by myself, and I only used the sources given at the end. Every part that was quoted word-for-word, or was taken over with the same content, I noted explicitly by giving the reference of the source.

The reviews of the dissertation and the report of the thesis discussion are available at the Deans Office of the Faculty of Electrical Engineering and Informatics of the Budapest University of Technology and Economics.

Budapest, September 23, 2022

______________________
Khalil Mebarkia
Abstract

Mobile communication networks have evolved through four generations to provide users a better quality of service and experience. The 3GPP introduced a new mobile core network architecture called Evolved Packet Core (EPC) for 4G Long Term Evolution (LTE) that allows mobile users to access multimedia resources in packet data networks external to the provider’s network, such as the Internet. With the witnessed boom in the popularity of smartphones and the appearance of high-resolution video streaming, the Internet of Things (IoT), and real-time control applications such as telemedicine and vehicular communication, a vast data traffic amount is expected. The current LTE technology is not equipped to satisfy the customers’ demands by increasing traffic exponentially.

In the last few years, networks have grown anonymously and have become very complicated. The networking world is now driven by a completely different set of business needs such as high performance, scalability, flexibility, high availability, and simplicity to operate and support mobility. The new requirements call for a need to automate and manage the network by emerging technologies such as Software Defined Networking (SDN) and network function virtualization (NFV), which are playing a vital role in the design of the fifth-generation mobile networks (5G). 5G is expected to solve the issues related to high data rate, capacity, reliability, and quality of Service (QoS) for the different services provided in the network. These Emerging technologies, SDN and NFV, and the concept of network Slicing in Service Function Chaining (SFC) are playing an enabler key and fundamental feature for the 5G mobile networks.

The main contribution of this dissertation is an analytic and performance evaluation by presenting two different models to be applied in QoS analysis and to consider options for the flexibility, which shall come with 5G.

In the first contribution group, I propose network models and evaluation methods that can be applied to compare 4G and 5G-based solutions to analyze the possible effects and advantages on the QoS properties when migrating to 5G. Besides, I propose Service Traffic Engineering (STE) solutions for the dynamic calculation of chains that avoid overloads the networking layer. The solutions aim to determine service chains that meet the required bandwidth and VNF order. The solutions consider the current network load to avoid the use of heavily loaded links when it is possible.

In the second group of contributions, I extend the previously proposed models to support network slicing. I address the challenges that network slicing slicing brings. Furthermore, I introduce heuristic service chaining solutions that consider shared slicing and apply preservation of network resources for other slices to hold the QoS expectations. I show the importance of network slicing for building up to ensure low latency and guaranteed bandwidth for different services. I demonstrate the impact of slices on each other while showing the importance of the proposed solutions to avoid IP link overload.

I believe that the results of this dissertation may contribute to the general understanding of service function chaining leverage and help maintain and preserve the network resources and ensure the quality of service.
Acknowledgement

I would like to express my sincere gratitude to Dr. Zoltán Zsóka for his intensive supervision and precious guidance and advice. Dr. Zoltán Zsóka has guided me in the direction of my research at the preliminary time and taught me how to approach and solve problems. Without his continuous supervision, I could not grow up to achieve my Ph.D. degree.

I would like to express gratitude to my examiners for their insightful comments and valuable suggestions to improve this dissertation. I acknowledge all the staff members at AddICT lab and the department of Networked Systems and Services in the Faculty of Electrical Engineering and Informatics. I am thankful for the help and enthusiastic support throughout these years.

Finally, I dedicate my sincere thankfulness to my parents and my sister for their love and intensive encouragement. I am also grateful to all my friends and family members who have supported me.
Abbreviations

3GPP  The 3rd Generation Partnership Project
eNB  Evolved Node B
EPC  Evolved Packet Core
ETSI  European Telecommunications Standards Institute
GRE  Generic Routing Encapsulation
ILP  Integer Linear Program
ISIS  Intermediate System to Intermediate System
LLQ  Low-latency queuing
LTE  Long-Term Evolution
MME  Mobility Management Entity
MPLS  Multiprotocol Label Switching
NFV  Network Function Virtualized
OSPF  Open Shortest Path First
PGW  Packet Data Network Gateway
QoS  Quality of Service
SC  Service Chain
SFC  Service Function Chaining
SGW  Serving Gateway
SNMP  Simple Network Management Protocol
SR  Segment Routing
STE  Service Traffic Engineering
TE  Traffic Engineering
UE  User Equipment
VNF  Virtualized Network Function
VPC  Virtual Private Cloud
WFQ  Weighted Fair Queueing
# Contents

Abstract

Acknowledgements

Abbreviations

1 Introduction

1.1 Overview

1.2 5G Networks

1.2.1 Software Defined Networks (SDN)

1.2.2 Network Function Virtualization (NFV)

1.2.3 Service Function Chaining (SFC)

1.3 Network Slicing

1.4 Motivation

1.5 Research Goals

1.6 Structure of the Dissertation

2 Methodology

2.1 Model Definition

2.1.1 Separation of networking and functional layers

2.2 Performance Evaluation

2.2.1 Link Level Evaluation

2.2.1.1 Link Load

2.2.1.2 Link Overload

2.2.1.3 Overload Reduction

2.2.2 Ensuring QoS with Queuing

2.2.2.1 Strict Priority Queueing

2.2.2.2 Weighted Fair Queueing

2.2.2.3 Low Latency Queueing

2.2.3 Packet Level Evaluation

2.2.3.1 Packet Loss
3 Overloads in Service Chaining

3.1 Related works

3.2 QoS Modeling and Analysis in 5G Backhaul Networks

3.2.1 Modeling 4G EPC

3.2.2 Modeling 5G

3.2.3 Forming Service Chains using the multilayer model

3.2.4 Performance Evaluation

3.2.4.1 Link Level Evaluation

3.2.4.2 Packet Level Evaluation

3.2.5 Conclusion

3.3 Avoiding Link Overloads in Service Chains

3.3.1 Classification of Service Function Chaining Solutions

3.3.2 Overload in Service Chains

3.3.3 Avoid Overloads in Service Chains

3.3.4 Solutions to avoid Overloads

3.3.4.1 The Optimal Solution

3.3.4.2 The Heuristic Solution

3.3.5 Analysis of provided solutions

3.3.6 Performance Evaluation

3.3.6.1 Algeria topology

3.3.6.2 BT Europe topology

3.3.7 Summary

3.4 Bandwidth-Aware Service Chaining

3.4.1 Bandwidth-Awareness in Service Chains

3.4.2 Bandwidth-Aware Solutions

3.4.3 Performance Evaluation

3.4.3.1 Overloaded Network Connections

3.4.3.2 End-to-end QoS results

3.4.4 Summary

4 Slice-aware Service Chaining

4.1 Related works
4.2 Network Slicing Model ........................................ 63
4.3 Service Chains in Network Slicing .......................... 67
  4.3.1 Slice-Awareness in Service Chains .................... 67
  4.3.2 Slice Limitation Algorithms ............................ 68
  4.3.3 Analysis of the algorithms ............................... 69
  4.3.4 Performance Evaluation ................................. 71
  4.3.5 Summary .................................................. 73
4.4 QoS Impacts of Slice Traffic Limitation .................. 75
  4.4.1 Adjusting Limitation Parameters ....................... 75
  4.4.2 Modelling the Overloads ................................. 76
  4.4.3 Performance Evaluation ................................. 79
    4.4.3.1 Small topology .................................... 79
    4.4.3.2 Larger Topology .................................. 81
  4.4.4 Summary .................................................. 83

5 Proof of Concept Implementations ......................... 84
  5.1 SDN-based Implementation ................................ 84
  5.2 Cloud-based Implementation ............................... 84
    5.2.1 Supporting Network Technologies .................... 85
    5.2.2 Concept Realization .................................. 85
    5.2.3 Network topology .................................... 85
    5.2.4 Performance Evaluation ............................... 87
    5.2.5 Evaluation Methods ................................... 88
    5.2.6 Measurement Results .................................. 90
    5.2.7 Summary .................................................. 92

6 Conclusion ................................................... 93
  6.1 Application of Results .................................... 93
  6.2 Summary of New Scientific Results ...................... 93
  6.3 Future Research Direction ................................ 95

List of Figures .................................................. 98

List of Tables .................................................... 99

Publications ....................................................... 100

Bibliography ....................................................... 101
Introduction

1.1 Overview

Mobile networks have grown faster in the last few years due to mobile phones’ rapid adoption and intense use. According to Statista [1], mobile traffic had continuously grown to have 54% of global website traffic in the fourth quarter of 2021. Since mobile technologies have become more widely available, more than 90% of Internet users worldwide have accessed the Internet via mobile phones. According to Ericsson, [2], the monthly global average usage per mobile phone reached 11.4 Gb at the end of 2021 and is expected to reach 41 Gb by the end of 2027.

In the coming years, a variety of heterogeneous use cases such as the Internet of Things (IoT), Artificial Intelligence (AI), Augmented Reality (AR), healthcare systems, and autonomous driving is foreseen to become more prevalent. Mobile networks need to adopt new technologies to provide a new set of services to support these cases more effectively. As a result, mobile networks are increasingly driven by a distinct set of business requirements, including the ability to deliver high performance, scalability, flexibility, high availability, and ease of operation. Internet Service Providers (ISPs) expand their businesses into new areas and industries to support a new generation of mobile networks. They intend to address a broad range of needs and application situations. Simultaneously, they seek to simplify operations and accelerate the deployment of new services by incorporating automated management and orchestration features that facilitate flexibility.

1.2 5G Networks

Industries, research institutes, and worldwide mobile operators are putting in enormous effort to standardize and prepare to introduce fifth-generation mobile networks (5G). The 5G networks are projected to enable the efficient and cost-effective launch of several new services, establishing an ecosystem for technological and business innovation. Then, gradually, the 4G era will come to an end of life and soon be replaced by 5G. At the moment, 5G subscriber uptake is predicted to be faster than when 4G was introduced; it is expected to reach 90% in Europe and North America by the end of 2027. [2].

5G technology primarily targets mobile broadband use cases, delivering increased system capacity and data rates. Telemedicine, for example, is critical for increasing healthcare access in remote rural and urban locations. With 5G networks, physicians and medical
professionals can access medical data containing high-resolution medical photos and videos anytime and from any location. Remote, real-time consultations with primary care physicians and specialists would result in cost-saving, higher convenience, and improved and more timely medical outcomes. Telemedicine and healthcare are not the only applications for which 5G networks can be used. Figure 1.1 illustrates other options too. For instance, automated control systems in the industry can manage repetitive operations with other technologies such as Internet of Things sensors, artificial intelligence vision cameras, and autonomous robots. Autonomous vehicles are another application of 5G networks, where future cars communicate and collaborate.

![Figure 1.1: An example of 5G use cases](image)

To achieve 5G standards, ISPs must address difficulties not adequately handled by 4G, including capacity, data rate, end-to-end latency, huge connections, and ensuring Quality of Service (QoS). The following challenges are predicted to be addressed by 5G networks:

- **Data rate:** when it comes to the rate at which data may be transported end-to-end, 5G allows data transfer rates exceeding 10Gbps.

- **Capacity:** to comply with an exponential rise in traffic, 5G networks will need to enhance their traffic-handling capacity. It is defined as the total amount of traffic the network element can handle.

- **Massive connectivity:** the exponential growth of heterogeneous devices will require improved global connectivity. It plays an essential role in IoT use cases.

- **Latency:** applications with very low delay tolerance shall be supported. For instance, in the communication of autonomous automobiles, 5G shall provide an end-to-end latency of 1-10 milliseconds (ms)[3].

- **Ensuring QoS:** high-priority applications with prescribed requirements shall be executed reliably. It can be ensured by deploying a collection of technologies that work together to enable the needed throughput despite constrained network capacity. The study of QoS is gaining growing importance, and 5G networks are expected to improve these characteristics significantly [4].
The rise of new use cases for 5G networks, such as IoT, has resulted in a surge of interest in Service Function Chaining (SFC), Software Defined Networks (SDN), and Network Function Virtualization (NFV). These enabling technologies make it easier for ISPs to offer new services on demand in 5G networks that can be virtualized and programmable. As a result, SFC, SDN, and NFV have attracted many academics and industry participants. Introducing these key enabling technologies is a critical step in the evolution and development of the infrastructure for 5G networks. The following subsections detail them from the 5G support point of view.

### 1.2.1 Software Defined Networks (SDN)

Software-Defined Networks (SDN) is an emerging technology that aims to simplify network monitoring and management to improve network performance. It reshapes the architecture to provide centralized management with automation of steering the traffic demands through the network. The Open Network Foundation standardized the SDN architecture by separating the Control Plane of conventional networking devices (e.g., switches, routers) from the Data Plane, as shown in Figure 1.2b. It enables more efficient management of ISP networks by making the configuration programmable. In such networks, the routes and settings can be altered dynamically on the routers without human interaction, enabling more precise control and hence better resource management. Additionally, SDN enables the implementation and testing of novel protocols that are difficult to install on legacy networks. As a result, it introduces the new potential for optimizing network resources, such as available bandwidth on lines, and increasing the ISP’s profit.

![Figure 1.2: Traditional vs SDN Networks](image)

The main components in the architecture SDN are separated into two layers as follows:

- **Data Plane** consists of primary networking devices (switches) connected via various links. It permits the transport of packets between hosts via different protocols.

- **Control Plane** contains one or more software entities called SDN controllers, which control a collection of data plane resources, allowing for dynamic access and administration. The control plane’s primary job is to decide where traffic should be routed and advertise this information to the switches.
Several implementations exist for the components [7], e.g., Open vSwitch is a basic switch, and ONOS is an example of an SDN controller. The standardized communication protocol among the components is OpenFlow.

While SDN presents some challenges, it also has some limitations [5], including maintenance, which is a crucial element of networking. Additionally, centralization implies the risk of having a single point of failure. Although SDN enables an automated, programmable method of managing the network, swapping a legacy architecture for SDN might be time and money-demanding because the entire set of network devices must be reconfigured. On the other hand, reconfiguration must also incorporate network operations since SDN’s goal can be accomplished when all the network management functions are similarly available.

1.2.2 Network Function Virtualization (NFV)

In traditional networking, such devices as Proxy Servers, Load Balancers (LB), Domain Name Systems (DNS), Firewalls (FW), and Intrusion Detection Systems (IDS) are placed at various Customer Premise nodes. In the nodes, proprietary and mostly specialized hardware are employed, as illustrated in Figure 1.3a. These devices provide various network functions required in services, bringing security, performance, and cost advantages.

The traditional implementation of network functions is inflexible since the hardware devices can only be connected and/or moved manually to the network, and their management can hardly be automated. Although incorporating the functions into specialized hardware devices increases performance and revenue, deploying and upgrading several devices is expensive and complex. Additionally, it has a short life span [8].

![Traditional vs NFV Approaches](image)

**Figure 1.3:** Traditional vs NFV Approaches

Network Function Virtualization (NFV) overcomes the challenges and limits associated with traditional network function deployments. The NFV method leverages the advancements of virtualization technology which implements network functions on servers, switches, and storage devices. These elements can be housed in Datacenters, or centralized locations [9]. NFV’s objective is to radically change how ISPs build, deploy, and maintain network infrastructures. NFV enables the deployment, execution, and management of network functions or operations on Virtual Machine (VM) instances. To deploy a new
network function or extend the capacity of a running one, the operator might establish a new virtual machine (VM) or scale up an existing one [10].

VMs can typically be instantiated on on-premises servers, and a single server can perform many functions as long as the server capacity is sufficient. Virtual Network Function (VNF) is a software-based implementation of the network function deployed in a virtualized infrastructure. In general, a VM could run more types of VNFs, but typically it is supposed that each VM is dedicated to only a single function. When the VM’s capacity is insufficient, even more than one VMs can be applied in a node to implement one function, and they handle the demands in a shared manner. For simplicity, in this work, I consider no more than one VM per VNF-type in a node and ignore the question of its serving capacity. According to this assumption, I mention only the VNF when speaking about the virtualized functions and neglect the VM instance that runs it.

On the other hand, it is not sure that any VNF-types are available in every node. This capability determines whether a VNF of a given type can be applied in a node or not.

The advantage of virtualization is the higher flexibility in resource usage and network management since NFV allows dynamically operating and orchestrating VNFs. Additionally, it can optimize network resources, lower expenses, and increase revenues [11].

ISPs provide network services to their subscribers, and as mentioned before, such a service can require network functions to pass through. From the NFV point of view, a network service requires a collection of VNFs in a given order according to its specification. Figure 1.4 illustrates an example of a network service.

![Figure 1.4: An example of Network Service](image)

The European Telecommunications Standards Institute (ETSI) specifies three modules [12] for the NFV architecture:

- **NFV Infrastructure (NFVI)** outlines the hardware and software components for computing, storage, and networking resources that can be utilized to create and run VNFs, such as hypervisor nodes and virtualization clusters.

- **VNF Manager** is responsible for VNF instances management, including scaling, changing operations, adding new resources, and communicating the states of VNFs to the orchestrator.

- **NFV Orchestrator** is responsible for resource orchestration, which is critical for ensuring sufficient computing, storage, and network resources to provide network services.

Figure 1.5 illustrates the connection among the modules.

Although they share numerous objectives and characteristics, SDN and NFV are two distinct and independent technologies that can operate independently. For example, the SDN controller optimizes traffic routing, whereas the NFV orchestrator optimizes the placement and management of VNFs.
1.2.3 Service Function Chaining (SFC)

According to the concepts mentioned above, a network service request is a demand of traffic that may require to be transmitted through a sequence of VNFs, i.e., the packets belonging to this particular traffic flow shall be processed by them. According to ETSI [13] and the RFC 7665 [14], the mechanism for creating such a VNF sequence, also called chains is referred to as Service Function Chaining (SFC). The required VNFs and, optionally, their order are inputs for SFC. Its key benefit is to determine the nodes where the VNFs run, automate the process of classifying traffic, and implement a certain policy to lead the traffic through the selected nodes [15], i.e., creating a service chain (SC).

A network node itself can be a router that might not be able to run a VNF, but I extend the VNF capability concept to routers to simplify the view. In this sense, a router is capable of a given VNF-type when a server is connected to it that can run the virtual machine which implements the given VNF.

Figure 1.6: An example of realizing an SFC

Figure 1.6 shows how SFC works in a network, where traffic between the source and destination should be routed through the given set of VNFs. In this example, the sequence is specified as the red, green and blue colored VNFs. The traffic flow passes the nodes a, c, e, b, d and finishes with the packet arriving at the node f. ISPs can offer different orders of VNFs if it is not specified for the service. This gives more freedom for SFC and a wide range of service chains for network services.
In the example network on Figure 1.6 I assume to have quite poor VNF capabilities in the nodes, i.e., the nodes $c$, $e$ and $d$ are capable to run only VNF $\text{red}$, $\text{green}$ and $b$ respectively.

In some cases, for example, when the objective is to lower a node’s energy consumption, combining numerous functions on a single node may be more convenient than separating them. Despite this, it may be advantageous to distribute VNFs among more nodes because, otherwise, some further services that use the same VNFs might overload the node or link resources. With more nodes of similar capabilities, starting and using a new VNF in a different location may also be possible, or relocating an existing VNF from one site to another can be applied when congestion happens. VNFs can be arbitrarily located inside the virtualized architecture according to the nodes’ capabilities. They can then be executed on demand, making the chains adaptive to the current traffic load. However, the capability distribution among the nodes is a question of network planning.

![Diagram](image)

**Figure 1.7:** SFC and service chains for different network services

Due to the limited VNF capabilities in the network in Figure 1.6, the SC selected by SFC uses for traffic flow the shortest path that passes the VNFs in the given order, as the red line shows it. In an extended scenario, where multiple demands of different services are considered, service chains apply the same VNFs. The example in Figure 1.7 shows SFC for three demands with different VNF requirements. The network nodes’ color refers to their VNF capability, which is set the same way as in the previous example.

Also, the SCs are created here using the shortest path to the destination while considering the demands’ requirements. The demand 1 requires VNFs $\text{red}$, $\text{blue}$ and $\text{green}$, and its SC follows the path $a-c-e-b-d-f$, just as before. It is important to recognize that the paths used by the second and third demands differ from this, although the source and the destination nodes are the same. The demand 2 requires only the $\text{green}$ VNF and is routed via the path $a-b-d-f$. The demand 3 has to pass through two VNFs, $\text{red}$ and $\text{blue}$, and is routed over the path of nodes $a-c-e-f$. 
1.3 Network Slicing

Numerous advantages, including extreme high-speed fast data transmission and lower latency, have already been mentioned concerning 5G. Depending on the use cases, many new Quality of Service criteria is expected to be met, which can enormously differ from traditional ones and each other. New and heterogeneous use cases appearing nowadays or in the near future, including the application of SDN and NFV, imply the need to create different and independent logical networks over the same infrastructure. The capacities and capabilities in these logical networks are designed to meet the specific criteria of the services provided in the individual use cases. The concept of Network Slicing is subdividing an existing network into more slices, i.e., unique logical networks that share network, storage, and computation resources [16].

On the one hand, the Next Generation Mobile Networks (NGMN) standard [17] defines the slice as a set of network services that can be executed on top of physical resources. The service set in the slice represents various types of traffic demands, each having its own statistical attributes and quality standards. It might require particular VNF sequence, SFC, and routing solutions to be handled appropriately [18].

On the other hand, besides network services, network link capacity and VNF resources should also be arranged into a logical network in order to meet specific needs. Network slicing is a technique that allows allocating resources to a specified slice of the network, although the exclusive resource dedication is not the only option [18]. In the case of a firewall, for example, assigning it to a specific slice rather than using a shared firewall can enhance security and performance. In the case of a load balancer function for outgoing traffic, the separation might not be so essential, and more slices can use the same VNF. As a result, the advantage of network slicing for ISPs is the ability to deploy in a slice only the VNFs that are required for services of specific users and market groups. Network slicing for 5G allows the ISPs to implement the concept of network-as-a-service, i.e., establish several virtual networks on a shared infrastructure and expand their business [19].

Multi-tenancy is an architecture that allows a single instance of software to serve many clients. Each customer is referred to as a tenant. Tenants may be given the ability to customize some aspects of the software. Multi-tenancy architecture in cloud computing refers to multiple cloud vendor customers using shared computing resources [20].

![Image: Single vs multi-tenancy]

Figure 1.8: Single vs multi-tenancy

In single-tenant architecture, the client is the tenant. Each user has supporting infrastructure and a dedicated server in the single-tenant environment. Users cannot share single tenant products, but they can customize them to their requirements. Single-tenant environments are more reliable because the activities of one user cannot affect anyone else. However, some drawbacks are associated with single-tenant environments, including the
cost, which costs more than multi-tenant cloud architecture. Each new user requires a new instance, and everyone has an associated cost [21].

Multi-tenancy has some level of customization, which could be possible, but it shares the same application. Thus, a multi-tenant cloud management platform has several benefits, such as low cost, efficient resources, low maintenance, and allowing for the exchange of applications, resources, databases, and services.

In order to allow the simultaneous operation of numerous logical networks over a single shared physical network, the implementation of Network Slicing is often based on SDN principles. This technology provides options for software-driven coordination of resources to manage the network and supply its behavior fulfilling the demands of the offered services [22]. Using network devices with a high degree of management and configuration flexibility enables the SFC process to take into account loads of available resources, which can potentially impact service quality in the slice [23].

1.4 Motivation

In order to respond to the challenges of new use case scenarios, 5G networks support new requirements for a wide range of services. ISPs are moving toward the use of enabling technologies to change the way how networks operate. The objective is to deliver the expected requirements and meet the new Quality of Service criteria.

It is essential to integrate SDN and NFV to establish a virtualized network architecture that can be used to deliver a variety of services. Allowing NFV, SFC solutions determine where and how many VNFs should be placed and chosen for serving a service demand. SFC introduces new challenges, one of which is in the context of resource usage since the VNF capabilities in nodes are not the only inputs that influence the service chain’s form. Additional limits, such as the link capacity and load, shall be involved to maintain network resources and assure QoS.

If many users request the same service at the same time, maybe along with additional requests from different services, then the SCs of the requests may use the same network link many times. It can lead to the link load increasing so much that it exceeds the link capacity. The link becomes congested or overloaded [24], and its transmission performance depreciates. Overloaded links in a network can seriously affect the network’s performance, resulting in a degradation of QoS.

This dissertation intends to solve issues associated with resource management in use cases involving 5G networks. It presents SFC solutions that consider the NFV concept and might be applied in environments where NFV orchestrator and SDN controller elements are available. These SFC solutions should enable the dynamic creation of service chains for a wide range of network services. Their main goal should be to consider the current state of network resources to ensure QoS. In the case of applying network slicing, the SFC should be enhanced with the option of preserving resources for the different slices when the service chains are calculated.

In order to present the dissertation’s substance of the matter, Figure 1.9 illustrates a summarized structure of my research work which uses the leverage of SDN, NFV, SFC, and network slicing.
Research Goals

In order to meet requirements associated with a service request, VNF-capable nodes and network connections with appropriate resources must be available in the network. On the one hand, the request for a specific service must go through nodes with VNF capabilities according to the ordered set of VNFs prescribed for the service. On the other hand, the traffic must be routed through network links. Both sets should be determined while creating a service chain, i.e., by performing the SFC process.

I aim to overcome the difficulties associated with creating service chains, which are discussed in Section 1.4.

In order to address this double challenge of SFC solutions, I propose a multilayer model that allows the creation of service chains considering both VNF and network capabilities. Furthermore, I offer several different architectures for classifying SFC solutions that apply this model. I propose SFC algorithms for the dynamic calculation of service chains that meet the bandwidth and VNF order requirements. As a result, the SFC algorithms consider the load of networking links to avoid overloads when at all possible. Note that the VNF capabilities are not provided by the solutions but are taken as input, while the VNF placement is performed by determining the service chain.

I extend the same multilayer model to include network slicing as a new feature. The aim is to handle traffic needs from multiple network slices; therefore, SFC will be based on the requirements assigned to the particular network slices. I propose slice-aware SFC algorithms for the dynamic computation of service chains while considering a load of slices. The goal is to keep the network resources of other slices less affected to meet their QoS requirements.

