VN Embedding in SDN-based Metro Optical Network for Multimedia Services

Faisal Ameen Zaman

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

Masters Of Applied Science

In Electrical and Computer Engineering

Ottawa-Carleton Institute for Electrical and Computer Engineering

University of Ottawa

Ottawa, Canada

January 2017

© Faisal Ameen Zaman, Ottawa, Canada, 2017

Abstract

Currently a growing number of users depend on the Edge Cloud Computing Paradigm in a Metro Optical Network (MON). This has led to increased competition among the Cloud Service Providers (CPs) to supply incentives for the user through guaranteed Quality of Service (QoS). If the CP fails to guarantee the QoS for the accepted request, then the user will move to another CP. Making an informed decision dynamically in such a sensitive situation demands that the CP knows the user's application requirements. The Software Defined Networking (SDN) paradigm enabled the CP to achieve such desired requirement. Therefore, a framework called Virtual Network Embedding on SDN-based Metro Optical Network (VNE-MON) is proposed in this Thesis. The use of SDN paradigm in the framework guarantees profit to the CP as well as QoS to the user.

The design concept of the SDN control plane, raises concerns regarding its scalability, reliability and performance compared to a traditionally distributed network. To justify concerns regarding the SDN, the performance of VNE-MON and its possible dependency on the controller location is investigated. Several strategies are proposed and formulated using Integer Linear Programming to determine the controller location in a MON. Performance results from the assessment of the VNE-MON illustrates that it is more stable compare to GMPLS-based network. It is evident that the controller location's attributes have a significant effect on the efficacy of the accepted VN request.

Acknowledgement

This thesis would have remained a dream, had it not been for my Professor Ahmed Karmouch. Whose unflinching confidence in my abilities metamorphosed my personality. "Good, Better, Best. Never let it rest. Till your good is better and your better is best" - St. Jerome. The above quotation is the hallmark of my Professor's work ethics. He always insisted to bring out the best, with a great sense of accomplishment. I would like to express my deepest gratitude to Professor Ahmed Karmouch.

I would like to express deepest appreciation and thankfulness to Dr. Abdallah Jarray. His attitude and genius continuously and convincingly through out the research work has enabled me to present this thesis. He inspired, guided and motivated me at all times of research work. I would not have imagined a better adviser and mentor.

My sincere thanks to Heli Amarasinghe, Venkatraman Balasubramanian and Wijay Ekanayake and other Lab colleagues, without their precious support, it would have not been possible to write this thesis. I am deeply indebted to my brother Mohammad Farooq Zaman for his moral support, encouragement and guidance. He has been a pillar of strength to me at all times. My sincere thanks to Hisham Veeran, Mudasser Noor, Yasir Noor, Fazal Noor, Mohammed Riyaz and Niranjan for their moral support.

My thanks to University of Ottawa for giving me access the resources to carry out my research and write this thesis. I also thank my many friends for their moral support and guidance. I also thank all those who directly or indirectly helped me complete this challenging task.

"As we express our gratitude, we must never forget that the highest appreciation is not to utter words, but to live by them", - John F Kennedy. Sincere thanks to my parents for their unconditional love and support.

Dedicated To Mom and Dad

List Of Acronyms

Acronym	Expansion
API	Application Programming Interface
CG	Column Generation
CG-ILP	Column Generation - Integer Linear Programming
CP	Cloud Service Provider
CPU	Central Processing Unit
CSP	Carrier Service Provider
DC	Datacenter
GLASS	GMPLS Light Agile Simulator
GMPLS	Generalized Multi Protocol Label Switching
GPU	Graphical Processing Unit
IaaS	Infrastructure as a Service
ICMP	Internet Control Message Protocol
IG	Information Graph
ILP	Integer Linear Programming
LDP	Lable Distribution Protocol
LINC -OE	Link Is Not Closed - Optical Extension
MEC	Media Edge Cloud
MON	Metro Optical Network
NP	Non- deterministic Polynomial

Acronym	Expansion
NV	Network Virtualization
O-E-O	Optical Electrical Optical
ONOS	Open Network Operating System
PaaS	Platform as a Service
QoE	Quality of Experience
QoS	Quality of Service
RAM	Random access memory
REST	Representational State Transfer
ROADM	Reconfigurable Add Drop Multiplexer
RSVP	Reservation Protocol
RWA	Routing and Wavelength Assignment
SaaS	Software as a Service
SDN	Software Defined Networking
SLA	Service Level Agreement
SOA	Service Oriented Architecture
SOA	Service Oriented Architecture
SSF	Scalable Simulator Framework
VN	Virtual Network
VNE-MON	Virtual Network Embedding on SDN-based Metro Optical Network
WAN	Wide Area Network

List of Figures

List of Tables

Contents

1 Introduction

This chapter provides a broad outline of the Thesis including an explanation of the motivation and the main objectives of the research. As well, a synopsis of the structure of the entire thesis is presented.

1.1 Motivation

Traditionally, Metro Optical Network (MON) was used to aid Wide Area Network (WAN) by aggregating the traffic from Access Networks. Recently, the trend in user-generated traffic has shown a tremendous growth towards multimedia [1]. Due to this traffic pattern change, Metro Network is required to be more than just aggregating middleware. It requires MON to be agile and support dynamicity. These services have a stringent requirement for resources [2]. To satisfy the insatiable demand for multimedia services, and to create a co-operative environment called multi-tenancy concept, Network Virtualization (NV) was introduced [3] - [5]. Network Virtualization was seen as an ultimate solution to solving the so-called *Internet Ossification* [7]. However, due to the lack of agility, and complete interaction between the application requirement and the network requirement, the advantages of Network Virtualization could not be completely utilized.

One of the major challenges associated with virtual network resource management is to dynamically and optimally embed the incoming VN requests. In a traditional system, the embedding was performed in a conventional data center (DCs) with prior intimation. Due to rigidity in deployment, it was not favorable to the user. To circumvent and satisfy the request and demands, a new class of cloud DC architecture called Edge Data Center

was introduced [8]. Edge DCs are deployed in large numbers in a Metro Network and they are installed near the user location. The concept of small distributed Edge DCs captured the attention of service providers, resulting in many innovative and efficient methods for delivering multimedia services to the user. Application Service Providers like Netflix, and Facebook rented these DCs to manage user data. Both the infrastructure service providers as well as the application service providers benefited from the concept of distributed Edge DCs [10, 39]. Until recently, IP technology was the most favorable form to connect DCs.

The cost incurred by the use of packet switching to serve dynamic allocation of resources to the incoming VN request, went beyond the practical limits. To overcome this challenge, the Infrastructure Service Provider shifted the focus towards Optical Network. The cost of switching associated with optical switching was very minimal, and the amount of traffic that could be transported using it was many times greater than packet switching [12, 42]. Considerable research has been directed towards photonics which has resulted in next generation Reconfiguration Add Drop Multiplexer (ROADMs) [13]. ROADMs are highly flexible optical devices which support Colourless Dimensionless Contentionless (CDC) feature. They enable a high degree of Dense Wavelength Division Multiplexing. Even though Metro Optical Network Service has all the above mentioned desirable features needed as a Carrier Service Provider (CSP), it still cannot be utilized dynamically. This is due to the rigidity of the switches and manual operations involved in the configuration of devices.

A dynamic resource allocation with an infrastructure consisting of hundreds of DCs and thousands of switches requires a system that can be agile, smart and autonomous. Many solutions were proposed to achieve a dynamic network control plane for the optical network. One such example which captured the attention of the infrastructure provider in the recent past was Generalized Multi-Protocol Label Switching (GMPLS) [27]. The proprietary and distributed nature of GMPLS was one of the main reasons for infrastructure providers to refer back to traditionally deployed architecture.

Recently, Software Defined Networking has been elevated to bring dynamicity and flexibility to a network. SDN consists of placing the set of network functionality associated with control logic in a centralized manner. Moving the control logic from all the switches to a centralized platform increased the visibility and coordination among different networks. It eliminated some of the traditionally faced issues with packet switching like irregular and unpredictable convergence time [28, 36].

The confluence of SDN and Next Generation DWDM Optical Network created the perfect opportunity for CSP to automate resource management for the VN requests. SDN's global view expedited the process of making an informed decision on embedding the incoming VN requests for the CSP. It also facilitated the rapid deployment of new and innovative models for VN embedding without modifying the substrate network. Although many types of research have been proposed for the SDN-based network to embed the incoming VN request efficiently, they fail to consider design principles of the SDN-based network. Some of the design considerations for the SDN-based network include controller performance and its location. Failure to address these issues will lead to inefficiencies in the network.

The above research trends and studies provided motivation to examine VN Embedding in SDN-based MON. The usage of SDN paradigm to perform VN embedding is manifested in this Thesis. The proposed architecture considers the limited CPU and resource availability in an SDN controller while performing VN embedding. The controller location problem presented in research articles and literature contributed motivation to address the study of *placement influence* on the accepted VN request [51] - [58] .

1.2 Thesis Objective

The main aim of the present Thesis is to propose a framework for Virtual Network Embedding for Edge DC in a Metro Optical Network. To guarantee the QoS for the VN requests and the efficient utilization of network, requires a control plane which is

agile, autonomous and capable of providing real time substrate information. To achieve such a feature SDN is used. Use of SDN requires careful consideration of the challenges associated with it. These include: Limited CPU, and the location attributes with respect to data plane. To fulfil the main purpose of this Thesis, the following investigations and design considerations were carried out.

A. **Guaranteeing QoS for Multimedia Services through Virtualization:** In a traditional virtualization context, a client requesting resource will not surpass the requirement. Presently, the use of the multimedia applications by users, their requirement changes dynamically. Multimedia requests have a stringent requirement for both networks as well as the DC resources. For efficient working of VNs, complete interaction of DC resources and the available network is required. In the present Thesis, the stringent requirement of the multimedia request is taken into consideration while formulating the VN embedding model. Media-aware cloud architecture is considered to circumvent the aforementioned challenge.

B. **Metro Optical Network:** It is a well-known fact that packet switching cannot cope with the increasing multimedia services. It is predicted that by 2020, 92% of the Internet traffic will be formed by multimedia [2]. To address these constraints, Optical Architecture was adapted. As a result of the revolution in the field of photonics like Dense Wavelength Division Multiplexing (DWDM) and ROADMs, it is possible to configure optical devices dynamically. Optical Network has its challenges compared to packet switching [42]. Thus it is not feasible to use solutions of packet switching to the optical network. The dynamic nature and the minimization of Optical-Electrical-Optical (O-E-O) conversion are taken into consideration.

C. **To Embed VN requests dynamically:** It has always been a concern on how to faster the provisioning of resources to the VN requests. To enable dynamic allocation of resource, we need an approach which is faster and gives optimal or near optimal

solution in the least amount of time. To propose an approach for dynamic allocation of resources firstly, an investigation on available approaches is carried out. Secondly, based on the considered scenario, a novel approach for embedding VN requests into Edge DC interconnected by MON using Integer Linear Programming and Column Generation is proposed.

D. **SDN-Based Network:** The complexities associated with legacy control plane focused the attention on SDN-based architecture [36]. Numerous applications were introduced with SDN. Routing, automated cloud orchestration, resource allocation, and load balancing to name a few [37, 38]. SDN-based architecture was considered to leverage its centralized nature and the global view in providing efficient resource allocation. Although SDN brings flexibility to the network, if not designed with a focus on the nature of SDN-based network, it will lead to the sub-optimal performance. Different challenges associated with the SDN-based network are considered in the present proposal. Two major constraints of SDN-based network considered in this proposal are: controller resource and the effect its location has on the performance of the accepted VN requests.

1.3 Thesis Contribution

The contributions of the present Thesis are summarized in the following paragraphs.

To efficiently utilize the network and the available resources in the Edge DC in a metro network, Network Virtualization was adapted. To solve the NP-hard problem of Virtual Network embedding on the substrate network, and to achieve dynamicity, a combination of Column Generation and Integer Linear Programming technique was followed. The VN embedding was solved in two phases. The first phase used Integer Linear Programming (ILP). ILP was concerned with allocating DC resources to the VN request. Whereas, the second phase used Column Generation and was responsible for light path embedding.

To interconnect these two modules, Software Defined Networking principles were

used. SDN facilitated the application service provider by providing global view of the network. This ensures a guaranteed QoS to the application service providers and it also enabled faster provisions of resources to VN requests. As a result, the advantages of virtualization were elevated. In addition, the SDN framework was adapted to include the features required for Metro Optical Network. In other words, SDN should support Dense Wavelength Division Multiplexing (DWDM). The proposed framework is termed as Virtual Network Embedding in SDN-based Metro Optical Network.

The use of SDN in the framework raised concerns regarding its scalability, and reliability with respect to the traditionally distributed network like GMPLS. To verify the advantages of inclusion of SDN, its performance is compared with GMPLS-based network. Furthermore, several optimal techniques to determine the optimal controller location for the SDN controller were proposed. The controller location problem is formulated using ILP technique and VNE-MON is accessed at these pre-determined locations.

1.4 Thesis Organization

This chapter serves as an introduction to the entire thesis. A brief synopsis of the remaining chapters follows.

Chapter 2 deals with the review of all the background and related proposals. It gives the structural differences between the traditional cloud DC and Edge DC. The architecture of SDN and its use for Optical Network is explained. The logic behind the use of SDN to manipulate the Edge DC resources and Optical network is included in the scope of this chapter.

Chapter 3 has a comprehensive focus on the whole architecture proposed for VN embedding on MON. The chapter presents an evaluation of suitable Optimization techniques and various constraints used for Resource Allocation of a VN request. Finally, mathematical formulation of a two-phase Integer Linear Programming and Column Generation technique is presented.

In Chapter 4, an evaluation dealing with concerns related to the design of the SDN-based metro network is performed. One such enigma associated with the design of the SDN-based network is the controller location for a small network. Different approaches are reviewed, and several strategies are proposed to solve the enigma of controller location. The scope of this chapter is to access VNE-MON performance at different controller locations.

Chapter 5, an evaluation of the concept proof using an in-house built simulator and standard emulators is presented. Different metrics used to gauge the proposed model and strategies considered for solving the Mathematical model are shown. An in-depth reasoning of the findings obtained from the evaluation for each scenario is discussed.

Chapter 6 concludes the Thesis with possible areas for future work. Various considered avenues given for future work are briefly highlighted in this chapter.

2 Background and Related Works

2.1 Objective

The purpose of Chapter 2 is to present technologies and standards relevant to the proposal of the current Thesis. The chapter begins with a look at the concept of traditional cloud and its evolution towards edge cloud. An investigation is conducted on the advantages of edge cloud computing and the adaptability for multimedia services as well as different proposed architecture for Cloud Service Provider. Further, an investigation is carried out on Network Virtualization's ability to share substrate resources with multiple users and to solve the so called *'Internet Ossification'*. Attention is also given to the use of Network Virtualization (NV) with Edge DC to host multiple users.

In addition, focus is directed to a background study on Software Defined Networking, its architecture and complementary protocols which aid in easy access to network elements. Finally, different 'use-cases' of SDN, specifically for Optical Network and VN embedding problem are reviewed.

2.2 Edge Datacenter

2.2.1 Traditional Datacenter and its Challenges

DCs has come a long way from its first implementation. They were privately owned by companies to share resources, files, and applications among employees. With the increase of user's reliance on the DC, Cloud Data center (CDC) was introduced by the

Cloud Infrastructure Provider. These CDCs were centrally placed, and users could rent resources from them *'as-a-service'*. Although Cloud Computing and Service Oriented models are different, they functioned interdependently. Cloud Computing deals with virtualizing the resource, so as to create a view of the infinite resource pool. Service Oriented Architecture (SOA) uses a cloud computing paradigm to deliver services by different avenues to the end user (Eg. WEB) [14].

Figure 2.1 shows SOA models employed in Cloud Computing. A cloud provider can be any agent or the owner of a cloud infrastructure. Users can request services like platform, infrastructure or software from the cloud provider. Once the cloud provider receives the request, it creates a profile for the user based on its requested service. The types of SOA prominently in use are:

Infrastructure as a Service (IaaS): It deals with providing virtual substrate resources like CPUs, Disk Drives, Storage and GPUs over the Internet. IaaS providers are highly scalable that adjust the resources based on demand. One of the main characteristics of the IaaS model is to pay only for services used. It reduces the Capital expenditure (CAPEX) of the client. Hence, it is a favorable technology for the users who have temporal needs. To handle a huge amount of requests, the IaaS service provider need to support automated allocation and maintenance, dynamic scaling, and maintain QoS.

Platform As A Service (PaaS): It is one level higher than IaaS in the SOA hierarchy. PaaS delivers Platform as an application over the internet. The users along with substrate resources also request platforms like databases and Operating systems. In addition, PaaS eliminates the inconvenience of resource maintainability, for the users. Users access PaaS through a web browser, and since user depends on the service provider for the platform, development will be affected by vendor lock.

Software As A Service (SaaS): It deals with renting software through the web or the internet. A user can request different software like Online video editing and players, IDEs, and Text editors. Traditionally SaaS was not feasible with the centralized architecture of cloud, but the introduction of distributed cloud facilitated the reduction of response time, thus enabling real-time interaction between them.

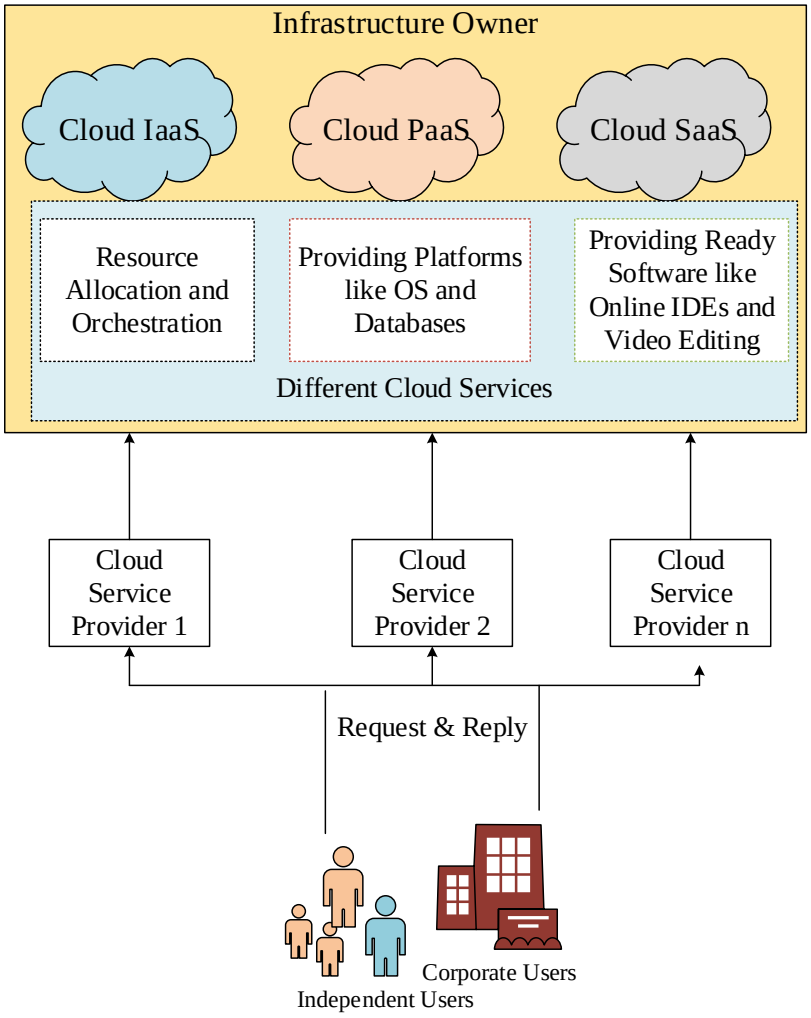

Figure 2.1: User Interaction for Services of Cloud Computing through standard SOA [14]

Limitations of the Traditional Cloud Architecture As mentioned earlier, SOA described was used with Centralized Cloud Computing. The use of SOA with Traditional Cloud architecture was only efficient for non-multimedia traffic and non-real time communication. Every time a user requested a computing resource, it was retrieved

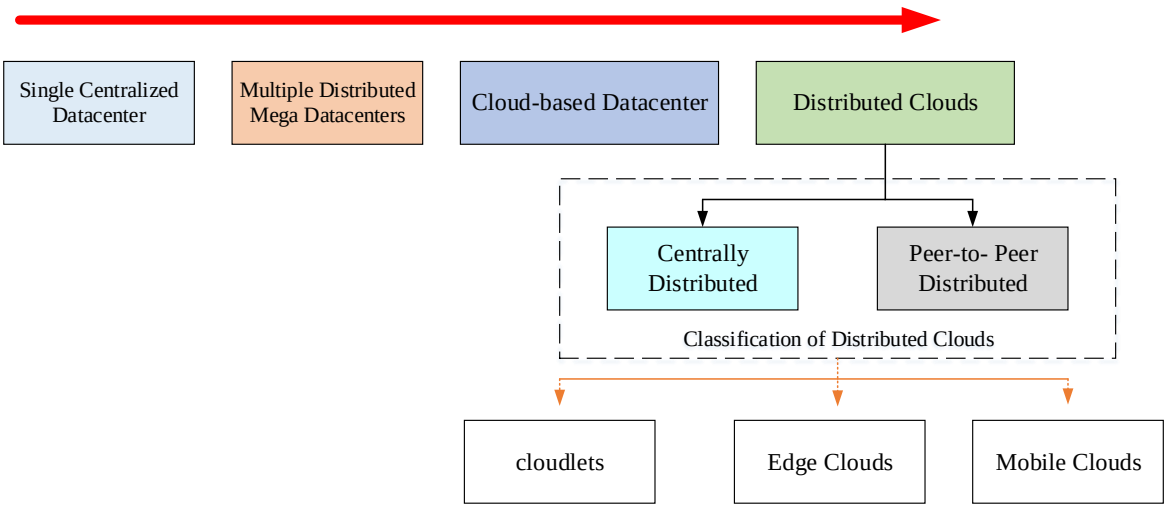

Figure 2.2: Evolution of Distributed Cloud DC

from a central location. This led to maximum delays and lacked Quality of Experience (QoE), resulting in inefficient utilization of the network. With the increase of users moving toward Cloud Computing, and the introduction of Web2.0 applications like video streaming, Netflix, and other multimedia services, the centralized Cloud system lacked what was needed the most, better QoS.

Several proposals were put forth to circumvent the limitations of centralized cloud architecture. The timeline for the evolution of different architecture is shown in Figure 2.2. When numerous small scale information industries started to invest money in renting DCs, for their computing needs, the cloud DC providers expanded their cloud infrastructure by building multiple clouds around the globe. For a given user the content was allocated in a single DC. To accommodate the multimedia application requirement, the concept of smaller DCs near to the user in a distributed fashion was introduced by cloud service providers.

The major types of distributed architecture are centrally distributed and peer - to - peer (p-p) distributed. Figure 2.3 and Figure 2.4 represent the same. A centralize distributed cloud DC synchronizes its content from a large central DC placed in a remote location. Moreover, the DCs are independent of each other; in other words, the content

stored in one DC is not directly shared with other DCs. Instead, DC will obtain content from the central master DC. Whereas in P-P distributed DC, the data is synched among proximal DCs using the principle of distributed consensus. In a few cases, if a data center does not have a content requested by a user, it checks upon the nearest DC rather than communicating with the remotely placed larger centralized DC. The advantage of such architecture is that it reduces the communication with the large centralized DC, but the disadvantage is, it will lead to increased inter-DC communication. The application of the choice of architecture for a given Metro network purely depends on the network's requirement.

To satisfy the stringent requirements of multimedia service, a novel concept called Multimedia aware Edge Cloud (or Media Edge Cloud) was introduced [8,16]. This cloud architecture enhances the Quality of Experience (QoE), and QoS provided to the end user. Various methods used to process the multimedia in an Edge cloud are stated in the subsequent section as proposed by [8].