I analyze the proposed solutions using calculation methods that consider connection capacities and packet-level parameters of the traffic and queues. In order to demonstrate the benefits of the proposed solutions, results on link utilization and several QoS metrics are evaluated. The dissertation aims to introduce new SFC solutions and demonstrate their effectiveness in preventing link congestion, boosting demand acceptance, and assuring QoS.

The research objectives can be divided into various categories:

- the first part investigates the migration from 4G to 5G and proposes SFC algorithms to create service chains,
- the second part investigates the SFC solutions that avoid the congestion of links while creating service chains,
the third part focuses on considering network slicing in SFC to avoid QoS degradation in the different slices.

The main contributions of the dissertation are illustrated in Figure 1.10 and can be summarized as follows:

- **Contribution 1**: To get insight into the network QoS in the 5G backhaul and the introduction of NFV, I put the 4G and 5G architectures under the scope. I offer multilayer models for these architectures, including the appropriate VNF-capable nodes. I present SFC solutions and evaluate the 4G and 5G cases to recognize the effects that may lead to decrease QoS (Section 3.2).

- **Contribution 2**: Motivated by the results on the influence of congested network links produced by the previous contribution’s workout, I extend the multilayer model to consider bandwidth constraints. I present different SFC solutions which consider the sequence of required VNFs and avoid network link overload (Section 3.3).

- **Contribution 3**: I propose bandwidth-aware SFC solutions which provide service chains using lower loaded network links to avoid congestion (Section 3.4).

- **Contribution 4**: The multilayer model is extended to support network slicing. I present slice-aware SFC solutions that consider the network link resources assigned to the slices during the service chain computation to preserve network resources (Section 4.3).

- **Contribution 5**: Starting from the SFC solutions introduced in the previous contribution, I introduce essential policies for setting up the parameters according to the packet-level processing properties. In order to get a more comprehensive view, I extend the calculation of some relevant QoS metrics to support slices (Section 4.4).

![Figure 1.10: Structure of the dissertation’s contributions](image-url)
Additionally, besides the evaluations made with simulation, I illustrate the effectiveness of SFC solutions in preventing network overload using real-world networks. I have implemented an environment on an on-demand cloud computing platform for this proof of concept, applying segment routing to realize service chains.

1.6 Structure of the Dissertation

In the previous sections, I briefly explored the background and new technologies related to the introduction of 5G networks. I discussed the main problems and challenges in Section 1.1. I gave the motivations of the dissertation in Section 1.4, and summarized the goals of my research in Section 1.5.

The rest of the dissertation is structured as follows. In Chapter 2, I introduce a network model with multiple layers; this model is then applied throughout my whole work. In addition, Section 2.2 introduces the methods and metrics used to evaluate and compare the solutions I propose in the other parts of the dissertation. I describe the followed evaluation process in Section 2.3.2, and introduce the hypothetical backbone network used as a generic case study scenario for evaluation.

I split my contributions into two groups. The first group is presented in 3. The objective here is to find service function chaining methods that help to ensure QoS and avoid overloads in network links. I go into the technical details of the 4G EPC and 5G Core network architectures to show the differences from the SFC point of view. I introduce models for their backhaul segment and SFC methods to be applied in Section 3.2. In Section 3.3, I introduce the effect of link overloads in the multilayer model and its possible consequences on QoS. I propose and evaluate service function chaining solutions that aim to avoid overloads. Then, I finish this group with the contribution of proposing bandwidth-aware solutions in Section 3.4.

The second group of contributions is presented in Chapter 4. It includes discussing the options for extending the multilayer model proposed in Chapter 2 to support network slicing in Section 4.3. I also propose slice-aware SFC solutions that limit the bandwidth used by the slices to preserve network resources for other slices. I propose policies for the slice-aware solutions and analyze them from the network QoS point of view in Section 4.4.

Chapter 5 presents two proofs of concept applications using SDN and cloud-based environments. It also lists the possible application of the solutions proposed in the dissertation.

Finally, in Chapter 6 I conclude the work by summarizing the contributions and looking at possible further research directions in this field.
Methodology

Various surveys were published in the research field of multilayer networks providing an in-depth examination of the technical literature and outlining the attributes of various multilayer architectures [25, 26]. It is important to note that the terminology used to refer to systems with multiple distinct relationships has not yet reached a consensus. Different works from various fields use similar terminologies to describe various models. In contrast, different works from the same field sometimes use different names to describe the same models, elements, and relationships.

This chapter presents a hierarchical multilayer model for computer networks that can be used to support SFC and be applied in a variety of situations. Each layer comprises nodes and connections, which are links between the nodes. Nodes and connections can be of different types.

2.1 Model Definition

Although it gives a general background to the dissertation, it was initially created to represent a potential implementation of 5G core networks or any other NFV-enabled networks. Let me start by giving the conceptual view through an example.

The suggested network model is represented from several distinct views in Figure 2.1. Figure 2.1a presents the hierarchy or 3D perspective of the layers in the model. In this example, three layers are placed on top of the other. Figure 2.1c illustrates the same three layers as network graphs that are connected at the same time.

In order to introduce the model’s concept, a hypothetical implementation of a 5G network, or any NFV-enabled network, is illustrated in Figure 2.1b. The model has three layers, and each layer has a distinct set of connection types and node types, like a router, VNF-capable node, or traffic endpoint. The traffic layer is composed of the traffic (or service) demands between two endpoints, shown by a blue line in the Figure. These endpoints are responsible for sending and receiving traffic. The illustrated example considers a service demand from endpoint (1) to endpoint (2).

The Functional Layer $\mathcal{L}_F$ contains functional connections that link the VNF-capable nodes, i.e., the VNFs $v_1$, $v_2$, and $v_3$ located at various sites can be connected through them. Functional links are also used for connecting endpoints to VNF-capable nodes, thus, providing a connection between an endpoint and a VNF. I assume that the traffic
demand from endpoint (1) towards (2) requires the series of VNFs \( v_1 \), \( v_2 \), and \( v_3 \) in this given order. A basic SFC solution is employed in the example scenario, which determines the shortest functional path considering the VNFs that must be traversed along the way. The functional links utilized in the chain are shown by the bold yellow lines, while further links in the functional layer, not used in the chain, are represented by the other yellow lines.

The Networking Layer \( \mathcal{L}_N \) is composed of the routing nodes, the endpoints, and the VNF-capable nodes. The network connections between them are the links that will carry the traffic in the Internetworking layer of the TCP/IP stack or the L3 layer in the ISO/OSI model. Figure 2.1b presents network connections as green dotted lines.

Dashed-red lines represent the path traffic takes during transmission across the networking layer. It passes through the nodes \( n_1 \), \( n_2 \), \( n_3 \), and \( n_4 \), then returns to \( n_3 \) and terminates at the endpoint (2). This path is derived from the applied functional connections and their mapping over the network connections.

In the formal description of the layered model, I consider a network represented by a set of graphs \( \mathcal{G}_l = (\mathcal{V}_l, \mathcal{E}_l) \), where \( \mathcal{V}_l \) denotes the set of vertices and \( \mathcal{E}_l \) to the edges in layer \( \mathcal{L}_l \). In order to support SFC and network link loads, I consider two essential graphs: the networking graph \( \mathcal{G}_N \) and the functional graph \( \mathcal{G}_F \). These graphs are constructed using the networking layers \( \mathcal{L}_N \) and functional layers \( \mathcal{L}_F \), as indicated in Figure 2.1c.

The networking graph \( \mathcal{G}_N = (\mathcal{V}_N, \mathcal{E}_N) \) is illustrated in Figure 2.2b, and consists of networking edges \( \mathcal{E}_N \) that represent the connections between the networking vertices \( \mathcal{V}_N \). The networking edges \( \mathcal{E}_N \) are characterized by their capacity \( C \). The value \( C_l \) is a duplex capacity of the networking edge \( l \in \mathcal{E}_N \).
The functional graph $G_F = (V_F, E_F)$ is illustrated in Figure 2.2a, and consists of the set of functional edges $E_F$ that represent the connections between functional vertices $V_F$. Since these connections shall be mapped over the networking layer, the functional nodes need a mapping to the networking nodes, i.e., $V_F \subseteq V_N$.

![Figure 2.2: Networking and Functional layers’ representation](image)

The path $H_f$ is the mapping of a functional edge $f$ in the networking layer $L_N$. It represents a set of networking edges $l \in E_N$, which are used to determine the functional path of $f$ in the networking layer $L_N$. For each functional edge $f$, one can determine its mapping to one or more networking edges $E_N$, for example, by using routing protocols like OSPF or IS-IS. I assume that the edges in each $H_f$ form a continuous and loop-free path from the starting vertex to the ending vertex of edge $f \in L_F$. The mapping defines the logical connection between the edges in the functional layer $L_F$ and the networking layer $L_N$.

Figure 2.3 shows the graphs of the functional $L_F$ and networking $L_N$ layers and some examples of the mapping $H_f$. The functional edges $f_1$ and $f_2$ have the mapping $H_{f_1}$ and $H_{f_2}$, respectively, both consisting of sets of networking edges in $G_N$.

![Figure 2.3: Functional and networking layers](image)

Let $U$ be a set of possible VNF types that can be implemented somewhere in the network. For each functional vertex of $v_f$, there can be a subset of $U$ that gives the capability of vertex $v_f$. The VNF-capable nodes are those functional vertices in $V_F$ where at least one VNF-type is available, i.e., a VNF instance can be started if needed. The vertex $v_f$ is considered to be a VNF-capable node if $U_{v_f} \neq \emptyset$, and $U_{v_f} \subseteq U$.

The traffic layer contains traffic or service demands or requests denoted by the letter $R$. $r \in R$ is a simplex demand of bandwidth that needs to be served from a traffic source to a destination across the functional layer $L_F$. It requires passing a sequence of VNFs, i.e., it should go through nodes that are VNF-capable and offer the given types of VNFs. The traffic request $r$ is characterized by bandwidth $b_r$ and the ordered set of VNFs $U_r$.

Performing an SFC solution in the model means finding a path for request $r$ from source to destination that meets VNF-capable nodes offering VNF-types according to the require-
ments specified in $U_r$. The applied SFC solution calculates the service chain, which is represented as path $P_r$ in the functional graph $G_F$, i.e., a series of functional edges. If two consecutive VNF types of the required sequence are available in the same VNF-capable, functional node, no functional link is needed to pass between them. It can be supposed to use one VNF right after the other in the same node, i.e., the length of $P_r$ can be less than the number of required VNFs. In this work, $P_r$ is sometimes referred to as chain or SC.

![Figure 2.4: Path $P_r$ in Functional Layer $L_F$](image)

A possible chain $P_r$ in the functional layer $L_F$ is illustrated by the bold-orange lines in Figure 2.4. The request $r$ is considered to require the use of VNFs $U_r = \{u_1, u_2, u_3\}$, and the VNF-type capabilities of the functional nodes are shown in the square representing the node.

The chain $P_r$ in this example consists of the functional edges $P_r = \{f_1, f_2, f_3, f_4\}$, it and satisfies the VNF sequence given in $U_r$. The set of the selected functional edges might be different for another request if it has a different source or destination node or requires a different ordered set of VNFs. A reason for the latter can be that the request is an instance of a different service.

In order to determine a route $P_r^N$ in the networking layer $L_N$ for the request $r$, the mappings $H_f$ of functional edges $f_i$ in chain $P_r$. In Figure 2.5, the brown edges represent the set of networking edges $E_N$ used to realize the route $P_r^N$ in the networking graph $G_N$. $P_r^N$ is defined as the concatenation of networking edges used in $H_f$ mappings:

$$\hat{P}_r^N = \bigcup_{f_i \in P_r} H_f$$

Since $H_f$ mappings for different $f$ functional edges can contain common elements, the route $P_r^N$ can not be considered as a simple set but rather as a list of network links. A link might appear in it at more than one position. In addition, the route $P_r^N$ takes into account the direction of each networking link and the sequence in which they are connected. This concept is essential for the proper calculation of the load in the networking layer $L_N$. For example, if a networking connection $l$ appears twice in the route $P_r^N$ in the same direction, the request loads it both times.

![Figure 2.5: Path in the functional and networking layers](image)
Different methods can be applied to determine the pathways in the two layers. Except in the case of the fully integrated management of the layers, the calculation processes can be separated as follows:

- First, in advance, the mapping of the functional links is determined in the networking layer.
- Then, for each arriving request, the service chain is determined as a set of functional links that satisfy the requirements of the sequence of VNF-types.
- Finally, the traffic is routed over the networking links used in the mapping of each functional link in the applied SC.

### 2.1.1 Separation of networking and functional layers

Figure 2.6 illustrates the problem of independently handling the functional and networking layers. Figure 2.6c shows the functional vertices $V_F$ and edges $E_F$ indicating the VNF capabilities for the vertices. The functional layer consists of functional vertices ($k$, $l$, $m$ and $n$) connected with functional edges $E_F$. The VNF-types $u_1$, $u_3$, $u_3$ and $u_4$ are offered in vertices $k$, $l$, $m$ and $n$ respectively.

Figure 2.6d illustrates the networking edges $E_N$ and the vertices $V_N$. As previously stated, the functional vertices ($k$, $l$, $m$ and $n$) in the networking layer $L_F$ are likewise considered to be networking vertices in the networking layer $L_N$. Besides these VNF-capable vertices, the vertices $a$ and $b$ are just pure forwarding vertices and do not implement any VNFs.

In this example two requests $r_1$ and $r_2$ are assumed, both between the vertices $s$ and $d$. However, the first demand requires the VNF-types $u_1$ and $u_2$, while the second demand requires $u_3$ and $u_4$.

![Functional and Networking Layers](image)

**Figure 2.6:** Example for the paths of two requests

The path shown in red lines is the set of functional edges that are used in the chain $P_{r_1}$ to fulfill the requirement of request $r_1$. It starts at the node $s$ and proceeds through $k$ and $l$ before terminating at node $d$. The path in blue lines represents the set of functional edges used in the chain $P_{r_2}$, starting from $s$ passing by $m$ and $n$ then ending at vertex $d$.

As mentioned earlier, the requests $r_1$ and $r_2$ use the mapping $H_f$ of each functional edge $f$ in chain $P_{r_1}$ and $P_{r_2}$ to determine routes $P_{r_1}^N$ and $P_{r_2}^N$ in the networking layer $L_N$. In this layer the requests $r_1$ and $r_2$ are routed over the links $(s-a-k-l-d)$ and $(s-b-m-n-d)$ respectively. These routes are shown as dotted red and blue edges in the networking graph in Figure 2.6d.

The functional edge between vertices $s$ and $k$ is mapped into the networking layer as a path of two hops through vertex $a$. The mapping of the functional edge between vertices $s$ and $m$ also uses two hops but through the vertex $b$. As one can see, in this case, the two sets of edges in the networking routes $P_{r_1}^N$ and $P_{r_2}^N$ are independent.
Figure 2.7 shows the networking route for the same two chains $P_{r_1}$ and $P_{r_2}$ created for requests $r_1$ and $r_2$. Identically to the previous case, the chains $P_{r_1}$ and $P_{r_2}$ consist of three independent functional edges. The difference comes from the mapping $H_f$ of edge $f$ between vertices $s$ and $m$ using the vertex $a$ instead of $b$ as illustrated in curved-dotted links in the Figure. In this case, although the functional layer paths are still independent, the networking routes $P^N_{r_1}$ and $P^N_{r_2}$ are no more independent but have the edge $s-a$ in common.

![Figure 2.7: Independency of functional and networking layers](image)

The two cases illustrated in Figures 2.6 and 2.7 highlight that the separated use of functional and networking layers can mislead the decision of SFC. Independence of paths might be supposed where they are not. Although functional links may not seem highly loaded by the traffics, they can cause even overloads if using joint edges in their mappings.

Another issue of managing the layers separated is the risk of not correctly handling the link loads if there are uncontrolled or controlled loops in the routes. Uncontrolled loops appear in networks due to configuration errors, leading to severe performance problems. In such a situation, the traffic at the looped nodes gets routed repeatedly in the same direction. This behavior leads to reusing the same resources practically, which causes an infinite multiplication of the traffic on them. An uncontrolled loop quickly overloads the resources, while the traffic never arrives at its destination.

A controlled loop is less problematic than an uncontrolled one. In this case, traffic will use some resources more than once but will not be trapped in an unending loop. Controlled loops may cause performance problems or slower traffic transmission due to network link congestions.

![Figure 2.8: Example for inevitable loop in SFC](image)

The controlled loops at the networking layer come from concatenating loop-free sections into a route that contains a loop. The control over it is solved by directing the traffic step by step through the sections. In my model, the sections are functional links with no loops in their mapping.

Additionally, a loop in the functional layer can appear in specific constellations of VNF capabilities’ location and VNF sequence requirements of the services. In such cases, the chain $P_r$ will contain inevitable loops to satisfy the VNF requirements, although these loops are also controlled in the above sense. If there are loops in the functional layer, there are also loops in the networking layer.

Figure 2.8 shows a simple example of having a situation in which a loop in the path is inevitable. In the topology, there is only one link $l$ between the nodes $i$ and $j$. The request $r$ requires the VNF sequence $u_1$, $u_3$, $u_2$, and $u_4$. When routing the request $r$ in the networking layer, it is impossible to prevent entering a controlled loop since $u_1$ and
are not consecutive in the sequence, and node $i$ has to be passed through more than once. As a result, the actual path of $\mathcal{P}^N_r$ in the networking layer contains a loop passing link $l$ two times, and this has to be considered when analyzing the loads and congestions on the link.

### 2.2 Performance Evaluation

The model described in Section 2.1 can not only be applied to the SFC problem but allows for the estimation of average load values on functional or networking links. Although with some of its derivative values, this link-level analysis gives a good point of reference to the network’s behavior, a more precise approach is required for performing a QoS analysis. The level of quality and its variation with the network load can be examined only using a packet-level evaluation of link and traffic behavior. Some of the Key Performance Indicators (KPIs) are identified in this section. They help to analyze the QoS for the solutions proposed in the dissertation. These KPIs are influenced by link capacities, packet-level traffic, and service descriptors and show a solid connection to the number of overloaded links and link utilization results. The following list organizes the values under the scope:

- **Link Level Evaluation**
  - Link Load
  - Link Overload
  - Overload Reduction

- **Packet Level Evaluation**
  - Packet Loss
  - Packet Delay

#### 2.2.1 Link Level Evaluation

Link-level performance evaluation is used to analyze the performance of networking links under various scenarios, including congestion on the link. I analyze the performance of networking links in the route $\mathcal{P}^N_r$ by calculating the load on these links to identify the overloaded links.

#### 2.2.1.1 Link Load

Allowing multiple parallel traffic requests may induce situations where the requests’ service chains share some functional links. Although the required VNF sequences $U_r$, and thus, also the chains $\mathcal{P}_r$, can be different, the latter may contain joint edges. These functional edges will be loaded by all the concerned traffics. The mapping $H_f$ of the functional edge $f$ determines which links in the networking layer will be used by a chain and loaded by the traffic. Moreover, as mentioned in Section 2.1.1, even if two chains are independent of the functional edge point of view, the dependencies coming from the $H_f$ can lead to loading the same networking edges.
To elaborate on the networking edge’s load, which is represented by the value $\sigma_l$, I calculate the sum of all bandwidth requests that use link $l$ in $L_N$ as follows:

$$\sigma_l = \sum_{r_i \in P_r} \sum_{j \in P_{r_i}} b_{r_i} I(j = l) \quad (l \in P_r^N)$$

(2.1)

$\sigma_l$ is the sum of all requests’ bandwidth $b_{r_i}$ which pass through the link, i.e., where $l \in P_r^N$. $I(j = l)$ is an indicator variable, which equals to 1 if link $j$ is exactly $l$. We must go through all requests $r_i$ that contain the link $l$ in their routes, and for each of these routes, we must go through all of the links and sum up the total bandwidth $b_{r_i}$ coming from request $r_i$.

If request $r_i$ uses network link $l$ more than once in its route, its bandwidth is summed up in the value $\sigma_l$ as many times as the link $l$ appears in $P_r^N$. In addition, the value $\sigma_l$ can be calculated for both the forward and backward directions, considering the bandwidth characteristics of the traffic demand.

The load $l$ is defined as a relative value representing the utilization of network resources at a networking edge $l$. It is defined as the sum of demands’ bandwidth $\sigma_l$ divided by the capacity $C_l$ of link $l$, and can be calculated as follows:

$$load_l = \frac{\sigma_l}{C_l}$$

(2.2)

For example, suppose the sum of demands’ bandwidth $\sigma_l$ that passes through the networking edge $l$ to be 55 Mbps and the edge’s capacity to be 100 Mbps, the utilization load $l$ of the link is 55%.

### 2.2.1.2 Link Overload

A networking edge is considered fully loaded if its load equals the capacity, i.e., $load_l = 1$. When further requests arrive and use this link $l$, the available capacity $C_l$ will not be sufficient to transmit the requests’ traffic properly. Additionally, if the load exceeds the capacity of the networking edge $load_l \geq 1$, the edge is considered an overload edge. In such cases, not only the request that caused the connection $l$ to become overloaded, but all the requests will be concerned by a throughput reduction.

The overloads in the service chain $P_r$ of request $r$ can be interpreted as having at least one functional edge $f$ in the chain that is mapped by $H_f$ to one or more overloaded networking edges. Specifically, it is about identifying a route in $P_r^N$ where there is at least one networking edge $l$ with relative load $load_l$ greater than 1, formally defined as follows:

$$\exists l \in P_r^N : load_l \geq 1$$

(2.3)

### 2.2.1.3 Overload Reduction

Since a network link cannot physically transmit more data than determined by its capacity, the actual bandwidth summed up on it can not exceed the link capacity in an overload situation. Overload reduction can be interpreted as a traffic loss caused by throwing away the part of the traffic that exceeds the capacity, i.e., reducing the actual bandwidth according to it. Although its interpretation is based on the bandwidths, it can be taken as
a part of the packet loss, assuming that reducing the bandwidth happens through throwing away some packets.

The overload reduction factor \( OvlRed_l \) is calculated on the networking link \( l \) with capacity \( C_l \) to evaluate how much traffic is thrown away. Its calculation implicitly considers the capacity and the requests that go through the link \( l \); it is defined as follows:

\[
OvlRed_l = \frac{load_l}{\sigma_l} - 1
\]  

(2.4)

For example, if we suppose that \( C_l = 10 \text{ Mbps} \), \( \sigma_l = 15 \text{ Mbps} \), and thus, \( load_l = 1.5 \text{ Mbps} \), then \( OvlRed_l = 1/3 \), since one third of the traffic will be thrown away to hold the link capacity as a physical constraint.

Besides network links, the overload reduction factor is also evaluated for traffic demands. For a single request \( r \), \( OvlRed_r \) is the maximum value of \( OvlRed_l \) over the links \( l \) that is used in the path \( P^N_r \). \( OvlRed_r \) is defined as follows:

\[
OvlRed_r = \max_{l \in P^N_r} OvlRed_l
\]  

(2.5)

To prepare the model for more accurate evaluations of various situations, I introduce the experienced bandwidth \( \beta_r \) value, often known as throughput. It is a quantitative measure for request \( r \), giving the bandwidth delivered successfully from its source to its destination. Following the overload reduction calculated for requests, \( \beta_r \) shall be the same on every link \( l \) in \( P^N_r \).

There are two main situations where the experienced bandwidth values will result differently. First, whenever the link \( l \) is not congested, the sum of experienced bandwidths on it may be lower, or at least not higher than the sum of requests’ bandwidth \( \sigma_l \). Second, when \( \sigma_l \) exceeds the link capacity \( C_l \), the link \( l \) gets congested, and traffic reduction happens. In this case, the sum of experienced bandwidths on it may be lower, or at least not higher than the capacity \( C_l \). This concept will be applied in Section 4.4.2.

### 2.2.2 Ensuring QoS with Queuing

Before introducing the packet-level KPIs in detail, let me quickly present how the packets are handled. The two most common QoS tools used to handle traffic in network nodes are classification and queueing. The traffic classification helps identify and mark traffic demands assigned to traffic classes. The traffic classification ensures that network nodes know how to identify data that will be prioritized. Queueing disciplines support the way of accepting, serving, and releasing packets.

The input queue accepts packets to be routed over the network links attached to the node. After determining the correct output link, the router dequeues or takes out a data packet from the input queue, and if there is enough space, the data packet is added to the output queue, known as enqueuing.

The packets are arriving in the output queue attached to the link. They are queued for transmission if the link is currently busy transmitting another packet. Suppose the buffering space is insufficient to hold the arriving packet. In that case, the queue’s packet discarding policy determines whether it will be dropped or other packets will be removed from the queue to make space for it [27]. I apply a queueing network analysis method to
calculate the packet-based KPIs, which support Strict Priority, Weighted Fair Queueing (WFQ), and Low Latency Queueing (LLQ) policies with multiple traffic classes.

### 2.2.2.1 Strict Priority Queueing

With strict priority queueing, the highest priority queue is served until it is empty, preventing lower priority queues from getting starved. This policy only allows the queue to exceed the committed data rate when other queues are not congested. Then, the lower priority queues are sequentially served until they are empty. When the queue of a strict-priority traffic class is full, and the class exceeds its limit or can not dequeue packets because of higher priority traffic transmission, packet drops happen, causing traffic losses.

I introduce an example in Figure 2.9 where different packets get classified and arrive in the queues of the output link.

Figure 2.9a illustrates how the queueing model serves the packets by classifying them into priority classes. The packet’s priority class may depend on an explicit marking in its packet header. Each priority class typically has its queue. When choosing a packet to transmit, the priority queueing discipline will transmit a packet from the highest priority class with a non-empty queue. The choice among packets in the same priority class is typically made FIFO. If there is more than one priority class, the packets are treated in different priorities.

![Classification of packets for strict priority queues](image1)

(a) Classification of packets for strict priority queues

![Operation of the strict priority queues](image2)

(b) Operation of the strict priority queues

**Figure 2.9:** Strict priority queueing model for high and low priority queues

Figure 2.9b illustrates the operation of a priority queueing with two priority classes. The packets 1, 3, 5, and 6 belong to the high priority class, and packets 2 and 4 belong to the low priority class. Packet 1 arrives and, finding the link idle, begins transmission. During the transmission of packet 1, packets 2 and 3 arrive and are queued in the low and high priority queues, respectively. After the transmission of packet 1, packet 3 with a high priority is selected for transmission over packet 2, which has a lower priority packet even
though it arrived earlier. At the end of packet 3, packet 2 begins the transmission. Packet 4 (a low-priority packet) arrives during the transmission of packet 3. Due to priority queueing discipline, the transmission of a packet is not interrupted once it has begun. In this case, packet 4 is enqueued in the lower priority queue and begins its transmission after packets 5 and 6 were completed.

### 2.2.2.2 Weighted Fair Queueing

WFQ is a dynamic process that applies multiple queues for buffering the packets and divides the link’s capacity among them on a weight-system basis. WFQ ensures that all traffic is treated fairly concerning the weight. Each type of traffic is associated with an independent queue. The policy allows assigning different weights to different service classes to specify the bandwidth allocated for a specific traffic class.

Prioritization of packets in WFQ might be interpreted as allowing schedulers to specify, for each flow, which fraction of the capacity will be given to it. This policy prevents a single flow from using all the link bandwidth, effectively denying other flows. The mechanism does not provide a strict priority system. If the system is highly loaded and queues are nearly full, the lower weight can lead to a higher likelihood of packet drops. If every queue is continuously loaded, the relation of data amount served from the different classes will accord to the weights.

The example in Figure 2.10 clarifies the WFQ model. The packets are classified into queues with weights 0.5, 0.3, and 0.2.

![Figure 2.10: Queueing model for WFQ queues](image)

Figure 2.10a shows how WFQ assigns the different arriving packets to queues and serves them based on the weights. Figure 2.10b illustrates the handling of the incoming packets based on the serving mechanism.

The WFQ process takes out some messages from the queue of class WFQ-0.5 by serving the packets 1 and 3. Then it moves on and takes some packets from the class WFQ-0.3 queue. Afterwards, the packets from WFQ-0.5 and WFQ-0.3 are served and then takes
some from queue of class \(WFQ-0.2\) etc. It goes back to queue \(WFQ-0.5\) after finishing a complete pass through the queues.

### 2.2.2.3 Low Latency Queueing

\(LLQ\) is the mixture of strict priority and \(WFQ\) mechanisms. Since some traffic might need high priority independently of link utilization, \(LLQ\) extends the \(WFQ\) system with a strict priority class and modifies the scheduler. The packets in the queue of this class are dequeued and sent before the lower priority packets. This policy allows delay-sensitive data such as Voice and Video applications to be preferred and treated over other traffic.

Similarly to \(WFQ\), \(LLQ\) allows for defining multiple lower priority traffic classes and assigning different bandwidth proportions to these classes. It is a flow-based policy for ensuring QoS, which schedules high-priority traffic classes first and splits the remaining bandwidth fairly between the other classes according to the weights.

![Classification of packets for LLQ queues](image)

![Operation of the LLQ queues](image)

**Figure 2.11:** Queueing model for \(LLQ\)

Figure 2.11 presents the priority queueing model for the mixture of \(WFQ\) and Strict Priority policies. In the example in Figure 2.11a, each class contains two packets. The packets arrive in order from left to right as illustrated in Figure 2.11b. In the first iteration through queues, one packet from the strict class gets served. It is followed by a packet from class \(WFQ-0.5\), then this iteration finishes with a packet from class \(WFQ-0.3\). The scheduling of further packets is performed according to the priority and weights.

### 2.2.3 Packet Level Evaluation

Calculating the average load on network links can determine if there are overloaded links in the route \(P_r^N\). Overload reduction refers to the traffic part, which is undoubtedly thrown away on these links due to congestion, and its calculation is also extended to requests.
Load and overload are essential indicators in the link-level evaluation, but they are not enough to exactly catch what happens on the network links with the not-thrown packets of the requests.

Starting from several statistical properties of different traffic classes and some packet-level properties of the nodes’ queueing system, such as packet length, inter-arrival time, and queue length, I use theoretical calculations proposed in [28, 29]. These methods provide the packet-level evaluation of priority queueing systems with finite-length buffers to calculate packet loss and packet delay. The packet-based evaluation methods are primarily based on the Markovian approach or its extensions like Markov Arrival Processes MAP, or Quasi-birth–death QBD.

According to Equation 2.5, the overload reduction calculation does not consider the priorities of the service classes and the exact queueing policy, but the packet level evaluations do. In my approach, the overload reduction must be applied first to let work properly the packet level calculation. This step is necessary because the applied packet-level evaluation methods are not stable when the queues are overloaded. As a result, the traffic is reduced to avoid network link overload before the packet-level queue analysis is performed. However, despite the reduction, the probability of having a full buffer is not zero, and packet losses can occur.

2.2.3.1 Packet Loss

*Packet loss* describes the rate of packets that do not reach their destination after being transmitted across a network. It is given as the number of thrown packets to the total number of sent packets and should represent the probability of throwing away a given packet. In this work, I only consider network link congestion as the cause of throwing packets. Congestion means having too much traffic to be transmitted across a link at a given moment, and it causes throws when the output buffers get full and further packets arrive. Similarly to the overload reduction, losses reduce the throughput of protocol or application traffic on the network link, which might cause a reduction of their quality, for example, in the case of streaming video or Voice over IP (VoIP).

The method mentioned above can calculate the queue-level traffic loss probability $PktLoss_l^q$, i.e., it gives the value for the individual queueing systems attached to the network links. Besides this, I also evaluate the value that concerns the service request. The equation in 2.6 shows how to calculate traffic-level packet loss $PktLoss_l^r$ based on $PktLoss_l^q$.

$$
PktLoss_l^r = 1 - \prod_{l \in \mathcal{P}^N_r} (1 - PktLoss_l^q) \quad (2.6)$$

As first, the queue-level traffic losses $PktLoss_l^q$ are determined for each link $l$ in $\mathcal{P}^N_r$ of request $r$ considering their direction. Then, a product is applied using the *inverse* values, i.e., the probability of not throwing the packet on link $l$, which results in the *inverse* of the traffic-level loss $PktLoss_l^r$.

2.2.3.2 Packet Delay

*Packet delay*, also referred to as latency, is the average time taken for a packet to be transmitted across a network from source to destination, i.e., the ratio of the total waiting
and serving time of packets to the total number of packets. Several factors introduce network latency. In this dissertation, I only consider packet delays caused by waiting for the turn while the transmitter is busy with other packets and the transmission.

The method mentioned above can calculate the queue-level delay $PktDelay^q_l$, i.e., it gives the value for the individual queuing systems attached to the network links. Besides this, I also evaluate the value that concerns the service request. The traffic-level packet delay $PktDelay^{tr}_r$ for request $r$ is calculated by taking the sum of queue-level packet delays on each link in $\mathcal{P}_r$ as shown in equation 2.7:

$$PktDelay^{tr}_r = \sum_{l \in \mathcal{P}_r} PktDelay^q_l$$

(2.7)

### 2.3 Result generation

#### 2.3.1 Evaluation Process

To compare the different cases and solutions, I generate numerical results for several performance indicators as introduced in Section 2.2. This process takes multiple steps. For each network topology, I consider predefined networking and functional layers, and functional nodes with VNF capabilities. Based on these, each functional link $f$ in the functional layer $\mathcal{L}_F$ is mapped into the networking layer $\mathcal{L}_N$ by the mapping $\mathcal{H}_f$ which consists of a set of network links. If it is not differently stated, e.g., in the case of different topologies, the mapping represents the shortest path in the networking layer.

The SFC solutions determine service chains in the functional layer $\mathcal{L}_F$ by finding a chain $\mathcal{P}_r$ for every traffic demand $r$ in order. The order of the traffic demands is determined before the evaluation process runs. The traffic demands are randomly ordered at the start of the analysis, which is then used in simulations. The results can depend on the order of the demands. However, I expect similar result trends with different randomized orders since the topology is rather generic, and the demands are distributed among the nodes. In contrast, suppose the traffic demands arrive in non-randomized order might mislead the evaluation of the results. For example, when starting with the traffic demands of classes requiring higher or lower bandwidth, the overloads on networking links could appear earlier or later in a rather deterministic manner. Few study cases have examined how the order of traffic demands could affect the KPIs; therefore, five different orders of demand are taken into account, each generated randomly.

The chain $\mathcal{P}_r$ presents the path for request $r$, which consists of a set of functional links $f$ in the functional layer $\mathcal{L}_F$. I first perform the link-level evaluation of the networking links in $\mathcal{P}_r$ to calculate the load $load_l$ and find which links are overloaded. After that, I calculate the overload reduction, i.e., the amount of traffic lost due to network link congestion, if there are any overloaded links. In some cases where multiple demand orders are evaluated, the minimum, maximum, and average values are examined. However, a detailed analysis of the multiplication of orders is not applied in every scenario since my experience is to get similar trends in the results.

In the last steps, the packet level results are calculated considering the load and overload reduction on the links and the packet level parameters for queues and traffic, such as packet length and inter-arrival time. I evaluate the packet loss and delay metrics for the network links first and then for the traffic demands. The process flow followed in this dissertation is illustrated in Figure 2.12.
Figure 2.12: Flow of evaluation process

The simulation and analysis operations are carried out in ARIADNE/NetAnalyst which is a middle-term design tool for telecommunication networks developed at the Networked Systems and Services Department of the Budapest University of Technology and Economics and the Flexiton company\[30, 31\]. NetAnalyst follows an iterative planning approach that encourages decision-making at every stage. The SFC is performed in this simulation tool considering the order of the requests’ arrival but not allowing any departures. The tool allows evaluations during the stages to support planning cycles. Based on the results, potential loopbacks can be applied, which are necessary for most networks in real-world situations.

I use the framework presented in \[32\] to handle network node transport and switching capabilities. It provides a collection of network nodes, connections, and services that can be flexibly configured.

After that, SFC solutions find the shortest path by considering the VNFs that must be passed. For each traffic demand, the solutions create a service chain that connects the source to the destination by transferring the traffic demand through networking links in the networking layer.

### 2.3.2 Network topologies

I intend to use both small and large topologies to generate results and compare the proposed solutions. The small toy topologies with few particular demands are applied to analyze the essential behaviors and differences. In contrast, I applied more complex topologies with different characteristics for a more general analysis.

The topology BT Europe with twenty-four (24) nodes and thirty-seven (37) links was chosen from the Internet Topology Zoo \[33\]. This network is rather sparse, i.e., it has a quite low average node degree of 3, and network links are identical, with 1 Gbps capacity. Figure 2.13 shows the networking layer topology.

I selected the six more central nodes as VNF-capable nodes and built the functional layer by connecting them in a full graph. The VNF capabilities are as follows:

- \( v_1 \) is placed in Budapest and London2,
- \( v_2 \) is placed in Paris and Madrid,
- \( v_3 \) is placed in Amsterdam and Frankfurt.

The traffic layer includes twenty-four (24) traffic endpoints distributed across the twenty-four (24) European capitals. There are five simplex traffic demand types to be transferred,
Interactive Video, Best Effort, Video, Voice and New Services which can be typical traffic demands to be supported in 5G networks for example. Each traffic demand must pass through VNFs (v3, v2, v1) in order. The New Services type represents the traffic of a new application using a central server. Its demands start from the endpoint in Amsterdam, and the destination is distributed among the other endpoints. The traffic demand sources and destinations are distributed uniformly among the endpoints used for all the other types.

The number of requests, the average of the requested bandwidth, and the main QoS properties, such as packet lengths and priority class for BT Europe topology, are summarized for each traffic type in Table 2.1.

<table>
<thead>
<tr>
<th>Traffic</th>
<th>Requests</th>
<th>Bandwidth Mbp</th>
<th>Packet length Kb</th>
<th>Priority class</th>
</tr>
</thead>
<tbody>
<tr>
<td>New Services</td>
<td>23</td>
<td>0 – 700</td>
<td>8</td>
<td>WFQ-0.3</td>
</tr>
<tr>
<td>Best Effort</td>
<td>275</td>
<td>3.98</td>
<td>500</td>
<td>WFQ-0.2</td>
</tr>
<tr>
<td>Interactive Video</td>
<td>275</td>
<td>3.03</td>
<td>500</td>
<td>Strict</td>
</tr>
<tr>
<td>Video</td>
<td>275</td>
<td>2.11</td>
<td>500</td>
<td>WFQ-0.5</td>
</tr>
<tr>
<td>Voice</td>
<td>275</td>
<td>1.16</td>
<td>80</td>
<td>Strict</td>
</tr>
</tbody>
</table>

Table 2.1: Traffic types characteristics of BT Europe topology

In order to analyze a denser network, I also created a hypothetical Algerian backbone network shown in Figure 2.14. The networking layer here has ten (10) core nodes and seventeen (17) edge nodes depicted as black-filled and grey-filled nodes, respectively. All the nodes are placed at different geographical locations to have a perspicuous view. The topology has sixty-eight (68) network links, and the node degree of this topology is 5. Edge links and core links are the two sorts of networking links used in the topology. The core nodes are connected via core links, forming a core part that ensures two-connectivity to provide higher reliability. At the same time, the edge and core nodes are connected by edge links, having at least two connections for each edge node. Unless otherwise noted, the edge and core links have a capacity of 1 Gbps and 10 Gbps, respectively.

The functional layer is constructed by connecting the various VNF-capable nodes in a full graph. There are three (3) different VNFs, and five core nodes in different cities are selected to provide the capabilities as follows:

- v1 and v2 are placed in the capital Algiers,
- $v_1$ is placed in Boussa\andalone{d}a city,
- $v_3$ is placed in Constantine city,
- $v_2$ is placed in T\eun{e}s city,
- $v_3$ is placed in Oran city (left).

Furthermore, the traffic layer contains twenty-seven (27) endpoints distributed across the twenty-seven (27) cities. There are five simplex traffic demand types, as also mentioned in the description of the BT Europe topology. In order, the demands must pass through the VNFs ($v_3$, $v_2$, $v_1$). The New Services demands start from the endpoint in Algiers. The number of requests, the average of the requested bandwidth, and the main QoS properties, such as packet lengths and priority class, are summarized for each traffic type in Table 2.2.

<table>
<thead>
<tr>
<th>Traffic</th>
<th>Requests</th>
<th>Bandwidth Mbps</th>
<th>Packet length Kb</th>
<th>Priority class</th>
</tr>
</thead>
<tbody>
<tr>
<td>New Services</td>
<td>26</td>
<td>0 – 1500</td>
<td>8</td>
<td>WFQ-0.3</td>
</tr>
<tr>
<td>Best Effort</td>
<td>702</td>
<td>3.98</td>
<td>500</td>
<td>WFQ-0.2</td>
</tr>
<tr>
<td>Interactive Video</td>
<td>702</td>
<td>3.03</td>
<td>500</td>
<td>Strict</td>
</tr>
<tr>
<td>Video</td>
<td>702</td>
<td>2.11</td>
<td>500</td>
<td>WFQ-0.5</td>
</tr>
<tr>
<td>Voice</td>
<td>702</td>
<td>1.16</td>
<td>80</td>
<td>Strict</td>
</tr>
</tbody>
</table>

Table 2.2: Traffic types characteristics of Algeria topology

In each thesis, I aim to study scenarios where the bandwidth of the demands of New Services grows linearly. Further input parameters for the methods that calculate packet-level QoS performance can be derived from these characteristics using the network link capacity and load information. Higher moments of the packet lengths and inter-arrival times are set to the default value, and the total queue length is set to 100 packets for each output queueing system. If not otherwise stated, I consider the WFQ weights as written in the name of the traffic class assigned to the service types.
Overloads in Service Chaining

The continuous evolution of the networks implies that new technologies shall be applied for the more and more effective provision of a broad set of services. These services require higher quality, with complex use of the available networking, storage, processing functions, and resources. The changes in requirements conclude in frequent reorganization and reconfiguration of the network functions and resources.

3.1 Related works

The concept of SFC is not dependent on SDN and NFV. SFC can be implemented between physical network functions through segment routing. This paradigm encapsulates packets in headers in a recursive manner for the packet to be sent through a predetermined set of intermediate routers until its destination. By combining segment routing and NFV, it is possible to make the management of SFCs much more flexible. In [34], the packets are sent, then VNFs will be chosen, and the packets are encapsulated in the corresponding headers to allow the service to be deployed. In [35], the authors propose an architecture for deployment on NFV networks using existing routing protocols (OSPF) and allowing the placement of SFCs without using SDN. Unlike [34], the ingress node is not responsible for setting the list of nodes to be visited to constitute the SFC. Instead, the decision is completely distributed within autonomous NFV routers. They propose a Linear Program (LP) to minimize costs (traffic and VNF) and compare their solution to a centralized SFC placement. The cost of their solution never exceeds two times the cost of the centralized solution.

It is sometimes more convenient to place several functions on the same node to reduce the energy consumption of a node, for example. Nevertheless, allocating the functions on different nodes can be more interesting because other chains can use these functions. It is necessary to avoid overloading the nodes and the links attached to them. Considering the traffic routing simultaneously and the placement of the functions can also be interesting. It avoids having too long routes and does not saturate the network unnecessarily. In this example, the functions are placed, so the flow goes through its shortest path.

Emerging technologies such as NFV are vital for 5G development since it improves the capabilities and services by adopting them in network architecture and services. Therefore the requirements of 5G networks will be twice as rigid for traffic, and the expected delay of the 5G Radio Access Network (RAN) network should be less than 1 ms. In particular,
several 5G studies, such as [3, 36, 37], focus exclusively on the RAN section, ignoring the fact that the backhaul segment causes a part of the end-to-end delay.

Numerous recent research initiatives have been done to meet the 5G Networks, including developing network design and services technologies. The multiple access techniques in the network are almost on the brink and require sudden improvements. As an instance, [38] addresses the requirements of 5G, such as increased capacity, improved data rate, decreased latency, and better QoS.

Several studies, such as those cited in [39, 40] address NFV as a promising architecture that can be used to increase the scalability and functionality of networks by leveraging virtualization technologies. NFV technology, on the other hand, facilitates the execution of software-based network functions on general-purpose hardware through virtualization.

The European Telecommunications Standards Institute (ETSI) provides an overview of the benefits, enablers, and constraints of NFV [12]. ETSI develops a baseline NFV architectural framework for controlling and coordinating network services while lowering equipment costs. In order to leverage these benefits of virtualized networks, challenges need to be addressed, such as automatic scaling of the VNFs when needed and ensuring the appropriate level of resilience to hardware and software failures. For example, NFV is applicable across many network functions, including stationary and mobile networks. A significant example of NFV applications is the Evolved Packet Core (EPC), which is a framework of 4th generation (4G) mobile networks for voice, data processing, and switching [41]. These functions can be virtualized into VNFs implemented in an NFV-based architecture as used by many ISPs.

Different approaches have been proposed by industries, research institutes, and mobile operators to standardize SFC. In [13], ETSI defines a network service as a chain of VNFs and emphasizes the demand for a new set of orchestration and management functions. In the RFC 7665 [14], IETF defines the service function chain as an ordered set of abstract service functions and ordering constraints. IETF also describes an SDN-based SFC architecture. An SFC classifier in the data plane performs a classification of end-to-end traffic to determine which VNF should be chained to process the traffic based on its requirements. In general, many different chains are possible, and the best one should be chosen to provide a high QoS (delay, loss, or throughput) level if the service requires it.

Various scientific works have been addressed in the context of the placement and chaining of VNFs. The survey in [42] is a remarkable example of a comprehensive summary of the extensive research on placement problems in edge computing systems. The authors studied many research papers based on several dimensions, such as the problem formulation, the applied mathematical methods, the objectives and constraints incorporated in the optimization problems, and the complexity of the proposed methods. Most research papers in [42] formalize the problem as linear programming where the objective function typically aims at optimizing latency, energy, or resource utilization in the function of optimization constraints, such as bandwidth capacity, latency, power, or memory.

Additionally, in the context of the placement and chaining of VNFs, the authors in [43] emphasize the importance of routing in the NFV network and propose a Routing Evaluation model. They aim to explore potentially better paths by selecting those with minimum costs, and they formulate the SFC problem showing that it is NP-hard. The same approach is applied in [44]. In [45], the authors focus on the VNF placement problem in SDN/NFV enabled networks. They formulate the problem as a Binary Integer Programming (BIP), aiming to minimize a weighted cost composed of the VNF placement and
running costs. Besides this, the authors propose a VNF Placement Algorithm using reinforcement learning to optimize network performance.

The authors in [46] propose an algorithm for service graph embedding which can jointly control and optimize cloud and networking resources focusing on modifiable objectives. The algorithm finds a greedy mapping of the service chains onto the infrastructure to maximize the number of service graphs and balance the network load. The algorithm selects a set of shortest paths that minimizes a composite metric of bandwidth, delay, and resource utilization.

The authors in [47] provide ILP and heuristic solutions for selecting data center nodes for the SFC. They consider VNF deployment and routing costs and capacity constraints of networking links. The model proposed in [48] also considers the reliability requirements in the demands description. The authors present a mathematical formulation for the SFC and routing problem assuming bandwidth, capacity, and reliability constraints. They introduce and evaluate greedy algorithms to search for the shortest paths to the next host where the next VNF is feasible. Although the search is performed bi-directionally from the source and destination and vice-versa, it is not assured that the shortest path will be found this way.

Nearly all methods presented in the literature consider the network topology with link bandwidths, e.g., in [47, 48, 49, 50]. Simple routers are assumed to have only forwarding functions, while some routers are extended to have VNF capabilities. This approach leads to the option of applying any IP link for any chain section in any service chain, i.e., the connection between any two VNF-capable nodes is allowed. The solutions in [47, 48] focus only on selecting the nodes for deploying a VNF for a specific traffic demand. They do not consider constraints on the topology of VNF-capable nodes and neglect most networking link characteristics, including the capacity and currently available bandwidth. These solutions do not support scenarios where the required VNFs’ order is prescribed for different traffic demands.

The other extreme is supporting the functional topology with some VNF-capabilities at each node but not assuming the bandwidth constraints on the forwarding network’s links. This approach is used for instance in [51].

In recent years, research reviews have been published that examine solutions for deploying and managing network services, mainly focusing on concurrently routing demands and supplying them with the required VNFs. The authors of [52] present and compare many efficient algorithms and optimization models that aim to minimize initial cost and consider some chaining limitations.

Papers [10, 53, 54] focus on QoS issues. The authors propose methods that assume latency limitations but only consider the delays related to applying the virtual network functions. This technique is insufficient for meeting QoS requirements because, besides the service quality at the VNF-capable nodes, the network also influences the total QoS. Using multiple networking links in the chains can result in a significant demand on the network resources and produce heavy or even overloaded links. Thus, the number of network links in the service chain selected for a given service demand and their load impact QoS.
3.2 QoS Modeling and Analysis in 5G Backhaul Networks

EPC has played a significant role in developing 4G LTE, allowing operators to deliver and operate packets across the network while providing converged voice and data. The 5G Core Network, as defined by the 3GPP, is becoming a rising star in core networks as the telecommunications industry continues to innovate at an accelerated pace.

This section investigates the migration from 4G to 5G architectures by presenting various network models that use multiple layers and several types of connections for modeling the virtual EPC (vEPC) and 5G network architectures, respectively. I investigate potential SFC solutions by presenting two algorithms to establish service chains of network services, which are then compared assuming the migration to 5G. I aim to examine how much traffic is being put on the lines and whether it could impact the QoS qualities.

3.2.1 Modeling 4G EPC

The vEPC is designed to separate control and user plane requirements into specific functional elements known as critical elements of EPC, such as the Mobility Management Entity (MME), the Serving Gateway (SGW), and the Packet Network Gateway (PNG or PGW). The architecture of EPC is illustrated in Figure 3.1.

In order to provide services, vEPC communicates with Packet Data Networks (PDNs) outside of the EPC, such as the Internet or private corporate networks. In this architecture, the only component of the access network is the Evolved Node B (eNodeB), which handles radio communications between the mobile and the EPC. The user Equipment (UE) and the core network communicate directly through the eNodeB, which is considered the endpoint of each type of traffic and is responsible for direct connection with the UEs.

The EPC includes a Home Subscriber Server (HSS) component, which serves as a central database for all of the network operator’s subscribers, storing their personal information. The MME serves as an essential control node for the LTE access network. It is responsible for handling the UE access network, managing mobility, and establishing the bearer path for UE. At the same time, SGW is a crucial entity in the EPC since it serves a specific set of eNodeBs for User Plane data and is responsible for routing and sending UE packets to the appropriate destination. SGW receives instructions from the MME on setting up and tearing down sessions for a particular UE, and it receives and transmits data between the eNodeB and the PGW [55]. PGW is the point of interconnection between the EPC and the external networks and is responsible for routing packets to and from the EPC.

As an illustration of how a mobile session is established, Figure 3.1 shows an example of traffic transmission from the UE to the PDN and then back to the UE. When the user equipment (UE) is switched to a mobile network, it indicates a handshake. MME receives the request message from the UE, which is sent through the eNodeB, and then the UE
initiates a mobile session. MME forwards the packets to SGW, and then SGW forwards the packet to PGW, which sends a reply packet to SGW towards the UE, indicating that the mobile session has been created. MME forwards the packets to SGW, and then SGW forwards the packet to PGW. Once the mobile session has been formed, a tunnel between the eNodeB and the external network is constructed by EPC; it passes the packets through SGW and PGW used in the process.

I use the multilayer model described in Section 2.1 for the EPC architecture and its elements. The 4G model comprises several types of connections, nodes with and without VNF capability, and traffic endpoints. The assumption is that SGW and PGW functionalities have been virtualized into VNFs. The signal communication between two eNodeBs is considered the traffic layer, and Figure 3.2 shows such a connection by the blue line. The communication between a VNF and another VNF available at another VNF-capable node or an eNodeB is represented as a functional link in $L_F$ illustrated by orange lines. The networking layer $L_N$ contains all the nodes, including the ones without VNF capabilities. The nodes in $L_N$ are linked together by network connections illustrated by green lines.