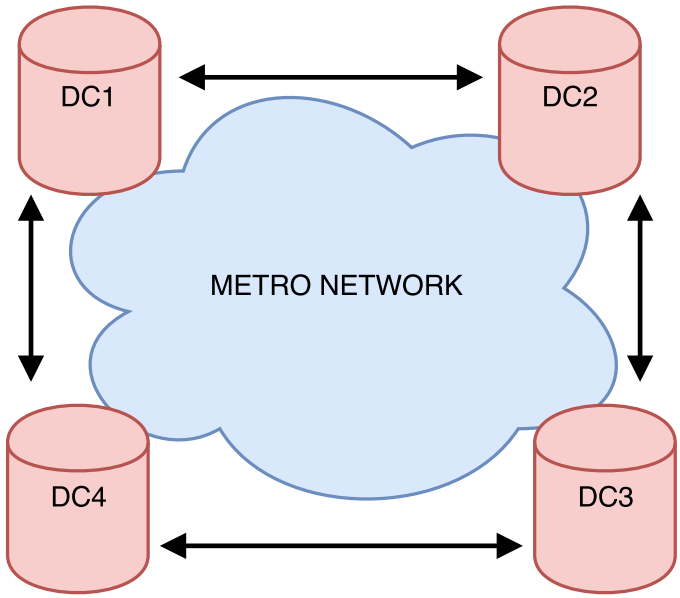

Figure 2.3: Peer- Peer Distributed Edge DCs [16]

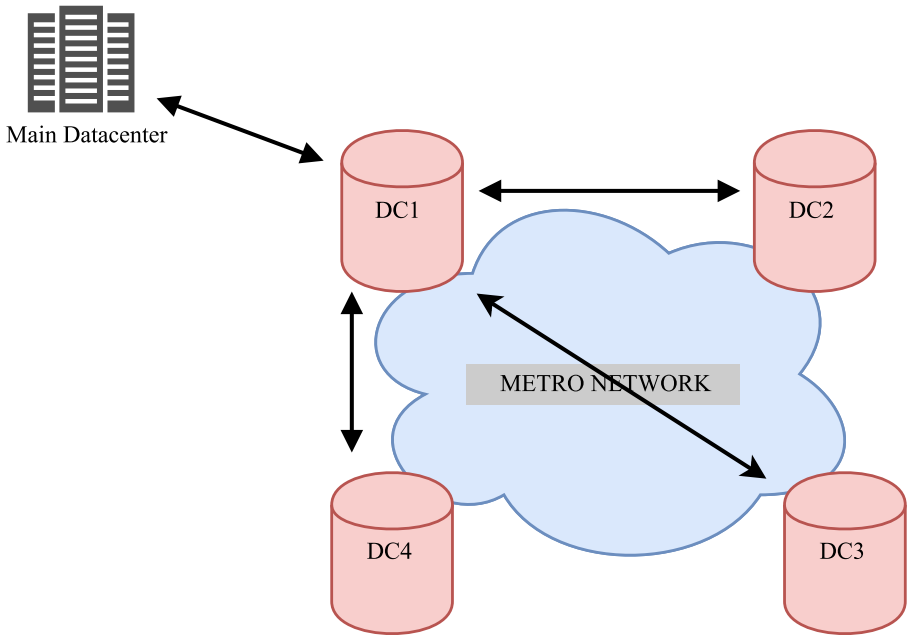

Figure 2.4: Centralized Distributed Edge DCs [16]

2.2.2 Edge Datacenter Architecture

Edge Computing, also referred to as cloudlet computing or fog computing, is a paradigm where the computing resources needed for processing the task are brought closer to the user [8]. The differences between the traditional cloud and Edge Cloud applications are shown in Table 2.1. Some of the motivational advantages for moving to Edge Cloud computing from the Traditional Cloud are presented in [17]. As highlighted, they are as follows:

- **Improving the QoS for the End User by Reducing Latency and Faster Provisions of Resources** The response time for a signal to travel from the DC at Berkeley to Canberra takes approximately 175 ms. This is beyond the limits of the QoS requirement for multimedia. To surmount the limitation of not being able to satisfy the QoS is one of the reasons for the Cloud Service Provider to move to Edge Cloud computing.

- **Dealing with Traffic Burst:** With the emergence of new classes of multimedia

Table 2.1: Comparison of Application in traditional Cloud and Edge Cloud

Traditonal Applications on Cloud	**WEB 2.0 Based Application**
1. Non Real Time Application	1. Mostly Real time Applications
2. Database Oriented	2. Continuous Computing
3. Less Resource Intense Task	3. Heavy Resource Intense Task
4. Sporadic and Bursts Transactions	4. Heavy Transactions all the time

applications like Google Plus, various media-sharing and streaming applications like Netflix, the amount of generated traffic is enormous. The traffic generated is characterized by occasional traffic bursts. As a result, it became imperative to move towards a Distributed Cloud Architecture to meet the resource requirements for these applications. Edge Cloud architecture not only ensures data availability but also guarantees the redundancy of data.

- **Innovative Computational Methods**: Several methods were proposed in order to place the users data near to the users' locations. Research on issues related to which data should be stored on which DC are well investigated [18, 19].

Other motivations like these Edge DCs consume less energy than the massive central DC. The Edge DC paradigm will save computation energy for the end user. [8] proposed an architectural design for satisfying the stringent demand of multimedia services. The computing of multimedia requests at the edge cloud has a slightly different requirement. These requirements are highlighted and different proposals made for Multimedia Cloud Computing are discussed in the subsequent sections.

2.2.3 Edge Datacenter for Multimedia Services

Multimedia Computing at an edge cloud possesses certain challenges. As pointed out by authors in [20] significant challenges are:

Heterogeneity of Services: The ever increasing various multimedia services like Voice over IP (VoIP), and Video Streaming apps pose new challenges to Service providers. Each Application has a unique requirement for the resources at the cloud. Overlooking this application requirement might result in the degraded working of an application. Heterogeneity also includes on the number of users and the type of end devices.

Distinct QoS Requirement: Cloud Infrastructure should seamlessly bear different QoS classes. This requirement results from the heterogeneity of application it needs to support.

Different Network technology: The clouds are kept at the last mile, closer to the user in a metro network. Metro networking is composed of different switching technologies at the Access Network End. It is important that the Edge Cloud architecture be designed by considering different switching technologies and its capability of being scalable.

Although many proposals have been made for providing QoS to an Edge Cloud, the proposal focusing on guaranteeing QoS for multimedia is [8]. The authors proposed two architectures called Media Cloud and Cloud Media. A brief summary of Media Cloud and Cloud media follows.

Media Cloud

Media cloud is responsible for providing required QoS to the request it is processing. Media Cloud maintains QoS by reserving the raw resources, distributing the processing of tasks and selecting the clouds close to the user. The architecture describing the Multimedia-aware Cloud or Media Cloud is shown in Figure 2.5. It is the responsibility of the service provider to ensure that the cloud selected for processing can satisfy the request's QoS. Media Cloud makes use of Edge Clouds to provide QoS requirement like latency, processing power and accessibility to the data ubiquitously.

Some features an edge cloud needs to pose in order to be called a Media Edge Cloud

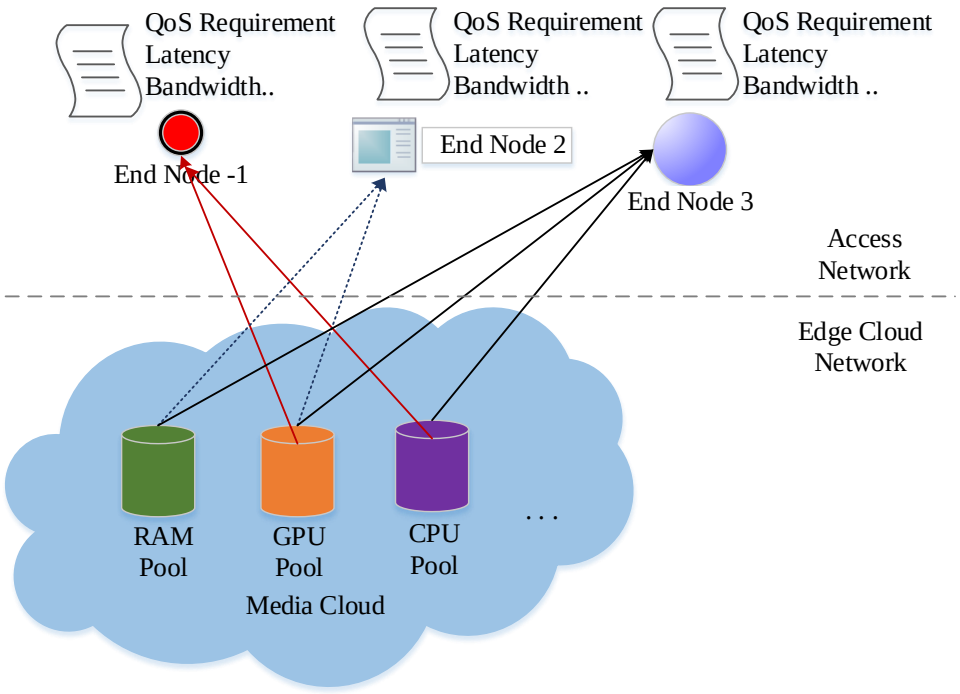

Figure 2.5: Multimedia Aware Cloud Architecture for Edge Cloud [8].

(MEC) are: Dynamic QoS provisioning for various services related to multimedia, support for Parallel Processing, and adaptation of network and DC resources based on heterogeneity of Devices.

Cloud Media

Cloud Media is a Cloud Aware Multimedia Application design. It is essential that the designing of this application utilize the distributed nature of cloud efficiently. Disregarding the limitations of cloud media while designing the application will lead to inefficient performance. Figure 2.6 shows the cloud media architecture, as seen by the application designer. The end nodes in cloud media need to be aware of the type of clouds they are dealing with. Whereas in the case of Media Cloud, it is the other way around, the end nodes do not need to be aware of the type of cloud they are dealing with.

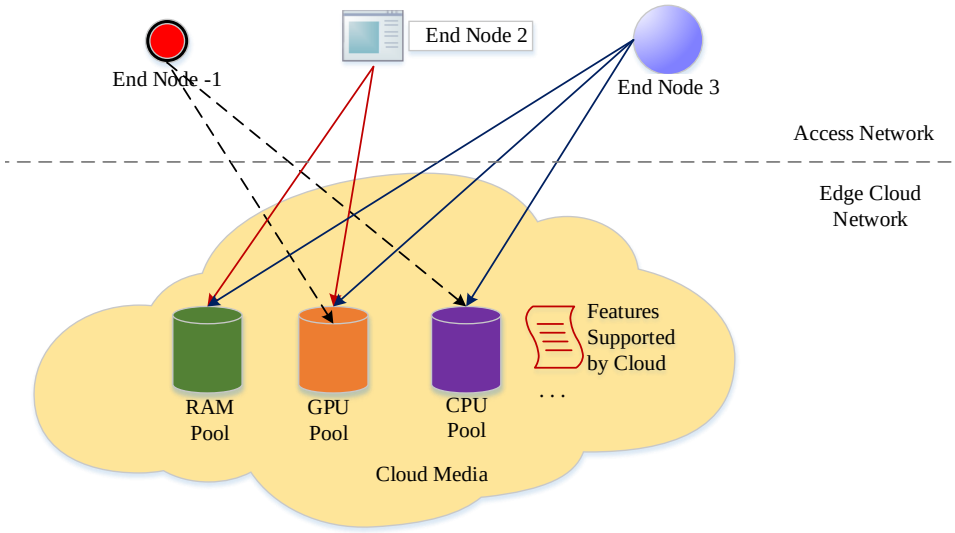

Figure 2.6: Cloud Aware Media Architecture for Edge Cloud [8].

2.2.4 A Use Case of Multimedia Processing at Edge Cloud

The proposed Thesis considers the architecture of Media Cloud from the perspective of application requirement while allocating resources; it is the duty of the cloud provider to ensure reliability and QoS for the end users. The difference in Media Cloud Computing is elaborated from a traditional Edge cloud through an example of Video Editing and Mashing. The steps involved in resource allocation to parallel processing are as follows:

1. Mashing is a process of joining multiple segments from different video sources to make one, whereas editing involves enhancing quality and cropping. The video mashing and editing require a high amount of resources which are limited to the end nodes of the personal computer and mobile phones. Media Cloud facilitates the processing of video mashing by providing sufficient resources.

2. The user requests for the resources to a cloud provider are sent through any web application that can perform video mashing, such as Jaycut (http: // www.jaycut.com). The media cloud's responsibility is to create an abstraction layer to hide the internal intricacies. At the same time, it needs to split the request into multiple sub-requests.

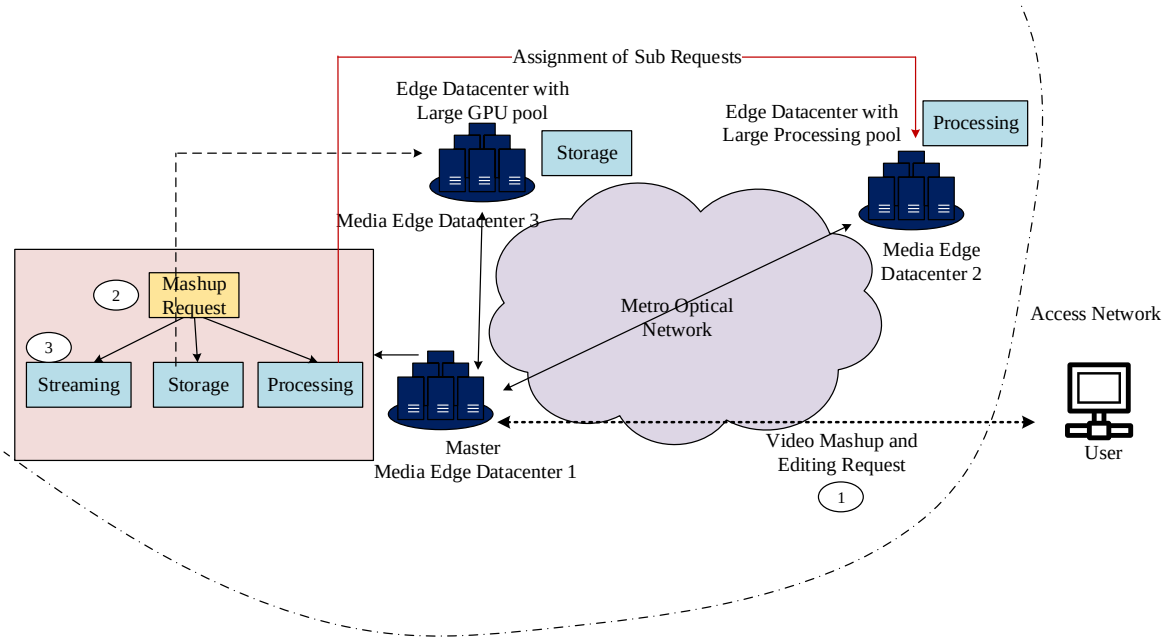

Figure 2.7: Task Sharing among Edge Clouds using the architecture of Media Cloud [8]

3. The divided sub-requests, depending on the type of resource they need, will be mapped to other Edge DC. For example, if a sub-request is a GPU intensive task then it will assign to an Edge DC which has the highest amount of GPU pool; at the same time, care is taken that the selected Edge DC is not too far from the end user. The streaming edge cloud will be the closest one to the user.

4. The interconnected Edge Cloud DC should be capable of handling requests by dividing the task into sub-tasks and giving the user a seamless experience.

5. Figure 2.7 represents the processing of a task at the inter-connected Media Cloud as referred to steps no. 1-4 inclusive.

A request is converted into sub-requests, these sub-requests which are interconnected, constitute what is called *Virtual Network requests* in the rest of the Thesis. It is important for service providers to incorporate Network Virtualization into their infrastructure to enable sharing of resources among multiple users. The following section describes the NV model and how multiple users are embedded in a given DC.

2.3 Network Virtualization

Network Virtualization (NV) is defined as sharing of the substrate resource with multiple users. It is performed in a way wherein the users are unaware of the existence of other users. It facilitates multiple heterogeneous networks to co-exists as a single network. NV brings advantages such as scalability, multi-tenancy, and flexibility, to the existing network without a change in any physical infrastructure [22]. In this section, the architecture of NV, design goals, and challenges associated with it are reviewed.

2.3.1 Network Virtualization Architecture

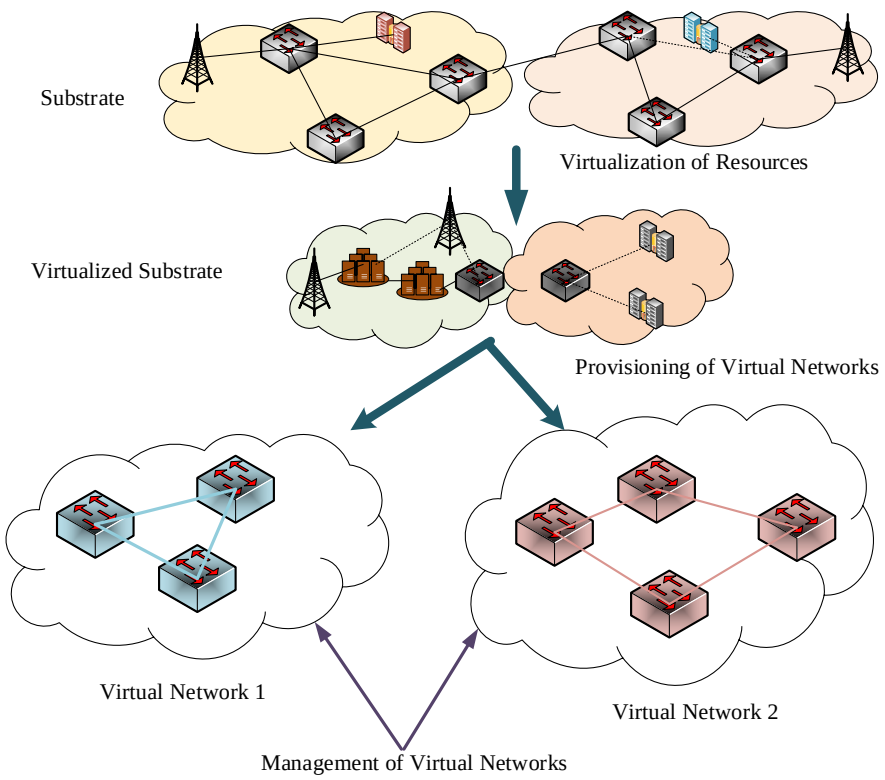

Figure 2.8: Network Virtualization Overview [21].

4WARD architecture [21] suggested one of the first proposals on the NV. Depending on the implementation of the core network, different proposals have been made [22]. Virtual Network (VN) constitutes a basic entity of an NV. VN consists of a set of virtual

nodes like switches, routers, and firewalls, along with interconnected Virtual Links. Virtual Links are mapped to the multiple substrate links. The mapping is done based on the QoS requested by the VN. Similarly, virtual nodes are mapped to different substrate nodes.

Figure 2.8 shows the mapping of VN to substrate network. In NV the VNs can be created by any service provider. VN1 and VN2 can belong to different service providers. It is VN owner's responsibility to define the type of QoS required, the protocols needed to utilize, and other constraints. Some of the architectural principles highlighted in [23] include: Co-existence, Recursion, Inheritance, and Revisitation.

1. Co-existence or multi-tenancy is defined as the existence of multiple VNs in the substrate network without interfering with each other's resources.

2. Nesting of VN request is called Recursion. In other words, it is a VN spanning between two different substrate networks.

3. It is possible to inherit the features of a parent VN to a child VN [24]. It is termed as Inheritance.

4. When multiple Virtual nodes from a given VN are hosted on a single substrate node it is called Revisitation.

2.3.2 Goals of Network Network Virtualization

Substrate networks are often composed of different switching technologies. A successful implementation of NV to such a network must address features such as flexibility, manageability, scalability, legacy support, and Heterogeneity.

A design requirement of NV is the ability to overcome rigidity and to provide user *flexibility* so as to define the type of required network. NV must support the freedom to set the type of topology, routing, forwarding and switching elements. It must also provide freedom of defining the type of physical layer attributes to the VN user.

By separating Service providers from infrastructure providers, resulted in what is called modularization of the network. It created a layer overlay models for different planes of a substrate network. By doing so, the NV should consider providing complete control of *manageability* of end-to-end layers to the Service providers.

The Infrastructure provider must design the NV such that it supports the increasing demands of users. That is, the NV should be *scalable* to incorporate the rising demands without major modifications to the architecture.

NV should support programmability, legacy devices and heterogeneity in the infrastructure. Heterogeneity is support for different infrastructure; for example, co-existence of optical and packet switching domain. At the same time, the virtualization should not lead to capital expenditure when upgrading the existing physical infrastructure.

2.3.3 Challenges Associated with Network Virtualization

Challenges associated with Network Virtualization pointed out by [24] are:

Interfacing: While creating isolation of resources for VNs it is the role of the service provider to establish a clear interface so that they can communicate with them.

Bootstrapping and Signalling: The CSP should have service connectivity to the network before establishing virtualization. This is because network connectivity is a prerequisite to itself. Bootstrapping should be included to provide customization of virtual nodes and links allocated to a service provider, which aids in creating a virtual network.

Resource Scheduling: Network Virtualization in a computer dialect is a multi-threaded environment. As VN share a common substrate resource, it is important that the Infrastructure provider can create an efficient method for providing resources to all the accepted requests. Resource scheduling is an NP-hard problem and research has been done concerning efficient scheduling of resources and tasks. [40] and [41] are a few examples of how to solve resource scheduling problem.

Virtual Network Embedding: It is another NP-hard problem, concerned with

optimal embedding or mapping of virtual nodes and links to a substrate network, with the achieving pre-determined objective. The objective can be to reduce the overall latency of the VN requests, to increase the network utilization, or to increase the service providers' revenue by increasing the profit. [4, 5] and [6] are some of the earlier proposals dealing with VN embedding. Selecting the constraints which matter the most, and reaching a feasible solution in a reasonable amount of time is a challenging task. To enable dynamic VN embedding over the present scenario makes it even harder for the CSP. In such a scenario, to respect a QoS and reach a feasible solution, the allocator needs to be completely aware of both Edge DC as well as network resources. The envisioned solution is to have a centralized controller which orchestrates the DC resources as well as being responsible for the allocation of network resources. In the subsequent section, Software Defined Networking (SDN) which is the most widely accepted solution that aids Network virtualization is presented.

2.4 Software Defined Networking

SDN is an emerging paradigm which promises to reinforce the currently distributed network without major modification to the existing infrastructure. SDN encompasses the network resource visibility, flexibility, and automation of resources by dismantling the vertical integration of control logic from the network elements and moving the control logic to a centralized location. In this section a comprehensive explanation on SDN is presented. Challenges associated with the traditional network, the motivation for SDN, and key building blocks for SDN using the layer model approach, are presented. A brief overview is also given of the analysis of the hardware infrastructure, southbound and northbound application programming interface (APIs) and Network Operating Systems. Finally, the chapter concludes by investigating different proposals for VN embedding problem using SDN.

2.4.1 Traditional Architecture and Challenges

In recent years, the Internet has grown both lengthwise and breadthwise connecting everything from a mobile phone to a DC, giving access from almost anywhere to anything. Despite this tremendous success, the traditional network is complex to manage [25]. It is both complex and difficult to configure the network to predefined policies and to reconfigure for adaption to dynamic changes in the traffic patterns. The vertical integration of the network control plane to the data plane has made the situation for development and improvement of the network even more complicated.

The existing networking infrastructure is distributed in nature, where a local controller running in each switch communicates with the neighboring switch to make informed decisions. To express desired network performance, the providers need to configure each switch separately using low-level, often vendor specific devices. Also, the system needs to incorporate the dynamic changes in network state and adapt to load changes. Automating reconfiguration and response mechanism are virtually non-existence in an IP-based traditional network. Enforcing policies and agile adoption to the network change in a dynamic environment is highly challenging.

It becomes even more challenging with the distributed nature. Different protocols like Open Shortest Path Fast (OSPF), Intermediate System - Intermediate System (IS-IS), Reservation Protocol (RSVP), and other Traffic engineering protocols face convergence issues due to the distributed architecture. These various networking protocols are bundled together and work in tandem to keep the network state in a stable condition. Any change in the protocol or sporadic traffic bursts could lead to unpredicted behavior in the network. These led to what is called *"network ossification"*, hindering innovation, and development of the internet. In the current network state, to design and implement a new protocol for IP would take around five to ten years to design, evaluate and implement [26].

Some limitations of the existing network which motivated the service provider to

consider SDN as a potential solution are:

1. **Highly Distributive Protocols:** Link State protocols like OSPF and IS-IS share, every few seconds, the network information with each of the routers present in the network. This occurs typically for every 10 seconds. Sharing the information in this fashion keeps the network active. Due to a massive number of *Hello* and *state information* packets interfacing between the network devices, and traversing through links, a major chunk of the network resources were consumed. With the tremendous growth in the Internet traffic, every bit of network resource is valuable to the service provider. It has always been and will continue to be a priority to find an ingenious way to reduce the network control messages and utilize it for data traffic.