In order to clarify how the service demand’s traffic travels from one endpoint to another, consider the following example. An application runs on a mobile device attached to $eNodeB_1$, and sends packets needing to pass through the SGW and PGW before being transferred to $eNodeB_2$. In the model, the traffic is transmitted through the chain of functional connections indicated by the bold orange lines in the Figure. The dashed red lines show the networking path that data packets traverse. This path depends strongly on the location of the VNF-capable nodes, contains a controlled loop, and differs significantly from the path in the functional layer.

### 3.2.2 Modeling 5G

The 5G system consists of three main key components: the 5G Radio Access Network (RAN), the User Equipment (UE), and the 5G Core Network (5GC). The 5GC’s Service-Based Architectures (SBA) enable the provision of services such as authentication, security, session management, and traffic aggregation from linked devices, all of which involve complex connectivity of network operations.

As a result of utilizing HTTP for communication, the SBA provides a modular framework from which typical applications may be launched using components from various sources and providers. As defined by the 3GPP, SBA is the control plane functionality of a 5G network, and standard data repositories are delivered by a network of interconnected NFs, each of which has the authorization to access the services of its neighboring networks [56]. The basic 5G network shown in Figure 3.3 is based on SBA architecture.

In addition, the 5G design is built on the vEPC architecture, but the Control and User Planes have been divided into separate roles to allow for more flexible deployment. The Session Management Function (SMF) contains the Control Plane functionality from SGW and a part of the MME functionality, particularly the session management part. The
Access and Mobility Management Function (AMF) is responsible for implementing the other MME functions in EPC architecture, such as essential derivation and security context management.

The User Plane function is composed of a single entity, referred to as the User Plane Function (UPF), which is regarded as a hybrid of the PGW and SGW components of the EPC architectural design. UPF can be deployed as one or multiple UPFs in the User Plane, depending on the configuration. It is in charge of packet routing and forwarding and ensuring QoS. As previously stated, it is also regarded as the point of interconnection between the mobile infrastructure and the PDN.

I apply the multilayer model introduced in Section 2.1 to represent the 5G architecture and its elements as illustrated in Figure 3.4. The Control and User Planes functionalities, such as UPF, AMF, and SMF, are represented as VNFs. The VNF-capable nodes forming the functional layer are distributed in different physical locations. The 5G model also includes various types of connections, traffic endpoint eNodeBs, and nodes incapable of running VNFs. However, the latter ones are not present in the example. The traffic layer is shown by a blue line representing the signal communication between two eNodeBs.

The network functions as UPF, AMF, and SMF are virtualized into the blue, yellow, and red VNFs, respectively. Functional connections originate from VNF-VNF pairs or eNodeB-VNF pairs and are presented as orange lines. The networking layer can contain multiple connections among VNF-capable or not-capable nodes and eNodeB presented by the green lines in the Figure.

An example as shown in Figure 3.4 on how packets transit through the 5G network, assuming a service demand from eNodeB1 to eNodeB2 requiring to pass through AMF, SMF, and UPF. Traffic is directed through the functional layer links, which are presented as orange lines. The traffic is transmitted through the networking links by passing, in this case, only VNF-capable nodes. Dashed red lines in the Figure represent the network path.
The SDN paradigm separates the Control plane from the Data plane in the 5G network architecture. The resulting flexibility and the application of virtualization allow the deployment of individual services with various performance requirements that share the same infrastructure. With the virtualization of functions, software and hardware implementations are effectively separated, allowing each function to be scaled separately and distributed as efficiently as possible while considering the available bandwidth capacity and latency requirements. As illustrated in Figure 3.5 too, the significant difference is based on this essence when comparing 4G and 5G network architectures.

![Figure 3.5: 4G/5G Comparison in deployment](image)

### 3.2.3 Forming Service Chains using the multilayer model

This initial part of my research aims to measure and demonstrate the benefits of migrating from using 4G EPC to using 5G Core Networks for the backhaul architecture in mobile networks. It is supposed that moderate network link loads and short paths are needed for achieving acceptable QoS.

This section introduces two essential SFC solutions for calculating service chains. The task is to determine a path in the functional layer for a service demand from an eNodeB to another one fulfilling the requirement of passing through the set of prescribed VNFs. In order to find such a path, only the VNF capabilities are considered. Dealing with the current placement and number of already running VNFs is out of the scope of my research. Both solutions are based on the Dijkstra algorithm developed for finding a path but running on an extended graph to support the VNF requirements. The following are the most significant distinctions between the two solutions:

- **HopBased**: HB is a model of the reference supplied in [57]. It identifies the shortest path for a service request in the functional layer $L_F$, resulting in the determination of the path $P_r$. The cost of every functional edge $f$ is considered here 1. This algorithm provides the fewest amount of functional hops in the chain $P_r$ without considering the networking layer $L_N$.

- **CostBased**: CB algorithm finds a chain $P_r$ that is calculated by determining service paths over the functional connections in $L_F$. It applies a cost for the functional connections calculated based on the length of the path $H_f$. CB allows transferring
the traffic along the path $\mathcal{P}_r^N$ in the networking layer in $\mathcal{L}_N$ that is assumed to be the shortest.

In both 4G and 5G models, HB solution is expected to choose the shortest path in the functional layer that meets all the required VNFs. On the other hand, CB is expected to choose the shortest path in the networking layer.

### 3.2.4 Performance Evaluation

The case study includes two possible topologies for both 4G and 5G networks and is based on a hypothetical Algerian backbone discussed in Section 2.3.2. The 4G network supports two VNFs: PGW and SGW, available in the two core routers. In the 5G network, the VNF capabilities are deployed in more significant topology locations.

The study scenarios assume that the bandwidth of the New Services type traffic connections increases linearly from 100 Mbps to 500 Mbps in four steps. The characteristics of each type of traffic demand are summarized in Table 3.1, including the number of connections, the average requested bandwidth and the QoS properties.

<table>
<thead>
<tr>
<th>Type</th>
<th>Bandwidth (Mbps)</th>
<th>Number</th>
<th>Priority</th>
<th>Packet Length (Kb)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Video</td>
<td>6.03</td>
<td>351</td>
<td>WFQ-0.5</td>
<td>500</td>
</tr>
<tr>
<td>Voice</td>
<td>3</td>
<td>351</td>
<td>Strict</td>
<td>80</td>
</tr>
<tr>
<td>Best Effort</td>
<td>11.49</td>
<td>351</td>
<td>WFQ-0.2</td>
<td>100</td>
</tr>
<tr>
<td>Int. Video</td>
<td>9.63</td>
<td>351</td>
<td>Strict</td>
<td>500</td>
</tr>
<tr>
<td>New Services</td>
<td>100 – 500</td>
<td>27</td>
<td>WFQ-0.3</td>
<td>200</td>
</tr>
</tbody>
</table>

**Table 3.1:** Traffic types characteristics

### 3.2.4.1 Link Level Evaluation

First, I evaluate the effect of increasing the number of VNF-capable nodes in the 5G network using the CB algorithm, as the VNF capability support is more flexible in SBA than in the EPC model. The number of VNF-capable nodes increases linearly, and different core nodes are supposed for their locations.

Figures 3.6b and 3.6a present the average relative load on edge and core links. As expected, for each case of traffic demand bandwidths, the load decreases when the number of VNFs in the network emerges. The first observation is that the more VNFs are deployed, the more functional links are created to route the traffic demand. Also, the monotonous and nearly linear decrease of the mean link load in Figure 3.6a is caused by the working mechanism of CB, which prefers the shorter networking path for the traffic demand. If there are more VNFs distributed in the network, the path through the VNF can be shortened for some traffic demands.

Figure 3.6b present the average loads on the edge links. The traffic load is moderate compared to the load in the core links, although many VNFs are deployed. This behavior comes from the distribution of the VNFs concentrated in the core nodes. As the functional layer topology consists of VNF-capable nodes, the load with CB solution is concentrated on the core part of the network. The flexibility of 5G networks allows a set of VNFs to be selected and reduces the load in the core part of the network, but there are few choices regarding the path sections beyond the core network.
The next evaluation scenario compares 4G and 5G networks having four different sites with VNF capabilities. Figure 3.7 shows the average relative load of the networking links, which increases nearly linearly with the elevation of the bandwidth of New Services. For both 4G and 5G networks, HB shows lower loads than CB because the chains use shorter networking paths to transfer the traffic demands and concentrate the traffic on fewer links in the core segment, which dominates the average.

Figure 3.8a and 3.8a illustrate another perspective on how CB and HB perform in an overloaded situation. HB applies only a small number of functional hops when building the chains and overloads some core links but only in the 4G network. On the other hand, Figure 3.8b shows that the edge links are loaded quickly, and the overloaded connections appear sooner, particularly in the case of HB in the 4G network. In the high load range, the CB algorithm shows a lower number of overloaded connections in both networks compared to the HB algorithm. Considering the cost of functional links in the chains allows the CB algorithm to use shorter networking paths compared to HB, resulting in lower traffic loads in the edge segment of the network.

Figure 3.9 shows the average values of overload reduction for the New Services traffic. The reduction increases since there are overloaded connections in the edge segment for the whole load range, and in the high-load range, these traffic flows suffer high overload reduction. This harmful effect is observable, especially with the algorithm HB in the 4G network, which offers no flexibility in choosing the required VNF-capable sites. In such
cases, this traffic class meets many congested networking links causing packet losses and high delays.

### 3.2.4.2 Packet Level Evaluation

Beyond the load-level analysis, I evaluate the packet-level results to analyze the link QoS properties related to the New Services traffic class. \(^1\)

Figure 3.10a compares 4G to 5G and HB to CB algorithms presenting the packet loss rates on networking links after the overload reduction. There is less packet loss observed in the 5G than in 4G for both HB and CB solutions, even when the network load gets higher. 5G networks use more VNF-capable nodes and a larger set of functional links to be selected in SFC, allowing them to apply more networking links with less traffic load. In comparison, in 4G networks, the selected chains are mapped to a smaller set of networking links, which get congested, causing packet drops.

Figure 3.10b presents the average queue-level delay for different studied scenarios. The delay for 5G is less than 4G in the low load range. However, 4G and 5G for HB show almost the same average delay in the higher range because this solution uses the lowest

---

\(^1\)The original Figures in [C1] differ from those inserted in the dissertation since an implementation bug in QoS analysis was discovered, and the Figures were revised.
cost in the functional layer and, therefore, causes high utilization of links in the networking layer. In this case, packets must be buffered for a long time in the queue until the link has the available capacity to transmit everything, while in the 5G case, SFC can use more functional links mapped to different network links.

![Figure 3.10: Average queue-level loss](image1)

![Figure 3.10: Average queue-level delay](image2)

Figure 3.10: Average queue level for delay and loss

Figure 3.11 presents 5G network average end-to-end delays calculated for non-strict priority demands Video, Best Effort, and New Services traffic types for CB and HB algorithms. When the New Services bandwidth emerges, each type of traffic suffers a rising delay.

![Figure 3.11: Average end-to-end delays](image3)

Figure 3.11: Average end-to-end delays

The strict priority demands will be severed first, and the remaining capacity will be divided among the other types according to the weights. Video has a higher WFQ weight than the other two traffic types, but larger packet sizes cause higher delay values. Although Video traffic could take more than half of the link capacity, it can not be reached when New Services’ bandwidth is elevated because the network links in the chains are congested.

### 3.2.5 Conclusion

This section discusses how the multilayer model can be applied for the backhaul portion of 4G and 5G networks. Based on this, I have investigated the migration from 4G to 5G architecture and provided SFC methods named HopBased and CostBased to establish service chains in various networks. Assuming different traffic types, I have analyzed
the network load and QoS metrics by calculating the networking layer’s delays, overload reduction, and packet losses.

I have compared the results to study whether the effects in the different architectures could fulfill the QoS requirements of 5G networks. The results show the QoS-related advantages of extending the core functionality toward 5G. The method CB performs better than HB because it considers the mapping of the functional links and uses the shorter ones in $P_r$. It involves fewer networking links from $L_N$ into the $P_r^N$ paths.

It is encouraging that the 5G architecture’s flexibility in VNF capability brings scalability to future network implementations. The results show shorter pathways in 5G than in 4G due to the more significant number of VNF-capable functional nodes scattered throughout the network.

The presented SFC solutions demonstrate varying performance levels when network links are overloaded, resulting in quality of service issues. Therefore, it is necessary to develop Service Function Chaining methods that avoid overloading links to ensure QoS.
3.3 Avoiding Link Overloads in Service Chains

SFC solutions that have been presented can support both the functional and networking layers; however, it is not sure whether all of the information is available to an operator that provides VNF service chaining. On the one hand, the approach used for service chaining can mean different requirements for the structure of the functional layer. On the other hand, the conceptual network layer is instead determined by the physical network topology. Since these pieces of information originate from various sources, i.e., maybe from different operators, the level of interaction cannot always be at its maximum.

Dynamic SFC solutions assign the service chains to network and VNF resources that have enough capacity to align the demand. However, as shown in [58], the traditional Traffic Engineering concept, which tries to use the lightly loaded part of the network, can also be extended for the SFC problem.

In this section, I define various architectures to classify SFC solutions based on how much information can be applied from the functional and networking layers when making decisions. Next, in this section, I examine the overloads on functional and network links, considering the service chains and the mapping of the functional links to the networking layer. I propose Service Traffic Engineering (STE) strategies to avoid overloading the networking layer, which will help to reduce network congestion. I propose solutions that dynamically calculate service chains to avoid excessively congested links wherever possible by finding a service chain that meets the required bandwidth and VNF order requirements.

3.3.1 Classification of Service Function Chaining Solutions

The SFC paradigm allows the definition of the required VNFs, and the corresponding order that must be applied in steering traffic flows to realize complex end-to-end services. The SFC brings new challenges to the network, one of which is in the context of resource allocation. It is necessary to use optimization models for SFC distribution and allocation to avoid networking link overloads, ensure optimal network performance, satisfy user needs, and reduce network costs.

I define three architecture models that differ in the level of information sharing between the management systems of the functional layer $L_F$ and the networking layer $L_N$, similar to the classification of traditional multilayer path selection for IP over optical networks in [59]:

- **Overlay**: works with separated layers, i.e., no information of $L_N$ can be applied in the decisions for $L_F$, and vice-versa.

- **Integrated**: allows full information exchange between layers, i.e., an integrated model can be applied for taking decisions in both $L_F$ and $L_N$ layers.

- **Augmented**: only a restricted set or aggregated information values can be exchanged between the layers. An integrated model is not possible to be built, but some information of $L_N$ (e.g., path lengths or routing dependences) can be applied in the decisions of $L_F$.

The architecture of a solution also depends on how the decision model is built from the two layers. For instance, the algorithm CB presented in Section 3.2 is based on the summed cost of the networking path used for the functional connection is also considered as augmented.
It is possible to determine the right path using several SFC algorithms, but many of these solutions consider the cost that must be assigned to the functional links. The authors aim to identify the shortest path that meets a particular SFC requirement by using static networking link delays represented as a cost in their algorithm.

The solution known as SFC-constrained shortest path is presented in [51] and is referred to as CSP. The algorithm can be used as a basis for comparison and for defining other algorithms. To run this SFC for a particular request, the original functional graph $G_F$ has to be transformed into a new graph $\bar{G}_F$ according to the VNF requirements. It is possible to transform back the shortest path that is found in the new graph $\bar{G}_F$ as a service chain over the original graph.

The CSP solution does not consider separated layers. However, the functional layer is identical to the networking layer, or, in other words, a one-to-one mapping is considered between them. CSP can be categorized as an augmented solution since the transformed graph $\bar{G}_F$ used for the decision in the functional layer $L_F$ is constructed by considering functional link costs that come from link costs in the network topology. In contrast, the heuristics in [49] and in [47] are considered to be with overlay architecture since they determine the paths in the two layers in two separated phases.

Based on what is known at this time, the SFC solutions considered in this section are either overlay solutions or augmented solutions. However, it is possible to define simultaneously the service chain over functional connections and the mapping $H_f$ over networking links using an integrated solution. For example, the method in [48] sets up service chains and $H_f$ together with the creation of the functional layer $L_f$.

### 3.3.2 Overload in Service Chains

I interpret SFC as the search for a chain that leads a service demand through network nodes with VNF capabilities. The chain shall go from the traffic’s source to its destination, passing through the VNFs according to the order predefined for the service. It shall have some functional links connecting the VNF capable nodes, i.e., SFC is searching for a functional layer path that matches the demand requirements. Many algorithms apply static input values, i.e., costs and other parameters are set up in advance and independently from the current state of the network.

In order to understand the problem of using only static link costs when there is an overload situation, I illustrate the behavior of CSP in Figure 3.12. I consider different VNF capabilities at different locations. Three VNF types are available, $(u_1, u_3)$ at node $a$, $(u_2)$ at node $d$, $(u_1)$ at node $c$ and the node $b$ remains without any VNF-capability. I assume a traffic flow demand $\delta$ that starts from the ingress node $a$, goes through some nodes to touch the required VNFs $(u_1, u_2$ and $u_3$), and finally exits at the egress node $c$. I suppose that the capacity of each duplex network link is 1 $Mbps$, and the simplex traffic flow requires 0.6 $Mbps$. As mentioned above, the functional layer is on-to-one mapped to the networking layer, and the static link costs are set as indicated in the figure.

The shortest chain that completes the requirements of the traffic request shall take the path $(a, b, c, d, c, b, a, b, c)$ and the cost equals to 14. The first observation is that the thick black links $(a \rightarrow b)$ and $(b \rightarrow c)$ are being passed in one direction twice by the traffic, forming a controlled loop in the networking path. This demand entirely consumes the capacity leading to link congestion along the path. In more general cases, when there are more demands, the static costs can be critical due to congestion on many SFC’s selected
When a network link is congested, i.e., it carries more data than it can comfortably handle, it degrades the network performance, and the QoS gets reduced. For example, the link will no longer be responsive to carry requests from source to destination as it fails to process them, and the service will no longer be available.

One of the problems studied in my research is avoiding overloaded resources in service chains. The objective of the SFC is slightly modified: it shall find a service chain that is good enough for the current state of the network and align better with the dynamic arrival of traffic demands. Therefore, I propose SFC solutions that avoid congestion on the networking links while still providing Service Chains considering the required VNF order.

Formally described in the above-defined multi-layer model, the SFC solutions find a service chain $P_r$ series of functional edges $f$ for the demand $r$ that requires bandwidth $b_r$ and a given ordered set $U_r$ of VNFs to pass through. The request $r$ is routed over networking links $P^N_r$ in the networking layer $L^N$ considering the mapping of functional edges $H_f$. The following conditions shall hold for the chain $P_r$:

- It has to form a continuous path in $G_F$ by creating a route in the functional layer $L_F$.
- For at least one ordered subset of the nodes in $P_r$, the supported VNF types have to fit $U_r$ to fulfill the service chaining requirement.
- As mentioned in Section 2.2.1.1, for each networking link $l \in P^N_r$, the load of networking link $l$ should not go over the networking link capacity $C_l$.

In the classic routing problem, Traffic Engineering (TE) means that traffic paths are chosen according to the requested bandwidth and the network state to optimize network performance and avoid overloaded resources. This latter objective matches the third condition; therefore, the above concept is named Service Traffic Engineering. Although TE solutions might also have further objectives, for example, limiting network link loads to a moderate level much below link capacities, I concentrate only on avoiding the network link overload in STE. Note that the first two conditions are essential to solve the original SFC problem.
3.3.4 Solutions to avoid Overloads

In order to avoid overloads in the networking layer, the mapping information must be involved when selecting the route in the functional layer. The functional graph $G_F$ is transformed into new graph $\tilde{G}_F$ built up on the basis of the functional layer $L_F$ of the model presented in Section 2.1 using the method introduced in [51].

The crucial point of the method is a transformation step that builds up a directed graph with several levels. The originally directed topology is multiplied by the number $|U_r|$ of VNFs in the set $U_r$ plus one. These copies form the levels, and the VNFs $u_i$ in $U_r$ can label the upper $|U_r|$ levels. The lower level gets the label $u_0$.

Some auxiliary edges are added to connect the levels. If a node $v$ has the VNF-capability for type $u_i$, then an extra edge will be added for each edge going into $v$ and starting in $v'$. The auxiliary edge will connect the copy of node $v'$ in the graph-level labeled by $u_{i-1}$ to copy of node $v$ in the graph-level labeled by $u_i$. More than one node can offer the same VNF type, and there can be capabilities for more VNFs in a particular node.

As one can see, a new transformation might be needed for each request since the set of required VNFs can be different. Once the transformed graph is built, a path is searched by the shortest path algorithm for SFC-constrained, but any algorithm could be used that finds a path in a directed graph. The path is starting and ending nodes will be the copy of node $s_r$ in the lower layer and the copy of $d_r$ in the upper layer, respectively.

When this method is applied to my multilayer graph model, the transformation process can be summarized as follows. In order to interpret the levels, sublayers are created with identical topology as the original functional layer $L_F$. Some additional links between the sublayers also represent the nodes’ VNF capability.

The transformation produces a graph $\tilde{G}_F$ with a structure that depends on the request $r$ to be chained. Based on the services in $U_r$, a directed graph is created with $|U_r|$ sublayers and cross-sublayer functional links. The functional edge $\tilde{f}$ in $\tilde{G}_F$ always has an originated functional link $f$ from the original functional layer $L_F$. The mapping $H_{\tilde{f}}$ contains the same networking edges as $H_f$ where $f$ is the originated functional edge. I assume that the mapping $H_{\tilde{f}}$ is a loop-free path of network edges.

Figure 3.13 illustrates transformation of the original graph $G_F$ (Figure 3.14a) to the transformed graph $\tilde{G}_F$ (Figure 3.14b) using the method described above. As examples for edge-copies and auxiliary edges, the bold edges in $\tilde{G}_F$ shown in Figure 3.14b are created by the transformation, and all of them originated from the functional edge $f$ in $L_F$.

In the following subsections, I propose an optimal solution and a heuristic for the STE problem, which are of the augmented architecture because they need to exchange information between the layers but not the complete databases. Remember that the aim is to find a service chain path $P_r$ in the functional layer $L_F$ for request $r$ that avoids overloads on network edges in $L_N$. The methods assume the dynamic arrival of requests, i.e., the chain is calculated for each request individually and according to the current network state.

Both proposed solutions use the same graph transformation method to transform the functional layer into a transformed graph. After the transformation of functional layer $L_F$ to the transformed graph $\tilde{G}_F$, I aim to find a loop-free path $\tilde{P}_r$ in $\tilde{G}_F$ considering the mapping $H_{\tilde{f}}$ as well. Suppose there are nodes in the functional layer with capabilities of VNFs that are required consecutively by the service request. In that case, the auxiliary edge between the sublayers is generated with empty mapping since its starting and ending nodes are copies of the same original node.
The path $\mathcal{P}_r$ from $\mathcal{G}_F$ can be mapped back to the original functional layer $\mathcal{L}_F$, i.e., the reverse job of creating $\mathcal{G}_F$ is done to get the service chain $\mathcal{P}_r$. Note that $\mathcal{P}_r$ can result in loops in the route $\mathcal{P}^{DN}_r$ in the networking layer $\mathcal{L}_N$, but these are considered controlled loops.

### 3.3.4.1 The Optimal Solution

I use an Integer Linear Program (ILP) to formulate the optimal solution for STE and refer to it as Overload Avoiding Augmented Linear Program (OdAALP). The notation for the graph model introduced in Section 2.1 is extended by some new elements and applied in the formulation of the ILP on the transformed graph. $s_f$ and $d_f$ stand for the source and destination of edge $f$, and $w_f$ is a weight or cost of it. The latter can be set up in arbitrary ways leading to different results. If not otherwise stated, I use the cost of the path that realizes the original functional link, i.e., of its mapping.

The source, destination, and bandwidth of request $r$ are noted as $s_r$, $d_r$, and $b_r$, respectively. $C_l$ is the total capacity networking edge $l$ while $c_l$ is the available capacity, which can be defined as subtraction of the capacity $C_l$ and load $\sigma_l$ on the link $l$. If the subtraction result is lower than zero, then the remaining capacity remains 0.

The values $C_l$, $c_l$, and $b_r$ are used as real constants in the program, but they are not the only ones. The binary constant $h_{f,l}$ is an interpretation of the mapping $\mathcal{H}_f$ that describes whether the networking link $l \in \mathcal{L}_N$ is in the mapping of the transformed functional edge $f$.

$$h_{f,l} = \begin{cases} 1, & \text{if } l \in \mathcal{H}_f \\ 0, & \text{otherwise} \end{cases} \quad (3.1)$$

The most important binary variable I define is $p_f$. It shows whether for request $r$ the path $\mathcal{P}_r$ in transformed graph $\mathcal{G}_F$ contains link $\tilde{f}$ or not.
The objective is to find a minimum weight set of transformed functional edges by forming a path and not overloading the networking layer. I formulate the objective as follows:

\[
\min \sum w_{\bar{f}} \cdot p_{\bar{f}} \tag{3.2}
\]

The objective shall be found as subject to the node equilibrium constraints that assure the continuity of the path. Equation 3.3 ensures that what traffic comes in should go out from all transformed functional vertices in \( \bar{G}_F \), except for the starting and ending nodes of the request \( r \). These exceptions are handled by Equations 3.4 and 3.5:

\[
\sum_{\forall f: s_f=v} p_{\bar{f}} - \sum_{\forall f: d_f=v} p_{\bar{f}} = 0 \quad \forall v \in \bar{G}_F \setminus \{s_r, d_r\} \tag{3.3}
\]

\[
\sum_{\forall f: s_f=s_r} p_{\bar{f}} - \sum_{\forall f: d_f=s_r} p_{\bar{f}} = 1 \tag{3.4}
\]

\[
\sum_{\forall f: s_f=d_r} p_{\bar{f}} - \sum_{\forall f: d_f=d_r} p_{\bar{f}} = -1 \tag{3.5}
\]

In order to avoid overloads, the network link capacity constraint shall hold. The Equation 3.6 ensures that the bandwidth of request \( b_r \) does not exceed the available link capacity \( c_l \) for the network links \( l \) that appear in the mapping of the transformed functional edges in path \( \bar{P}_r \).

\[
\sum_{\forall f} p_{\bar{f}} \cdot h_{\bar{f}l} \cdot b_r \leq c_l \quad \forall l \in \mathcal{L}_N \tag{3.6}
\]

With an ILP solver, the program of OdAALP will find a chain if the constraints can be satisfied. However, in higher load cases, it can return without a solution.