2. **Convergence Issue**: The Convergence affects almost all the routing protocols in IP an network paradigm. When a network link fails, the node updates its nearby devices about failure, which propagates till all the devices in the network are aware. To reduce and eliminate false alarms, the industry follows a dead time configuration. When a device does not reply for more than a fixed time, then the network link is considered to have failed. According to industry standards, the dead timer value is usually four times the hello time, which amounts to approximately 40 sec. Other than the huge wait time, the converge state after the recovery is unpredictable in the IP network. It causes overloading as well as under-loading of links.

3. **Slow Adaption to New Services**: Due to the proprietary nature and vendor specific protocols, introducing new features to the infrastructure became a daunting task.

4. User experience, inefficient service deployment, physical layer protocol-centric issues, and difficulty with managing are a few other few major concerns.

Prior to SDN, many attempts were made to circumvent the above mentioned limitations as mentioned above, but most failed for numerous reasons. Implementations like

GMPLS and OpenSig were part of the the early research to improving the IP network. As pointed out by [25] some of the reasons why these technologies could not see a wide adaptation among the infrastructure providers are:

1. Data plane-centric rather than providing flexibility to the control plane.
2. Little focus placed on practical issues and efficient utilization of network
3. Could not guarantee the programmability of the network it advocated to the end user.

SDN is an emerging paradigm that breaks the vertical integration of data plane and control plane, promoting a logically centralized control plane. In addition, SDN introduced substrate Network and vendor agnostic programmability to the user. The separation introduced between the control plane, and data plane is the key to the desired flexibility. SDN technology enable creating the abstraction in networking, simplifying the network management and facilitating network evolution. In the subsequent section, a comprehensive study of SDN and its complementary protocols is carried out.

2.4.2 SDN Architecture

Software Defined Networking is an emerging paradigm where the network control logic is separated from the switching devices and is managed from a remote location. The control plane can be physically centralized or logically centralized. Different SDN architectures have been proposed to support the network traffic load balancing. Separating the control logic from the network elements makes the device a plain forwarding blocks called *white boxes* [28]. Overview of SDN architecture is shown in Figure 2.9.

SDN devices forward the packets based on flow rules rather than the lookup table. These flow rules are composed of numerous QoS factors associated with each flow between a device. Each flow rule is unique and represents a fingerprint behavior. Based on the flow rule, the devices decide the traffic engineering and rerouting principles to be followed.

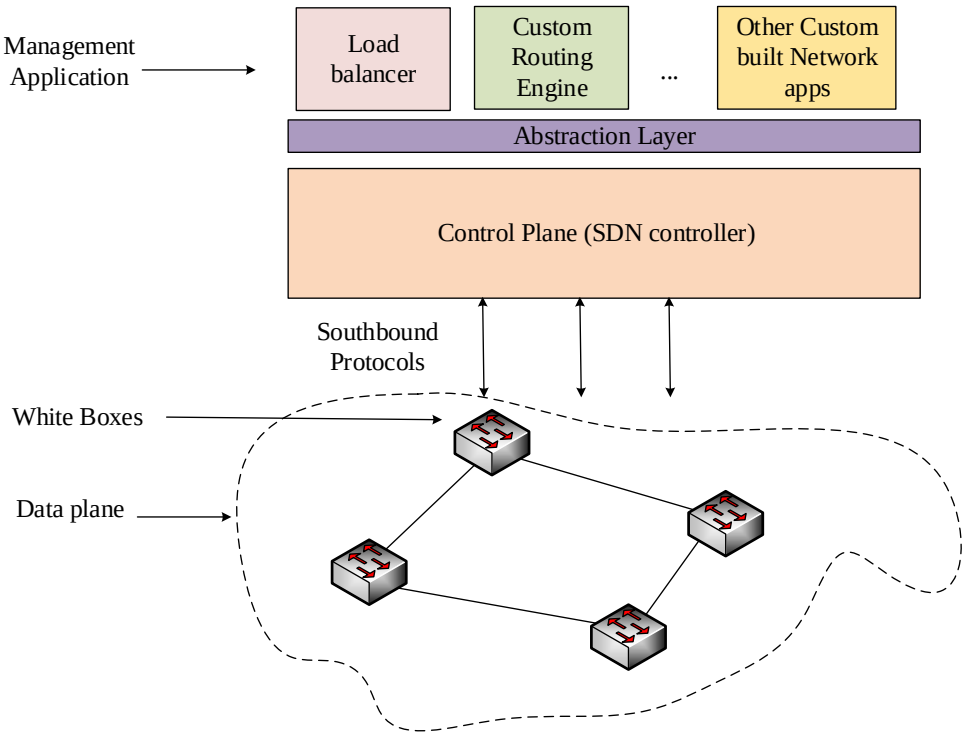

Figure 2.9: SDN Architecture [28].

Externalization of the controller enables placing it inside a DC or dedicated hardware. The controller can span in multiple locations. If the controller is placed at different locations, it is called Distributed Control Plane. A Distributed Control plane can be logically centralized or decentralized. In contrast, the physically centralized plane is one in which a single location is selected for the controller placement. The operating system controlling the resources of the network is called Network Operating System (NOS). The network is programmable through applications running on top of NOS.

The centralized nature of SDN offers three major advantages compared to current status quo. Firstly, the abstraction layer enabled by SDN architecture allows the use of high-level languages (C++, JAVA, Python). This in turn facilitates reducing the errors and faster deployment of services. Secondly, SDN enables automation of the device configuration. Thirdly, the global view provided by the SDN paradigm enables deployment and innovation of sophisticated algorithms to the network.

Based on the functionality associated with different modules of SDN architecture,

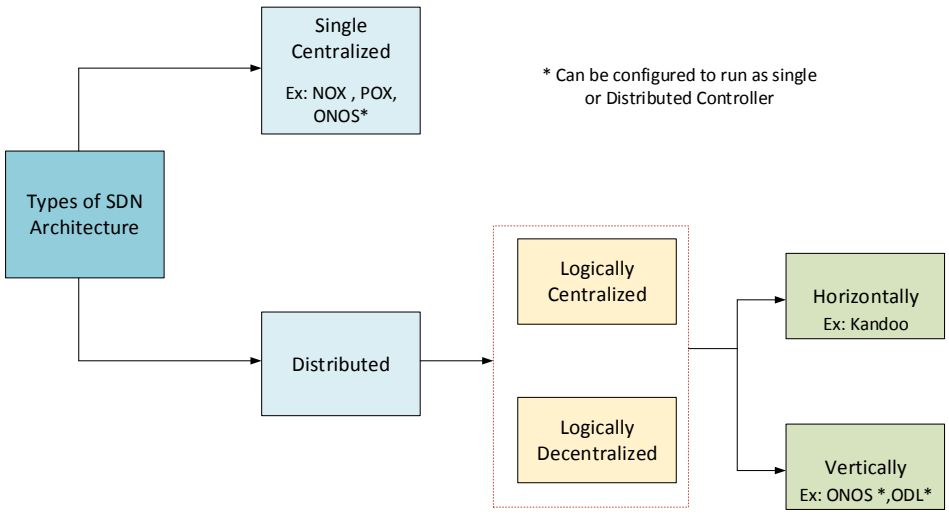

Figure 2.10: SDN Classification based on its Implementation Architecture.

they are classified into five distinct layers and each layer has a functionally associated with it. The layered model of SDN is shown in Figure 2.11. A brief outline of protocols, roles, and responsibilities associated with each layer is given in the following sections.

Data Plane

Data Plane constitutes the white boxes in a pure OpenFlow-enabled network [28] or elements like switches and routers. The devices could be optical, legacy or packet devices. This plane is also referred to as the forwarding plane. Each device in the forwarding plane has an OpenFlow table and an OpenFlow agent. Depending on the version of OpenFlow, the device can have different features. For example, devices supporting OpenFlow V1.0 do not support multi-table, and priority lookup table, whereas these features are supported in OpenFlow V1.3 onwards. The controller regularly sends *"keep alive"* messages to network devices to maintain track of their state. The state can be available flow table size, packets traversing through them or the failure notifications. Many protocols were introduced for the controller to switch communication. Netconfig, OpenFlow, or any other proprietary protocols can be used for this purpose. The scalability and vendor agnostic nature of OpenFlow promoted the research community to further develop

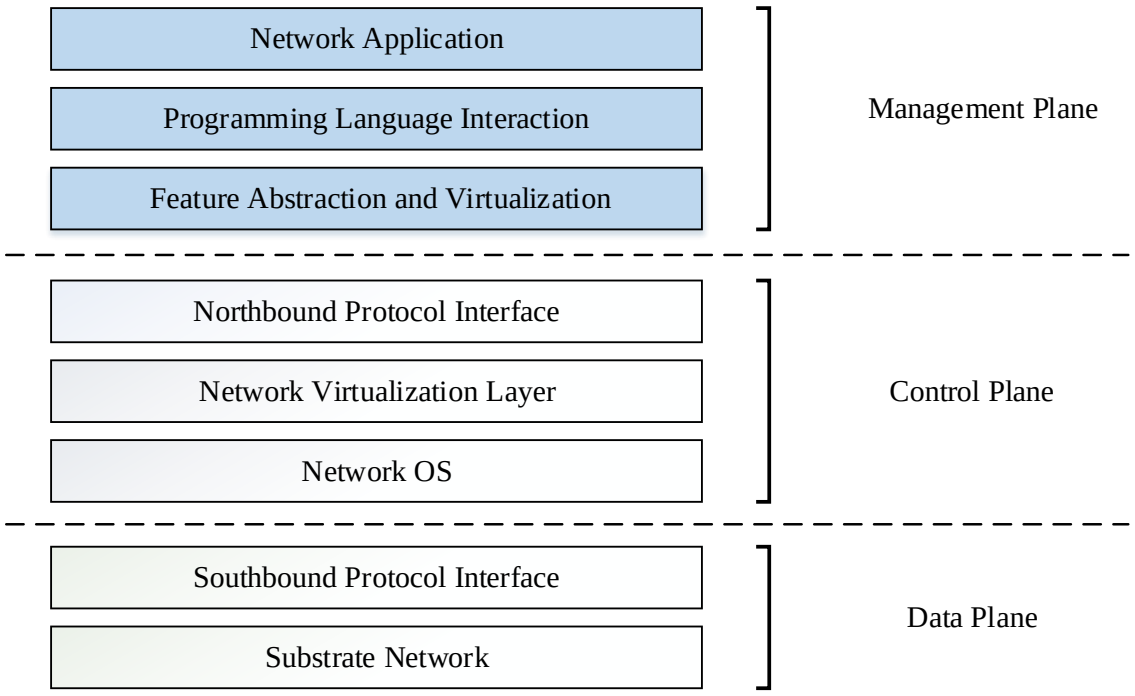

Figure 2.11: Layer representation for SDN architecture [26].

OpenFlow.

South Bound Protocol Interfaces: The Instruction set used to communicate with the forwarding devices, and the controller is defined in South Bound APIs. It formalizes the way the control and data plane interact. An example of such a protocol is OpenFlow [28]. A detailed architecture of OpenFlow as a South Bound Protocol is examined in the following paragraph

OpenFlow Protocol: Figure 2.12 represents the OpenFlow's Flow table architecture and Figure 2.13 accounts for the communication between the control plane and white boxes, through OpenFlow protocol. OpenFlow is considered to be the standard communication interface between the control plane and the forwarding plane [26]. OpenFlow gives access directly to the elements of forwarding plane. The network elements can be switches or routers. It enables seamless configuration of devices based on the application's need for setting bandwidth and policy enforcement to the devices.

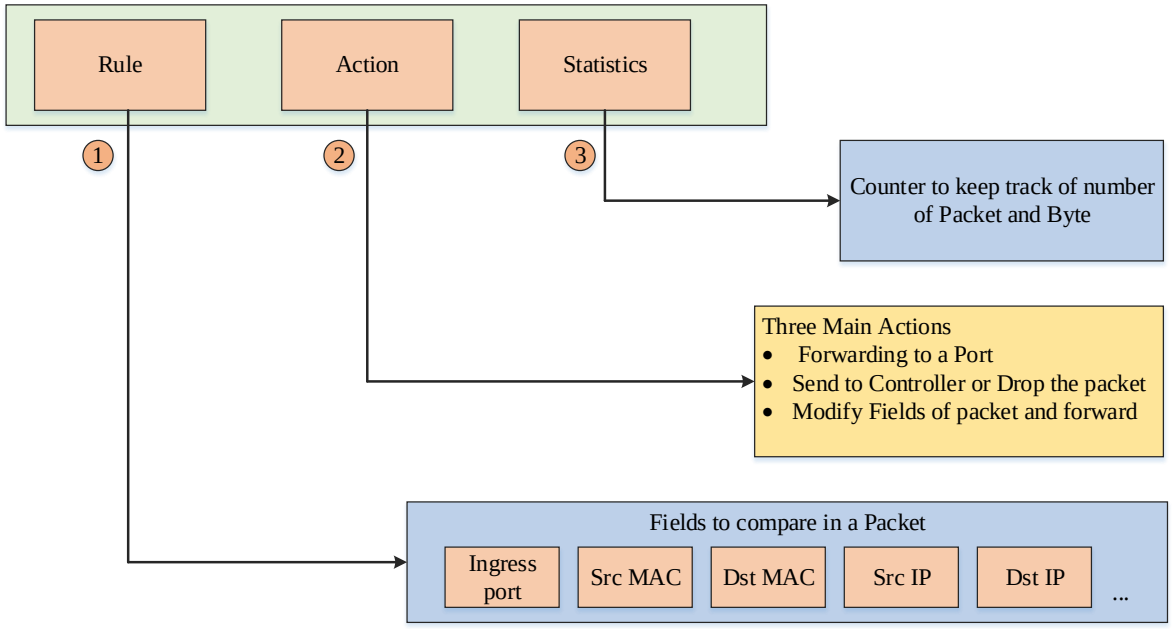

Figure 2.12: OpenFlow Flow table in V1.0 [28]

The major fields of OpenFlow table and the process of flow matching and forwarding are described below:

1. *Rule:* It defines which field is to be matched or verified before evaluating a flow. For example, if a flow needs to be matched with the device ID (DPID) it checks for the DPID field of incoming flow with its flow table. If the DPID of the flow is available on the table, it moves to the action section. If the DPID is not available, then the incoming packet is encapsulated and sent to the controller. The controller then determines the action to be taken. Once the packet of the particular format is assigned a Rule in a switch, it is considered as a flow, not as a packet. The reason for this is because all the packets of the same pattern are forwarded exactly in the same fashion. In the case of packet switching, there is no guarantee that the same packet will be processed the same way.

2. *Action:* Once the Flow rule has been matched the device checks for the action to be performed. By default, if there is no match the packet will be encapsulated and forwarded to the controller. If there is any specific action, it will be performed.

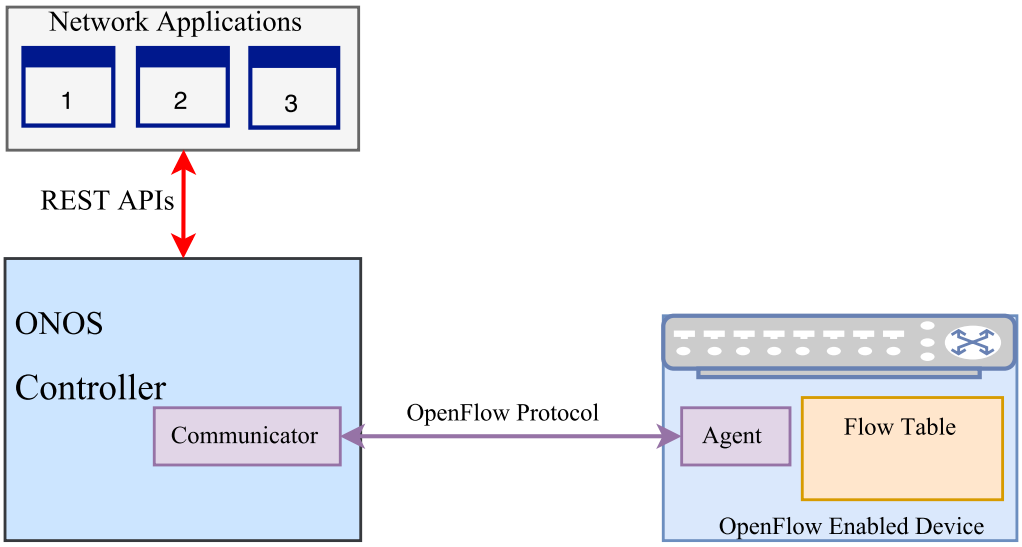

Figure 2.13: Controller Communicating with the White Boxes using OpenFlow Protocol.

The basic actions are, forward to particular physical port, drop the packet or encapsulate and forward to the controller. Other actions like to modify fields, or any custom-built action rules can be installed in OpenFlow protocol.

3. *Statistics:* This column is used to keep track of the number of packets processed by each flow rule and a device. It provides valuable information to build automated load balancing applications. It is also utilized to verify the flow rule activity and authenticity.

Control Plane:

Control plane can be viewed a Network Brain. It performs all the decision-making tasks of the OpenFlow devices. Network Operating System, North Bound Interfaces and necessary abstraction and virtualization layers together constitute the control plane. Many architectures are proposed for the control plane. Figure 2.10 shows the different control plane architectures with examples. ONOS is used as the control plane for the evaluation in proposed Thesis. A brief note on ONOS is given in the next paragraph.

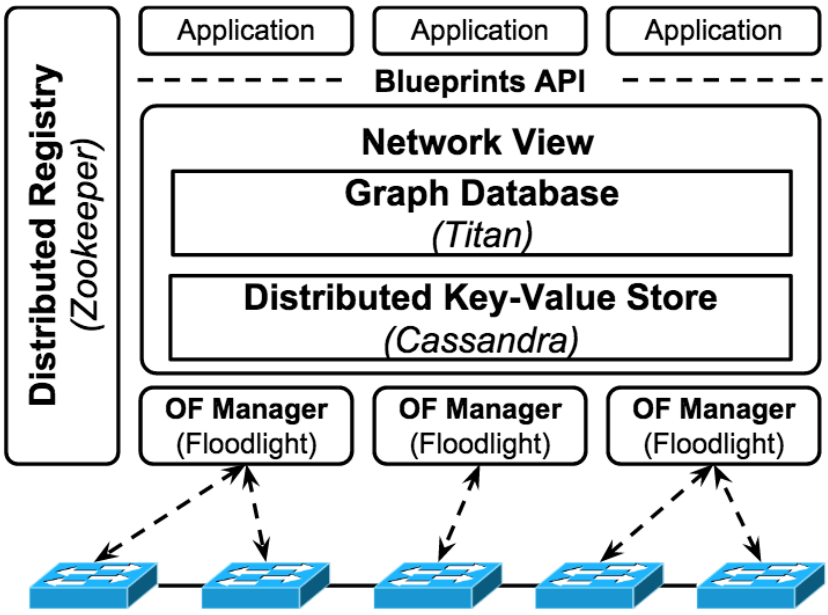

Figure 2.14: Overview of ONOS controller [63]

Open Network Operating System [63]: The reason for the use of ONOS is its structural support for optical and carrier grade networks. An overview of ONOS architecture is shown in Figure 2.14. Some of the core features introduced in ONOS which distinguishe it from other available controllers are: Intent Based flow installation, Central Office Re-Architecture as DC (CORD) support, carrier grade SDN controller, scalability, and horizontally distributed and logically centralized controller.

Network Virtualization and North Bound Interfaces: It is responsible for the easy integration of user built apps and it hides the intricacies of the underlying infrastructure. As well, the presence of this layer enables building platform agnostic network applications. Many innovative network applications have been proposed so far. The northbound interface offers APIs to the developers through which they can request the infrastructure information needed for their implementation. In this way, the network application developer can ignore unnecessary information from the network. Some of the most commonly used Northbound APIs are Java API and RESTful APIs. In this proposed Thesis, RESTful APIs are used for communicating to the controller.

Rest stands Representational State Transfer (REST) Commands: REST is an architectural approach to communications that is often used for communication with the server. It is a preferred method for communication between client-server models on the web because of its light weight, consumes less bandwidth and runs on HTTP protocol. The required information is collected through unique URLs. The modification to a resource is performed by use of REST commands such as: GET, POST, PUT, and DELETE. Every command has a specific function.

Management Plane

The user developed apps constitute the management plane. The apps can override the default inbuilt controller applications like routing engine and network configuration module. The main enabler for such a platform is the Abstraction layer.

Abstraction and Application: The abstraction layer is also known as the virtualization layer. Although it is not part of the SDN controller architecture, it is considered as one of the main complementary characteristics of SDN. The Application layer contains applications like a Load balancer, monitoring, VN Embedding and User defined UI to name a few.

All the discussions detail with regards to SDN architecture were for a Packet switching network. Few proposals were made using OpenFlow V1.0 to extend and include the Optical network in the SDN framework [29], [30] and [34]. This extension of the SDN framework enabled a single controller for different forwarding plane. The changes made to the SDN controller to include Optical Network and extend to include Metro Networks are noted briefly in the subsequent sections.

2.4.3 OpenFlow for Optical Network

Implementation of SDN-based IP network on top of the static optical network may result in inefficiencies and instabilities in the network operation. As an example, setting up

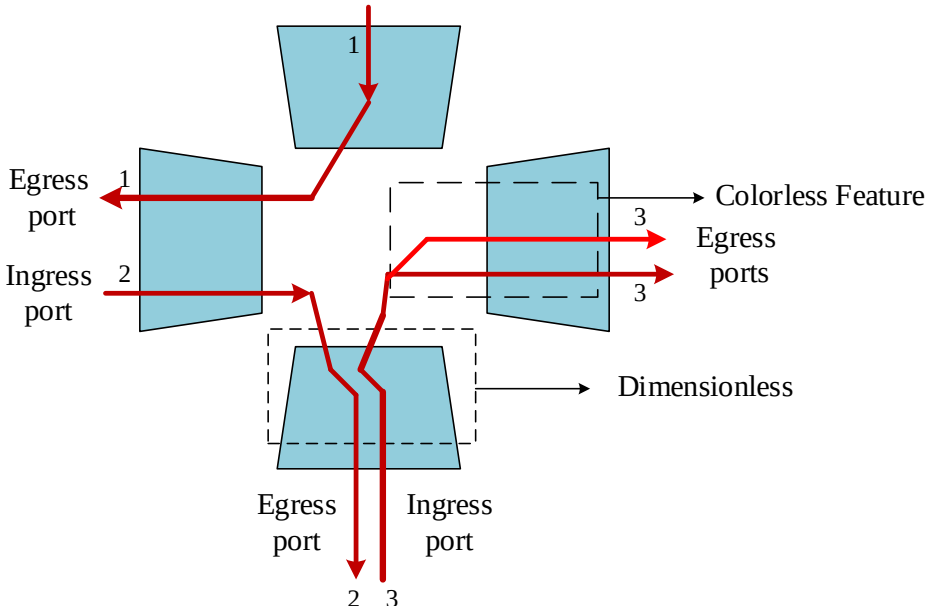

Figure 2.15: Colorless Dimentionless Contentionless ROADM Architecture

dynamic bandwidth allocations between two remote locations might lead to a bottleneck by the static wavelengths between them. If the lambda routing between them could be automated and controlled dynamically, then a significant level of utilization and flexibility can be achieved.

The main factor to consider for implemented SDN to any network is that the network elements be software programmable. Until recently the optical device programmability was not possible and required human intervention for the network's configuration. However, with the introduction of Re-Configurable Optical Add Drop Multiplexer (ROADM) [13], it is possible to dynamically and programmatically adjust and groom the wavelengths.

Colorless Dimentionless Contentionless ROADM

Figure 2.15 explains the characteristic Colorless Dimentionless Contentionless (CDC) of Reconfigurable Optical Add Drop Multiplexer (ROADM). A ROADM is said to be *Colorless* if any available port is capable of transmitting any wavelength from the c-band. A ROADM is said to be *Contentionless* if at any given time the same wavelength

can be transmitted as well as received at two different ports. A ROADM is said to be *Dimentionless* if all the ports available can be used as in-port as well as out-port, and that they can be configured accordingly, depending on the traffic behaviour. Although a complete realization of CDC ROADM is still not practical but theoretically several approaches have already been made [13].

Showin in Figure 2.16 is is possible to manipulate all the rules associated with the Rule section depending on the optical node's capability. For considered test-bed in this Thesis, we only manipulate the channel spacing and for cental frequency c-band spectrum is used whereas, for rest of the fields constant standard values are used.