### 3.3.4.2 The Heuristic Solution

OdAALP provides an optimal solution, but its complexity can be high, like ILP-based solutions in general. Since the graph transformation, setup of constants, and problem solution have to be performed separately for each demand, and this method can be time-consuming in real-life situations. The time of SFC computation significantly enlarges when having more edges in the functional graph and more required VNFs for the service requests.

I introduce the heuristic method **Overload Avoiding Augmented Shortest Path (OdAASP)** as a modified version of the Dijkstra algorithm [60]. As well-known, the original algorithm determines the shortest paths from a source to reach all other nodes in a graph. CSP applies the Dijkstra algorithm for the transformed graph \( \bar{G}_F \).

In the case of OdAASP, the main idea is to dynamically manage the set of edges that may connect new vertices to vertices already reached from the source. If a functional edge \( \bar{f} \) is mapped on a networking edge \( l \), which would get overloaded if \( \bar{f} \) would be used in the route of the request, \( \bar{f} \) is eliminated and not applicable in creating the chain. Figure 3.14 illustrates a situation where overload can be avoided using this modification of the Dijkstra algorithm. The vertices \( a, b, \) and \( c \) are already reached from \( s_r \), and vertex \( d_r \) could be reached by the lowest cost when using the edge \( f_2 \).
However, if both the mappings $H_{f_1}$ and $H_{f_2}$ contain the network link $l$, and the requested bandwidth is larger than the half of $c_l$, edges $f_1$ and $f_2$ can not be applied in the same route without overload link $l$. In such a situation, the overload avoiding algorithm $\text{OdAASP}$ eliminates the edge $f_2$, and only edge $g$ remains to reach vertex $d_r$. Note that this greedy method can not find any path between the source and destination due to the elimination steps.

The method steps are summarized in Algorithm 1.

```
Algorithm 1 $\text{OdAASP}$
1: $p_s \leftarrow 0$
2: $p_v \leftarrow \inf$ for all $v \neq s$
3: $\mathcal{P}_{sv} \leftarrow \{\}$ for all $v$
4: $R = \{s\}$
5: $Q = \{f \in \bar{G}_F : b_r \leq c_l, \forall l \in \mathcal{H}_f\}$
6: while $d \notin R$ and $Q$ is not empty do
7: for all $f \in Q$ where $s_f \in R$ do
8: $r \leftarrow s_f$
9: $n \leftarrow d_f$
10: if $p_r + w_f < p_n$ then
11: $p_n \leftarrow \mathcal{P}_r + w_f$
12: $\mathcal{P}_{sn} \leftarrow \mathcal{P}_r + f$
13: $m \leftarrow \arg\min(p_n : n \notin R)$
14: $R = R \cup m$
15: for all $f \in Q$ do
16: if $d_f = m$ then
17: remove $f$ from $Q$
18: if $s_f = m$ and $\text{OvlRisk}(f, \mathcal{P}_{sm}, b_r)$ then
19: remove $f$ from $Q$
```

The algorithm manages the path $\mathcal{P}_{sv}$ from $s$ to $v$ where $p_v$ is the cost of path $\mathcal{P}_{sv}$ for each vertex $v_f$ in $\bar{G}_F$. $R$ is the set of vertices for which the path $\mathcal{P}_{sv}$ is already determined, i.e., they can be reached from the source. The set $Q$ contains the edges that still can be used for reaching vertices that are not in $R$ yet without the risk of overloading any networking edge. The function $\arg\min$ gives back the node $n$ where $p_n$ takes its minimum.

$R$ gets augmented, and $Q$ diminishes each algorithm iteration. The objective is to insert the destination vertex $d_r$ into $R$ and thus determine $\mathcal{P}_{sd}$ for the demand $r$.

The first for-loop is used to select the vertex $n$ with the lowest distance from $s_r$ from the candidates not yet reached. After inserting $n$ in $R$, in the second for-loop, the edges are eliminated from $Q$ due to ending in $n$ or risking overload on any network links.
The overload risk is evaluated based on the mapping and done by the subroutine OvlRisk defined in Algorithm 2. It sums up the load on each networking edge \( l \) used in the mapping \( \mathcal{H}_f \) for the functional edge \( f \) starting in node \( m \). The calculation considers the mapping of functional edges that form the already determined path \( \mathcal{P}_{sm} \). If the remaining capacity \( c \) on any such \( l \) goes below the requested bandwidth, then the use of \( f \) would cause an overload, the subroutine returns true, and the edge gets removed from the set \( Q \) in Algorithm 1.

Algorithm 2 OvlRisk

Input: \( f, \mathcal{P}_{sm}, b_r \)

Output: True/False

1: function OvlRisk
2: for all \( l \in \mathcal{H}_f \) do
3: \( c \leftarrow c_l \)
4: for all \( \bar{f}_i \in \mathcal{P}_{sm} \) do
5: if \( l \in \mathcal{H}_{\bar{f}_i} \) then
6: \( c \leftarrow c - b_r \)
7: if \( c < b_r \) then
8: return True
9: return False

The complexity of the heuristic can be derived from the complexity of the Dijkstra algorithm for a graph with \( |V| \) nodes and \( |E| \), which is \( O(|V| \times |V|) \) in the naive implementation. The complexity can be reduced; for example, the Fibonacci heap implementation is of \( O(|E| \times \log(|V|)) \) complexity which is a good starting point.

The OdAASP solution transforms the functional graph into a graph where each VNF type required by the service request means another copy of the topology, and auxiliary edges are added to connect these copies according to the VNF capabilities in the nodes. The nodes’ number enlarges due to the multiple copies. However, the transformed graph’s special structure limits the number of edges to be considered in the steps of Dijkstra. The complexity of finding the functional path \( \mathcal{P}_{sd} \) in the transformed graph is:

\[
O((|\mathcal{E}_F| + N_{VU}) \times \log(|\mathcal{V}_F| \times |U|))
\] (3.7)

where \( N_{VU} \) is the maximal number of functional nodes that provide a given VNF type. \( N_{VU} \) depends on the applied VNF-capability placement but is limited by \( |\mathcal{V}_F| \).

Beyond the Dijkstra steps, it needs to be considered that in each step, the network links in \( \mathcal{H}_f \) will be analyzed from the load point of view by the subroutine OvlRisk. Since the number of links in the mapping is limited by the number of nodes in the network layer, the resulting total complexity is:

\[
O((|\mathcal{E}_F| \times |\mathcal{V}_N| + N_{VU}) \times \log(|\mathcal{V}_F| \times |U|))
\] (3.8)

3.3.5 Analysis of provided solutions

In order to understand how the proposed solutions work, I analyze a case where algorithms CSP, OdAALP and OdAASP provide different service chains for the same request.
The network topology is presented in Figure 3.15 and contains seven bidirectional networking links of 1 Gbps capacity. The VNF-type $u_1$ is available in node $c$, $u_2$ in node $a$, and $u_3$ in nodes $b$ and $e$. In this case, the functional layer $\mathcal{L}_F$ has an identical topology to the networking layer $\mathcal{L}_N$, and a one-to-one mapping is applied between their links. The costs of the functional connections are indicated in the Figure. Request $r$ starts in $s_r$ and ends in $d_r$ requiring 700 Mbps bandwidth and the ordered set of VNFs $u_1$, $u_2$, $u_3$.

![Figure 3.15: Simple topology for analyzing OdA algorithms](image)

After performing the three methods for the request $r$, the service chains are remapped to the original network, and paths $P_r$ are listed in Table 3.2.

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Route</th>
<th>Cost</th>
<th>Overld. Connections</th>
</tr>
</thead>
<tbody>
<tr>
<td>CSP</td>
<td>$s_r - a - b - c - b - a - b - c - d_r$</td>
<td>8</td>
<td>2 ($a - b$ and $b - c$)</td>
</tr>
<tr>
<td>OdAALP</td>
<td>$s_r - a - c - b - a - b - c - d_r$</td>
<td>16</td>
<td>0</td>
</tr>
<tr>
<td>OdAASP</td>
<td>$s_r - a - b - c - b - a - c - c - d_r$</td>
<td>17</td>
<td>0</td>
</tr>
</tbody>
</table>

Table 3.2: Result comparison for SFC algorithms

The algorithm CSP solves the problem without caring about the overloads in layer $\mathcal{L}_N$. On the other hand, STE methods can provide results only in cases where a service chain exists that does not overload any network link. In these cases, OdAALP indeed will find the service chain. At the same time, this is not always sure for the OdAASP due to its greedy behavior. An example of this can be the above scenario but excluding from the topology the network link between nodes $e$ and $c$.

When using STE, managing infeasible demands depends on the service providers’ policy because, primarily, they should be refused. If the refusal is not an option, a fall-back on a secondary service chain can be applied. The CSP algorithm is a simple and good choice for such a fall-back. I apply this option in the simulations made for performance evaluation. This will result some overloaded networking links even for OdAALP or OdAASP, because the fall-back solution CSP does not take into consideration the loads in $\mathcal{L}_N$.

From the analysis results, one can also see the other main drawback of the STE methods OdAALP and OdAASP. For getting a service chain that avoids overloads, we pay by using more, sometimes even much more links. On the one hand, this means a higher total load on the network. On the other hand, using longer paths also affects the QoS because it can lead to higher traffic-level (end-to-end) latency.

### 3.3.6 Performance Evaluation

I evaluate the proposed solutions using case studies based on a hypothetical Algerian backbone and BT Europe topologies, which are introduced in Section 2.3.2.
I aim to study a scenario where the bandwidth of each traffic demand of the type New Services grows linearly from 100 Mbps up to 1600 Mbps and from 0 up to 700 Mbps for Algeria and BT Europe topologies, respectively. The number of connections and the average bandwidth of each type of demand are summarized in Tables 2.2 and 2.1. The links with zero load are not considered in the average calculation. It is acceptable for our aims since links without load do not affect the QoS experienced by any traffic flow.

I analyze the link level evaluation to compare the solutions, the number of overloaded links in the network, the average length of the networking paths, and the average relative load on used networking links. These values are among the most critical factors that affect the QoS to be provided for a traffic flow. The packet-based evaluation is not performed for these scenarios. The lower the link-level values, the better QoS could be expected since a high relative load or overload on a link will cause a longer serving queue in the router and, thus, a higher delay and higher probability of packet loss. The proposed augmented SFC methods are expected to overcome the simple static-cost shortest path-based method CSP.

In order to study the effect of having different orders of traffic demands, I evaluate not only the average but also the minimum and maximum values of the main results for five inputs of the Algeria topology differing in demand orders. The collected results interpreted and represented as error bar plots show that the order of arrival does not significantly influence the trends.

When the network load gets high, the overload of some links cannot be avoided, and thus the overload-avoiding methods might fail to find a service chain, i.e., the method refuses it. For such cases, I applied a fall-back strategy and used the simple CSP to find a chain for the refused demand without caring about the unavoidable overloads.

### 3.3.6.1 Algeria topology

**Overloaded networking links**

Figure 3.16a and 3.16b present the number of overloaded links and the length of the networking paths, respectively. The fall-back option will bring overloaded network links even for OdAASP and OdAALP when the traffic raises above the point where the first refuses happen (at 600 Mbps).

![Figure 3.16a: Number of overloaded networking links](image)

![Figure 3.16b: Length of the networking paths](image)

**Figure 3.16**: Length of networking paths and number of overloaded links in Algeria topology
In the low-load range, all algorithms perform the same in which there are no overloaded links. In the mid-load range (400 – 1000 Mbps) OdAASP and OdAALP perform much better than CSP. At 1100 Mbps, it can be observed that OdAASP causes more overloads than CSP, which comes from using more networking links and thus causing higher loads on some links as shown in Figure 3.16b. However, at a bit higher load OdAASP performs better again, which comes from the behavior that it starts to use different functional links quite early during the chaining of all the demands.

The SFC solutions perform worse in the high load range (over 1300 Mbps) than CSP. This behavior is again normal since they use more network resources to avoid overloads (Figure 3.16b). After a point of saturation when all the links used in chaining are overloaded, it is intolerable to the network, just as in the case of TE routing solutions. Note that in this range, nearly all used networking links are overloaded in all three methods.

**Average relative load**

Figure 3.17b and 3.17a present the average relative loads of 1 Gbps and 10 Gbps links respectively. Due to the load distribution mentioned earlier, the OdAASP and OdAALP show lower averages than CSP, although the total sum of the absolute load on network links of the former ones cannot be smaller than that of the latter.

In the case of the core links (10 Gbps) shown in Figure 3.17b, and within the low-load range (below 600 Mbps) OdAASP and CSP algorithms almost perform the same. On the other hand, OdAALP performs better in this range due to choosing the optimal path, which are shorter paths than those of OdAASP as it is shown in Figure 3.16b.

In the mid-load range (between 600 and 1300 Mbps) the average relative loads of OdAASP and OdAALP show little differences. The interesting property here is that the values show some bouncing and do not monotonously grow with the network load. This behavior again comes from the behavior of the solutions that, after finding low-loaded resources, these methods use them until they get overloaded. When the load emerges, they involve further low-loaded resources, which results in a jigsaw-like behavior in the average values. The bouncing can also be seen in Figure 3.16b, where the length of the networking paths decreases due to involving other edge links in the paths.
In the high-load range (above 1300 Mbps), the network resources get exhausted, and the solutions cannot avoid overloads. The overload-avoiding methods refuse many demands, apply the fall-back option, and perform better than CSP only due to the load distribution, i.e., using more network links in the different chains. I suppose that the values of the QoS properties fall drastically in this case since, for the not refused traffic, the number of used networking links gets higher. The relative loads on the core links are significantly smaller for the SFC methods than for CSP, where even the average value grows over 1.0 at the end of the mid-load range.

In the case of the edge links (1 Gbps) shown in Figure 3.17a, there are lower relative loads, but with a minor difference between SFC methods and CSP. It can be observed that the bouncing effect is present, relatively strong in the mid-load range.

3.3.6.2 BT Europe topology

Figure 3.18a and 3.18b show the number of overloaded and the relative load of networking links of BT Europe topology. In the beginning, below 150 Mbps, the solutions almost perform the same, showing almost the same load and number of overloaded links. Within the range 100 and 250 Mbps, the OdAASP and OdAALP solutions start to use more network links, resulting in more networking links that eventually get overloaded. Due to overloaded links, the relative loads are also higher than with CSP, and the solutions start using new networking paths. After a while, at 300 Mbps, the relative link load drops, and ultimately, the number of overloaded links also gets lower.

The number of overloaded links in Figure 3.18a bounces this time to the upside in the case of OdAASP and OdAALP as the bandwidth of New Services gets higher at around 500 Mbps. The relative load results also follow the rise, then drop lower, and continue increasing linearly.

In order to better understand the bouncing behavior, I present the length of the networking paths and the number of loaded networking links in the networking path shown in Figure 3.19a and .

The average length of the networking path in the overload-avoiding methods makes impulsive moves to the upside because these methods do not consider only the shortest networking paths. The traffic gets transferred along more links resulting in a higher num-
ber of loaded links compared to CSP in Figure 3.19b. At a certain level of bandwidth, the impulsive move shows up again, but towards the downside. The lengths decrease because the network resources get exhausted, and the OdAASP and OdAALP solutions can not avoid overloads and often apply the fall-back option, i.e., the shortest path is used to transfer the demands. Above the bandwidth of 500 Mbps, the solutions start to use other longer networking paths as observable from the lengths shown in Figure 3.19a, leading also to fewer overloaded links. There are not many changes in the set of the used paths when network load increases, and both the length of the networking paths and the number of loaded links continue to have a constant value over 700 Mbps bandwidth.

![Figure 3.19: Length and number of networking links in BT Europe topology](image)

The differences between the results in the two networks come from the topological differences. BT Europe topology has a much lower node degree than Algeria topology, and some nodes are connected only with a single link. The traffic coming from or going to such nodes must be transferred through these singular links because there are no alternative networking paths, and these links likely get overloaded when traffic increases.

### 3.3.7 Summary

In this section, I have proposed different architectures to classify SFC solutions according to the amount of networking layer information applied in the decisions. I have also investigated the problem of overloaded links used in service chains and studied the effect of having overloaded links in the network, which lead to consequences in QoS.

To overcome these issues, I have proposed overload-avoiding SFC methods. I formulate the overloading problem in service chains as an integer linear program (ILP) and propose a heuristic algorithm. Their operation is analyzed to show the primary benefits, and the performance is also evaluated in larger topologies. The numerical results show that the proposed solutions perform better than the CSP method in terms of link loads and overloads when the network load is not too high, and the topology is dense. The reason behind this behavior is that OdAALP and OdAASP predict when a network link might get overloaded and use other functional links in the chain, also causing a kind of load distribution.

In addition, Algeria topology seems to use the resources of the core networking segment more than the edge segment due to the distribution of VNF capabilities allowing alternative functional connections using the core and applied to the chains. BT Europe topology has fewer network links, and the proposed solutions have fewer possibilities to find alter-
native resources. The overload-avoiding methods do not perform so well in such topologies but are still better than CSP from the network link load point of view.
3.4 Bandwidth-Aware Service Chaining

The SFC solutions presented in Section 3.3 create service chains that meet the required bandwidth and VNF order of the demands according to the STE concept, i.e., considering the topology and current load of the networking layer below. These solutions aim to find service chains that avoid the overload of networking links. They are based on weights, i.e., they first use the shorter functional links without considering other properties of these links. The short functional links mapped to fewer networking links are also preferred because of their lower delay properties. However, the network links in their mapping can get overloaded soon, as seen in the evaluation results.

In this section, I propose bandwidth-aware methods for the dynamic calculation of service chains by considering the current network load to avoid networking links overload, similarly to the objective of STE in Section 3.3. Overload avoidance is not directly applied in these methods; they only apply those functional links that seem to have more adequate resources in the networking layer.

3.4.1 Bandwidth-Awareness in Service Chains

Supported by the two-layer model introduced in Section 3.3.2, SFC algorithms can be created that consider the free capacities of links in the networking layer. These methods are bandwidth-aware because they suppose each node in the networking layer knows the current local available bandwidth on the connected links at any time. This information can be applied in the chain selection. The bandwidth-aware methods are created using a similar concept to STE solutions introduced in Section 3.3, but there is a significant difference. Although the main aim would be to create chains that avoid network link overloads, these methods will not necessarily succeed in doing it.

The bandwidth-aware methods are considered as augmented solutions since they work with specific properties of the networking layer while making a decision in the functional layer. The information to pass between the layers is on the same level as in the case of the STE solutions, where the mapping of the functional edges $H_f$ is supposed to be determined in advance, and only the available capacity values are considered in functional link’s selection.

In order to support the requirements of the ordered set of VNFs, these SFC algorithms apply transformed graph $\bar{G}_F$, which is generated from the functional layer $L_F$ by the transformation method described in Section 3.3.4. The determination of the service chains uses a cost value $\nabla_{\bar{f}}$ calculated for each functional edge $\bar{f}$ based on the mapping $H_{\bar{f}}$ as follows:

$$\nabla_{\bar{f}} = \sum_{l \in H_{\bar{f}}} F(C_l, c_l, b_r)$$  (3.9)

where $C_l$ and $c_l$ stand for the total and remaining throughput capacity of networking link $l$, respectively, and $b_r$ is the bandwidth required by request $r$. The cost $\nabla_{\bar{f}}$ depends on these values because the main idea behind the bandwidth-aware methods is to punish the functional links mapped on saturated or nearly saturated network links. If any networking link in $H_{\bar{f}}$ contains an overloaded networking link, the cost is set to a constant $F_{sat}$ value.
which is higher than any possible cost that could result by 3.9 for any functional edge in the network, e.g.,

\[ F_{\text{sat}} = 2 \times |\mathcal{E}_N| \times \max_{l \in \mathcal{V}_N}(F(C_l, c_l, 0)) \]  

(3.10)

The bandwidth-aware solutions do not prohibit the use of overloaded links, and the requests will not get refused if there are not enough available networking resources. In every possible \( \mathcal{P}_r^N \) path, at least one network link that would get overloaded when selecting this path as a service chain path for request \( r \), the network is saturated. The STE solutions would refuse the request and turn to the fall-back option, but the bandwidth-aware methods can still select a chain. It is certain that the cost of the chain in the saturated case is higher than \( F_{\text{sat}} \). Note that the consequences of resource overload are QoS violations.

The overload of a network link can occur in non-obvious situations in which there would be a controlled loop in the networking route \( \mathcal{P}_r^N \). STE solutions handle this issue properly, but the bandwidth-aware ones do not consider it because the costs are independently calculated for each functional edge. The punishment by using cost \( F_{\text{sat}} \) is applied only when the link is already overloaded or adding the bandwidth \( b_r \) would overload it.

### 3.4.2 Bandwidth-Aware Solutions

Referring back to the algorithms introduced in Section 3.2, \( \text{HopBased} \) considers simply the number of functional links in the chain \( \mathcal{P}_r \). \( \text{CostBased} \) takes one step beyond and considers a simple cost calculated for each functional link as the number of hops in the mapping \( \mathcal{H}_f \). I propose bandwidth-aware solutions which are based on looking for the minimal cost heuristic also used in \( \text{CSP} \), but consider different cost functions \( \nabla \bar{f} \) applied for a functional link \( \bar{f} \in \bar{G}_F \).

The aim is to identify and prefer links with enough capacity to transmit the requested bandwidth, which may leave enough available capacity for further requests. I define three cost functions \( F(C_l, c_l, b_r) \) summarized as follows:

- **Inverse of Absolute Remaining bandwidth (IAR)** aims to use the “widest” \( \mathcal{P}_r^N \) route and only considers how much bandwidth is still available on networking link \( l \). \( \text{IAR} \) applies the inverse of the absolute value of the link’s remaining capacity \( c_l \), i.e., the cost function used in \( \nabla \bar{f} \) is calculated as follows:

\[ F(C_l, c_l, b_r) = \frac{1}{c_l} \]  

(3.11)

- **Inverse of Relative Remaining bandwidth (IRR)** does not aim to determine and apply the widest route. It selects a chain with not so much used network links, considering whether the remained bandwidth on a link means a large or a small amount by simply relating the remaining bandwidth to the capacity. For two network links with different total capacity \( C_l \), the same absolute value of \( c_l \) can mean a very different situation, and the costs should differ. For instance, having two types of network links, one with 1 Gbps and another with 10 Gbps capacity, then 800 Mbps remaining bandwidth means 80% on the 1 Gbps links, while only 8% on the 10 Gbps links. The cost function applies the inverse of the relative remaining bandwidth and is defined as follows:

\[ F(C_l, c_l, b_r) = \frac{C_l}{c_l} \]  

(3.12)
• **Inverse of the Estimated Absolute Remaining bandwidth (IEAR)** aims to consider the state of the network links after possibly deploying the current demand and then find the widest route. This way, IEAR tries to avoid the strong loading of network links. For example, if the requested bandwidth \(b_r = 50 \text{ Mbps}\), deploying it has little impact when the remaining capacity \(c_l = 500 \text{ Mbps}\), but the impact is high when it is only 100 Mbps. IEAR estimates the total remaining bandwidth as the difference between the current remaining bandwidth, and the cost function is defined as follows:

\[
F(C_l, c_l, b_r) = \frac{1}{c_l - b_r}
\]

(3.13)

Remember that if the remaining capacity \(c_l\) or its estimated value is equal to or lower than zero, then the link \(l\) is considered overloaded, and the bandwidth-aware solutions assign a high punishment cost to each functional link \(f\) that is mapped onto link \(l\). Since the solutions determine the chain with the lowest cost, these functional links will not be preferred to use in the chain.

### 3.4.3 Performance Evaluation

I evaluate the proposed solutions using case studies based on a hypothetical Algerian backbone and **BT Europe** topologies, which are introduced in Section 2.3.2.

I aim to study a scenario where the bandwidth of each traffic demand of the type New Services grows linearly from 0 up to 800 Mbps and from 0 up to 700 Mbps for **Algeria** and **BT Europe** topologies, respectively. The number of connections and the average bandwidth of each type of demand are summarized in Table 2.1 and 3.3 for **BT Europe** and **Algeria** topologies, respectively.

<table>
<thead>
<tr>
<th>Type</th>
<th>Bandwidth (Mbps)</th>
<th>Priority</th>
<th>PacketLen (Kbit)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Video</td>
<td>2.11</td>
<td>WFQ-0.5</td>
<td>500</td>
</tr>
<tr>
<td>Voice</td>
<td>1.16</td>
<td>Strict</td>
<td>80</td>
</tr>
<tr>
<td>Best Effort</td>
<td>3.98</td>
<td>WFQ-0.2</td>
<td>100</td>
</tr>
<tr>
<td>Int. Video</td>
<td>3.03</td>
<td>Strict</td>
<td>500</td>
</tr>
<tr>
<td>New Service</td>
<td>0 – 800</td>
<td>WFQ-0.3</td>
<td>200</td>
</tr>
</tbody>
</table>

*Table 3.3: Traffic types characteristics*

In order to compare the bandwidth-aware solutions with SFC solutions in Section 3.3, I calculate the average relative load of the networking links to analyze the number of overloaded links in the network. I use the theoretical study in Section 2.2 to analyze end-to-end QoS properties. To compare the results with bandwidth-aware solutions, I use the algorithm **CSP** as a referenced solution.