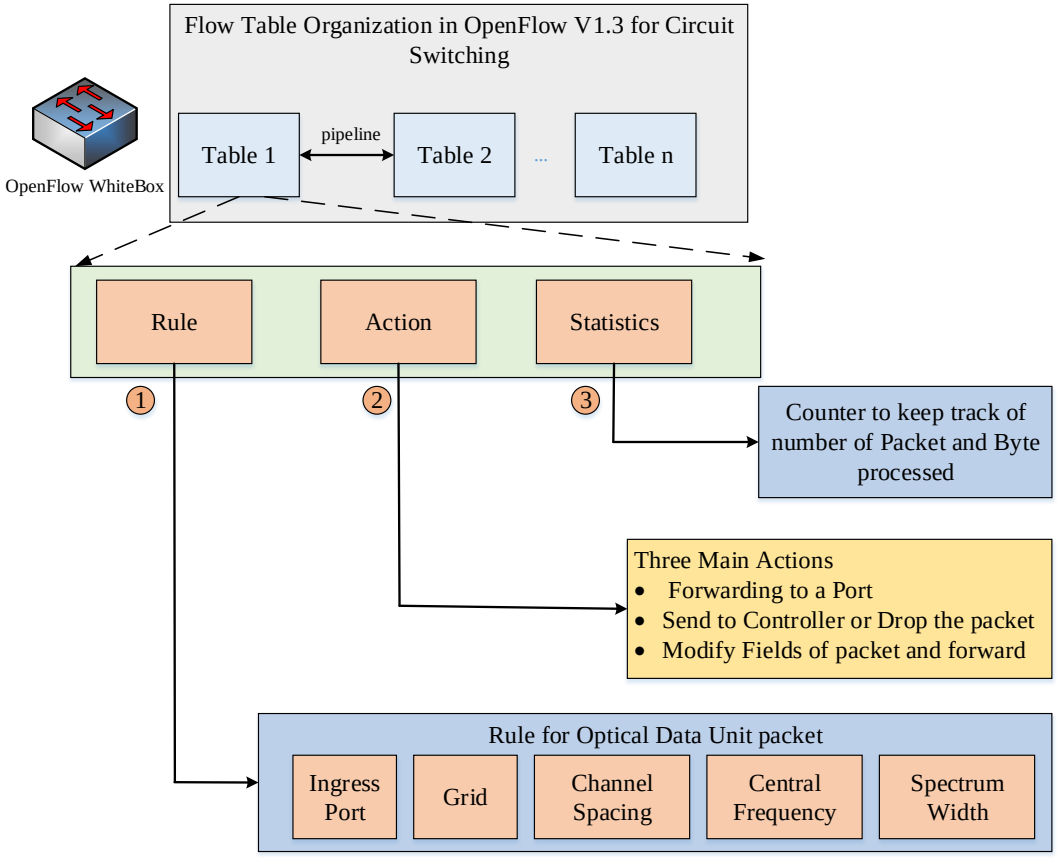

Figure 2.16: OpenFlow Flow Table Format for Optical Network [35]

The confluence of ROADMs and standardization of OpenFlow to include optical network, created the perfect opportunity to enhance the network utilization. Figure 2.16 outlines the extension of OpenFlow for the inclusion of optical network. [35] made an

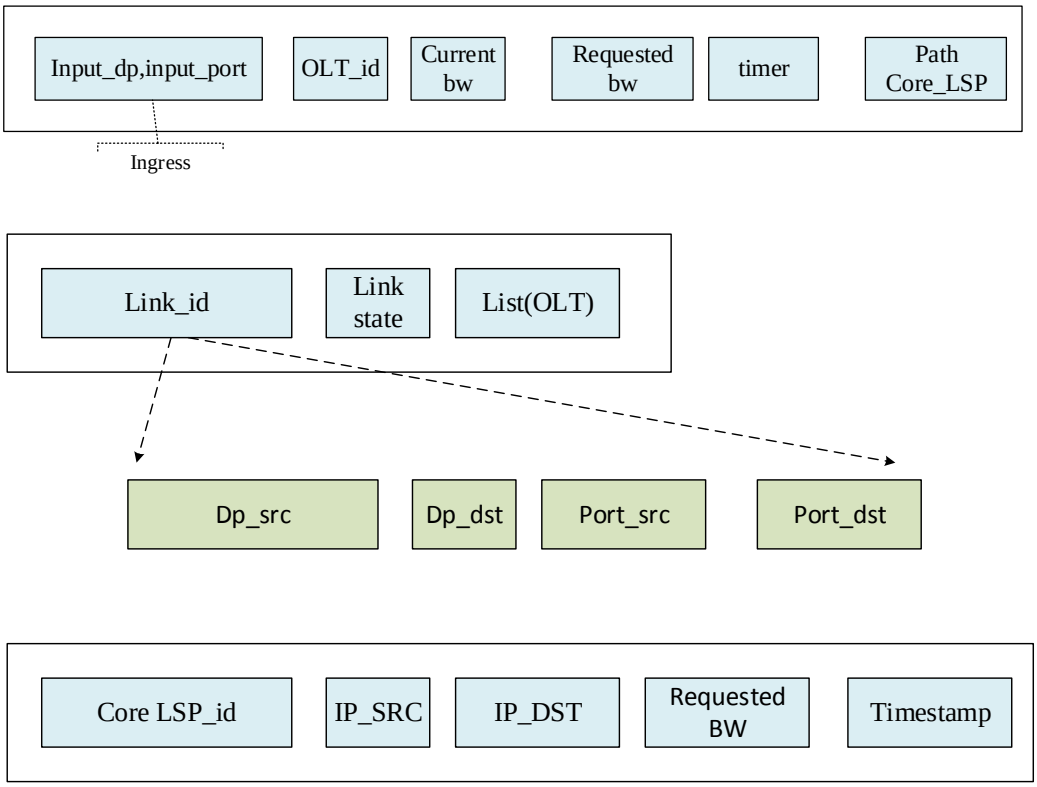

Figure 2.19: Data structure for LSP [32].

initial attempt at extending OpenFlow V1.0 to include circuit switching. This was carried out by introducing flow level abstraction into the switches flow table. With the introduction of OpenFlow V 1.3 [31] the circuit switching became part of the standard OpenFlow protocol. Several modified versions of OpenFlow to support unified circuit switching have been proposed. One such proposal for the flow format was introduced by [32] to bring unification of different networks is shown in Figure 2.19

2.4.4 SDN for Metro Optical Network

Metro Optical Network is composed of Flexi-grid Optical Network. In other words, these optical devices support fine tuning of wavelength selection. The optical network can be tuned with the least granularity of 6.25 Ghz compared to 50 GHz in a rigid Optical Network. Flexi Grid facilitated the Metro Network traffic to be dynamic and provide

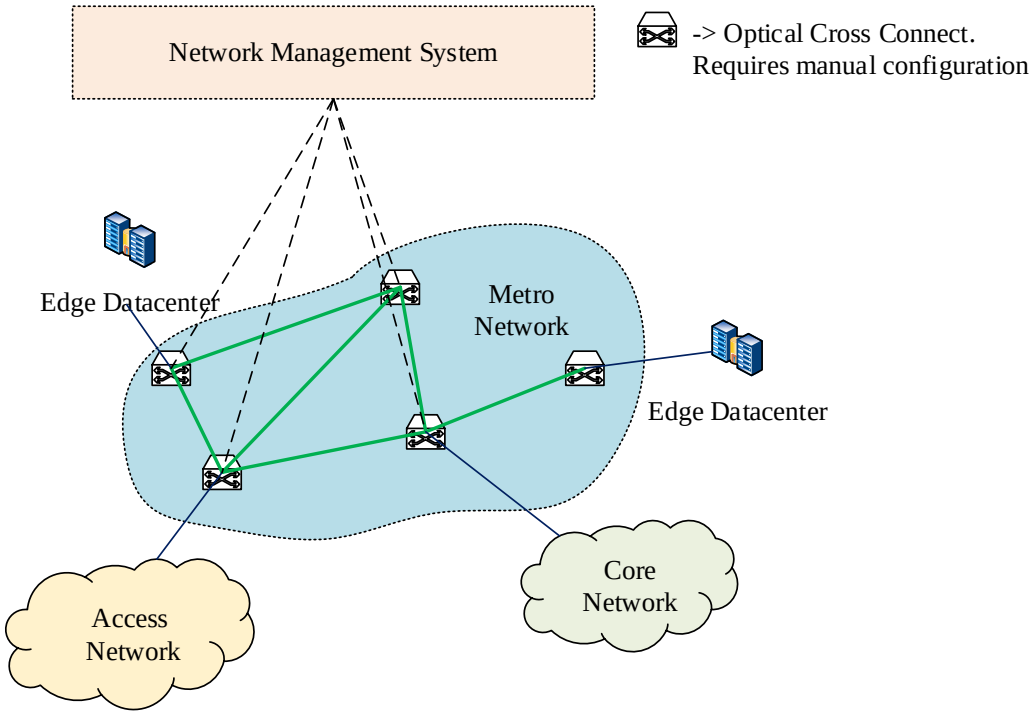

Figure 2.20: Network Management Based Control Plane for Optical Network

better QoS to the users.

It is a common misconception that the MON is a miniature incarnation of Wide Area Network (WAN). Also, the solutions proposed for WAN can be used for MON. However, WAN and MON differ with regard to wider protocol, switching support, loosely aggregated data, the density of network and traffic bursts, to name a few [39]. An optimal performance of MON will aid in efficient working of WAN. Traditionally, MON was used for aggregating different access networks into WAN. With the introduction of multimedia services and web 2.0, the need for MON to be more agile and flexible became inevitable [2]. Different Control planes used for MON are described in Fig. 2.20 - Fig. 2.22

Fig.2.20 depicts the use of Network Management System (NMS), based on static configuration. All the configurations were done manually, involving human intervention between the control plane and data plane. Due to its rigidity, the use of MON for multimedia services was not feasible. To bring automation into the system, GMPLS

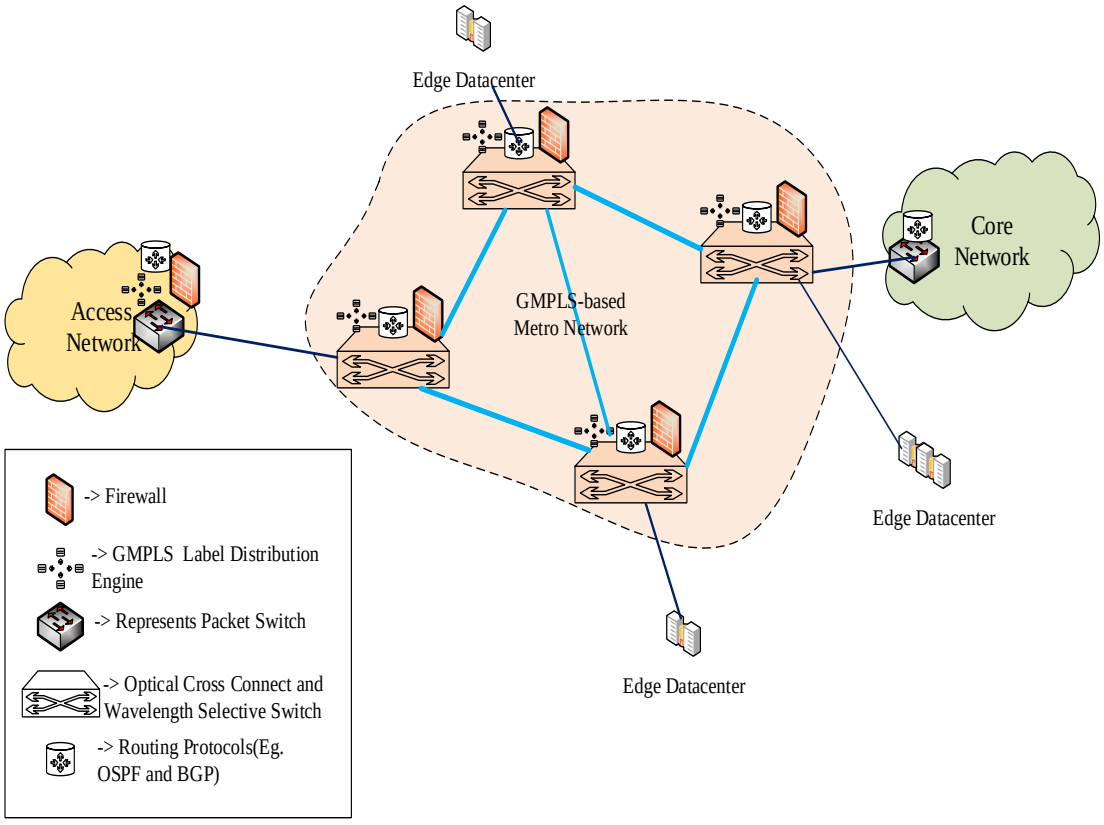

Figure 2.21: GMPLS-based Metro Network

was introduced. Fig. 2.21 depicts a general architecture of the GMPLS-based network. GMPLS is a not just a mere extension of MPLS protocol, but it also acts as a control plane to the network and allows dynamic configuration. GMPLS was a distributed control plane architecture, and the presence of handshaking signals, RSVP-TE, MPLS-TE and other protocols led to increasing complexity. Due to the proprietary nature of these protocols, scaling the existing architecture becomes impractical. GMPLS is responsible only for controlling and routing of devices and cannot handle a task like automation of network resource orchestration. The complications and sophistication involved with the number of protocol interactions led to the windfall of GMPLS paradigm [61]. The SDN-enabled network can virtualize the data plane and can run many user defined applications to configure different QoS, based on the application's need. SDN-based architecture for MON is shown in Fig. 2.22. The centralized nature of SDN architecture facilitated easy

integration of new technologies, improved flexibility and automated the orchestration of network resources to VN requests. [32] are some of the attempts put forth to leverage the global view and abstraction provider by SDN to enhance the performance of Metro Networks. VN embedding is an NP-hard problem; several approaches for solving this

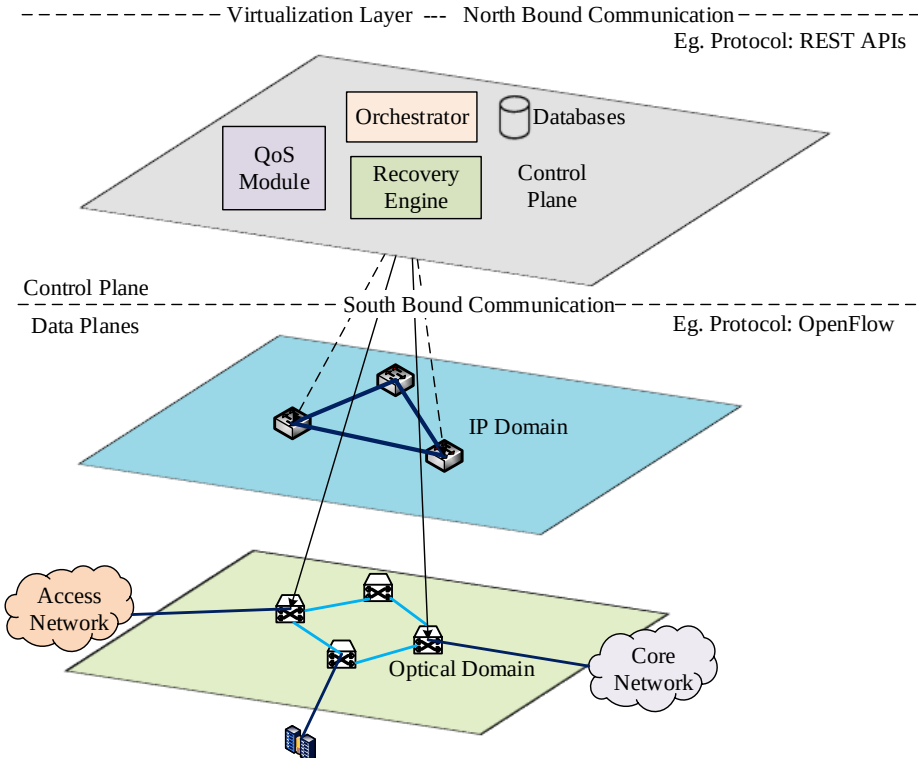

Figure 2.22: SDN-based for Metro Network

problem using SDN are discussed in the following section.

How Metro Network Different from WAN

It is a misconception that MON is a miniature incarnation of Long haul network. MON is more like a synergic model which aids the service provider to efficiently utilize and facilitate the ultra dynamic, highly variable network traffic of the end users. Some of the key unique characteristics of MON network are:

1. Needs a Multi Protocol Support

2. Optical Transparency
3. Mesh Scalability
4. Multi survivability option
5. Sub rate bandwidth provisioning
6. Cost
7. Network Management
8. Reliability and Modularity
9. Fast Intelligent Provisioning

1-5 are mostly related in designing the topology and use of physical ROADMS such as using DWDM technology, CDC ROADMs, Mesh Type network to name a few, assuming such a physical infrastructure is already available, the post work is to manage and efficiently utilize such a network. This is characterized by (6) - (9). A brief detail about 6 – 9 is discussed following paragraphs.

Cost: With the increase in service provider, and the CAPEX cost of installing new devices and adopting new technologies, it is inevitable to follow an avenue which in long run helps reduce cost. Therefore, an orchestration and optimization technique is needed which can make use of network resources by efficiently reducing the cost for the network provider without affecting the QoE for the end user.

Network Management The MON characterized by a highly dynamic, volatile and highly unpredictable traffic. To manage and sort this type of traffic needs a very agile system. The existing network plane cannot efficiently manage such a system.

Reliability and Modularity: In physical domain it is taken care by proper mesh design and use of DWDM network. But to efficiently utilize and provide those resource dynamically by understanding the needs of the end user is of paramount.

2.4.5 Related Works on VN Embedding on SDN-based Network

This Section deals with an overview of proposals related to the VN embedding on substrate resources. The global view provided by SDN/OpenFlow architecture is leveraged by many researchers to optimally utilize the available network resources. Authors in [43] have proposed VN mapping on the optical network using the reactive approach, where the allocation is triggered during a network failure. Authors in [44] have made use of the failure probability factor in their VN embedding model to decide on the Virtual Optical Network Location. The aforementioned work are focused towards reimplementing the VN requests after a network. They follow Integer Linear Programming and Heuristic technique to determine the optimal VN embedding solution. Heuristic approach provides solution in less time but at the cost of suboptimal solution, whereas the ILP provides optimal location at the cost of time to reach a solution. In the proposed work of this Thesis, Integer Linear Programming and Column Generation techniques are utilized.

[45] proposes a dynamic VN mapping based on switch load, and switch factors like memory and storage. [46] highlights the fact that the number of 'flows' in a switch is vital for efficient embedding of VNs. Both the approaches [45] and [46] used Integer Linear Programming technique and focused on increasing the utilization of data plane. These consideration are important while designing an SDN-based network. The work does not consider the type of VN requests and does not differentiate the allocation based on the QoS requirement of the VN requests.

A framework of VN embedding for a converged network of fiber and wireless network was proposed in [47]. The model accepts VN requests if the computational requirement is met. A heuristic approach is adopted for reallocation of channels if link embedding

is not satisfactory. The proposal by authors of [47] does not consider the VN requests Datacenter requirement while embedding to substrate network and is focused towards only network utilization. Authors in [48] considered the computational attributes of VN request along with link and switch requirement. The key objective of their proposal is to increase the acceptance ratio and infrastructure service provider's revenue. [49] proposed a Mixed Integer Programming technique for mapping VN requests to a substrate network. [48] and [49] model allows for partial embedding of VN requests, and least preference was given to QoS with the objective of increasing the provider's profit.

In the light of aforementioned works, the main contribution from the proposed VN embedding model in Chapter 3 are: (a) A Virtual Network embedding approach on a Metro Optical Network consisting of Media Edge Clouds (MEC- DC) to provide guaranteed QoS. The proposed model explicitly considers the multimedia requirement. The proposed model uses the concept of generating independent configuration columns using the branch and bound paradigm so as to reduce computational complexities with respect to link embedding. ILP was used to determine the optimal location for the placement of VN's DC requirement. (b) Proposed an SDN framework for integrating VN embedding and to ensure proper coordination between different embedding modules.

2.5 Summary

It has always been a major concern among the research community and the industry to find a way to enhance utilization of network and DC resources. In this chapter, basic architecture of cloud, the evolution of cloud and SOA used in the cloud were discussed. Limitations of traditional cloud and various approaches used to serve the increasing traffic in Access and Metro Networks were reviewed. Furthermore, virtualization by enhancing the utilization of network and DC resources was evaluated. Factors affecting the complete realization of virtualization were also examined. How SDN can aid virtualization and resource management by automation was studied.

Lastly, some of the current state of the art approaches on using SDN paradigm for VN embedding, were investigated. Three main observations that can be made at this point, from problems resulting from the proposals for VN embedding are: Firstly, focus was mainly directed towards the data plane utilization and very little or no focus was given towards overhead associated with running CPU intensive orchestration algorithms on an SDN controller. Secondly, the majority of the existing VN embedding technique does not consider the stringent multimedia requirement. Finally, consideration is not given to the SDN controller location attributes while performing VN embedding. By considering these research gaps a novel architecture for VN embedding on SDN-based Metro Optical Network is proposed in the following chapter.

3 VN Embedding in SDN-based Metro Optical Network

3.1 Objective

It is important that the available DC and network resources be efficiently utilized. This chapter proposes a mathematical model for dynamic VN embedding for SDN aware MON. To give a better idea of the adapted techniques, details are given on the different constraints that affect the VN embedding. SDN paradigm is proposed to bring better resource visibility and flexibility for allocation of available resources to Carrier Service Providers. The Proposed framework for SDN, interaction between its components and the adapted link recovery module are explained in detail.

3.2 Overview of the Adapted Architecture for VN Embedding using SDN

3.2.1 Problem Description

A VN embedding problem can be defined as attempting to find the most *optimal* locations to embed the incoming VN requests to a substrate network. VN embedding is a Non-Polynomial deterministic (NP) hard problem. The optimal solution can be stated distinctly as depending on the objective the service provider intends to reach. For example, if the service provider wants to achieve the maximum utilization of substrate

resources, then his objective function will be to accept those requests whose requested resources sum will be equal to the available substrate resources. The service provider, in this case, does not care about the QoS. Similarly, if the goal is to achieve maximum profit, then request who agree to pay comparatively more for the usage of resource will be accepted.

Often, in reality, the targeted object needed is governed by many constraints. Considering the right constraints in a mathematical model is vital to justify the optimality of the solution obtained. One more challenge while considering the constraints is that, if all the known constraints are included for determining the solution then the time required to solve the problem will be beyond practical time limits. Thus considering the right constraints to reach an optimal or near optimal solution in a reasonable time frame, is the aim of the proposed of VN embedding technique.

3.2.2 Translation of Media Request into a Virtual Network Request

The process for the translation of media request to a VN request is shown in Figure 3.1. The order of steps involved from requesting a service to granting access to a service is numbered from 1 to 8. User requests the service to the cloud provider for processing a video or for streaming a video or any multimedia application. It is led to the freedom of the user to choose the cloud service provider based on the incentives and the services required for the user. The cloud service provider can be any third party agent who has leased the cloud architecture, or the cloud infrastructure owner itself, such as Amazon and Google. Once the cloud service provider receives the user request they classify it based on the type of service the user is demanding. For example, if it is a video editing or processing request, it will require high bandwidth lines and VMs which are capable of streaming and processing of multimedia content. Once the multimedia request is obtained and profiled according to the QoS requirement, will be send to the Resource requester.

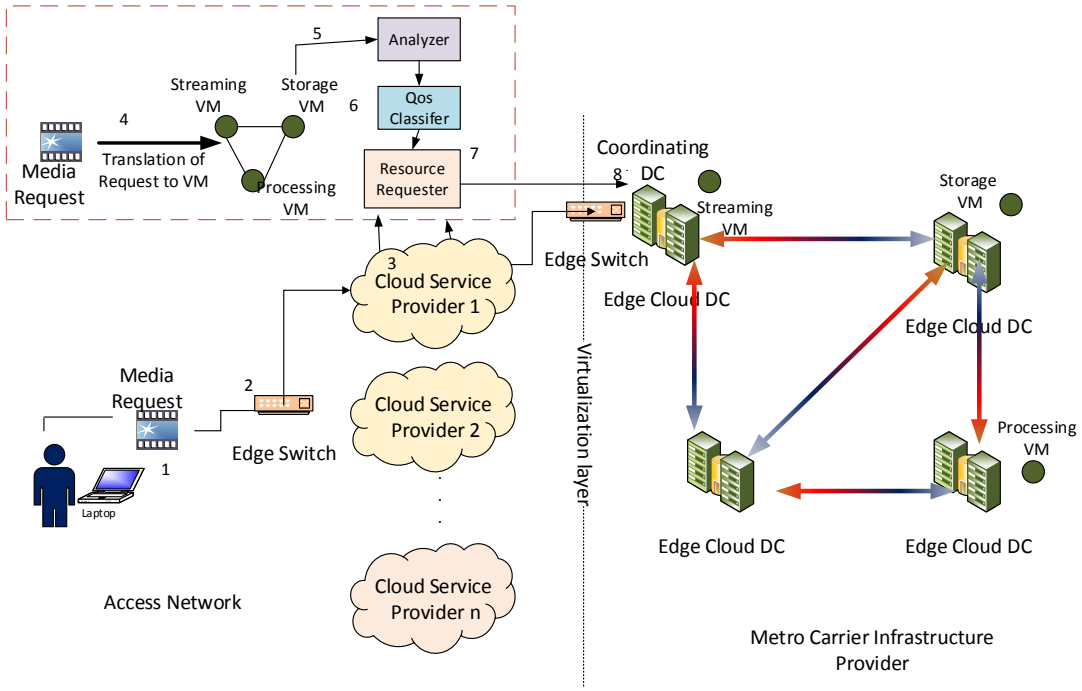

Figure 3.1: Steps involved in Translation of Media Request to a Virtual Network Request

The Resource requester is responsible for communicating with the carrier infrastructure provider. Virtualization of resources will be performed by both cloud as well as carrier service providers. From here on the request of the user is seen as a VN request. In the case of the considered example, there are three DC nodes, one each for streaming, storage and processing. Depending on the user requirement, the number of VMs and the capacity can vary. These VMs are fed to what is called the coordinator Edge Cloud. This cloud is responsible for communicating with the carrier providers and reserving the network resources, such as particular paths and bandwidths in them. After the conversion of the user request to a VN request, it the responsibility of the carrier provider to ensure that the agreed SLA is maintained during the course of lease. In this scenario, due to a lack of complete interaction of the user requirement and the type of network service being offered for it, resulted in poor QoE and inefficient network utilization. To improve this deficiency in the network architecture, the envisioned solution to managing the cloud as well as carrier infrastructure is SDN. In the following section an explanation is given for the role of SDN framework proposed to embed the VN request from the

carrier point of view.

3.2.3 Overview of VN Embedding Model using SDN

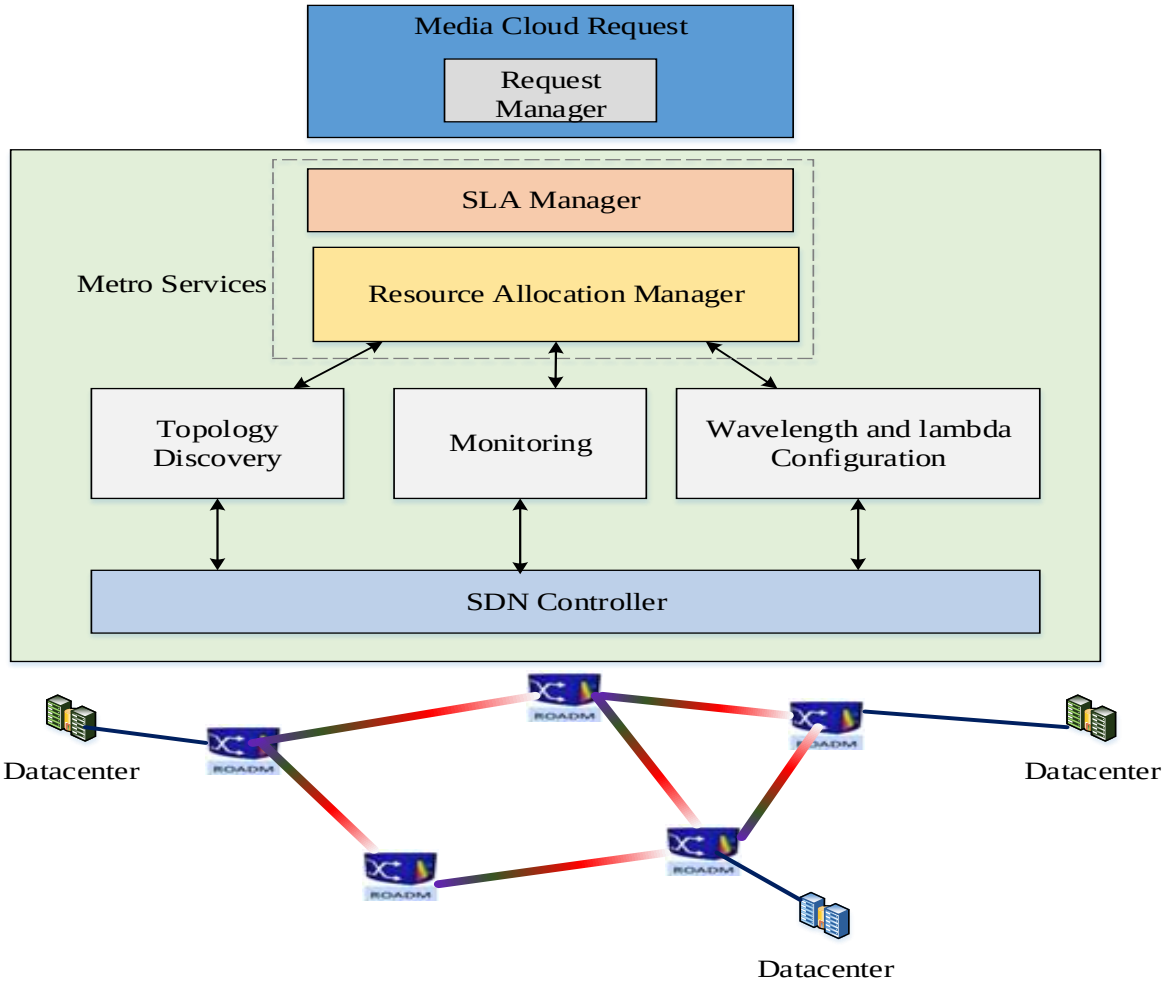

Figure 3.2: Overview of SDN framework and VN embedding

Figure 3.2 represents the framework for the VN embedding in the SDN framework. The SDN is responsible for maintaining the resource databases and providing any required network information needed for the resource allocator in real time. The real-time behavior and availability of all the information necessary for the resource allocation manager is the key for enabling dynamic embedding in the network.

In the case of MON monitoring, Topology discovery, wavelength, and lambda config-

uration module together form the main services which the Resource Allocation Manager required. Topology and Monitoring belong to the class called *active listener*. This means that they are actively listening to any network change and respond immediately based on the policies defined in SDN controller by the infrastructure provider. Whereas Wavelength and Lambda, configuration module belong to a class called *passive listener*. It is triggered when there is any link failure, or a new request needs to be configured.

SLA configuration can be done manually or defined as an automated application. In the case of automated SLA manager, the decision on how to evaluate a particular VN request, depending on the QoS it requires, will be handled accordingly. By defining different policies and algorithms, it is possible to modify the objective of the VN embedding problem. SDN facilitates faster implementation and testing of new SLAs to the network. Once the resource allocator makes the decision on the location and paths, the next step is to embed the requests to the substrate network.

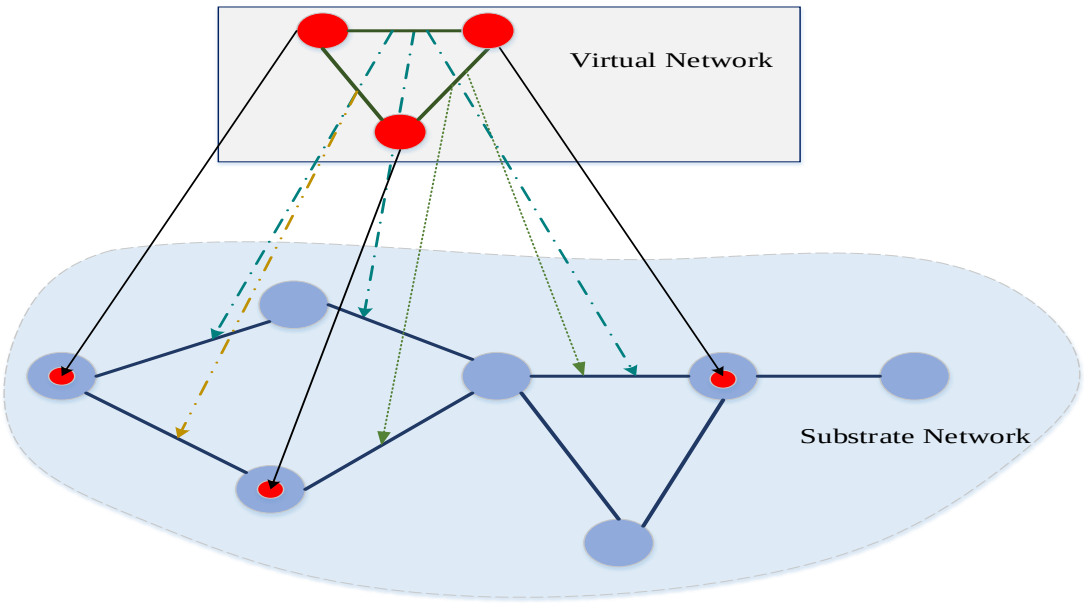

Figure 3.3: VN Embedding on Substrate Network

The process of embedding a VN request is described in Figure 3.3. It shows a VN request having three nodes interconnected with three virtual links. Each node has a

QoS associated with it. One node is for streaming purposes, one for storage and one for processing needs to be assigned in the substrate network. The SLA manager based on the client's requirement decides the QoS that it is suitable to, and passes this information on to the Resource allocator. The resource allocator decides according to the defined policy where to embed the request. To further understand the optimization technique used in a resource allocator is presented in the next section.

3.3 Overview of Optimization Technique Adapted

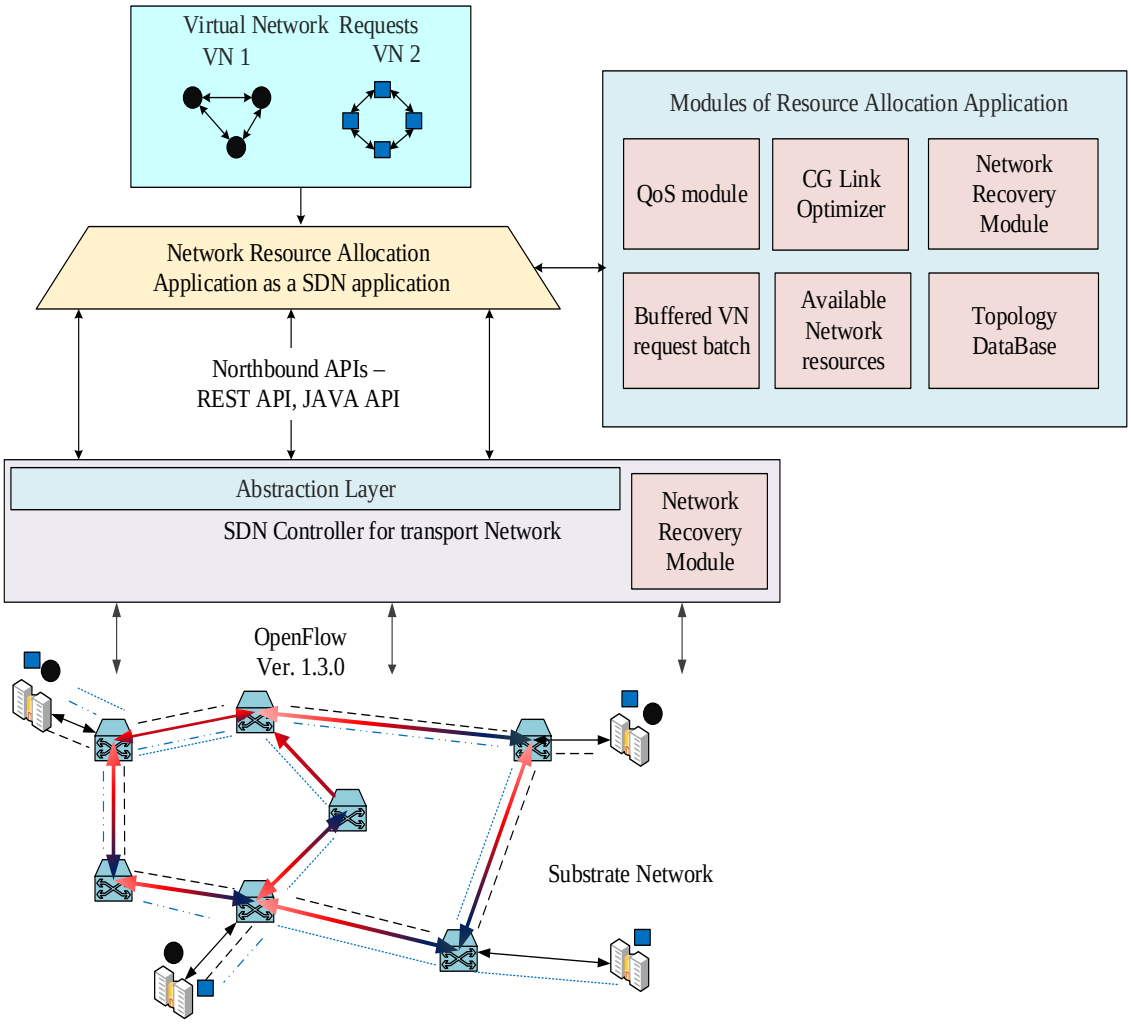

Figure 3.4: Overview of Optimization of VN embedding

The adapted Optimization technique is depicted in Figure 3.4. There are several

approaches to formulating the VN embedding problem, such as Integer Linear Programming, Heuristics, Branch and Bound, and Mixed Integer Linear Programming to name a few. It is important to choose the right combinatorial technique for solving a problem. Because each technique has certain advantages and limitations associated with it. The proposed combinatorial technique for solving VN embedding in an SDN-based MON is a combination of Integer Linear Programming (ILP) and Column Generation (CG). Subsequent paragraphs cover brief details on the ILP and CG techniques and the constraints used.

There has been an increase in user dependability on the cloud and the number of CPs. This has led to a tight competition among the CP to win the VN request. When many VN requests need to be embedded, it is important that agreed QoS is guaranteed. If not, the client will jump to another CP. This competitive scenario puts the CP in a dilemma as to which request will ensure the highest profitability, as well as an increase in the network utilization. At the same time, they also need to ensure that QoS is guaranteed. Motivated by the dilemma of the Cloud Service Provider, the proposed VN embedding objective is to increase the service providers profit.

The Constraints considered facilitate in ensuring the agreed QoS for a request will be fulfilled. The most important QoS for any request is Bandwidth, Link Latency, and DC resources. For a multimedia request, the need for GPU in a DC resource is mandatory. The ILP model which is concerned with DC resource embedding for a VN request only accepts a request if it has sufficient amount of RAM, CPU, GPU, and Storage. Along with this, it also needs to ensure that no partial embedding and overlapping transpires. The constraints not only aid in reducing the search space for the optimal solution but also ensure that the solution is feasible from practical standpoint.

The constraints considered in ILP formulation are only responsible for DC resources. For light path-embedding, the Column Generation technique is used. CG follows the principle of the branch and bound technique, where a huge problem bifurcates into two smaller problems, i.e., master and slave (or pricing) problem. The master problem

generates duality constraints. Using the duality variable column is generated in the slave problem. The Generated Column is used to populate a matrix. This matrix is used with the master problem to achieve the targeted objective. The splitting of the problem into many smaller problems facilitated in reaching the solution faster with a less number of CPU cycles. A simple equation decomposition using branch and bound is shown in figure Figure 3.5.

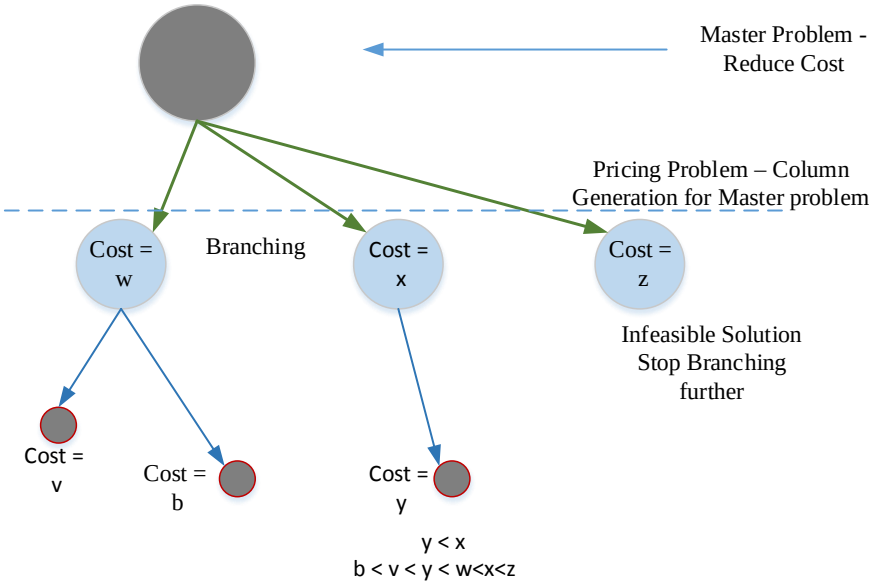

Figure 3.5: Decomposition of a Problem in Column Generation based on Branch and Bound Technique

To solve a problem based on column generation requires two sets of constraints, one for the master problem and the other for pricing or slave problem. Constraints related to the master problem are directly concern the VN embedding problem, whereas in the case of the pricing problem, they concern the master problem. The objective function of a master problem is to reduce the cost of using a link for VN request. By doing so, it will ensure maximum profitability to the service provider determined in the ILP model. Cost is the unit cost of bandwidth in a link multiplied by the total quantity used by a VN request. Another cost associated with the optical network is the use of Optical Cross Connect and Wavelength Selective Switch or ROADMs. However, these devices are very

expensive, and conversion of a wavelength from Optical - Electrical- Optical (O-E-O) is expensive regarding capital investment. In addition, too many O-E-O conversions will degrade the quality of the signal. Therefore, the objective will be to select those links for hosting VN light paths where the cost for unit bandwidth is the least, as well as selecting those paths where the least number of ROADMs are present.

Constraints like available bandwidth, the total number of wavelengths, and the possible Independent Configuration (IC) govern the master problem. IC represents the complete set of combination of a resource that can suffice a VN request. The constraints of the master problem are formalized as duality constraints and employed in the pricing problem. The Pricing problem mainly deals with the generation of least cost columns based on the duality constraints. The constraints for generating columns in a pricing problem requires that no partial embedding of light paths be allowed. The reason for this is that the considered environment is an optical network, the ingress and egress path between DCs should be on the same links. Another constraint is that minimum bandwidth should be satisfied by the considered wavelength. Table 3.1 shows the relation of bandwidth to wavelength. Optical links should not be overloaded because if so, will lead to wavelength contentions. A contention is a case where two requests race toward using the same wavelength resulting in uncertain conditions. Thus to avoid such cases, a grooming factor is considered as a constraint while formulating the pricing problem.The detailed interaction between modules of SDN architecture and the optimizer are presented in the following section.

3.4 System Architecture

The proposed SDN-based VN embedding is as shown in Figs. 3.6 and 3.7. Fig. 3.6 shows an overview of the whole system. Whereas, Fig. 3.7 shows the major components involved in finding the optimal locations for the VN requests. The proposed architecture is called Virtual Network Embedding in SDN-enabled MON; it is referred to as VNE -

Table 3.1: Maximum Bandwidth that can be pushed into a given wavelength based on ITU standards for a C-band wavelengths.

Signal Type or Standard followed for Optical Data Unit	**Maximum Data Rate in Gbps**	**Example of Application**
ODU0	1.244	Transfer a stream of packets following timing transcoded signal.
ODU1	2.498	Transport two ODU0 signal.
ODU2	10.037	Transport up to 8 ODU0 streams together or a 10GBASE-W channel. Can be used to send a stream of packets from ethernet, MPLS or IP
ODU3	40.319	Transport up to 32 ODU0 or 16 ODU1 or up to four ODU2 signals.
ODU4	104.794	Transport up to 80 ODU0 or 40 ODU1 or ten ODU2 or a 100Gigabit Ethernet Signal.

MON. A brief description and interaction between the major components of VNE-MON are explained in the subsequent paragraphs.

- *Request Handler and Processed Request Queue* manages the user request and processes it to generate Information Graph (IG). This module facilitates the proposed system by providing detailed information of the VN request. It consists of DC resource and link requirement needed for VN request.

- *VN request queue* is a buffer where IG of incoming VN requests are kept on hold until the active process is complete.

- *Orchestrator* is a central processing unit responsible for managing and mapping the incoming VN requests. Orchestrator is in charge of converting the cumulative IGs to set a of flow rules and messages, which then inform the DC to block resources needed for the VN request. Once the request arrives, the Orchestrator communicates it to the *Routing Engine (RE).*

- *RE* identifies the class of QoS through *QoS Classifier* and then communicates to the *databases* for available resources in the substrate network. After the RE

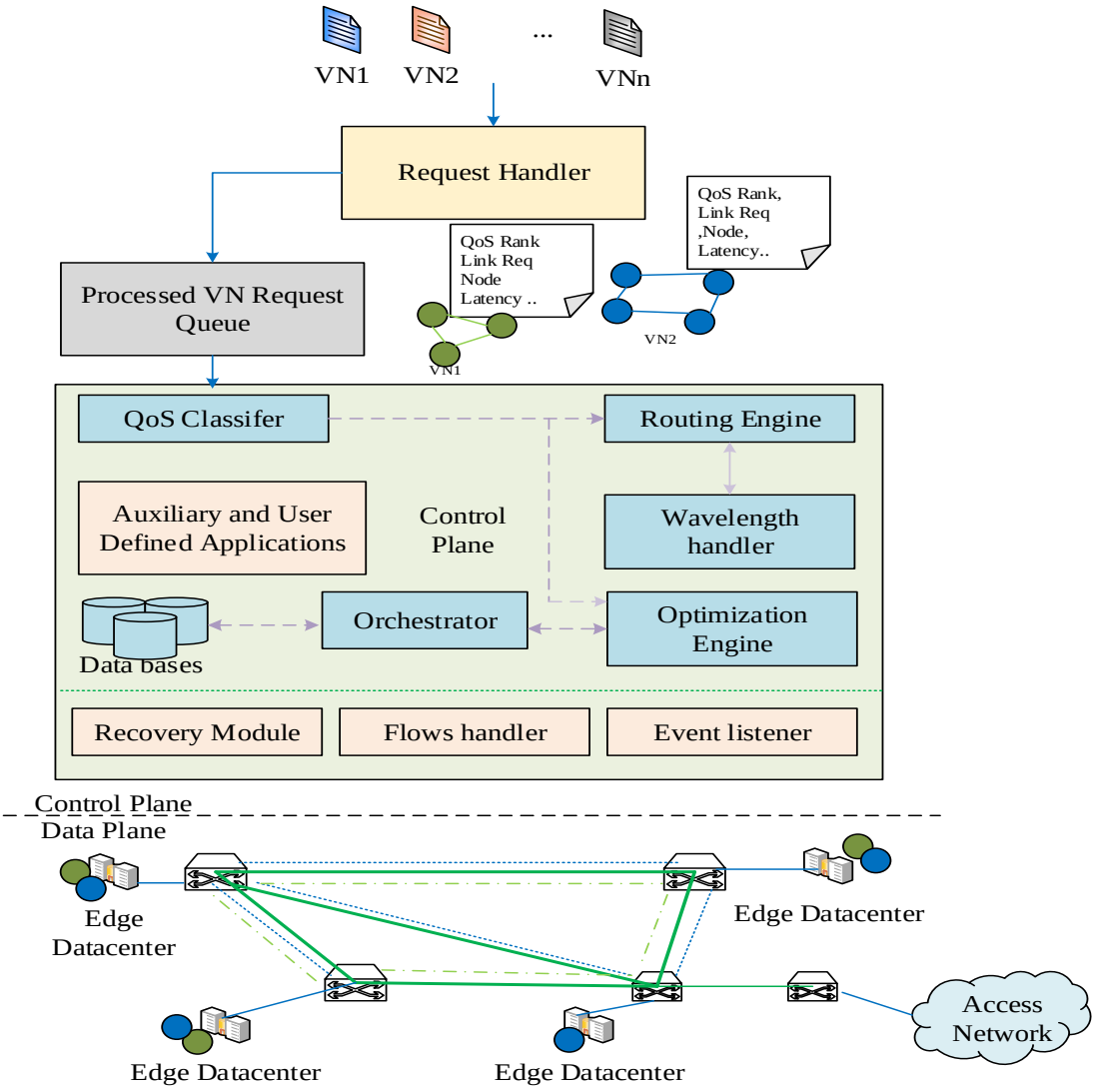

Figure 3.6: Interaction of different modules in VNE-MON

determines the paths between the VNs, it replies back with the set of paths to the Orchestrator. Upon receiving the information from RE, Orchestrator passes it to *optimizer*.