#### 3.4.3.1 Overloaded Network Connections

Figure 3.20a and 3.20b present the number of overloaded networking links for both **Algeria** and **BT Europe**, respectively. In Figure 3.20a within the low-load range (below 200 Mbps), IAR, IRR, and IEAR are performing the same, and there are no overloaded network links. While in Figure 3.20b, IAR and IEAR show no overloaded links but IRR shows a single overloaded link, due to the sparse topology.

In Figure 3.20a within the mid-load range (300–600 Mbps), the bandwidth-aware methods perform similarly, and significantly better than CSP. In the high-load range (over 600 Mbps), CSP experiences a higher number of overloaded links, indicating a less efficient allocation of bandwidth resources. IEAR, IRR, and IAR, on the other hand, maintain a relatively low number of overloaded links, demonstrating their effectiveness in managing network loads.
(a) *Algeria* topology

(b) *BT Europe* topology

**Figure 3.20:** Number of overloaded networking links

Mbps), the number of overloaded links is elevated when the absolute remaining capacity is considered, i.e., for IAR, but it is still moderate for IRR and IEAR. This behavior of IAR comes from having too many network links in the chains and causing higher loads on some links. In the high-load range, the proposed solutions perform worse because the network link overload is not avoided but only punished by higher costs.

Figure 3.20b shows the number of overloaded links in the sparse topology. For this case, the bandwidth-aware solutions cause more overloaded links than CSP does in nearly the whole range of network load. More effects cause this behavior. The network contains singular links inevitable to being used in service chains for some traffic relations and getting overloaded soon when the load emerges. On the other hand, the bandwidth-aware solutions do not use the fall-back option and do not return to the shortest path when overload is inevitable, i.e., they can use more network links even in overloaded situations. Loading more links can lead to more potential overloads. Method IRR performs better at higher loads because it considers the relative network link load in the cost function.

### 3.4.3.2 End-to-end QoS results

Figures below present the average end-to-end packet delay for different priority classes in different topologies. Each figure represents the delay calculated for only one priority class in the function of increasing the bandwidth of New Services in each iteration.

Figure 3.21a and 3.21b present the average end-to-end packet delay calculated for the demands of *strict* priority class. In Figure 3.21a, the delay increases with the load because the network paths get longer, but the proposed methods still show good performance compared to CSP in the dense topology. Figure 3.21b shows that the delay received for the proposed methods is higher than that of CSP in the BT-Europe topology due to the combination of having more overloads and longer network paths.

For CSP, the delay is moderate and independent from the network load. The packets of the *strict* class are served first in the queue, and since CSP uses only static link costs and does not use any alternative networking paths, the load does not influence the service.

In the mid-load range, the delay of IAR and IEAR rises faster than IRR, while in Figure 3.21b IAR and IEAR show almost the same performance. The solution IAR and IEAR start to use low-load networking links, which might be applied in the chain of other traffic
Figure 3.21: Average traffic-level delay for strict class

demands. However, on these links, traffic of class strict is prioritized to be served in the queue, restricting the other traffic in the queue. The same packet level behavior works for the IRR algorithm, except that it prefers the low-loaded links and chooses shorter paths for the chains.

Figure 3.22a and 3.22b present also the average end-to-end packet delay calculated for priority class WFQ-0.3, i.e., the New Services. The trends are similar to the previous ones, but the delay elevates stronger because the service of this class is also affected by other classes. Figure 3.22a and 3.22b do not show the delay value for the case of 0 Mbps bandwidth because there are no demands and the weighting time equals to zero.

The delay for CSP increases in both topologies, giving higher values than any other solution in the low and mid-load range of the dense topology.

In the sparse topology, bandwidth-aware solutions provide high delay values by using longer paths with different network links in the selected chains. On the other hand, the elevation of New Services bandwidth may cause the WFQ-0.3 traffic to fill or overrun the weighted portion of the capacity remaining after serving by the strict class, and packets have to wait in the queue. In high-loaded classes, the capacity of some network links
can be fully consumed, and then there are not enough resources to handle any upcoming packets from this class.

Figure 3.23: Average traffic overload reduction for WFQ-0.3 class

Figure 3.23a and 3.23b present the average overload reduction factor for the class WFQ-0.3. Similarly, as above, the proposed methods are performing well in the low-load range, where there are no overloads or at least not fully loaded links because the demands of this class are routed over different network links. In contrast, the traffic gets reduced also in this region when using the CSP according to its higher number of overloaded links as observed in Figure 3.20a and 3.20b. By elevating the New Services bandwidth, the reduction factor increases rapidly for all the solutions up to intolerable values as a consequence of overloading a significant part of the networking links. However, the bandwidth-aware solutions show lower overload reduction values compared to CSP.

3.4.4 Summary

This section presents the motivation and limitations of using STE solutions, which consider the topology and current load of the networking layer and the required bandwidth and VNF order of service demands. I have proposed heuristic bandwidth-aware solutions for the dynamic creation of service chains. These methods consider the total and remaining bandwidth of network links in the mapping of functional links during the chain selection by using different costing functions.

The numerical results and the evaluation of QoS results show that in the low and mid-load regions of the dense topology scenario, these solutions perform better than the reference algorithm CSP that misses the bandwidth awareness. Except in the mid-load range, the average delay values are acceptable and better than those of CSP until a point of saturation where IEAR turns to go worse. Unfortunately, the methods do not perform so well in a sparse topology due to the low number of alternative paths.
4.1 Related works

Network Slicing is one of the most crucial parts of 5G core networks. Its definition has never been unique, clear, and precise. It varies from different perspectives of the various service providers. For instance, NGMN [61] defines a network slice as a set of network services that consists of three layers, Service Instance Layer, Network Slice Instance Layer, and Resource Layer. The network slice runs on top of physical resources, where both network services and resources conform to a logical network to deliver specific requirements. Slice can be defined as a set of network and VNF resources that can support one or more services, each with a prescribed series of VNFs that the service traffic shall pass. The supported services can be told to be in the slice and might also require a level of QoS. The most exciting question is how to assign resources to the slices and map the demands to them.

The authors of [62] investigate the service chain embedding problem for diversified 5G slice requirements, considering the sharing property of VNFs. They develop a fine-grained approach that considers resource requirements and the limited traffic processing capacity of VNFs, which can be shared (or not) among slices depending on VNF functionalities.

The authors of [63] formulate the problem of statically embedding service chains into slices while also considering network link capacities. In [64], another MILP formulation is given to optimize slices over multiple domains and accept multiple services in each slice. The authors also present a heuristic that can guarantee the QoS requirements for the services by allocating the needed resources for the slices.

Paper [65] considers slices supporting demands with uncertainty in number and requested resources. The presented model involves a probabilistic approach of provisioning the node and link resources to fulfill the requirements. The problem of mapping the uncertain demands on the resources is formulated as a nonlinear constrained optimization problem, and then it is reduced to a parameterized MILP problem. Considering the uncertainty allows mappings that might be used in dynamic scenarios too.

Paper [66] extends the slice demand mapping problem to include guarantees on the end-to-end traffic latency and to use a combined objective for the optimization. The authors consider the option of flexible routing, or in other words, load balancing of traffic via multiple paths, and present an MBLP formulation for the problem. In their model, the latency calculation considers only the propagation delay and a static NFV delay while
neglecting the queueing delays at the network nodes. Due to the high complexity of the complete problem formulation, a reduced version is contributed too.

The slice mapping solution presented in [67] also applies multiple paths for a different reason. The paper takes another important requirement for SFC under the scope and design slices with guaranteed availability.

The authors in [68] present models for sliced networks to investigate cost reduction promises when using NFV and network slicing technologies. The models aim to improve network efficiency by allocating network costs to deploy slices where the authors interpret a network slice as supporting only one service demand. However, according to the definition above, a slice is given as a service set with the required VNF set.

Moreover, the authors of [69] focus on the end-to-end network slice life-cycle management performed on different sites using a single management and orchestration entity and present a coherent proof of concept. They propose algorithms for efficiently activating, deactivating, and decommissioning the network slices, using their real-time status information from Network Slice Management Function (NSMF). The results show that by adopting a better strategy for controlling various phases of a slice’s life-cycle, the algorithms can reduce the response time for a user request by 50%.

4.2 Network Slicing Model

From the previous chapters’ results, it can be seen that to handle the service demands’ QoS requirements, the network infrastructure and its relations to the functional links must consider the mapping of functional links on network links. In network slicing scenarios, the challenge is extended since the question is how to manage the resources of different slices. Therefore, the multilayer model proposed in Section 2.1 needs to be extended to support network slices.

From the service requirements point of view, a slicing concept can define a traffic type with an ordered set of required VNFs and QoS values for each slice. Moreover, this traffic type might determine the high- or low-level traffic parameters, such as request arrival rate, inter-arrival time, and packet lengths. For example, it can refer to a slice supporting voice and another supporting real-time video calls. It is worth mentioning again that not like some other works, my slicing model concept does not restrict the slice to only one possible pair of traffic end-nodes. Instead, it supports a set of such relations, i.e., traffic requests of the same slice can have different endpoints. The important is that they are of the same type.

The slicing can be embedded in the multilayer model considering a slice as a subset of VNF-capable nodes and links of the functional layer $L_F$ that connect these nodes. The subsets can be optimally or sub-optimally selected from the whole sets of VNF-capable nodes and functional links.

In a strict dedication model, network resources such as VNFs or network links’ capacity can be assigned to one or more slices. Besides this resource assignment, the slicing model shall consider the current network load measured on links distinguished for each slice, i.e., the load of service requests coming from a given slice $S_i$ must be summed up. This concept suggests that a part of the functional graph gets assigned to slice $S_i$ that might be selected in advance according to the slice’s needs. The reservation of network links can be neglected since the assigned resources usually do not belong directly to the Infrastructure Providers (InP). However, the tenant of the slice, and their reservation can be dynamic.
However, I assume no strict dedication of resources but a shared model, i.e., all the slices can use any network resources. A mechanism for coordinating the use of resources in $L_F$ and $L_N$ is needed to support the requirements. Politics and methods for efficiently assigning VNF resources like CPU, time-slots, or memory to traffic requests of different slices are out of the scope of my research, i.e., my model does not consider any related limitation in the functional layer.

In Section 3.3, I distinguish three architectures of cooperation between layers, *overlay*, *integrated*, and *augmented*. These architectures are also valid in network slicing scenarios. However, another perspective should also be considered here because the relation to other slices is not obvious during the SFC process. I define three basic schemes that can be applied when the service chain is calculated for a service request:

- The *independent* scheme does not allow any reservation. A load of other slices gets ignored, i.e., during the SFC of a slice’s demand, only the functional resources of the slice are considered. This scheme matches well in the overlay architecture but can cause quality degradation due to network links’ overload.

- In the *static* scheme, network links’ bandwidth is statically divided and reserved for the tenants, and each tenant considers only the reserved resources. The static assignment can lead to inefficient usage because the offline optimized bandwidth reservation is hardly applicable when user demands arrive dynamically.

- In the *dynamic* scheme, each tenant considers the current network load measured on links. This way, the traffic load of the slices or even other tenants can be considered in determining the chain. This scheme also requires integrated cooperation between the physical and network layers.

Slicing requires cooperation between networking layer $L_N$ and functional layer $L_F$ in which the functional links are assigned to slices statically or dynamically. From the resources point of view, the slice $S_i$ is defined as a set of functional edges assigned to it in a shared or exclusive manner. The slice’s relation to the network links is derived through mapping functional links to network links that provide the essential transport service for the request $r$ of $S_i$.

Formally, for each slice $S_i$, the functional layer $L_F$ contains the graph $G_F^i \subseteq G_F$, and only $G_F^i$ is considered when SFC is performed for demand coming up in $S_i$. For each functional edge of $G_F^i$, there is a mapping $H_f$ on the edges of networking graph $G_N$ in the networking layer $L_N$.

I propose a shared slicing model using five important assumptions about the subgraphs for the case of a single tenant:

- the functional graph $G_F$ is sliced up into $G_F^i$ subgraphs using the same set of vertices but different sets of edges,

- VNF capabilities in node $v \in G_F^i$ can be different for each $i$,

- the mapping $H_f$ of edge $f \in G_F^i$ on edges of networking graph $G_N$ is not dependent on $i$. In other words, two slices that use the same functional link in their chains use the same network resources,

- current capacity and load values of network link resources are available for all slices,
• Optionally, the node capacities can also be shared, i.e., decisions should consider the current capacities and loads of VNFs used by more slices, or VNFs can be assigned to individual slices.

On the other hand, if there are multiple tenants ($M > 1$) which may not share VNF resources and support slices, slightly different assumptions are required:

• An extension of the single-tenant case is possible by applying a functional graph $G_{F,t}$ for each tenant $t \in 1..M$,

• The functional links belonging to different tenants can be mapped on either the same or different network links,

• The mapping scheme depends on the way that the InP uses in sharing its network resources among the tenants: it can be based on a strict reservation or statistical multiplexing, applying queueing techniques with weighted service for the packets and optionally with priorities,

• The set of supported VNFs might differ for the tenants, and their availability in the nodes can also be different,

• Sharing VNF resources among the tenants is not suggested,

• Functional graph $G_{F,t}$ can be then sliced into $G^i_{F,t}$ subgraphs according to the slices of tenant $t$.

Figure 4.1 illustrates the concept of using the layered graph with functional subgraphs for slices of a single tenant. The edges of the networking graph are indicated with dashed lines and those of the functional graph with solid lines.

Figure 4.1: The layered graph model of an example with two slices

The example contains two slices: $S_1$ supports a service that requires VNFs $u_1$ and $u_2$ to pass, while the service in slice $S_2$ requires VNFs $u_2$ and $u_3$ to pass. The available VNFs are indicated near the nodes. Figure 4.1 shows how $G_F$ is sliced up into $G^1_F$ and $G^2_F$ subgraphs that share the functional link between nodes $c$ and $d$.

A rather important case is the use of edges $a-c$ in subgraphs $G^1_F$ and $b-c$ in $G^2_F$. These functional edges are dependent through the mapping on $G_N$ since both will use the network link $n-c$.

The networking graph $G_N$ of an InP is usually not full, but the overlaying functional graph $G_F$ can be a full one. In such a case, both subgraphs $G^i_F$ and $G^j_F$ of slices $i$ and $j$ can be full graphs and may differ only in the VNF capabilities of the nodes. My concept does not exclude this particular case, but there are several reasons to avoid it:
Many VNF placement solutions include creating a functional graph, sometimes called a virtual network. This network contains only a moderate set of edges and links, interconnecting the VNF-capable nodes. These graphs are not full graphs and accord more to the requirements of user demands in the slices.

The shorter functional links are mapped to fewer network links, so they are preferred because of their lower delay properties.

The functional graph often fully matches the networking graph itself.

Having more edges in the functional graph enlarges the time of SFC computation, especially when more VNFs are required in the service.

From the management point of view, the number of functional links can be critical when they are statically mapped on network routes since each requires static route entries, tunnels, or SDN flows to be set up and updated in the network. My concept does not allow dynamic mapping since it can cause shorter visibility on QoS. Most of the solutions for VNF capability placement also provide the mapping and consider its use in the SFCs.
4.3 Service Chains in Network Slicing

To serve a rising number of services, many service providers and research institutes use a single network architecture to connect their facilities and customers. Consequently, the capacity to tailor transportation solutions to specific application requirements is essential. Creating network slices with various characteristics that can stand on top of shared network infrastructure is one example of how this could be accomplished.

The 5G mobile networks include slicing [70] implied as allocating network and computation resources into software- or hardware-separated sets based on the services provided by the network. Various services may require different VNF and network architecture. Thus VNF resources may or may not be shared among them. On the other hand, since network slicing shall ensure low latency and guaranteed bandwidth for different services, performing SFC and orchestration for the service traffic must be done within the constraints of the slice’s allotted resources.

In Section 4.2 I extend the original multilayer model presented in Section 2.1 with the slicing approach. In this part, I propose slice-aware algorithms for the dynamic calculation of service chains similarly to the objective of Section 3.3 and 3.4 by taking into account the current network load of slices to avoid networking links overload. In order to allow network resource preservation for other slices, these slice-aware solutions determine chains by limiting the current load of the different slices. I expect that slice load limitation helps to avoid violating the links’ capacity and overloading them.

4.3.1 Slice-Awareness in Service Chains

When it comes to network slicing, the separation of resources supported by a strict allocation mechanism is widely applied. VNF-capable nodes, functional, and network links are devoted to each slice of the slice-set $S$ in such scenarios. This dedicated slicing may exclude the use of previously unavailable resources.

It is simple to see that exploitation increases when slices can share network resources. However, in the case of shared slicing, operators may consider congestion of different service traffics, which can result in unexpected QoS issues when the load increases. To demonstrate this point, I consider a multi-slice scenario with a simple network in the example presented in Figure 4.2. Functional layer links are assumed to use one-to-one mapping on the network links.

The slices match different services with different required VNF series and QoS requirements. Due to the latter, I consider a weight $w^j$ assigned to each slice $S_j$. The weights are used in the packet scheduling applied for the traffic on the network links. They determine the planned usage weights of the network resources with loads coming from different slices.

I consider the VNF $u_1$ available on the nodes $K$ and $N$, and $u_2$ on the nodes $L$ and $M$, while slices $S_1$ and $S_2$ are served with weights $w_1=w_2=0.5$, and requiring VNFs $u_1$ and $u_2$, respectively. The capacity of each link is 1 Mbps.

The bandwidth of requests ($r_1$, $r_2$ and $r_3$) from slice $S_1$ is 0.25 Mbps, and requests ($r_3$ and $r_4$) from slice $S_2$ is 0.3 Mbps. The requests, illustrated with blue and red solid lines, arrive in the indexes’ order, and I use a simple SFC resulting in the chains shown in the third column of Table 4.1.

The service chains of requests from slice $S_1$ and $S_2$ are shown in Figure 4.2 with dotted blue and red lines, respectively. Regarding the fourth column of Table 4.1 one can observe
Figure 4.2: Example network supporting slices

<table>
<thead>
<tr>
<th>Requests</th>
<th>Slice</th>
<th>Service chains</th>
<th>Load on link K – L Mbps</th>
</tr>
</thead>
<tbody>
<tr>
<td>$r_1(A - C)$</td>
<td>$S_1$</td>
<td>A-K-L-C</td>
<td>0.25</td>
</tr>
<tr>
<td>$r_2(B - D)$</td>
<td>$S_1$</td>
<td>B-M-N-D</td>
<td>0.25</td>
</tr>
<tr>
<td>$r_3(E - F)$</td>
<td>$S_2$</td>
<td>E-K-L-N-F</td>
<td>0.55</td>
</tr>
<tr>
<td>$r_4(E - F)$</td>
<td>$S_2$</td>
<td>E-K-L-N-F</td>
<td>0.85</td>
</tr>
<tr>
<td>$r_5(A - C)$</td>
<td>$S_1$</td>
<td>A-K-L-C</td>
<td>1.1</td>
</tr>
</tbody>
</table>

Table 4.1: Service chains and load results in the example for slicing

the overload of link K-L after the chaining of $r_5$. Taking this situation under a flow level analysis, we find that the traffic of slice $S_1$ will get 0.5 Mbps according to the scheduling weights, i.e., it will not suffer losses. In the case of the traffic of slice $S_2$, we get a packet loss of more than 18 percent.

Since some links in the network are lightly loaded, the overload can be avoided by using a kind of load balancing, i.e., applying alternative chains for some requests. The traditional way is to choose a different chain for request $r_5$ that does not contain the link K-L, e.g., A-K-M-N-L-C. However, note that this way, the traffic of $S_1$ had to use a longer chain and put more load on further links because the traffic of slice $S_2$ was too greedy.

4.3.2 Slice Limitation Algorithms

I aim to reduce the networking links load by applying link preservation to overcome the overload in slices. I apply allocation-based service chaining solutions that limit the load of the slice on the networking links. It should reduce the possible differences between the relative load coming from slice $S_i$ and the weight $w_i$ applied in WFQ for serving the packets on queues attached to network links.

The primary concept behind load limitation-based service chaining is to use a higher cost for functional link $f$, when the slice traffic load would override a given limit on it or any network link $l$ in the mapping $H_f$. I introduce two algorithms SLF and SLN, differing from the considered layer point of view. SLF uses the slice limitation with limit $SL(S_i, f)$ on functional link $f$ in $L_f$ considering the total capacity $C_f$ and the remaining capacity $c_f$. SLN uses the limit $SL(S_i, l)$ on network link $l$ in $L_N$. The algorithms determine a chain for each request coming from slice $S_i$ by imposing an appropriate limit value for $S_i$ that is between [0, 1].
• **Slice Limitation in** $\mathcal{L}_F$: **SLF** modifies the cost $\phi_f$ for each functional link $f$ where the relative traffic load originating from a slice $S_i$, including the current request form the same slice would exceed the predefined limit $SL(S_i, f)$.

Equation 4.1 formalizes the condition to be checked. The sum of requested bandwidth $b_{r_j}$ is calculated for the requests $r_j$ that are in slice $S_i$ and have the functional link $f$ in their service chain. This sum, plus the requested bandwidth $b_r$ of the current request $r$ is compared to the product of slice limit $SL(S_i, f)$ and the capacity $C_f$ of functional link $f$.

$$b_r + \sum_{r_j \in S_i, f \in P_r} b_{r_j} > SL(S_i, f) \times C_f$$  \hspace{1cm} (4.1)

If the condition is true, the cost $\phi_f$ gets modified to a relatively high value, like $10^6$ times greater than its original cost. The multiplier constant has to be chosen so that the new cost overruns the sum of the costs of all the functional links where the condition is false.

After modifying the costs, the algorithm calculates the lowest cost chain $P_r$ for request $r$ from slice $S_i$ based on solution CSP introduced in [51]. Applying the modified costs **SLF** will avoid choosing the functional links with higher costs as long as there are other usable links with a moderate load of slice $S_i$ traffic.

• **Slice Limitation in** $\mathcal{L}_N$: **SLN** works similarly to **SLF**, but the relative load is limited on network link basis. The limit $SL(S_i, l)$ is defined for slice $S_i$ and for network link $l \in \mathcal{L}_N$ with similar motivation as the $SL(S_i, f)$ values in **SLF**. It gets applied in the condition described in Equation 4.2 to decide whether the link should be used in the network path of the current request $r$.

The requested bandwidth $b_{r_j}$ is summed up for requests coming from slice $S_i$ and already using link $l$. The sum plus the bandwidth $b_r$ of the current request and compared to the limit level multiplied by the capacity of network link $l$.

$$b_r + \sum_{r_j \in S_i, l \in P^N_r} b_{r_j} > SL(S_i, l) \times C_l$$  \hspace{1cm} (4.2)

If the condition holds for link $l$, the cost $\phi_f$ is modified for each functional edges $f$ that are mapped to the networking link $l$, i.e., $l \in \mathcal{H}_f$. The new cost is high enough to allow **SLN** to prefer any other chains not containing such functional links.

Similarly to **SLF**, the cost modification is followed by selecting the lowest cost service chain.

Both solutions can also be applied in the case when the ISP’s politics forces independent handling of slices since the algorithms’ essential property is to evaluate the conditions for only the slice of the current request. The limitation allows a kind of bandwidth reservation for all the other slices. Cost modifications are valid only for the current service chaining; once it is performed, costs are reset to their original values.

4.3.3 Analysis of the algorithms

In this section, I present a scenario in which the algorithms **SLF** and **SLN** lead to different service chains to understand their differences better.
I consider the functional topology shown on the left side in Figure 4.3 where both the VNF capabilities $u_1$ and $u_2$ are placed in node $d_2$, only $u_1$ in node $d_3$, and all the other nodes are incapable to run any VNFs. The functional links are mapped to the networking topology shown on the right side in the same Figure. In each mapping, the lowest cost path is used, and the functional link costs accord to the total cost of the network links in this path. Costs are indicated near the links. Each network link has 1 Mbps capacity, which also induces each functional link to have 1 Mbps capacity.

![Figure 4.3: The topology of the functional and networking layers](image)

The requests $r_1$ and $r_2$ coming from slice $S_1$ require 0.3 Mbps bandwidth, and need to be transferred between the source $s$ and destinations $d_1$ and $d_2$, respectively. Request $r_3$ is in slice $S_2$ and requires 0.5 Mbps bandwidth between source $s$ and destination $d_3$. The slices $S_1$ and $S_2$ require to pass the traffic through VNF $u_1$ and VNF $u_2$, respectively. The requests arrive in the following order: $r_1$, $r_3$, and then $r_2$.

In this example, I consider the algorithm SLF and SLN to use uniform slice limits for each functional and networking links, as shown in Table 4.2.

<table>
<thead>
<tr>
<th></th>
<th>SL for slice $S_1$</th>
<th>SL for slice $S_2$</th>
</tr>
</thead>
<tbody>
<tr>
<td>functional link $f$</td>
<td>0.4</td>
<td>0.6</td>
</tr>
<tr>
<td>networking link $l$</td>
<td>0.4</td>
<td>0.6</td>
</tr>
</tbody>
</table>

Table 4.2: Slice limitation values for SLF and SLN

The solutions SLF and SLN determine service chains that use the paths in the functional and networking layer as shown in Figure 4.4 and 4.5. The dashed blue and red paths demonstrate the chains for requests in slices $S_1$ and $S_2$, respectively.

Both SLF and SLN select the chain $s - d_3 - d_1$ for request $r_1$ and $s - d_2 - d_3$ for request $r_3$. At the arrival of these requests, the slice limits are not yet validated, and no modification of functional link costs is needed because the loads on the functional and network links are low. The method SLF ignores the network layer and considers the limits only in the functional layer, and selects the chain $s - d_2$ for request $r_2$ using a common functional link with request $r_3$. The chain is independent of the chain of request $r_1$, but only in the functional layer $L_F$. However, SLN is transferring the request $r_2$ along a different path because at its arrival, considering the dependencies among the mappings, the load on the network link $s - a$ is over the slice limitation value of $S_1$. The cost will be modified for all the functional links that have this network link in their mapping $H_f$, i.e., in this example, for $s - d_2$ and $s - d_3$. Although not demonstrated, the requests coming after $r_2$ and from slice $S_2$ may find the functional links $s - d_2$ and $s - d_3$ with moderate loads.
Slice-aware methods limit the slice’s resource usage to spare bandwidth for later traffic from other slices instead of choosing the currently best chain for a request. SLN adjusts better to the network link capacities, while SLF may take fewer steps on the functional layer, leading to lower resource usage in the nodes.

### 4.3.4 Performance Evaluation

I evaluate the proposed solutions using case studies based on a hypothetical *Algeria* backbone topology, which is introduced in Section 2.3.2. The proposed solutions have been tested and analyzed in *BT Europe* topology, but due to the node degree, the solutions perform more or less the same as CSP.

I aim to study a scenario where the bandwidth of each traffic demand of the type New Services grows linearly from 0 up to 1500 Mbps. The numbers of demands, the required bandwidth, and slice limitation value are summarized for each traffic type of demand in Table 4.3.

I analyze the link-level evaluation introduced in Section 2.2.1 to compare the proposed solutions and the previously introduced CSP and OdAASP methods. The number of over-loaded links and the average relative load on used links is evaluated since these values are among the most critical factors that affect the QoS for traffic demands coming from different slices. The links with zero load are not considered in the load-related average.
Table 4.3: Traffic types characteristics

<table>
<thead>
<tr>
<th>Type</th>
<th>Bandwidth (Mbps)</th>
<th>Number</th>
<th>Slice limitation</th>
</tr>
</thead>
<tbody>
<tr>
<td>BT Europe</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Best Effort</td>
<td>3.98</td>
<td>275</td>
<td>0.25</td>
</tr>
<tr>
<td>New Services</td>
<td>0 - 700</td>
<td>23</td>
<td>0.35</td>
</tr>
<tr>
<td>Int. Video</td>
<td>3.03</td>
<td>275</td>
<td>0.2</td>
</tr>
<tr>
<td>Video</td>
<td>2.11</td>
<td>275</td>
<td>0.1</td>
</tr>
<tr>
<td>Voice</td>
<td>1.16</td>
<td>275</td>
<td>0.1</td>
</tr>
<tr>
<td>Algeria</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Best Effort</td>
<td>3.98</td>
<td>702</td>
<td>0.4</td>
</tr>
<tr>
<td>New Services</td>
<td>0 - 1500</td>
<td>27</td>
<td>0.6</td>
</tr>
</tbody>
</table>

calculations since links without load do not affect the QoS. Besides these values, the length of the networking layer paths and the number of loaded links are analyzed to see the cost of using paths differing from the shortest one.

adjusted to packet serving.

Figures 4.6a and 4.6b show the relative load and number of overloaded network links, respectively.

![Average relative load networking links](image1)

![Number of overloaded networking links](image2)

(a) Average relative load networking links  
(b) Number of overloaded networking links

**Figure 4.6:** Networking links evaluation for Algeria topology

Within the low-load range, below 450 Mbps, the algorithms SLN, SLF and OdAASP perform nearly the same, and there are no overloaded links at this load. In general, one can observe in the higher load ranges that OdAASP and OdAASP perform between OdAASP and OdAASP from the relative load point of view.

Within the mid-load range, between 450-1000 Mbps, the slice limitation algorithms show some bouncing, and the number of overloaded links does not monotonously grow. Alongside, the relative load of SLF nearly converges with CSP, while SLF keeps linearly increasing. Such behavior of the slice limitation algorithms comes from the transfer of traffic demands along networking paths where the slice limitation condition could be violated, and the slice methods modify the functional link cost. The slice limitation algorithms find alternative paths that are very different from the shortest one. When the slice load is high, the demands will be transferred along the paths that violate the slice limit, which can overload network links.

The network resources get exhausted in the high-load range, over 900 Mbps, and none of the algorithms can avoid overloads. However, slice limitation algorithms, trying to find
a chain excluding the links where the slice traffic would be over the limit, still perform better than CSP and OdAASP.

Figures 4.7a and 4.7b present the average length of the networking path and the number of loaded networking links, respectively.

![Graphs showing average length of networking path and number of loaded networking links](image)

**Figure 4.7:** Networking path analysis for Algeria topology

Comparing the average length and number of loaded networking links can help to understand better the causes presented above. OdAASP and CSP do not inherit the features of slice allocation limit, and as expected, OdAASP uses longer networking paths compared to SLN and SLF. Note that these results are not shown in the Figures but can be derived from the results in section 3.3.

In the case of SLF, the networking paths are expected to be shorter than SLN because there could be many shortest chains to be selected in the functional layer where the functional links load does not violate the slice limit. Mapping these links to the networking layer will induce no longer paths than that of SLN, which tries to avoid slice violations of the network links. Moreover, as illustrated in Figure 4.7b, SLF shows a fixed number of loaded networking links except for the high-load region, suggesting that it uses the same set of links.

The expected behavior of SLN accords to the results in Figure 4.7a and those on loaded network links. Figure 4.7b shows that SLN uses many loaded networking links in the mid-load range, and the violation of the slice limits is also more probable. In such cases, SLN excludes the violating network links by punishment in the costs and may find a longer path for the chain. Note that the difference in path length averages between the two slice-aware methods is not too significant.

### 4.3.5 Summary

This section considers the slicing approach and extends the multilayer model to support multiple slices. I have investigated the basic effects of slicing and proposed heuristic service chaining solutions considering multiple slices.

The basic idea of the slice-aware methods is to limit the slice traffic on the links at the arrival of a request and preserve network resources for other slices to meet the QoS expectations. The limitation in the SFC is solved by using punishment costs on the functional links that would violate the limit.
The proposed solutions show better performance than the reference algorithm CSP in terms of overloaded links and network link load when the network load is moderate.
4.4 QoS Impacts of Slice Traffic Limitation

Network Slicing divides a single infrastructure into many logical network slices, where VNFs can be flexibly configured in response to the demands of applications to meet their QoS requirements. Most algorithms proposed for service chaining handle traffic requests independently, considering only the required resources like bandwidth or VNFs and ignoring other slices.

Some of the previously proposed SFC algorithms, like CSP, concentrate only on the number of used networking resources and ignore the overloading effect. Other methods like OAAASP try to avoid overloaded network resources if possible. Still, they do not consider that resource sharing is usually done on a class or slice basis. It can lead to inefficient use of the links and higher traffic loss when the load increases.

In order to ensure better QoS for applications, in Section 4.3, I propose SFC methods preserving network resources for other slices. The concept is told to be slice-aware since the methods concentrate on limiting the network link usage of the current request’s slice and consider only the nodes’ VNF capability, neglecting the exact VNF resource limits. SLF and SLN algorithms use the limitation of resource usage and can control the loads coming from the slices on the Functional or Networking Layer links, respectively. The applied mechanism is simple: the link cost gets vastly increased before the path selection if a special limit rate SL for the given slice would be overridden on the link by leading the chain or the path of the current request through it.

In this section, I analyze these slice-aware SFC mechanisms from the network QoS point of view. I propose different policies for adjusting the slice limit values with the queue-level settings of service and evaluate their behavior. In order to support analyzing packet-level results for individual queueing systems in overloaded scenarios, I extend the methods proposed in Section 2.2.1 and 2.2.3. I aim to show results on overloaded networking links and identify the effects on the traffic coming from different slices.

4.4.1 Adjusting Limitation Parameters

It seems obvious that the performance of the algorithms SLF and SLN presented in Section 4.3 depend strongly on the slice limitation parameter $SL$. I consider two basic cases of setting $SL(S_i, f)$ or $SL(S_i, l)$ for slice $S_i$ and functional link $f$ or network link $l$:

- in the simple case, the same values on each link $f$ or $l$ uniformly, e.g., according to the ISP’s preferences and politics regarding slice $S_i$,
- in a more generic case different values can be set on every links $f$ or $l$, e.g., according to the estimated load on that link.

When the settings for the queueing weights are uniformly applied on each link, the use of simple setting of $SL$ is recommended. The slice limitation might precisely adjust the limits to the set of $WFQ/LLQ$ weights realized in queues and reflect the provider’s preferences. However, this case may lead to poor performance and QoS quality if loads of slices are very dissimilar on the various links or the traffic is directed using traffic engineering. In the latter scenarios, possibly supported by measuring and feeding back the link loads, it may be worth applying the different case and setting up $SL$ values adjusted to the link’s load components coming from the slices.
In my research, the *simple* case is applied since the link loads are not known in advance, and a uniform LLQ setup is applied to the links. The next question is how to choose the values for $SL(S_i, f)$ or $SL(S_i, l)$ for the different slices $S_i$, and I propose three policies for adjusting these limit parameters. In order to give a clearer view, I assume normalized $w_i$ weights in WFQ, i.e., $\sum_{S_i} w_i = 1$. The basic policies for *simple* setting of $SL$ are as follows:

**Conservative (Cons)** The value:

$$SL^i = \min_{j \in S_i} w_j$$

on each link, i.e., the bandwidth of each slice is intended to be pushed below the load of the least weighted traffic class. It induces that SFC starts to use low loaded links quite early. This policy is supposed to work nicely only for the case when the weights are similar.

**Weight-aware (WeiAw)** The value:

$$SL^i = w_i$$

on each link, i.e., the links shall be loaded with slice traffics according to their weights. This policy should preserve as many bandwidth resources for a slice as it would take with the queueing. This policy is supposed to work well in generic networking cases when the weights mostly correspond to the slice loads.

**Liberal (Lib)** The value:

$$SL^i = \max_{j \in S_i} w_j$$

on each link, i.e., the limit for a slice can be higher than its weight. It induces SFC to start to use low-loaded links quite late. This policy should work well in cases where the traffic request pattern is not aligning well with the queueing weights.

I compare the different setting policies for the slice-aware algorithms in this section.

### 4.4.2 Modelling the Overloads

I assume that the QoS experienced by the users is highly affected by the chains created with SFC solutions since both the number of hops and the link loads impact the metrics. The higher number of hops in routes can enlarge the end-to-end delays and lead to more queues where the traffic appears, loads the resources, and might suffer packet losses. In the case of moderate network load, the slice-aware SFC solutions provide lower values in link-level results for load, delay, and packet loss. However, when the network slice loads elevate strongly, more and more functional links get higher costs. After involving many links in the chains, the SFC solutions will return to using links in the shortest chain, i.e., those with the minimum number of network hops. If this happens for many traffic requests, overloads can appear on links of these paths.

The evaluation model presented in 2.2.1.1 determines average bandwidth values even for overloaded situations, and a more accurate approach is proposed in 2.2.3 for packet-based QoS analysis. However, the latter is based on stable models only when the relative link load is less than 1, i.e., there is no overload on the queueing system. On the other hand, studying the network performance in scenarios with overloaded links cannot be avoided, and for this sake, I introduced the overload reduction measure.
The overload reduction is essential for network performance analysis since it helps to avoid unstable calculations in the queueing-based packet-level calculations. Assuming Network Slicing, WFQ and LLQ are weight-based serving policies for queueing, which allows the share of link capacity among the traffic of different classes joint to slices. In order to provide stability to the calculation methods, the concept of overload reduction needs to be extended to support traffic coming from different slices. Therefore, the calculation of link level measures such as relative load, experienced bandwidth, and overload reduction shall be extended too.

The concept is based on the assumption that the reduction of slice traffic in overload cases depends on how much the total load is over the link capacity and on the WFQ weights. The overload reduction $OvlRed_{S_i}^{l}$ represents the relative amount of reduction for slice $S_i$, i.e., the rate of its traffic that will be thrown away and not transferred along the overloaded link $l$.

Figure 4.8 presents three different load use-cases for a link shared among three slices $S_1$, $S_2$ and $S_3$. The slices are represented by color red, yellow and green, and have the WFQ serving weight 0.2, 0.3 and 0.5, respectively. The weight can also determine each slice’s limitation according to the Weight-aware policy. The first three columns in the Figure show the relation between the total traffic and the link capacity. In low and middle load ranges, resource sharing is based on the arrival of the requests, except when there is a violation of slice limitation. The slice limitation solutions preserve bandwidth for other slices and try to find new paths that will not overload any link. However, when the load is higher, overloads happen, and some of the traffic is thrown away, i.e., overload reduction occurs. The fourth column in Figure 4.8 illustrates the high load situation but showing which parts of traffic are thrown away. As one can see, slices $S_1$ and $S_3$ suffer the reduction because they require more bandwidth than expected according to their serving weight, while $S_2$ does not.

The value of overload reduction can be formally calculated for a link as follows. Let us assume that only one request $r$ coming from each slice $S_i$ represents its whole traffic, and its requested (or offered) bandwidth is $b_r^{S_i}$, while the average experienced bandwidth or throughput for this request is $\beta_r^{S_i}$. On each networking link $l$, with capacity $C_l$, the sum of experienced bandwidth does not exceed the capacity $C_l$, described in Equation 4.3.

$$C_l \geq \sum_{S_i \in \mathcal{S}} \beta_r^{S_i}$$  \hspace{1cm} (4.3)
In the case of a lower or middle link load where no overloads happen, i.e., for underloaded links, the weights do not play a significant role in the average load level. However, in the case of an overloaded link with capacity $C_l$, a particular subset of the slices can be defined. Let $S^{ul} \subset S$ be the subset of slices requesting less bandwidth than would be acceptable according to its weight. The condition for slice $S_i$ being in this subset is given in Equation 4.4.

$$b_{r}^{S_i} \leq w_i \times C_l$$  \hspace{1cm} (4.4)

The experienced bandwidth $\beta_{r}^{S^u}$ can be easily calculated as $b_{r}^{S^u}$ for slice $S^u$ in subset $S^{ul}$. For the slices not in the subset of slices $S^{ul}$, the capacity remaining on the link is shared according to the WFQ weights on this subset of slices.

$$\forall S_j \notin S^{ul}: \beta_{r}^{S_j} = \left( C_l - \sum_{S_u \in S^{ul}} \beta_{r}^{S_u} \right) \frac{w_i}{1 - \sum_{S_u \in S^{ul}} w_u}$$  \hspace{1cm} (4.5)

Equation 4.5 formulates the calculation of the experienced bandwidth $\beta_{r}^{S_j}$ for each slice $S_j$ that subset $S^{ul}$ does not contain. The sum of requested bandwidths for each slice $S_u \in S^{ul}$ is subtracted from the capacity $C_l$ of link $l$, and multiplied by the fraction of weight $w_j$ of slice $S_j$ and the sum of weights for each slice $S_u \notin S^{ul}$.

The model can also be extended to consider a strictly prioritized slice $S_p$, i.e., to support LLQ. First, the experienced bandwidth $\beta_{r}^{S_p}$ is calculated for slice $S_p$ as the minimum of its requested bandwidth and the link capacity, as shown in Equation 4.6. As the next step, the capacity of the link $C_l$ has to be decreased by $\beta_{r}^{S_p}$ before the subset $S^{ul}$ gets selected.

$$\beta_{r}^{S_p} = \min(b_{r}^{S_p}, C_l)$$  \hspace{1cm} (4.6)

According to the extended model described above, for each slice, the difference between the required and experienced bandwidth will be lost on an overloaded link. Therefore, it can be interpreted as slice load reduction coming from overloading, and I define the value of $OvlRed^{S_i}$ as the fraction of lost bandwidth and the requested bandwidth for slice $S_i$, as given in Equation 4.7. The higher the value of the overload reduction factor, the more significant part of the whole traffic is lost.

$$OvlRed^{S_i} = \frac{b_{r}^{S_i} - \beta_{r}^{S_i}}{b_{r}^{S_i}}$$  \hspace{1cm} (4.7)

To avoid instability, I apply the stochastic model with input parameters mimicking a reduction of loads by $OvlRed$ factors, i.e., the analysis will be done for traffic that does not overload the link. The simplest way of varying the inputs is to enlarge the mean of inter-arrival time, although higher moments might be affected too. Due to input data alteration, the waiting times and packet loss results are valid for the part of the traffic that is not thrown due to overloading. These waiting time results are also valid in general. However, when analyzing the loss, the overload reduction shall be considered as a macroscopic or flow-level traffic loss.
4.4.3 Performance Evaluation

I evaluate the packet-level properties for the policies in topologies of different scales. Based on the link-loads experienced in the simulator and considering the assumed QoS characteristics of slices, I calculate the metrics applying the theoretical models.

4.4.3.1 Small topology

First, let me introduce the analysis using the network topology illustrated in Figure 4.9, which also shows the VNF capabilities in nodes. Network links are of 1 Gbps capacity, and for the sake of simplicity, a one-to-one mapping is applied between the functional and networking layers. I assume two slices, traffic requests of slice $S_0$ (red) and $S_1$ (blue) require VNFs $u_2$ and $u_1$ respectively. In this scenario, a four-phase elevation of slice traffic is evaluated, i.e., more and more requests get added to slice $S_0$ between the node pair $E - F$. The traffic of $S_1$ is less increasing, and its requests are between node pairs $A - C$ and $B - D$. The number of requests to chain, the average of the demanded bandwidth, the mean packet lengths, and the assigned WFQ weights are summarized for each slice in Table 4.4. We uniformly apply the WFQ weights of each link to the simple method. The topology contains four edges in the center that are inevitable to use in the chains of both slices. I expect high traffic on these links, and the effects of congestion shall show the differences among the behaviors of methods and policies.

![Toy topology with service requests](image)

**Figure 4.9:** Toy topology with service requests

<table>
<thead>
<tr>
<th>Traffic</th>
<th>Number</th>
<th>Bandwidth</th>
<th>Packet Length</th>
<th>Weight</th>
</tr>
</thead>
<tbody>
<tr>
<td>Slice $S_0$</td>
<td>1-8</td>
<td>0.3 Mbps</td>
<td>8 Kbit</td>
<td>0.6</td>
</tr>
<tr>
<td>Slice $S_1$</td>
<td>2-3</td>
<td>0.35 Mbps</td>
<td>500 Kbit</td>
<td>0.4</td>
</tr>
</tbody>
</table>

**Table 4.4:** Traffic types characteristics

Figure 4.10 presents on its two y-axes the average values of overload reduction and packet loss calculated for the $S_1$ traffic requests. Due to the one-to-one mapping from the functional layer to the networking layer, methods SLF and SLN are identical. In the Figure, the policies proposed for setting up the SL limit values are shown for the slice-aware algorithm SLN to be compared with each other and with the results of simple CSP.

As expected, all the metrics are moderate in the case of relatively low and mid-range traffic load. The load elevation induces a packet loss increase, except for the CSP algorithm, where the overload of links appears already. The best-performing policy here is the conservative one.
In the higher load ranges, it can be seen that due to the overload of networking links, the overload reduction plays the primary role in the loss, while the packet loss decreases. The overload reduction of traffic ends with less loaded links and lower packet loss values. The chains are static in CSP, while in SLN with the liberal policy, the higher slice load on links is accepted before using alternative chain paths. This behavior leads to higher overloads.

The traffic suffers from a different effect in the case of SLN with weight-aware and conservative policies. The higher load induces a pretty early use of alternative chain paths and balances the load among more links, but these chains might be of more hops. More hops in the networking layer mean more links where the traffic and the packets can get lost. Furthermore, nearly all network resources can get overloaded when the network load gets high because traffic flows everywhere. For extra high load cases, the best performing policy is the weight-aware, where the $SL_i$ limits are adjusted to the weights used in WFQ.

The above explanations clarify the results on the end-to-end delay of $S_1$ requests presented in Figure 4.11. Although significantly higher losses, algorithm CSP, with its relatively short chain paths, performs well from this perspective. The SLN goes better with conservative and weight-aware policies, while the liberal one performs poorly.


4.4.3.2 Larger Topology

I evaluate the proposed solutions using case studies based on a hypothetical Algeria backbone introduced in Section 2.3.2. I aim to study a scenario where the bandwidth of each traffic demand of the type New Services grows linearly from 0 up to 1500 Mbps. The numbers of demands, the average of the demanded bandwidth (in Mbps), packet length, and WFQ weights are summarized for each traffic type in Table 4.5. I apply the Weight-Aware policy in the slice limitation algorithms due to its advantages observed in the small topology scenario.

I analyze the packet-level evaluation introduced in Section 2.2.3 and extend it to fit the network slicing model. I analyze the QoS properties to compare the slice-aware solutions with other service chaining solutions as follows:

- \( \text{OvlRed}_{S_i}: \) factor of reduction,
- \( \text{PktDelay}^g_{S_i}(W): \) packet delay,
- \( \text{PktLoss}^g_{S_i}: \) packet loss,

<table>
<thead>
<tr>
<th>Type</th>
<th>Bandwidth Mbps</th>
<th>Packet length Kbit</th>
<th>Weight</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Algeria</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>New Services ((S_0))</td>
<td>0 - 1500</td>
<td>8</td>
<td>0.6</td>
</tr>
<tr>
<td>Best Effort ((S_1))</td>
<td>3,98</td>
<td>500</td>
<td>0.4</td>
</tr>
</tbody>
</table>

**Table 4.5:** Traffic types characteristics

I consider two simplex traffic demand types, in other words, slices to be served in Algeria topology: New Services and Best Effort, referred to as \( S_0 \) and \( S_1 \), respectively.

**Average packet delay**

Figure 4.12a and 4.12b present the average end-to-end packet delays calculated for slices \( S_0 \) and \( S_1 \).

In the low-load range, below 450 Mbps, CSP and slice limitation algorithms perform nearly the same while values for OdAASP are slightly higher. In the mid-load range, 450-900 Mbps, on the one hand, the slice limitation methods show higher waiting times, which come from finding and using links that are low-loaded by the corresponding slice. The network resources get exhausted in the high-load range, over 900 Mbps, where very high waiting times are observed. There is also a descending trend in OdAASP and SLN, which comes from returning to choose often service chains with the shortest network paths.

Similar behavior can be observed for the other slice in Figure 4.12b, where the slice limitation algorithms perform equally or similarly to OdAASP. The descending trend at high network loads is not present because this slice’s load does not change, and the SFC for its requests will not return to the shortest network paths. In the case of CSP, the waiting time values are low because the chain paths contain few links in both the functional and networking layer.

**Factor of reduction and packet loss**

Figure 4.13a presents the average overload reduction calculated for slice \( S_0 \). One can observe that the demands suffer a pretty high overload reduction with CSP. The overload
avoiding algorithm OdAASP performs worse than the slice limitation ones in the low or middle load phase due to the high number of demands in slice $S_1$. Unlike SLF and SLN, OdAASP does not spare resources for $S_0$ traffic and might chain it over short but overloaded paths.

Figure 4.13 presents the end-to-end packet loss on links after throwing away the overloading parts. It can be observed that the slice limitation algorithms perform better than CSP and OdAASP, although the difference is not that large for higher loads. The cause of the fall-back effect at about 300 Mbps is that the number of network links applied in the chains increases allowing fewer relative loads and overloads on average.

SLN with a weight-aware policy performs better than any other method from the overload reduction and packet loss aspects. As the slice load limitation is considered on the links of the networking layer, this method takes more chains that can be mapped on a networking link. SLN service chaining is expected to be more adjusted to packet serving, since prioritization is performs on networking layer’s resources.
4.4.4 Summary

This section focuses on service function chaining methods, which support slices, and considers their packet level handling while calculating the appropriate service chain. I have proposed different policies for setting up the parameters of the SFC methods. The model behind the methods ignores the load and latency details or limitations of VNFs but considers link capacities and network loads coming from the different slices, which share the available resources according to the implemented queuing. It allows the systematic evaluation of QoS properties that can be experienced on the links or service requests. Result values are calculated with a packet-level model of network links extended by the overload reduction concept that shall handle overloads. The concept is not strictly coupled to the NFV and might apply to containerized functions.

The numerical results show that the proposed algorithms perform better from several QoS metric aspects than those missing the slice-aware property. From a set of results seems that the conservative policy works well when the load is moderate, while for higher loads, the weight-aware policy can perform better. Another result-set demonstrates the advantages of method SLN in the middle load range. However, the evaluation is not straightforward due to the complex dependencies among the different kinds of measures.
Proof of Concept Implementations

The networking world is now driven by a completely different set of business needs, such as the ability to fully use network links, high performance, scalability, flexibility, high availability, and simplicity in operation. This change of requirements needs to benefit from the numerous advantages of network automation. Network automation manages network resources and services through the use of programmable logic. It enables network operations to configure, scale, protect, and integrate network infrastructure and application services more quickly than manual configuration, scaling, protection, and integration handled by users [71]. To show the importance of avoiding overloads on networking links, I implement the proposed solutions in SDN and in cloud-based environments.

5.1 SDN-based Implementation

Some SFC methods proposed in the previous chapters of the dissertation are applied in a Master’s thesis [72]. The methods are implemented there as SDN-based solutions in OpenDayLight, a tool that offers network monitoring and configuring forwarding rules. The network to be controlled is created in the network virtualization tool Mininet. The work’s main contribution is to provide particular control scripts on the NorthBound Interface of the SDN controller for adjusting the forwarding tables in the SDN-based devices according to the decisions of SFC solutions. The proposed solutions show better performance for the cases with a potential risk of link overloading problems.

5.2 Cloud-based Implementation

This section presents a proof of concept implementation of the multilayer network model described in Section 3.3.2 based on an automated network environment. The aim is to analyze network loads and packet losses in simple scenarios using static and the SFC solutions presented in Section 3.3 that try to avoid network link overloads. The scenarios under the scope are not real-life but should demonstrate how the proposed SFC solutions work in real networks. However, the implementation of the multilayer model is robust and applies the Segment Routing technique for realizing service chains in the network.
5.2.1 Supporting Network Technologies

The Network Programmability concept has been steadily growing in the IT and networking communities. The YANG data modeling language is introduced in [73] and has become famous among networking vendors for various data modeling use cases. It can be used as a data modeling language for automation and reconfiguration. However, for such purposes, YANG data models need to be accompanied by NETCONF [74] or RESTCONF protocols [75]. These protocols enable the control and management of YANG data models, and also describe how to map a YANG specification to RPC or a RESTful interface running over HTTP.