- The *Optimizer* processes the incoming VN requests and facilitates the CP to increase network utilization. The Optimizer sends back the processed information to the orchestrator. The Optimizer consists of two engines as depicted in Fig 3.7. *ILP Based Engine for Node Embedding* and *Column-Generation Based Link Embedding Engine*. The orchestrator controls these optimization engines. When a

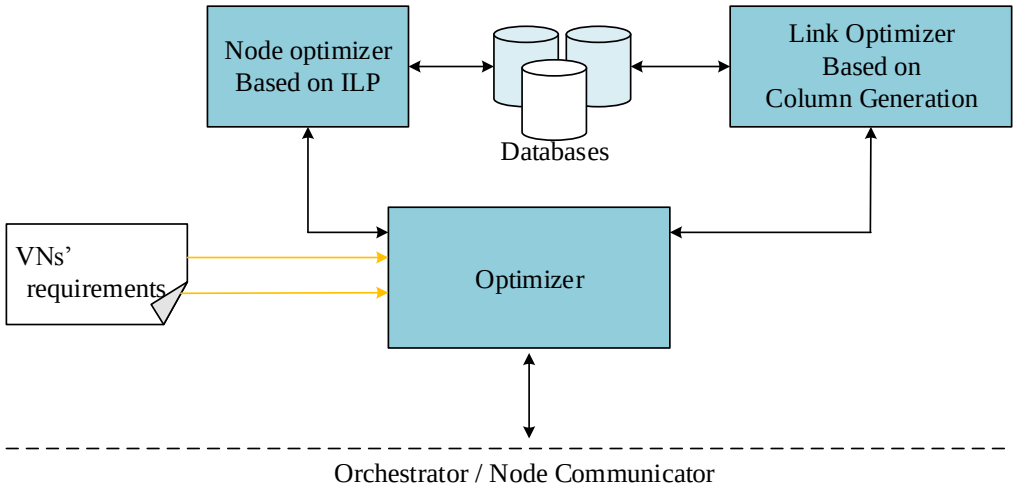

Figure 3.7: Interaction between ILP and CG for determining optimal solution

request arrives, the Orchestrator communicates with the ILP Node Engine to find the optimal location for the VN node. ILP Node Engine also provides the set of paths for each request. The information is then sent to the CG Link Engine. The CG Link Engine finds the most suitable wavelengths which can reduce the Optical-Electrical-Optical Conversions. Then the orchestrator evaluates the paths provided by ILP Node Engine and gives the most optimum among the set. The Node optimizer and the Link Optimizer are kept separated because the implementation of ILP is relatively straightforward, but the limitation is that many CPU cycles are needed to calculate the optimal solution. The ILP optimizer has to function when a new batch of requests arrives. As a result, it will not be a burden on the controller resources. The link optimizer is formulated based on the CG technique of optimization. CG module gets executed whenever the link congestion is sensed. It is of particular importance in MON because it often faces sporadic traffic bursts and needs to divert the traffic to alternate paths without losing the QoS agreement. CG approach is far more CPU efficient compared to the ILP approach [5].

- The Orchestrator communicates to the *Flow Handler* with the final set of paths and node. The Flow handler is responsible for communicating directly with the

network elements. Once the embedding is done, the controller has to keep track of these services to make sure that the QoS is not affected while the respective VNs are active.

- The *Event Listener and The Recovery module* are responsible for recovery and maintaining resiliency in the network. The proposed work has kept the implementation of these modules at a minimal and it is beyond the scope of this article to evaluate a detailed response. Nonetheless, the adapted architecture consists of segmented routing concepts available in the ONOS controller and the proposal made in [43]. This will ensure that spectrum contentions are avoided, which are common in the GMPLS-based network.

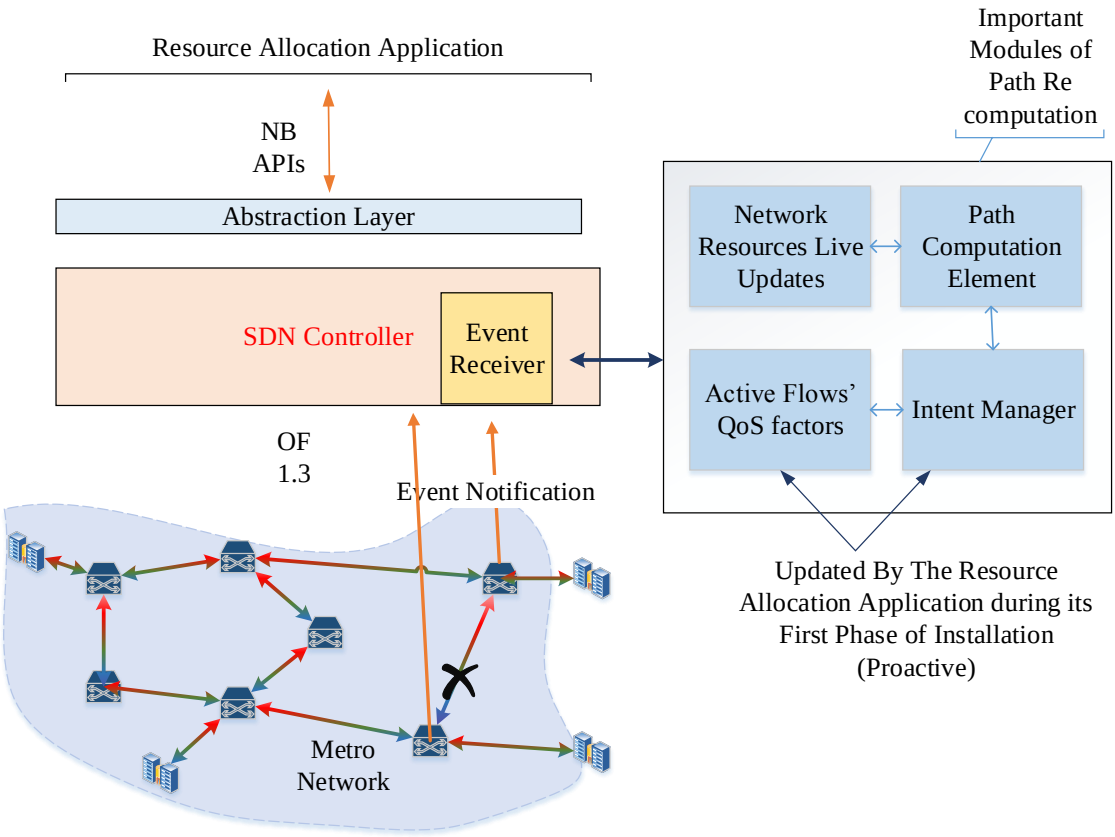

Figure 3.8: Working of Path Re-computation Module

The working of link recovery in the proposed architecture is represented in Figure 3.8.

When the event listener receives a link failure update, it communicates with the recovery module. A recovery module is associated with an intent manager, a path computation element, assigned QoS and current network available resource database. The intent manager, based on the intention of the failed link determines the next best possible path. The intents are nothing but the QoS associated with a link. Intent manager co-ordinates with the other modules present to determine the next best path.

Table 3.2: Notations Used in CG-ILP model for VN embedding

Symbol	**Definition**
H_s, H_v	Set of Substrate Nodes and Virtual Nodes.
L_s, L_v	Set of Substrate Links and Virtual Links
t	Time Period
N	Total number of requests to be embedded.
u	Substrate Node $\in H_s$
$P_u^{r/s/g/c}(t)$	Available Resources in node u at time period t.
$b_l(t)$	Bandwidth of substrate link l.
$t_c(u)$	Potential location that satisfy the QoS requirement of the request.
$p_a^{r/s/g/c}(t)$	Required Resources at node u.
$(QoS)_{a/e}$	QoS requirement for virtual node a and link e.
b_e	Bandwidth Requirement of virtual link e
f_u, f_v	Cost of using DC resources at location u and v.
c_l	Unit cost of substrate link l.
P_e^n	Total Revenue generated by use of substrate resources.
z_n	Boolean Variable, 1 if virtual network is accepted. 0 otherwise.
x_a^u	Boolean Variable, 1 if virtual node a is assigned to substrate node u.
α, β, γ	Dual variables for master problem constraints.
$\pi(sd)$	Set of links forming shortest paths between source s and destination d.

Symbol	Definition
Π_{uv}^{e}	Path between substrate node u and v which is used for virtual link e.
M_l	Mapping function of substrate virtual path to substrate path.
c	Independent Mapping configuration $\in C$.
$Cost_c$	Cost of using using c.
λ_c	Binary variable, 1 if VN request is satisfied by c.
c_{ROADM}	Cost of using ROADM.
$B^c(l)$	Bandwidth used in link l.
a_c^n	Binary variable if c satisfies the VN request $n \in N$.
x_n	Binary variable, 1 if VN request is served by configuration c.
y_u	Binary variable, 1 if ROADM is installed in location u.
x_e^l	Binary variable, 1 if virtual link e is assigned to light path l.
a_c^n	vector used to track configurations c service request.

3.5 Mathematical Model Adapted for VN Embedding

The proposed mathematical model performs optimization in two phases. The first phase uses ILP technique to determine the best set of DCs to satisfy the traffic demand, along with partially finding the many possible paths between those DCs. The second phase performs Column Generation to select wavelengths and paths between the selected nodes which can lead to a reduction in the cost for the CP provider. The following subsections explains briefly the proposed Column Generation - Integer Linear Programming (CG-ILP) model for optimization.

The ILP model was inspired by [67]. This model was extended and modified as described below:

1. Extended to include the Multimedia Requirement. Although the system considers

the model for including CPU requirement, one of the key requirement for the new class of multimedia requirement such as graphical processing unit was not included. To extend so as to satisfy multimedia requirement constraints such as graphic requirement and stringent bandwidth requirement are added.

2. The model described in [67] was a one shot solution which is good for calculating the solution mathematically. But to build as an application for SDN controller will be highly time consuming to run as and will be complex. To reduce the complexity a two phase approach was followed. Two Phase approach enabled us to run only link optimizer or node optimizer if the solution was not favorable to the network state in a SDN based scenario. Phase I followed ILP approach whereas Phase II followed a well know branch and bound technique based on column generation.

3. The use of Dense Wavelength Division Multiplexing introduced the use of new class of optical devices called Reconfigurable Add Drop Multiplexers. The constraints and the cost associated with these devices are considered rather than optical blades which were considered in the case of [67].

The Substrate network can be defined by an undirected graph:

$$K_s = (H_s, L_s) \tag{3.1}$$

This equation is characterized by a set of substrate nodes H_s and a set of links L_s.
The substrate node $u \in H_s$, can be defined by:

$$u = \{R_u, S_u, G_u, C_u\} \tag{3.2}$$

where, R_u, is the Memory, S_u is the storage capacity, G_u is the GPU, and C_u is the CPU capacity of the node u. The residual CPU, Memory, Storage and GPU capacity is given by, $P_u^{r/s/g/c}(t)$ the available memory/RAM, storage, GPU, CPU capacity at time period

t. Hence, at any given time the available node capacity can be defined by a set

$$Q_u = \{P_u^r(t), P_u^s(t), P_u^c(t), P_u^g(t)\} \tag{3.3}$$

A VN request can be represented as $K_v = (H_v, L_v)$ where, H_v represents set of virtual nodes and L_v represents directional Links. The QoS requirement of a virtual node a $\in H_v$ can be defined as:

$$(QoS)_a = (p_a^r, p_a^s, p_a^g, p_a^c) \tag{3.4}$$

$P_a^{r/s/g/c}$ represents required RAM, Storage, GPU and CPU at a selected substrate node for processing the selected VN request. Similarly, for a virtual link $e \in L_v$, can be defined by a set:

$$(QoS)_e = (b_e, d_e) \tag{3.5}$$

b_e and d_e define the minimum required bandwidth and delay for the selected virtual request respectively. x_a^u and z_n are binary variables. x_a^u is 1 if a virtual node a is assigned to substrate node u, 0 otherwise. Similarly, z_n is a binary variable which is 1 if a virtual network request is accepted, 0 otherwise.

Objective Function : To increase the revenue by maintaining QoS requirement for VN requests, i.e., maximizing profit. Eq (6) defines the objective function.

$$f_{objective} = \sum_{n \in N} \left\{ \sum_{e \in L_v; e=(sd)} P_e^n * z_n - \sum_{(u,v) \in H_s * H_s} x_s^u * x_d^v * (f_u + f_v + \sum_{l \in \pi(uv)} c_l * b_e) \right\} \tag{3.6}$$

Where, $P_e^n * z_n$ is the total revenue generated by use of substrate resources, $\pi(uv)$ is the shortest path for (e = sd) between source u and destination v, c_l is the unit cost of using a substrate link l and $f_u + f_v$ is the cost of using DC resources at u and v. The calculation of revenue is based on the standard bidding approach followed in cloud resource cost computing.

The Resource allocation of VN request on a DC is governed by the following constraints:

1. All accepted virtual requests should be embedded on the substrate network, which is given by

$$z_n \leqslant \sum_{(u,v)\in H_s * H_s} x_s^u * x_d^v; (sd) = e \in L_s \tag{3.7}$$

2. No more than one virtual node for a given VN request can be assigned to the same substrate node. $t_c(u)$ is a subset of H_s, represents the potential locations that can satisfy the VN's DC requirement. At any given period of time t, DC requirement to be satisfied for a given virtual node a are governed by the Eqs. (3.8) - (3.13).

   (a) No overlapping of embedding

$$x_a^u = 0; \forall u \in H_s, u \notin t_c(u), \forall a \in H_v, n \in N \tag{3.8}$$

$$\sum_{u\in t_c(u)} x_s^u \leq z_n; s \in H_v, n \in N \tag{3.9}$$

   (b) Available CPU capacity on a selected substrate node u should be more than the requested for the virtual node a.

$$\sum_{n\in N} \sum_{a\in H_v} x_a^u * p_a^c \leqslant p_u^c(t); u \in H_s \tag{3.10}$$

   (c) Available GPU capacity on a selected substrate node u should be more than the requested for the virtual node a.

$$\sum_{n\in N} \sum_{a\in H_v} x_a^u * p_a^g \leqslant p_u^g(t); u \in H_s \tag{3.11}$$

   (d) Available storage capacity on a selected substrate node u should be more than

the requested for the virtual node a.

$$\sum_{n \in N} \sum_{a \in H_v} x_a^u * p_a^s \leqslant p_u^s(t); u \in H_s \tag{3.12}$$

(e) The Available memory on a selected substrate node u should be more than the requested for the virtual node a.

$$\sum_{n \in N} \sum_{a \in H_v} x_a^u * p_a^r \leqslant p_u^r(t); u \in H_s \tag{3.13}$$

Light path embedding for the virtual request using Column Generation Technique

CG formulation for link embedding is based on our previous work [68]. CG technique is used as it facilitates in reducing the CPU cycles for optimization. Mapping of VN link based on CG approach can be defined as:

$$M_l : L_v \mapsto \Pi_{uv}^e \tag{3.14}$$

Where, Π_{uv} is the shortest path for the virtual link $e = (sd) \in L_v$ between substrate source u and destination v. The $\Pi_{uv} \in \pi(uv)$ where $\pi(uv)$ is calculated using K-shortest path algorithm [69]. CG approach is decomposed to master problem and pricing problem. The formulation of CG is discussed in subsequent Sections.

Master Problem:

Objective Function is to reduce the cost of link embedding and to maintain the QoS.

$$Min \sum_{c \in C} (Cost_c * \lambda_c) \tag{3.15}$$

Where, $c \in C$ is defined as independent mapping configuration for each VN request. It consists of set of substrate resources and links ensuring that each mapping of a VN request is independent of each other in a WDM based optical Network. Set of c serving the VN requests is defined by a vector a_c^n. It is 1, if configuration c serves the multimedia request n. λ_c is a binary variable for master problem, it is 1 if configuration used to satisfy the VN request, 0 otherwise. $Cost_c$ defines the unit cost of configuration c:

$$Cost_c = \sum_{u \in H_s} T^c(u) * c_{ROADM} + \sum_{l \in L_S} B^c(l) * c_l(b) \tag{3.16}$$

Where, c_{ROADM} is the unit cost of using a ROADM, $c_l(b)$ is the unit cost of bandwidth, $T_c(u)$ is the total number of ROADMs used in selected configuration c and $B^c(l)$ is the used bandwidth by the link l in configuration c.

Master problem is governed by the following constraints:

$$\sum_{c \in C} \lambda_c \leqslant W \qquad (\alpha) \tag{3.17}$$

$$\sum_{c \in C} \lambda_c * T_c(u) \leqslant N_{ROADMs}(u); u \in H_s \qquad (\mu) \tag{3.18}$$

$$\sum_{c \in C} \lambda_c * a_c^n \geqslant 1; n \in N. \qquad (\beta) \tag{3.19}$$

Eq. (3.17) defines that configuration used by a VN request should be unique. The maximum number of configurations supported by the system is equal to total number of wavelengths W. Eq. (3.18) assures that the signal quality is not degraded due to Optical - Electrical - Optical conversions. Eq. (3.19) ensures that a maximum of one configuration can be used for mapping of each VN request.

Pricing Problem:

It facilitates in generating additional columns for the constraint matrix of master prob-

lem. Pricing problem deals with constraints related to embedding of VN request path. Consider α, μ and β as dual variables for the master problem constraints (Eqs. (3.17) - (3.19)). Then *objective function* can be defined as:

$$\overline{Cost_c} = Cost_c + \alpha_o + \sum_{u \in H_s} \mu_u * T_c(u) - \sum_{n \in N} a_c^n * \beta_u \tag{3.20}$$

The relation between master and pricing problem is defined by the following equations:

$$x_n = a_c^n \tag{3.21}$$

$$T_c(u) = 2 * y_u \tag{3.22}$$

Therefore, $\overline{cost}$ can be defined as:

$$\overline{cost} = \sum_{u \in H_s} 2 * y_u * c_{ROADM} + \sum_{l \in Ls} B^c(l) * C_e(b) + \alpha_o + \sum_{u \in H_s} \mu_u * 2 * y_u - \sum_{n \in N} x_c * \beta_u \tag{3.23}$$

Where, x_n, y_u and x_e^l are the binary decision variables for pricing problem. x_n is 1 if VN request n is served by configuration c, 0 otherwise. Similarly, y_u is 1 if ROADM is installed in the node u, 0 otherwise. x_e^l is 1 if virtual link e is assigned to substrate link l, 0 otherwise.

Pricing problem is governed by following Constraints:

1. All the virtual links have to be accepted for a given VN request, otherwise VN request will be rejected. This is defined as:

$$x_n \leq \sum_{l \in L_s} x_e^l; e \in L_v; e \in L_v^n;\ n \in N \tag{3.24}$$

2. The assigned substrate link should satisfy the bandwidth and wavelength require-

ment of a given VN path.

$$\sum_{n \in N} \sum_{e \in L_v^n} \sum_{l \in L_s} x_e^l * b_e \leq B_l; l \in L_s \tag{3.25}$$

Where, B is the bandwidth capacity of the light path.

3. Candidate Light path:

$$L_e = l \in L, N_l \leq N_e; e \in L_v, n \in N \tag{3.26}$$

Where, L_e defines the set of feasible light paths, such that no link among used light path exceeds the maximum allowed link length. As well, the overall path latency is not more than the allowed limit.

4. Grooming factor (G) is defined as the total amount of bandwidth that can be pushed into a given wavelength. It is dependent on Optical Transport Unit (OTU) scheme used. For the sake of simulation, the grooming factor is kept constant. OTU-1 is used for each VN request and each substrate link has a capacity defined by OTU-3 standards. This makes a grooming factor equal to16. But the grooming factor can be adjusted by Eq. (3.27).

$$\sum_{n \in N} x_n \leq G \tag{3.27}$$

A reference table for the symbols used in the formulation of CG-ILP is given in table 3.2.

3.6 Summary

This chapter proposed a novel technique for VN embedding in a Metro Optical Network interconnected by Optical links. Methodologies for selecting different optimization tech-

niques and the constraints on relevant problem are presented. The fact that Column Generation is one of the efficient techniques for solving an NP-hard problem, helped direct the research methodology in this chapter towards adapting the technique for solving lambda embedding. Integer Linear Programming is used because of its simplicity in solving a problem. ILP is used to embed VN request to edge DC. Multimedia request requirement was considered while formulating the ILP model so as to guarantee the QoS. Each combination of wavelength and grooming capacity is represented as an Independent Configuration. As a result, the Independent Configuration ensured that wavelength contentions did not occur while embedding a request. In addition, the use of SDN and other protocols along with SDN's interaction with the optimizer is proposed in this chapter.

The use of SDN framework to perform VN embedding on substrate network, posed another question on the placement of controller. This consideration on controller placement is to ensure that agreed QoS for the accepted VN requests is not affected due to contingencies in the network. In other words, determine the placement of controller so as to enhance the performance of Data plane. Depending on the type of services offered by the metro network what will be the best possible location in a metro network? To evaluate and determine the solution for this question we proposed several strategies in the next chapter. The following chapter proposes ILP formulation for determining the optimal location for the SDN controller in a metro network.

4 Controller Location for Metro Optical Network

4.1 Objective

The SDN architecture proposes the separation of the control plane and the data plane. This SDN architecture leads to many concerns regarding its reliability, scalability and performance compared to a traditionally distributed network. In this chapter, the consequences of controller's placement on the performance of VN requests in a metro network is investigated. The investigation is carried out by looking at the current research proposals for the optimal number of controller required for the Metro Network. Furthermore, several two-phase strategies are proposed to determine the optimal controller location for different performance parameters in a Metro Network. The proposed formulation considers the possibilities of contingencies such as unexpected high delay and inadequacy of resources during a traffic burst in the metro network.

4.2 Related Proposals to Determine Optimal Controller Location

It is a well-known limitation of SDN architecture that a non-optimal physical location of the controller in a network can lead to the inefficient performance of the network. [50] addressed the scenario of controller placement to improve the overall network resiliency.

Authors in [50] proposed several split architectures for determining the optimal number of controller required for a given network. A Split architecture-based approach uses the least number of the controller for a given topology, while maintaining the reliability of the data plane. Their main objective was to provide resiliency at any cost. [51] made an in-depth evaluation of the controller placement on the wide area network. They also proposed an approach for determining the controller location based on worst case and average case latency. The authors define latency as the time taken by a signal for a round trip from the controller to the switch. Based on their evaluation, it was proven that to satisfy most of the network requirements a single controller is sufficient.

[52], [53] and [54] have proposed a dynamic location for the distributed controllers in a large network. Authors in [53] considered the placement problem as a warehouse location problem. The location is decided based on many factors such as the current traffic demand vs. the amount of load on the controllers. The controller is capable of cloning itself based on the needs of the network demand to maintain stability. They are all based on heuristic approaches as having an Integer Linear Programming or any other optimization for dynamically recalculating the location will be highly time-consuming. Some of the researchers proposed optimization techniques to determine the controller location based on standard combinatorial formulation such as ILP. [55] proposed a mathematical model to determine the least number of controllers required that can satisfy the network requirements. The controller locations are selected such that the cost for communication between controller to controller and controllers to switches is minimum. Also, it ensures that the cost of hosting a controller is least. The authors conclude that as the network size grew, due to the many constraints considered the system could not determine the controller location in deterministic time. [56] proposes an optimization technique to determine the k number of controllers such that it reduces the effect of propagation delay, increases network reliability and improves load distribution. In other words, the authors proposed a model to improve the QoS between the control plane to data plane communication. Both the previous approaches focused on distributed control plane. By using

different graph theory concepts, authors in [57] and [58] have proposed a robust and resilient location for controllers. [59] disregards the idea of following the warehouse location optimization technique as the proposals made in those fields are highly theoretical and cannot be used for SDN controller placement. The authors proposed a Pareto Optimal concept for the formulation of their model. The proposed model was solved using the heuristic approach. The solution and constraints considered are for the controller placement in a large scale core network.

In the light of the above proposals, it is clear that most of the proposals are focused towards the wide area network. Also, the proposal for determining a single controller location is based on brute forcing without considering the effect of dynamic change in network resources. Another observation is that the effect of the controller location on the accepted VNs has not been completely evaluated. Depending on the type of service offered by metro network the controller has a specific requirement such as the processing capacity, latency requirement and resiliency requirement to name a few. The impact of controller placement and the type of traffic in the metro network is studied in this chapter. The proposal in this chapter attempts to close this research gap.

4.3 Optimal Number of Controller for a Metro Optical Network

[51] has done an in-depth evaluation on the number of controllers required for a given topology. In conclusion, the authors proved that in most cases, one controller could suffice the network requirement. Similarly [53, 54] and [59] have advocated that a single controller will be sufficient for a smaller network unless the targeted objective is to have maximum resilience at any cost in the system. However, the disadvantage of having multiple controllers for the metro network is that it will increase the unnecessary inter-controller communication leading to excess usage of links. This is highly unfavorable for a small size metro network. Also, it will lead to challenges associated with designing

efficient frameworks for proper data synchronization and data authenticity among the controllers.

Therefore, for a metro network, it is not advisable to have multiple controllers as it will incur a high cost. An assumption that the controller is inside an Edge DC is made to assure resiliency for the control plane. These DCs are known to be highly available, and as a matter of the fact, they are known to have the least availability of 99.9 %, and the maximum of 99.99% [70]. Based on the above mentioned facts, further investigation is focused on determining a single optimal controller location.

4.4 Importance of Controller Location for Metro Optical Network

Metro network is highly dynamic in nature and time independent constraints govern them. As a result, the carrier Service Provider should be inclined to expect the unexpected contingencies. If the controller is not sufficiently agile and efficient, it can bring down the performance of the whole system.

The proposal for using the SDN framework to enhance the utilization and to enable agility in the network requires dealing with certain challenges. The most significant of these challenges for a given topology involves deciding where to place the controller [59]. Other challenges include the overhead associated with the use of OpenFlow and security between the controller and the switch. As well, depending on the type of SDN framework utilized for the implementation (i.e., hierarchical or vertical distributed) dealing with inter-controller communication can also be problematic.

In viewing a standard topology, taken from the topology zoo, as shown in Figure 4.1, the highlighted locations are the optimal location for one of the parameters. To better understand the importance of the controller location, the following examples illustrate how the network is dependent on the location attributes.

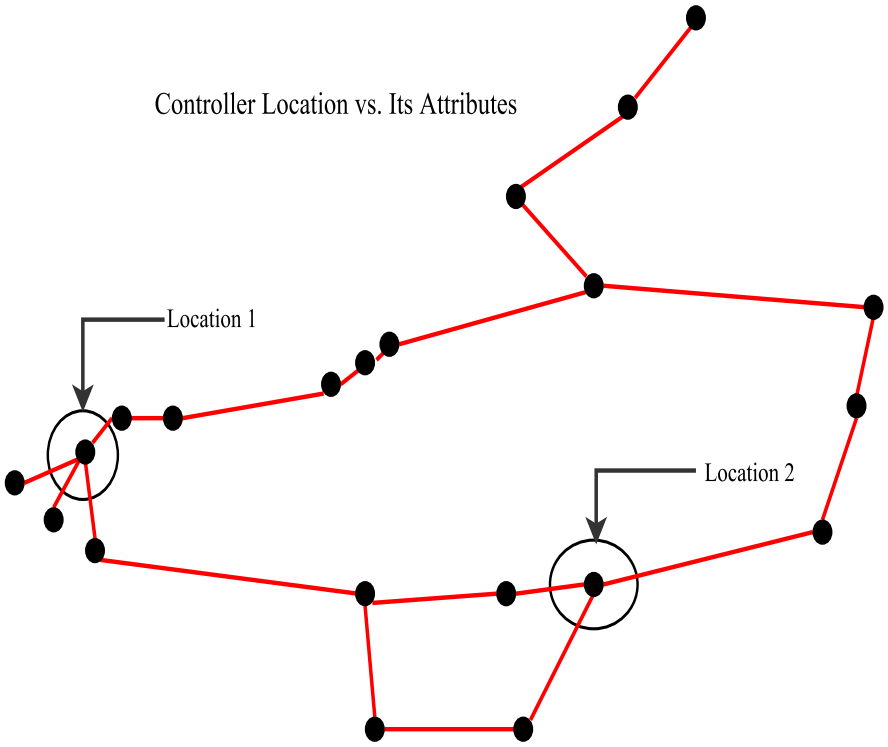

Figure 4.1: Variation of Controller Performance depending on the attributes location

Location 1: This is determined by the switch which has maximum ingress and egress links. Most of the time for a topology it will be near an edge router. Selecting this location for hosting the controller will ensure maximum reliability from the link failure point of view. Since it is on the brink of the network, it will have a maximum number of flows comparatively. On the contrary, the location fails to cater to the needs of switches present at the other end of the network, particularly during a traffic burst. It is difficult to prognosticate which part of the network will be responsible for the traffic burst at a given time. Thus this location is not optimal for all the scenarios.

Location 2: This location represents the place calculated based on the centrality of the node. The centrality considers link latency when ranking the nodes. The nodes determined by following the ranked based approach will ensure that the network experiences the least latency. Although the considered parameter is average latency, it is difficult to calculate the expected latency of the system. Also, if only the worst or the least

latencies are considered, even then it is like disregarding the better locations for hosting the controller. Due to the unexpected behavior of latency, it is not advisable to make a decision purely out of link latency. These examples serve as models for investigating and formulating the algorithms to determine the controller location in an MON.

Running sophisticated algorithms for dynamic VN embedding into DC requires the controller to be agile and have enough resources to host multiple VN virtual controllers for accepted requests. In such a sensitive scenario, placing the controller without any evaluation of the topology can lead to the sub-optimal performance of the network. In the case of MON, it is not the link latency which matters but the response time associated with each switch. The constant sporadic traffic bursts associated with a metro network traffic makes a controller's response time vital. Network resource availability can affect the performance of the controller. If sporadic burst traffic is not considered during the placement of the controller it can lead to contingencies like unusual delay or a high response time in the network. These factors determine the optimal controller location on the network.

4.5 Overview of the Approach Adapted for Determining Controller Location.

Figure 4.2 shows the methodology considered for determining the controller location. The process of locating the controller involves analyzing the network topology and its use. Topology Analysis can be broadly classified into three types: Firstly, heterogeneity of Network; Metro network is placed in between Access Network and Core Network (also known as WAN). Access Networks are composed of diverse switching technologies and end nodes. Whereas, the WAN is composed of either optical or IP. It is the role of Metro Network to seamlessly aggregate the Access network to WAN. Consideration should be given to the type of Access Network and the WAN the metro network is connected to, in the formulation of Controller Location determination. This ensures that if in the future

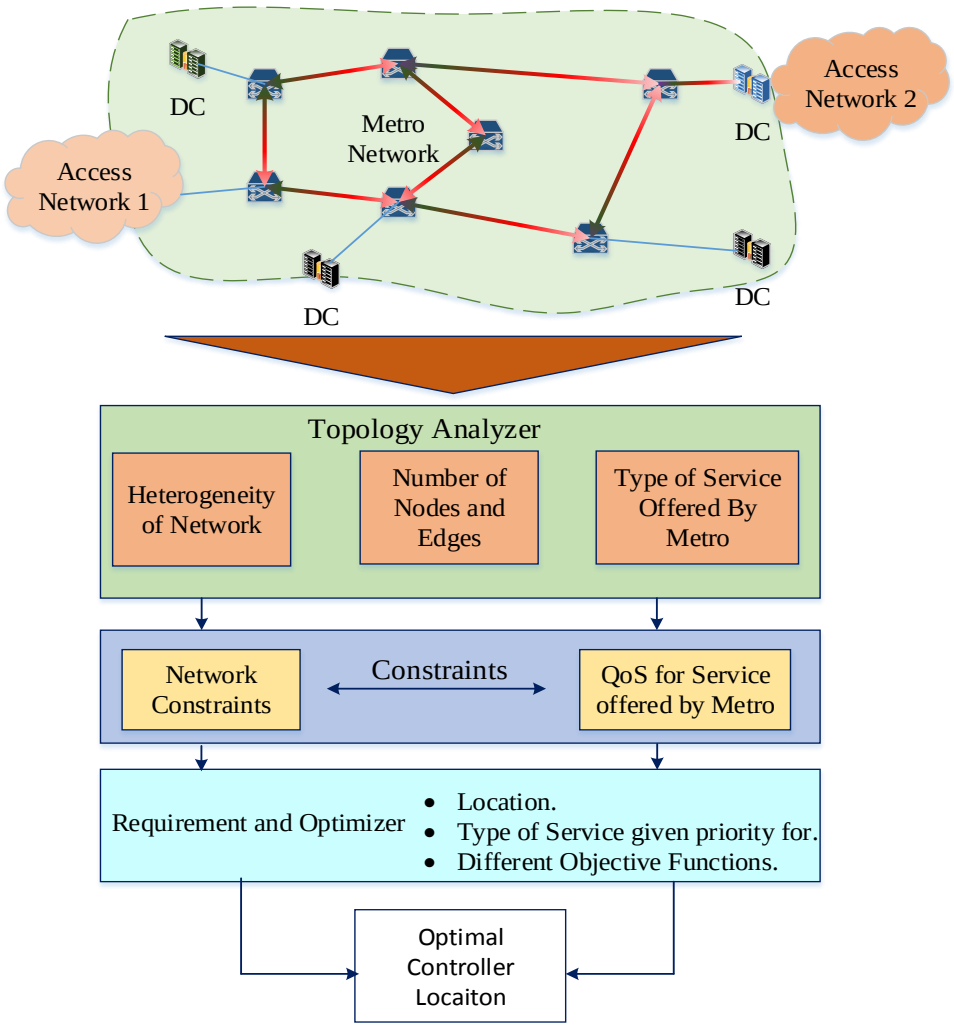

Figure 4.2: Overview of Controller Location Optimization Model

the controller needs to be scaled or integrated with the WAN or the Access controller, this can be done seamlessly. Secondly, attention should be given to the density of network elements i.e., number of Nodes and Edges. This density of network elements governs the maximum processing the controller can perform. The Number of flow rules the controller can concurrently process and the time taken to reinstall a corrupted flow are some of the factors depend on the number of nodes and edges. Thirdly, analysis should include the type of services offered by the Metro network. The type of service includes: real-time Multimedia applications, Ethernet services, and Virtual Network services, to name a few. The type of Service offered by a Metro network is important to consider because

the controller should be capable of satisfying all the services' requirements.

Once the network infrastructure is analyzed, the constraints which might affect the controller performance should be addressed. Constraints for determining the location of the controller can be classified as Network-related and QoS related. Network related includes links attributes whereas, QoS related includes DC and processing attributes. Once all the information is available, based on the formulation presented, the optimal controller location is determined. The considered constraints and their effect on the controller performance is evaluated in the subsequent section.

4.6 Constraints Responsible for Optimal Performance of Controller

For a case of WAN, the latency of communication between the controller and the switch is the most prominent factor, whereas the unexpected changing in the network traffic is considered to be a most significant constraint on MON. It is thus not possible to justify the optimality of the location based on one constraint. The sporadic traffic bursts can cause sudden overloading on links, processors and switch buffers. It is necessary to consider the availability of resources which can reduce such cases is taken into account when determining the optimal location.

Bandwidth: According to the OpenFlow, the standard size of each packet is, on average, equal to 160KB. Based on this fact it is possible to expect the required bandwidth during the worst case scenario.

Latency: To avoid unexpected behavior of the network, it is vital to be aware of the worst case latency between the control plane and the switch.

Resiliency: It is important that the considered node for hosting the controller has a sufficient resiliency on the node as well as with path. That is selected node to host the controller to the switch should have at least one additional unique backup path. This additional path from the controller to the node ensures that the network will be

resilient even during the worst case scenarios. Shared Risk Link Factor (SRLG) is used to measure resiliency of a location.

Node Weight: The Node weight is the magnitude calculated by the multiplying centrality of a node, based on latency and the used flow table size. This weight ensures that the controller is placed near to the location which are active and comparatively more prominent in the network. It also implies new requests are generated near to this location and requires constant contact with the controller to make informed decisions. A detailed discussion of the proposed mathematical model is discussed in the subsequent section.

4.7 Mathematical Model for Optimal Controller Location

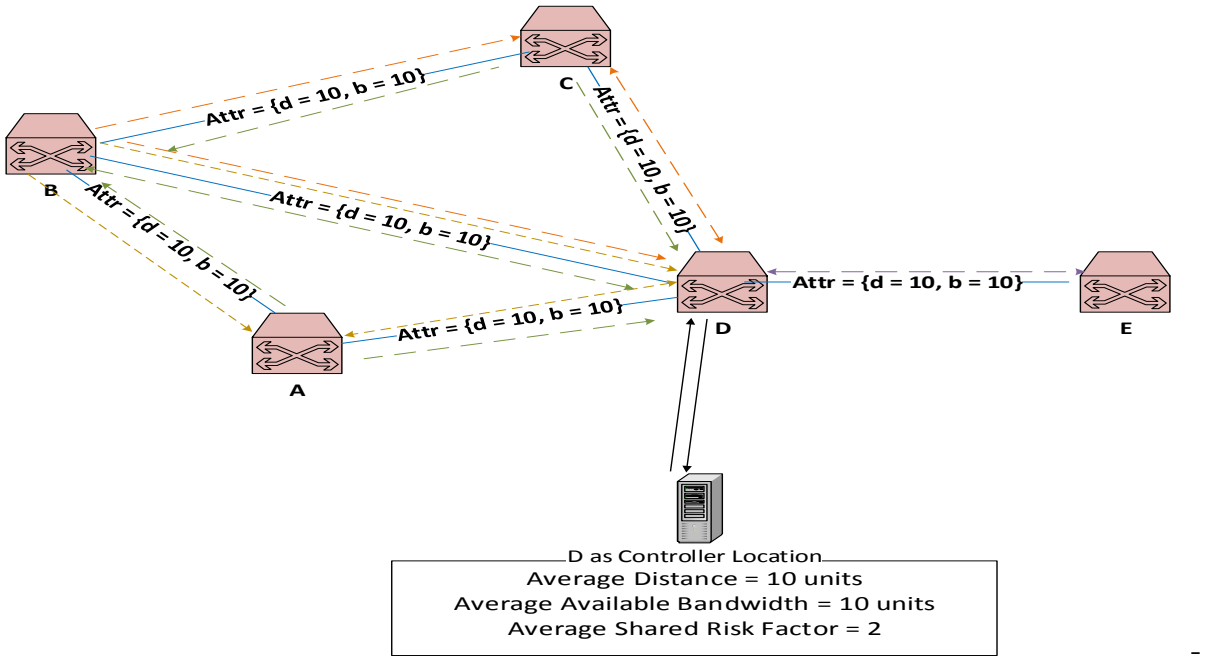

Figure 4.3: Calculation of Parameters used for the Determination of Controller Location

Table 4.1: Notations Used in ILP based Optimization of Controller Placement

Symbol	Defination
C	Subset of H_s potential location for the controllers.
u, l, c	representation of active element in set H_s, L_s and C respectively.
z_c	Binary variable. 1 if node u can satisfy controller requirement, 0 otherwise.
α_l	Unit cost for using link l.
R_u	Maximum tolerable response time for node u.
R_{cu}	overall Response time taken for the communication between controller and node u during an event.
Π_{cu}	shortest path from controller c to node u.
π	A single path in the set Π_{cu}
$cost_{cu}$	cost for communication from location c to node u.
O_c^u	cost of using node u for controller c.
δ_π^l	Binary parameter. 1 if link l is used by path π, 0 otherwise.
Π_{cu}	set of K paths from controller c to node u.
B_l	Available Bandwidth at link l.
N_{max}^{cu}	Maximum hops between controller to node to have efficient communication.
$N(\pi)$	Total number of intermediate nodes in the path π.
$\Phi(\pi_{cu})$	Total number of paths from controller c to node u.
b_{cu}	Bandwidth needed for the communication between controller c to switch u.
$Cost_{cu}$	Cost for communication from node c to u.
F^c	Average disjoint paths from controller c to all other nodes in the network in set π_{cu}.
$Cost_c$	Cost for placing a controller at location c.
CPU_c	CPU needed for the controller c.
RAM_c	RAM needed for controller c.
CPU_O^u	CPU that a node u can spare at a worst case scenario.
RAM_O^u	RAM / Fast Access Memory that a node u can spare at a worst case scenario.
y_c	Binary variable. 1 if controller from the phase I is selected as the final controller location.

Calculation of Performance Parameters Considered in Determination of Controller Location.

Several standard parameters that were modified and adapted to determine magnitude-metrics for a selected node are explained in this section. The parameters used in the determination of controller location and the methodology used in their calculations are

Table 4.2: Continuation: Notations Used in ILP based Optimization of Controller Placement

Symbol	Defination
$D(lat, lon)_{cu}$	Function that calculates the distance between two points on the surface of earth.
L_m	Average of Maximum response time of the path Π_{cu}
γ_{cu}	Paths $\in \pi_{cu}$ that satisfy the QoS requirement between the controller and node.
I_c	Katz centrality factor based on the node weight of flow table utilization.
L_m	Average of Maximum response time of the path Π_{cu}
γ_{cu}	Paths $\in \pi_{cu}$ that satisfy the QoS requirement between the controller and node.
L_m	Average of Maximum response time of the path Π_{cu}
γ_{cu}	Paths $\in \pi_{cu}$ that satisfy the QoS requirement between the controller and node.
I_c	Katz centrality factor based on the node weight of flow table utilization.

shown in Figure 4.3. For the sake of explanation, we consider the location 'D' as node under study. The metrics such as determine the link redundancy using Shared Risk Link Factor (SRLG) factor, determine the importance of node using Katz centrality are explained in the subsequent paragraphs.

Calculation of Shared Risk Link Factor:

SRLG technique is used as a metric to quantify the resiliency and to determine the link redundancy from the selected node for controller to all the other nodes in the network. This is calculated by taking the average number of independent paths from location 'D' to all the other nodes. In other words, by how many unique paths the selected location can communicate to the other node. In the case of 'D', it can communicate to 'E' by one unique path. From 'D' to 'A' can be communicated by two paths they are: D $\rightarrow$ A and D $\rightarrow$ C$\rightarrow$ B $\rightarrow$ A. Similarly, the number of unique paths from location 'D' to all the other nodes is calculated. The average the number of unique paths divided by the number of nodes, excluding the one in study.

Calculation of Average Bandwidth:

The bandwidth calculation depends on the path selected for communication. The bandwidth form the location under study (in this case 'D') to all the other nodes (i.e., 'A', 'B', 'C', and 'E') are calculated and the average is taken.

Average Distance:

The considered distance is a geodetic distance. That is the distance between two points on the surface of earth. The average of the distance is taken by adding the distance from the node under study (in this case 'D') to all the other nodes, divided by the total number of nodes, excluding the one in study.

The formulation of two-phased technique for solving the controller location problem is discussed in the next section.

Centrality based on prominence of the node in the network:

The prominence of node is calculated based on its relevance in the network. That is by determining what happens if we take out the node from the network? To do so we used the concept of Katz centrality. Katz centrality was modified to calculate the prominence of the selected location based on number of flows available in that node.

4.7.1 Two-Phased Approach for Solving Controller Location

The ILP model is formulated to determine the controller location and two-phase approach is used to determine controller location. The use of two-phase approach is followed because it aids in minimizing the search space in determination of final optimal location for the controller. Phase I deals with the minimization of cost for placing a controller. It provides a set of nodes as a potential location for the controller in a given network. Since the objective of the proposal is to study the effect of controller on the accepted VN requests and, to determine what will be the best location for a given type of VN

requests? Strategies to enhance a single objective are proposed. Which is done in Phase II. Phase II is assisted by the Phase I output. Phase II determines the optimal controller location for a given performance parameter. Fig 4.3 shows the calculation of different metrics used for the evaluation of the controller locations.

4.7.2 Phase I - Cost Based Optimization of Controller Location

$$Minimize \sum_{c \in C} z_c * \left(Cost_c + \sum_{u \in H} Cost_{cu} \right) \tag{4.1}$$

Where $Cost_{cu}$, is the cost of communication from controller location c to switch location u and $Cost_c$ is the cost of controller location. This can be defined as:

$$Cost_{cu} = \sum_{\pi \in \Pi_{cu}} \sum_{l \in \pi} \alpha_l * b_{cu}, \ \forall c \in C, \forall u \in H \tag{4.2}$$

$$cost_c = O_c^u * z_c, \ \forall u \in H, \ \forall c \in C \tag{4.3}$$

The constraints governing the controller location problem are:

Response Time:

It is defined as the time taken for an event generated at the data plane to receive an update from the control plane. This constraint depends on type of controller used, link bandwidth, switching latency and signal conversion latency.

$$z_c * R_{cu} \leq R_{max}^u, \ \forall u \in H, \ \forall c \in C \tag{4.4}$$

Where R_{cu} is the response time from location selected controller c to switch u. R_{max}^u is the maximum tolerable response time for the switch u.

Maximum Hops:

The hop constraint is vital in case of MON because, noticeable amount of time is taken by signal during O-E-O conversions. This is defined in Eq. (4.5).

$$z_c * \frac{\sum_{(\pi \in \Pi_{cu})} N(\pi)}{\phi(\Pi_{cu})} \leq N_{max}^{cu}, \quad \forall u \in H,\ \forall c \in C \quad (4.5)$$

Where $\phi(\Pi_{cu})$ gives the total k possible paths from controller c to node u or total paths in set Π_{cu}. N_{max}^{cu} is the maximum intermediate nodes between controller and node u to maintain QoS. And $N(\pi)$ is the total number of intermediate nodes in path π.

Contingency:

It is defined as failure or delay in communication between controller and switch which can occur due to the shortage of network resources. Selecting paths and nodes which have sufficient available resources can minimize the effect of network contingencies. It assures that the network is stable even during the worst scenarios. The network contingencies are prominent during the traffic bursts leading to bottlenecks at the links or the node. Some of the resources which can reduce contingencies are sufficient amount of bandwidth and processing power of the selected node for the controller. The formulation of resources requirement which can minimize the contingencies are defined below:

Bandwidth Available: Higher the bandwidth, lessen the chances of getting delayed during the peak hours. The selected path for communication between the controller and the switch should satisfy minimum bandwidth requirement. The available bandwidth in the selected link should be more than the required bandwidth for the node to controller

communication.

$$z_c * \sum_{u \in H} \delta_\pi^l * b_{cu} \leq B_l, \ \forall l \in L : l \in \pi, \pi \in \Pi_{cu}, \forall c \in C \tag{4.6}$$

Where, δ_π^l is a binary parameter, 1 if $\pi \in \Pi_{cu}$ uses link l for the communication between controller c and node u, 0 otherwise. The minimum bandwidth required for controller to node communication is represented by b_{cu}. B_l is the available bandwidth in link l.

Resource Requirement at the node: The resource needed for optimal performance of controller are Memory and CPU. Available resources at the node should be more than the resources needed for seamless performance of the controller. These are defined in Eq. 4.7 and Eq. 4.8.

$$CPU_c \ * z_c \leq RAM_O^u, \ \forall u \in H, \ \forall c \in C \tag{4.7}$$

$$RAM_c \ * z_c \leq CPU_O^u, \ \forall u \in H \ \forall c \in C \tag{4.8}$$

Where, CPU_u and RAM_c are the CPU and RAM needed for controller c for its optimal working respectively. RAM_O^u and CPU_O^u are the maximum RAM and CPU the node u can spare during sporadic burst events.

Maximum Number of Controllers:

This constraint can be relaxed if the number of controllers given as an output are not equal to the number of nodes initially present. If all the nodes can satisfy all the constraints, then it is good to select only best of the nodes. Then the value m will limit the number of controller locations.

$$\sum_{c \in C} z_c = m \tag{4.9}$$

4.7.3 Phase II - Potential Location for the Controller

Phase II is implemented for optimizing the controller to node communication for different performance parameter. The different strategies proposed for different performance parameters are:

1. Select the controller location by minimization of distance between controller to data plane. Distance considered is a geodetic distance. (Case 1).
2. Select location based on average maximum response time and distance. (Case 2).
3. Select the location for the most reliable and least Response time for controller to data plane. (Case 3).
4. Select the location near to a switch which has maximum utilized flow table. (Case 4).
5. Select the location near the switches which has maximum utilized flow tables and has the least response time. (Case 5).

A binary decision variable to represent the feasibility of the location called y_c is used. y_c is 1 if controller location c $\in$ C is selected for the placement of the controller, 0 otherwise. Formulation of the strategies based on different performance parameters are described in the subsequent sections:

1. **Select controller location by minimization of distance between controller to data plane:**

   If $D(lat, lon)_{cu}$ gives a geodetic distance between two points on the surface of the earth. *lat* represents latitude of two points, whereas *log* represents the longitude. Distance is calculated using Eq. (4.11).