Segment Routing (SR) is an architecture based on the source routing paradigm evolved from Multi-Protocol Label Switching (MPLS). SR seeks the right balance between distributed intelligence and centralized programmability by steering a packet through an ordered list of instructions. ISPs can deploy SR without extra investment or a need for new hardware to overcome the lack of flexibility, path control, and essential routing solutions to performance issues. SR enables effective TE solutions such as Resource Reservation Protocol with Traffic Engineering (RSVP-TE) while simplifying control plane operations. It operates over either an MPLS (SR-MPLS) or an IPv6 control plane. SR-MPLS encodes a path as a stack of labels inserted in the packet header by the ingress node. The SR path is composed of a succession of Segment Identifiers (SIDs) which can be associated with each router and link. They are globally unique and represent the shortest path to a router as determined by the IGP.

5.2.2 Concept Realization

Figure 5.1 shows the building blocks of the concept’s implementation and the steps of request handling. The most important block is the Controlling module, which works with service demand requirements and network information parameters. The other important element is Collecting module connected directly to the network infrastructure and responsible for the querying of load and routing information using RESTCONF on each router. This separation of the functional blocks is needed because our concept considers neither the SFC methods’ implementation on the links nor the pure, fully-centralized SDN approach. In step 1, RESTCONF APIs help to get the network’s information, such as the loads and capacities of links and the list of VNFs available in the Head-end and other routers. The information is forwarded to the Controlling module running on a separate server. This server executes a Python script realizing the SFC algorithms, and in step 2, the computed paths are posted to the appropriate routers using RESTCONF APIs.

According to the selected chain, a segment list is created, in which the segments realize the whole series of the functional links in the chain, i.e., not only the nodes with applied VNF instances shall be listed. Without this comprehensive information, the mapping of functional links on the networking layer cannot be adequately considered.

5.2.3 Network topology

The above-listed modules are implemented in a network set up over Amazon Web Service (AWS) cloud platform, which includes virtual routers and servers. The testbed is composed from four Cisco Cloud Services Routers (CSR 1000V) available on AWS marketplace with IOS XE 16.10.b software. The CSRs offer routing, security, and network man-
agement as cloud services, including the option of programmable configuration through \textit{RESTCONF}.

The traffic endpoints are \textit{Ubuntu 18.04 LTS Servers} where UDP traffic is generated using the \textit{iperf} tool, which can produce standardized input for performance measurements in networks. For collecting and organizing the information about routers on the IP network, I use Simple Network Management Protocol (SNMPv2c) in a separate server within 	extit{AWS}.

Since I focus on the loads and losses on networking links, real VNF capabilities are not implemented in separate servers or attached data centers. I assume the routers as VNF capable nodes since CSR 1000v is a virtual-form-factor router that delivers comprehensive WAN gateway and network service functions into virtual and cloud environments. If needed, a virtualized Linux-based environment can be used to implement VNFs even on a router, e.g., \textit{Guestshell}, which is designed to run custom Linux applications on Cisco devices.

\textbf{Figure 5.1:} Block diagram of the concept realization

\textbf{Figure 5.2:} Region and Availability Zone for the deployed instances
The implemented topology of the networking graph $G_N$ can be seen in Figure 5.2. There are four Virtual Private Clouds (VPCs): (VPC₁, VPC₂, VPC₃ and VPC₄) connected via VPC Peering. The network also contains connections between VPCs, which enable to transfer of the traffic using private IPv4 or IPv6 addresses within the CIDR range offered by AWS. Each VPC deploys an EC2 instance that hosts a single CSR. The links of $G_N$ can be realized in the cloud environment as generic GRE tunnels between the routers.

The GRE tunnels are advertised by the IS-IS routing protocol to be used as parts of the segments, which are the realizations of the functional links in $G_F$ graph assigned to slice $S_i$. Although the concept allows arbitrary mapping between the two layers, for the sake of simplicity, I use one-to-one mapping where each functional link matches one networking link. Thus, the calculated VNF-SFCs will be mapped directly on a series of networking links realized by GRE tunnels. As mentioned before, a segment route shall be configured for the chain and implemented as an MPLS-TE tunnel referred to as SR-LSP.

SR-LSP requires an SR policy configured on the Head-end router. In this implementation, I use an explicit path which accords to the SFC calculation. Allowing segment routing solves the problem of leading the traffic through the network by touching the given series of routers or other nodes representing VNF-capable nodes. The configuration is needed only in the Head-end node. Note that this solution also allows controlled network route loops in the SR-LSP.

In the observed scenario, the GRE tunnels’ capacities are configured to 10 Mbps. The reasons behind this setting are that the outbound data transfer pricing in AWS would be cheaper, and the UDP traffic generated by iperf limits the bandwidth for UDP clients to 1 Mbps by default. However, since the GRE tunnels are virtual links, the tunnel might normally perform without packet losses even if the throughput reaches or overrides the maximum capacity. In order to overcome this issue, the configuration of traffic policies is required. These policies allow us to control the maximum rate of traffic sent or received on an interface by applying multiple classes of service and are involved only to realize the virtual link’s capacity.

I use Segment Routing IPv4 (SR-IPv4) for the segments’ implementation because the available cloud environment suffers incompatibility issues with SRv6 of IPv6. SR-MPLS data plane must be enabled on all IPv4 interfaces in the IS-IS domain to allow (SR-IPv4). In order to direct the traffic, a static route entry with the correct segment route’s destination address shall be given to the head-end router.

Note that SR-MPLS with IPv4 addressing on Linux requires Kernel version 4.5 which is not supported on AWS and Segment Routing IPv6 (SRv6) can be implemented on the kernel version 4.15 or higher [76]. Moreover, the IPv6 data plane cannot be enabled in CSRs deployed in AWS [53], since the currently applied IOS XE software does not support SRv6 implementation.

### 5.2.4 Performance Evaluation

Figure 5.3 shows the network topology indicating only the network links that will be loaded in our experiments. According to the one-to-one mapping, segments (functional links) are created only between neighboring routers.

The traffic of a user demand instance in slice $S_i$ starting in $PC_{src}$ in subnet $LAN_{src}$ and ending in $PC_{dst}$ in subnet $LAN_{dst}$ takes the following path in the networking layer:

---

1. The current kernel version supported on AWS is 4.1.5
In the (edge) router $R_{src}$, which is connected to $LAN_{src}$, the traffic gets classified and assigned to the route configured via an MPLS-TE tunnel. The classification is not connected to a reservation process and is restricted to taking decisions only on the base of the destination IP address.

The dynamic routing is applied only for the networking links of $G_N$, which connect the routers, i.e., the GRE tunnels. The subnets of the PCs are not advertised. The route for the SR-LSP is built from these, taking the link costs into account. The SR-LSP ends in the (edge) router $R_{dst}$, which is connected to $LAN_{dst}$.

In router $R_{dst}$, the traffic has to be routed to the destination. In this case, the subnet of the destination $PC_{dst}$ is directly connected to the edge router.

Note that, in provider networks, the last step can be performed using Virtual Route Forwarding (VRF), a technique widely applied for creating virtual private networks (VPN) over the same infrastructure. However, slicing is not obliged to VPNs, and in general cases, there are no private networks that might use overlapping address ranges and need to be separated. Instead of the complex configuration of VRFs, we can use static routes to forward the packets to their destinations.

### 5.2.5 Evaluation Methods

Let me present the different experiment setups used for evaluating the performance of the applied SFC algorithms. I propose two scenarios, both for analyzing the effects of serving dynamically arriving, semi-permanent, i.e., never closing service demands. The semi-permanent behavior is modeled by stepwise elevating the traffic generation rate.

Figure 5.4 introduces *Scenario 1* with a single slice $Slice_0$ and one service in the slice that requires to pass VNFs $u_1$, $u_2$ and $u_3$ (grey, yellow and blue).

According to the VNF requirements of the service, there are two possible chains implemented as *SR-LSPs*. One is based on the reference solution CSP algorithm proposed in [51], considering to find the shortest path that satisfies the given SFC constraint. The other is applied exclusively by the OdAASP algorithm, which considers the network’s state and the mapping of functional links to find a short chain that should avoid overloads on network links. The *SR-LSPs* is organized as the following:

- *SR-LSP*$_1$ corresponds to the shortest path based CSP algorithm passing through network nodes $CSR_1$, $CSR_2$, $CSR_3$, $CSR_2$ and $CSR_3$.
- *SR-LSP*$_2$ corresponds to the OdAASP algorithm that tries to avoid overload on network links, and uses the path through nodes $CSR_1$, $CSR_4$, $CSR_1$, $CSR_2$ and $CSR_3$. 

---

**Figure 5.3:** Sample Network Topology
When the load is low, no overloads occur, and also OdAASP would use SR-LSP\textsubscript{1} since that is the shortest chain. However, with this scenario, I aim also to mimic the fallback option of OdAASP described in Section 3.3.5. When the method cannot avoid link overloads due to high traffic load, the SFC for a new traffic demand returns to use the shortest path chain solution, i.e., to the chain of CSP.

Figure 5.5 shows how the chain choice is driven by generating two different types of traffic and elevating the traffic load with the following phases:

- **Phase.I (CSP growing)** consists of generating a linearly growing UDP traffic load that takes the SR-LSP\textsubscript{1} path. This phase finishes when at least one of the network links would be overloaded, assuming further elevation of network load. It is expected that the link $B1$ gets overloaded sooner than others as the traffic passes twice on it.

- **Phase.II (CSP constant, OdAASP growing)** consists of generating a linearly growing UDP traffic load that takes the SR-LSP\textsubscript{2} path. On the other hand, modeling the unchanged SCs of earlier arrived demands, the client keeps generating a fixed amount of traffic, which is being sent on path SR-LSP\textsubscript{1}. This way, we avoid the overload of link $B1$ for a while. The phase finishes when at least one of the network links would be overloaded, assuming further elevation of network load.

- **Phase.III (CSP growing, OdAASP constant)** models the fallback option, i.e., the return to the shortest path SR-LSP\textsubscript{1} when the overload cannot be avoided.

Figure 5.6 introduces Scenario 2 that extends the previous case of VNF-SFC to two slices ($Slice_0$ and $Slice_1$). The demands in the second slice require passing VNFs $u_1$ and $u_3$ (grey and blue) and can only apply the SR-LSP path SR-LSP\textsubscript{3}.
As shown in Figure 5.7, the traffic generation from the second slice follows a more straightforward way. It takes three different rate levels for this slice in the three phases, while the phases for the first slice are determined similarly to Scenario 1. Note that the starting points of the phases here can differ from those implied in Scenario 1 since the total traffic of the two slices might overload the network links earlier.

5.2.6 Measurement Results

The aim of using the 0dAASP algorithm is to avoid network link overloads leading to a better QoS, e.g., suffering fewer packet losses on links. I analyze the load and packet loss results based on the Collecting module’s statistics in this proof of concept. Since these values are measured and not just calculated in a model, the loss result mixes the above-introduced overload reduction and packet loss metrics. It is worth mentioning that waiting times and end-to-end delays are not considered in the analysis because they are pretty hard to measure and interpret on the virtualized devices of the cloud-based environment.

Figure 5.8 and 5.9 present the values measured in Scenario 1. In the phase (0-25min), one can observe that the load is increasing on links $A$, $B_1$, $B_2$ and $C$. Particularly on link $B_1$, the increase is faster because the SR-LSP$_1$ path passes link $B_1$ twice. Therefore, after 25 min, there are packet losses on link $B_1$ as a consequence of reaching with the throughput the 10 Mbps link capacity, i.e., turning into overloads as shown in Figure 5.9.

Within the following period of time (25-45min), the load increases on links $D_1$ and $D_2$ as the Head-end router imposes a new MPLS label stack that corresponds to SR-LSP$_2$, which involves the links $D_1$ and $D_2$. Furthermore, it can be observed that the throughput stops increasing on link $C$ since SR-LSP$_2$ excludes it. Another interesting effect is the moderation of load and packet losses on link $B_1$. This moderation occurs because packet
drops increase on several other links leading to the decrease of the traffic that should arrive a second time at the link, which participates in a controlled loop. At 45min, the link A gets overloaded, and as a consequence, packet drops can be observed.

The effects in period (45–90min) are easier to understand: links A, B₁ and B₂ are fully loaded, cannot bring more traffic and drop packets. The throughput of the links C, D₁, and D₂ does not reach the links’ capacity during the entire experiment, and no packet loss happens on them.

Although the scenario focuses on measuring OdAASP, it is easy to see that using CSP the traffic would load only SR-LSP₁. This case induces that a linear increase of load with losses in the overload region would be observable on links A, B₁, B₂, and C during the whole measured period.

In the evaluation of Scenario 2, I only focus on the throughput and packet loss changes on the links. I aim to find similarities to the results of Scenario 1 because some links’ load is not affected when adding the SR-LSP₃ path. The differences are shown in Figure 5.10a and Figure 5.10b.

Figure 5.10a shows a linearly increasing load on the links A, B₁, B₂, C and D₁) in the period (0–30min), similarly to the previous experiment. Links C and D₁ are loaded
because the SR-LSP₃ path imposes the MPLS label of these links. However, the load on link D₁ is constant according to the traffic generation scheme of the second slice.

At (30min), the links A and B₁ start getting overloaded and consequently packets are being dropped as it can be observed in Figure 5.10b. Within the period (30-45min) of time, the load on links A and B₁ further increases causing them to drop packets at an increasing rate.

In the next period (45-90min) of time, the values of both load and loss for all links seem to be relatively high but stable. One can see particular intervals on both Figure 5.8 and Figure 5.10a, where the load of link B₁ exceeds its capacity. This overload implies states where the link itself consumes all the capacity allocated in the cloud, and packet losses occur, as can be followed in Figure 5.9 and Figure 5.10b too.

5.2.7 Summary

In this chapter, I have shown proof of the previously introduced concepts in a small network realization applying Segment Routing. This technique solves determining the packet path to follow from source to destination in real or virtualized networks applied for Service Chaining scenarios. I have presented two scenarios with multiple slices to observe the impact of slices on each other. The comparative results have shown the importance of the proposed solutions to avoid overloads in networking links, at least in the moderate network load regions.
Conclusion

The 5G technology has mainly focused on mobile broadband use cases, providing enhanced system capacity and offering higher data rates. However, future wireless networks should offer wireless access to anyone and anything. Thus, in the future, wireless access will go beyond humans and expand to serve any entity that may benefit from being connected (Internet of Things). For instance, Telemedicine is an essential tool for improving healthcare access both in remote rural and urban areas. With 5G, medical records containing high-resolution medical images and videos can be made available to physicians and medical professionals anytime and anywhere. Remote, real-time general physician and specialist consultations would also contribute to cost savings, convenience, and better and timelier medical outcomes. 5G will also propose new solutions for various new application scenarios. Besides, the development of innovative or wearable devices also motivates the realization of a telemedicine system.

6.1 Application of Results

I believe that my results may have a significant impact not just in the academic sector but also in the industry. I hope my results may contribute to the general understanding of avoiding overloads in Service Chains. I believe these results would impact network or cloud infrastructure providers to overcome the overload issues using the proposed methods. Moreover, network slicing and segment routing provide intelligent routing and traffic differentiation required to support this distributed architecture efficiently. My solutions may help use these technologies for better performance and guarantee Quality of Service.

Another direction of applying the proposed solutions can be in open standard cloud computing platforms such as OpenStack or OpenShift to build an NFV infrastructure. The combination between SDN and NFV on the OpenStack cloud platform could establish resource management for the proposed solutions, making packet forwarding more flexible and efficient.

6.2 Summary of New Scientific Results

This dissertation proposes a multilayer model composed of multiple layers based on the potential implementation of 5G core networks or any other NFV-enabled networks. The
dissertation focuses on studying and developing Service Chaining solutions based on creating service chains with given VNFs order and traffic requirements to avoid networking links overload with and without considering Network Slicing.

The dissertation identifies solutions for several problems related to the overload of networking links. To ensure the QoS, the evaluations for these solutions are based on the network link loads. From many points of view, the proposed solutions show better performance than the referenced service chaining method. The solutions preserve the network resources by avoiding an overload of network links. Some solutions will not only avoid the overloads but will help to reduce network congestion. The results achieved in the dissertation are summarized in two contribution groups. The author’s publications where the actual thesis groups were published are indicated in square brackets.

Thesis Groups I: Overloads in Service Chains

The main challenge of Service Function Chaining is to create a chain according to demand requirements as the ordered set of VNFs or bandwidth while adjusting the traffic to the networks’ capacities. From the traffic QoS point of view, the current load on the network links is crucial and should be considered at the chain’s selection. However, it is not always possible to avoid overloading network resources when the load is elevated. This group of contributions focuses on how overloads can be avoided and handled.

Thesis I.1: QoS Modeling and Analysis in 5G Backhaul Networks [C1]

I introduced a multilayer graph model supporting the Service Function Chaining problem. Unlike previous models, it separates the layer connecting the VNF-capable nodes (Functional Layer) and the network infrastructure (Network Layer). I applied the model for the 4G EPC and 5G Core network architectures and proposed a Cost-Based SFC method. The method can reduce the traffic losses on IP links by up to 40% compared to the method considering only the nodes’ VNF capabilities (Section 3.2).

Thesis I.2: Avoiding Link Overloads in Service Chains [J1]

I extended the multilayer model to take into account overloaded links not considered in previous models. I proposed ILP-based and heuristic SFC solutions that consider the ordered set of required VNFs, and avoid network link overloads if possible. I showed that in dense networks, the proposed solutions perform better than a reference solution from the literature in terms of link loads and traffic reduction due to overloads (Section 3.3).

Thesis I.3: Bandwidth-Aware Service Chaining [C2]

I proposed bandwidth-aware SFC solutions considering the link loads and the remaining capacity to select links with moderate load for the chains. I showed that these solutions perform better than a reference algorithm from the literature regarding link-level and end-to-end QoS metrics (Section 3.4).

Thesis Groups II: Slice-aware Service Chains

Network Slicing supports different services applying common or separated resources in the network. The services may differ in requirements, and their traffic is handled differently by the queueing services at the network nodes to ensure or improve service quality. This
thesis group focuses on the network slicing challenges and the ways of preserving resources for slices to ensure better QoS.

**Thesis II.1: Service Chains in Network Slicing [C4]** I extended the multilayer model by the option of multiplying the elements in the Functional Layer to support network slicing. I proposed SFC solutions applying slice load limitation on links to preserve network resources for other slices. I showed that these slice-aware methods cause lower link loads and overloads in the core segment of the network besides handling the slices’ traffic more adequately to the packet serving policies than a reference algorithm from the literature. (Section 4.3).

**Thesis II.2: QoS Impacts of Slice Traffic Limitation [J2]** I proposed a set of policies for setting the limitation parameters of the slice-aware SFC methods to adjust them with WFQ-based queueing systems. I extended the analysis of the link overload problem to support slices and showed that the proposed slice-aware solutions outperform other existing methods from QoS aspects (Section 4.4).

### 6.3 Future Research Direction

The work presented in previous chapters opens up several exciting research directions for better Quality of Service. Here, I highlight some interesting directions for continuing the research.

The proposed SFC solutions consider the network link loads and capacities and try to select chains to ensure high QoS but do not deal with prescribed QoS requirements. As a straight next step, the methods could be extended to consider such requirements directly applying reservation schemes.

I believe that an exciting aspect missing from the slice-aware SFC approaches is considering fairness among slices. Optimizing the SFC mapping separately for each slice or dynamically handling the requests may unfairly handle the slices. Further analysis from the QoS point of view and the extension of the proposed policies are needed to catch this issue. A similar direction is to focus on more complex scenarios improving the proposed solutions by considering different allocation mechanisms. This option is required on each network link since some of them might be used more by one slice and less by the others.

During my research, I have not focused on the problem of placing the VNF capabilities in the network nodes and the deployment of VNF instances. Several papers deal with these problems, considering the network traffic and service requirements as inputs. However, the solution based on static inputs might not perform flawlessly in the case of dynamically arriving demands and dynamic SFC decisions. The capabilities could be placed considering the properties of the applied dynamic SFC solutions, i.e., integrating solutions for multiple problems should be worked out. This integration can also be essential from the VNF instance placement problem point of view since the capabilities strongly influence the decisions there. TR-521
List of Figures

1.1 An example of 5G use cases ........................................ 2
1.2 Traditional vs SDN Networks ........................................ 3
1.3 Traditional vs NFV Approaches ..................................... 4
1.4 An example of Network Service ..................................... 5
1.5 ETSI NFV architecture ............................................... 6
1.6 An example of realizing an SFC .................................... 6
1.7 SFC and service chains for different network services .......... 7
1.8 Single vs multi-tenancy .............................................. 8
1.9 Substance of the matter ............................................. 10
1.10 Structure of the dissertation’s contributions .................... 11

2.1 The multilayer model ................................................. 14
2.2 Networking and Functional layers’ representation .............. 15
2.3 Functional and networking layers ................................ 15
2.4 Path $P_r$ in Functional Layer $L_F$ .............................. 16
2.5 Path in the functional and networking layers .................... 16
2.6 Example for the paths of two requests ........................... 17
2.7 Independency of functional and networking layers .............. 18
2.8 Example for inevitable loop in SFC ............................... 18
2.9 Strict priority queueing model for high and low priority queues 22
2.10 Queueing model for WFQ queues ................................. 23
2.11 Queueing model for $LLQ$ ........................................ 24
2.12 Flow of evaluation process ....................................... 27
2.13 BT Europe topology and VNF placement ......................... 28
2.14 Algerian topology and VNF placement .......................... 29

3.1 Basic vEPC Architecture ........................................... 33
3.2 Model overview for 4G ............................................. 34
3.3 Basic 5G Core Architecture ....................................... 35
3.4 Model overview for 5G ............................................. 35
3.1 4G/5G Comparison in deployment ............................................ 36
3.6 The average load of networking links in function of number of VNFs .... 38
3.7 The average load of networking links ........................................... 38
3.8 Number of overloaded core and edge links ................................. 39
3.9 Overload reduction for New Services ......................................... 39
3.10 Average queue level for delay and loss ..................................... 40
3.11 Average end-to-end delays ..................................................... 40
3.12 Example for overload in VNF-SFC with shortest path .................... 44
3.13 An example of $G_F$ transformed graph .................................... 46
3.14 Avoiding overloads using the extension of the Dijkstra algorithm ....... 48
3.15 Simple topology for analyzing OdA algorithms ............................ 50
3.16 Length of networking paths and number of overloaded links in Algeria topology .............................................................. 51
3.17 Relative load of core and edge links in Algeria topology ................. 52
3.18 Relative load and number of overloaded links in BT Europe topology .... 53
3.19 Length and number of networking links in BT Europe topology ......... 54
3.20 Number of overloaded networking links ..................................... 59
3.21 Average traffic-level delay for strict class .................................. 60
3.22 Average traffic-level delay for class WFQ-0.3 ............................. 60
3.23 Average traffic overload reduction for WFQ-0.3 class ..................... 61
4.1 The layered graph model of an example with two slices .................... 65
4.2 Example network supporting slices .......................................... 68
4.3 The topology of the functional and networking layers ..................... 70
4.4 The paths in the functional layer ............................................ 71
4.5 The paths in the networking layer ............................................ 71
4.6 Networking links evaluation for Algeria topology .......................... 72
4.7 Networking path analysis for Algeria topology ............................. 73
4.8 Link capacity sharing in different load cases ................................ 77
4.9 Toy topology with service requests .......................................... 79
4.10 Average traffic-level loss for slice $S_1$ .................................... 80
4.11 Average traffic-level waiting time for slice $S_1$ .......................... 80
4.12 Average delay for the Algeria topology ...................................... 82
4.13 Average overload reduction and packet loss for Algeria topology ....... 82
5.1 Block diagram of the concept realization .................................... 86
5.2 Region and Availability Zone for the deployed instances ................. 86
5.3 Sample Network Topology .................................................... 88
5.4 Creating chains in Scenario 1 ........................................ 89
5.5 Traffic generation in Scenario 1 ................................. 89
5.6 Creating chains in Scenario 2 ....................................... 90
5.7 Traffic generation in Scenario 2 ..................................... 90
5.8 Scenario 1: network link load ..................................... 91
5.9 Scenario 1: network link packet loss ............................ 91
5.10 Scenario 2 .......................................................... 92
## List of Tables

<table>
<thead>
<tr>
<th>Table</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.1</td>
<td>Traffic types characteristics of BT Europe topology</td>
<td>28</td>
</tr>
<tr>
<td>2.2</td>
<td>Traffic types characteristics of Algeria topology</td>
<td>29</td>
</tr>
<tr>
<td>3.1</td>
<td>Traffic types characteristics</td>
<td>37</td>
</tr>
<tr>
<td>3.2</td>
<td>Result comparison for SFC algorithms</td>
<td>50</td>
</tr>
<tr>
<td>3.3</td>
<td>Traffic types characteristics</td>
<td>58</td>
</tr>
<tr>
<td>4.1</td>
<td>Service chains and load results in the example for slicing</td>
<td>68</td>
</tr>
<tr>
<td>4.2</td>
<td>Slice limitation values for SLF and SLN</td>
<td>70</td>
</tr>
<tr>
<td>4.3</td>
<td>Traffic types characteristics</td>
<td>72</td>
</tr>
<tr>
<td>4.4</td>
<td>Traffic types characteristics</td>
<td>79</td>
</tr>
<tr>
<td>4.5</td>
<td>Traffic types characteristics</td>
<td>81</td>
</tr>
</tbody>
</table>
Publications

Journal Publications (10 pts)


(WoS, IF=3.24, Q1 ) [6/1 = 6]


(WoS, Q3) [4/1 = 4]

Conference Publications (12 pts)


(IEEE) [3/1 = 3]

C2  Z. Zsóka, **K. Mebarkia** "Layered Solutions for Dynamic Service Chaining” 22nd Conference on Innovation in Clouds, Internet and Networks and Workshops (ICIN), 2019.

(IEEE) [3/1 = 3]


(IEEE, Best Paper Award) [3/1 = 3]

C4  Z. Zsóka, **K. Mebarkia** "Slice-aware Service Chains” 24th Conference on Innovation in Clouds, Internet and Networks (ICIN), 2021.

(IEEE) [3/1 = 3]