   Formulation: Objective function is defined as in (4.10).

$$min \sum_{c \in C} \sum_{u \in H} y_c * D(lat, lon)_{cu} \tag{4.10}$$

Where

$$D(lat, lon)_{cu} = R_{earth}\sqrt{(\Delta lat)^2 + (cos(\phi_m) * \Delta lon)^2} \tag{4.11}$$

Where, Δ represents a function which gives the difference of two values. In this case Δlat gives the difference of latitude between controller location and switch. ϕ_m is the mean latitude of location c and u and, R_{earth} is the radius of earth.

2. **Select location based on average maximum response time and distance:** Most of the time, the least distance cannot provide the least latent paths, hence forth the location needs to be optimized such that it does not affect the QoS of data plane to control plane.

   Formulation: The Objective function is defined as in Eq 4.12.

$$min \sum_{c \in C} y_c * \left(\frac{\sum_{u \in H} R_{cu}}{N-1} + \sum_{u \in H} D(lat, lon)_{cu} \right) \tag{4.12}$$

   Where, R_{cu} is the response time for path Π_{cu}. N is the total number of nodes available in the topology or in the set H. -1 in denominator is to exclude the node under study. $\Pi_{cu} \in \gamma_{cu} \subseteq \Pi_{cu}$. γ_{cu} are the set of paths which satisfy the QoS of control plane to data plane.

3. **Select location for the most reliable and least response time for controller to data plane.**

   Strategy to select controller location which minimizes the response time of the controller to switch communication is not always optimal. It does not guarantee the resilience from the link failures or node failures. To circumvent this limitation, a location which can reduce response and provide resilience during failure is selected.

Formulation: The Objective function is defined as in Eq. 4.13

$$\sum_{c \in C} \frac{y_c}{F^c} * \left(\sum_{u \in H} L_m(\Pi_{cu}) \right) \tag{4.13}$$

Where, L_{Max} and F^c is the average of maximum response time of the selected path and average shared risk group respectively. Given total number of possible paths between controller and node as k, they can be defined as:

$$F^c = \frac{k}{N-1}$$
$$L_m = \frac{\sum_{u \in H} R_{cu}}{N-1}$$

4. **Select location near to a switch which has maximum utilized flow table:** As proved in paper [46] it is important the flow table space of the switch be considered while designing and planning an SDN-enabled network. Also, it is the switches which have their flow table full more likely to contact the controller for updating their flow tables.

   Formulation: Placing controller near the switches which are prominent in the network can defined as in Eq. 4.14 :

$$\max \sum_{c \in C} I_c * y_c \tag{4.14}$$

   Where I_c is the node weight factor of the switch. This is calculated by adapting the Katz centrality factor [62]. The node weight in Katz formulation is modified to represent the flow table size. More the value of I_c higher are the number of flows in node.

5. **Select location near the switches which has maximum utilized flow tables and has the least response time:**

   When selecting a location based on the flow table memory utilization of a switch,

it is possible that it can have a high response time. It is illustrated in the Fig. 4.3 where the location E will be selected if only flow table utilization was considered as it is an edge switch. This can lead to sub-optimal location. This can be circumvented by considering the response time of controller to nodes into the formulation.

Formulation: Objective Function can be defined as in Eq. 4.15.

$$min \sum_{c \in C} \frac{\sum_{u \in H} y_c * L_m(\Pi_{cu})}{I_c} \tag{4.15}$$

Constraints For the Phase II Formulations

The phase II techniques share the common constraints from the data plane to control plane perspective. The constraints which govern the phase II strategies are:

1. Limit the total number of Live controller at any given time to be one.

$$\sum_{c \in C} y_c = 1 \tag{4.16}$$

2. The latency from controller to the node should satisfy at least maximum latency requirement. In other words, the latency should be less than the maximum allowed latency.

$$y_c * R_{cu} \leq R^u_{max}, \ \forall u \in H, \ \forall c \in C \tag{4.17}$$

3. There should exist at-least one path from controller to switch which satisfy the QoS parameter of controller to node communication.

$$\gamma_{cu} \geq 1, \forall u \in H, \ \forall c \in C \tag{4.18}$$

Where, $\gamma_{cu} \subset \Pi_{cu}$ which satisfy the QoS requirement of controller to switch.

4. The path should have minimum bandwidth.

$$y_c * \sum_{u \in H} \delta_{\pi}^{l} * b_{cu} \leq B_l, \ \forall l \in L : l \in \pi, \pi \in \Pi_{cu}, \forall c \in C \tag{4.19}$$

List symbols used in the formulation is given in table 4.1.

4.8 Summary

The importance of the SDN controller, the number of the controllers required, and the placement of controller were investigated and evaluated in this chapter. Several novel strategies for determining the controller location in a metro optical network were proposed. The formulation for determining the controller location was based on the Integer Programming Technique. Some of the factors that can lead to contingencies in a network are highlighted and considered while determining the controller location. Following this section is Chapter 5 which deals with the methodologies of implementation and evaluation of the techniques proposed up to this point.

5 Implementation And Evaluation

5.1 Objective

This chapter deals with providing the proof of concept for the architecture and formulation proposed in Chapter 2 and Chapter 3 respectively. The Software used to perform simulation is explained. The solvers like Gurobi and CPLEX which adapted to solve the combinatorial problem are briefly shown. Finally, an in-depth inference on the obtained results is given.

5.2 Implementation Methodology

5.2.1 Simulation Setup for GMPLS-based VN embedding

Generalized Multi-Protocol Label Switching requires designing multiple protocols to work in tandem. This was achieved by using the present simulator, GLASS (GMPLS lightwave Agile Switching simulator) is used [66]. This simulator is an extension to a well know network simulator called SSFNet(Scalable Simulation Framework Network). It consists of many MPLS modules like RSVP (Reservation Protocol), LDP (Label Distribution Protocol) and other complementary protocols which are mandatory for the proper functioning of the network. GLASS is written in Java and whenever an element in the topology is added it creates an object for the respective element. Depending on the type of network element and the features it supports, the object attributes are given to the network element. For example, if a link with support to 80 lambdas is added between

two cross-connects, then an object for link with 80 different lambdas is added into the code block. Since GMPLS does not interact with the applications, it is not possible to orchestrate the DC resources. Therefore, only the end nodes are provided. In case of GMPLS; the routing module then determines the required path.

5.2.2 Simulation Setup for SDN-based VN embedding

It is a well-established fact that SDN brings flexibility to the network. Flexibility in case of SDN can be regarded as easy integration of user-built custom network application to an existing network. This existence of flexibility facilitated the developers to experiment with new features into an existing architecture. To realize the SDN-based network, Mininet with LINC-OE (Link Is Not Closed - Optical Extension) switch was utilized. Mininet is responsible for emulating the switches in a personal computer or a server. LINC-OE is an optical emulator switch built for realizing virtualization in optical switches. Open Network Operating System (ONOS) is used as an SDN controller. Other similar controller such as Open Day Light, RYU to name a few are not very well compatible for optical network and require lot of modification to include optical network. Since ONOS was developed with the focus of carrier service providers' requirements, led to selecting it as an SDN controller.

Due to the flexibility enabled by the SDN paradigm, it is possible to integrate the CG-ILP module as a part of the SDN controller. The role of SDN controller is to maintain the current version of network state, which will be utilized by the CG-ILP module. The selection of wavelength is adapted by following *'first-fit'* algorithm as shown in Algorithm 1.

Several topologies were used for the evaluation of the proposed model. Along with standard internet2 topology, and because of the unavailability of standard metro networks, several custom-built topologies were built, and are used for the evaluation purpose. The whole setup was run on a computer with Intel Core i7 processor, 16 GB RAM, and 3.1 GHz clock speed.

Algorithm 1: Selecting Links and Wavelengths for between the Substrate Nodes

```
BestPaths(a_s, a_d, π_{u,v})
begin
    foreach Π_{u,v} in π_{u,v} do
        foreach Link in Π_{u,v} do
            cost[]← Calculate Cost of Each link;
        end
        foreach cost in cost[] do
            Π_{u,v} ← Path with least cost in π_{u,v};
        end
    end
    Return {Π_{u,v}, cost Π_{u,v} }
end
BinPacking(Π_{u,v})
begin
    foreach link in Π_{u,v} do
        foreach Wavelength in link do
            AvailableWavelength[] ←getWavelength(link);
        end
    end
    foreach wavelength in AvailWavelength[] do
        compare if all the Links in path Π_{u,v} in
        AvailWavelength[] has the wavelength available.
        if available wavelengths in path Π_{u,v} is true for all links in it then
            Return( Selected Wavelength)
            break;
        else
            Update
            Π_{u,v} ← BestPaths(a_s, a_d, π_{u,v})
        end
        if no path found in (a_s, a_d, π_{u,v}) then
            Return Embedding Not Possible
            Select New VN Request
        end
    end
    Return ( Selected Wavelength)
end
```

5.2.3 Setup For Solving Mathematical Models

To solve the formulated model CPLEX and GUROBI were used. Gurobi is a fairly new solver compared to CPLEX but due to its wider community base, and the support to many programming languages, this solver has recently captured the attention of researchers. On the other hand, CPLEX remains as a standard software for solving the linear programming. The formulation of CG-ILP model was solved using CPLEX, due to its stability and it was found to be more appropriate for using it as an integral part of SDN controller. Due to the agility of Gurobi, and Wider support provided by Python libraries, it became more practical to use Gurobi than CPLEX for solving controller location problem. Which we found out during the course of research. The CPLEX communicated to the SDN controller through the use of REST APIs. Gurobi is used to determine the controller location. Gurobi is run once for a given set of different objective function to be enhanced between the control plane to data plane. Whenever we change the type of network and the type of service offered by the metro network the Gurobi solver needs to be run again to determine the new controller location. A detail evaluation on the performance is discussed in the following section.

5.3 Discussion of Results

5.3.1 Definition of Metrics Used For the Evaluation

In this section, a comparison of QoS metrics for VNE-MON and distributed GMPLS testbed is performed. The QoS metrics used for measurement include: (a)Resilience factor, (b) Topology Discovery time, (c) Throughput, (d) Jitter and (d) Acceptance of VNs ratio versus cost of wavelength conversion. The proposed mathematical model is evaluated on the following three main metrics for network performance:

1. Network resilience factor: This is measured by calculating the time taken by an ICMP packet to complete the round trip when there is a link failure. In other words,

the network resilience is the time the network takes to re-calculate the network path after a link failure notification has been detected by the network controller. It is one of the important metrics to consider when promising a QoS for a multimedia request. The initial setup after the request has been accepted doesn't affect the QoS, but where as having a very poor convergence time might not keep the QoS agreement.

2. Topology Discovery time: This measures the time the control plane takes to find all the links and switches in a network. In other words, the time taken to build the primary forwarding table with default flow rules. This result is directly proportional to the scalability of the network.

3. Throughput: This is determined by calculating the total percentage link bandwidth utilized for the communication and by the windowing size and switching capacity. The higher the throughput the better is the system. Depending on how fast the controller can process the new flow, it can have an impact on the overall network throughput. In the present Thesis, the throughput is evaluated between two virtual nodes embedded in a substrate node. During the evaluation, numerous stress scenarios were introduced.

4. Acceptance ratio: It is defined as the ratio of the total number of accepted requests to the total number of requests requesting the resources. This calculation provides a perspective on the controller's agility to process a new request. This result also facilitates in inferring whether the adapted VN model provides feasible solution.

5. Conversion Cost: In the case of an Optical Network, many challenges are associated with converting a signal from optical to non-optical and then back to optical. The Service providers prefer to select the paths which lead to the least number of O-E-O conversions. This result provides a perspective on the advantages of centralized over traditionally distributed network architecture.

6. Stress Condition: It is a simulated scenario wherein multiple links which are part of an active path have failed. Mega ping packets are irregularly flooded into the system. Link failure and flooding ping packets irregularly constitute what is called stress condition in the present Thesis.

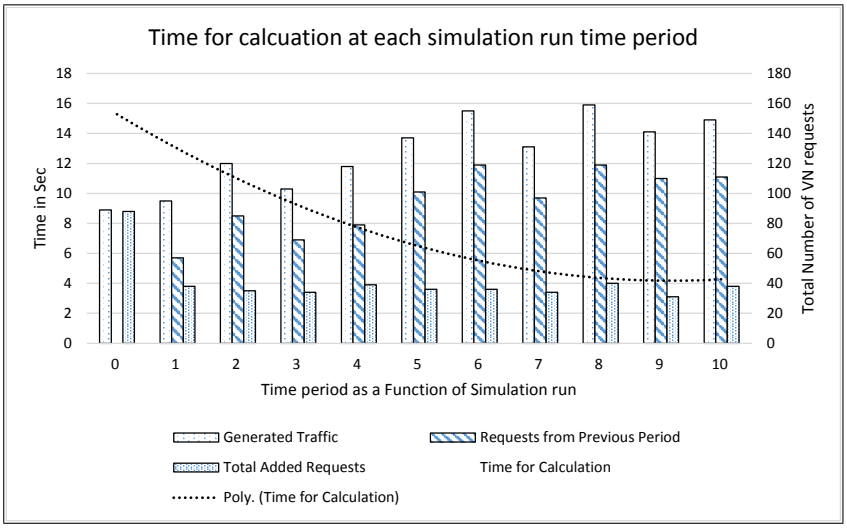

Figure 5.1: Time Taken by CG-ILP Solver

5.3.2 Performance of VNE - MON and GMPLS

Preliminary Evaluation

Due to unavailability of standard simulators or emulators for generating the Virtual Network traffic, an in-house built simulator was modeled and built. The traffic generator was written in C++. The traffic generator simulator generates virtual requests as a set of virtual node and virtual links in the form of text files for each time period. One-time period is defined as the run case of the whole system setup for one set of VN requests i.e. from generating VN requests to embedding the VN requests. These Virtual node and links are defined by the QoS requirement such as minimum required RAM, GPU, CPU, Storage, bandwidth and Maximum tolerable path latency by the simulator. The amount of request that needs to be generated can be modified. VNE-MON uses the traffic, generated by the traffic generator in a batch-wise fashion to solve according to

the defined CG-ILP model. Figure 5.1 shows the time taken by the CG-ILP to solve the optimization equation. Initially the number of request to be solved were high. But in later time periods a constant drop and add of request was performed. Because of the less number of requests being added than initially, the solver has a less number of samples to solve; hence, the time taken in the later intervals is constantly reduces and becomes constant. Figure 5.2 shows the acceptance ratio of the optimizer. The acceptance ratio depends on the available resource and the resource requested. Because of the randomness of the request generator, the exact output of the simulator cannot be predicted. The traffic generator at each run generates different types of traffic having different QoS requirement. However, the traffic generator was tuned to show overall stability in terms of number of requests generated. This ensures that a global steady state is maintained during the simulation runtime.

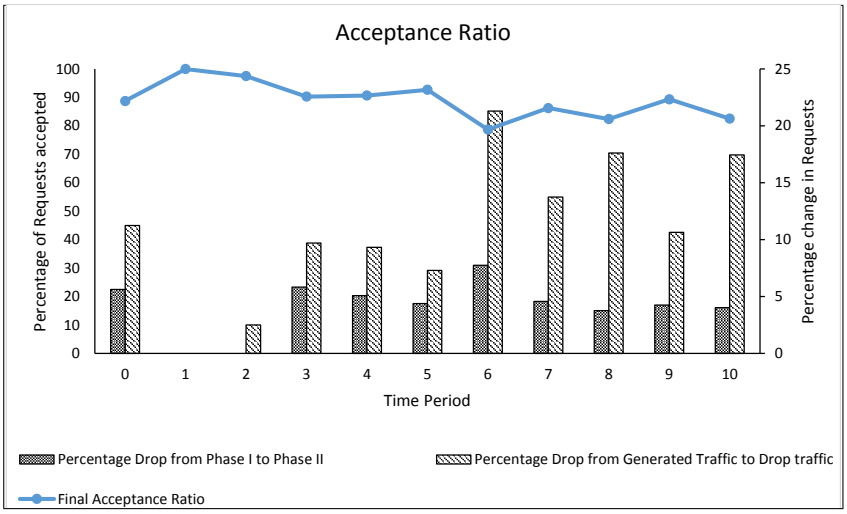

Figure 5.2: Acceptance Ratio of the Solver

Once the simulator was tuned it was configured and connected to the SDN controller. The time taken to install flows from the optimizer to the LINC-OE switch was recorded and this is depicted in the Figure 5.3. Initially, the time taken was substantial because the controller needs to gather the topology information. The time taken becomes constant and very minimal after the topology information has been gathered. Many factors which affect the flow setup time such as buffer size at each port, different processing power

at each switch to name a few in practical scenarios could not be considered due to the limitations of mininet.

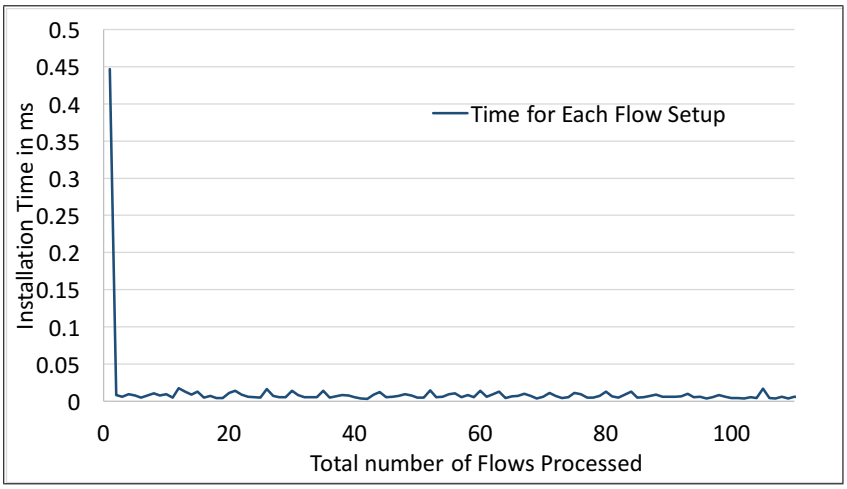

Figure 5.3: Flow Setup Time taken from Controller to Node through the use of REST APIs.

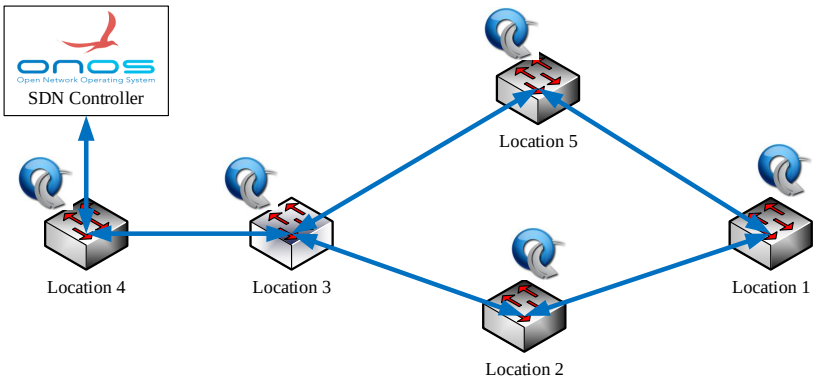

Figure 5.4: Verification of Controller Placement effect on a Small Topology

The controller placement problem was initially evaluated with a small topology before performing it with large topologies. The topology used for verification of the controller placement effect on the accepted VNs is shown in Figure 5.4. First the location 3 was selected to host the controller, and packets were sent from location 1 to 5. The link failure was simulated in the test bed to verify the controller's response time. Similarly, the test was performed by considering location 4. The results obtained from both locations were recorded and are depicted in Figure 5.5.

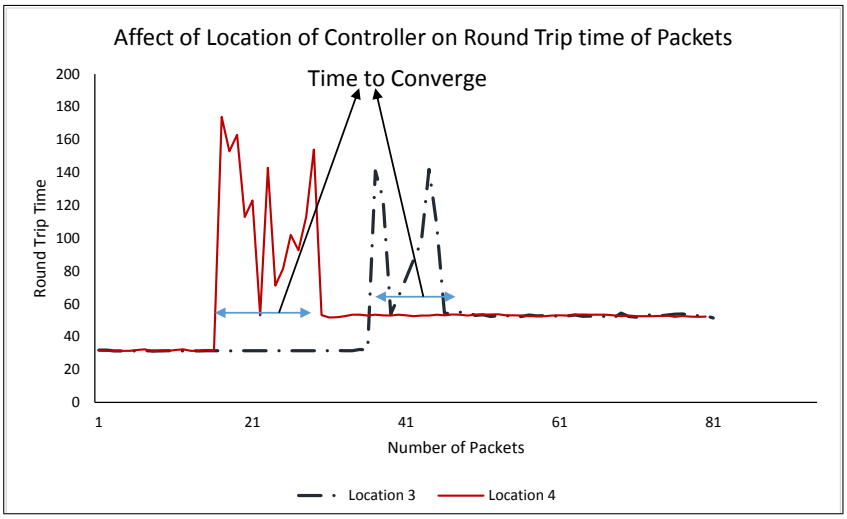

Figure 5.5: Effect of Controller Placement on the Ping (ICMP) Packets

The obtained results show a clear sign of the controller's role in the network performance. To assess VNE-MON in detail a bigger metro topology is used for the evaluation in the following sections. The topology used for further evaluation consisting of 42 nodes and 46 edges is shown in Figure 5.6. The dataset for the topology can be downloaded from Topology zoo website [65].

Evaluation of Setup Time

The evaluation of VNE-MON is performed for various setup times. Setup time is defined as the time taken by the system from responding to a signal to arriving to a final state. For example, convergence time is the time needed for a system to reach equilibrium after receiving a link failure signal. Similarly, several setup time such as time to install flows, time to reconfigure a new path, round trip delays are used for evaluation of VNE-MON with respect to GMPLS.

The round trip time (RTT) taken by an ICMP packet during a link failure for the VNE-MON and GMPLS based network is shown in Figure 5.7. The link failure event is simulated by triggering the port_down event in the test bed. The results illustrate that the maximum RTT for the VNE-MON reached 5.3 ms, whereas in the case of GMPLS, it was 12 ms GMPLS's poor performance is because of OSPF's convergence issue and

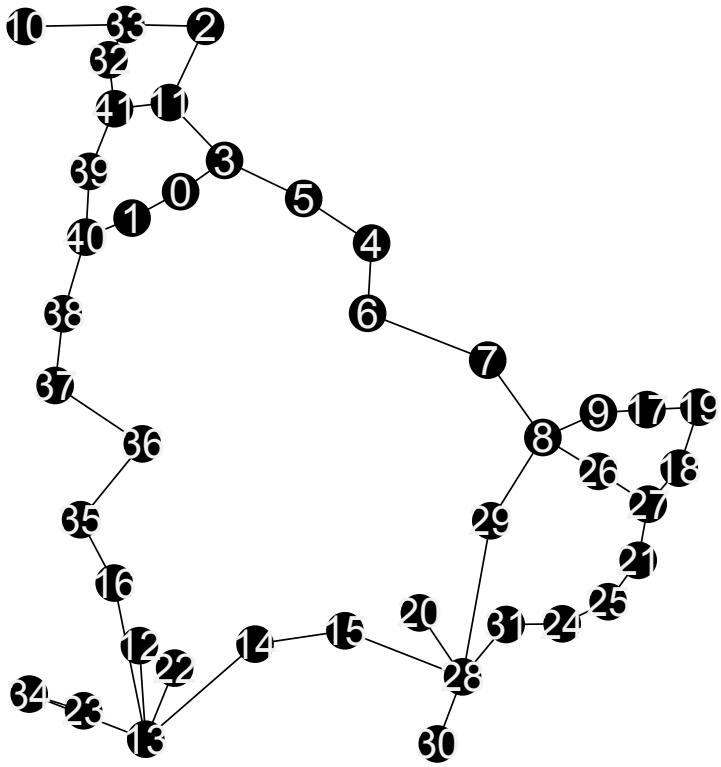

Figure 5.6: Topology used for the Evaluation

overhead related to Resource Reservation Protocol (RSVP) for the bidirectional link. The convergence time for OSPF is mainly contributed by the hello timer interval (say h_t). The dead timer interval by industry standard is $4 * h_t$ plus the path recalculation time. The path re-calculation time is a CPU intensive work, where network topology is exchanged among nodes. Label Distribution Protocol (LDP) and RSVP have a role in setting up the new connections by the brute force method.

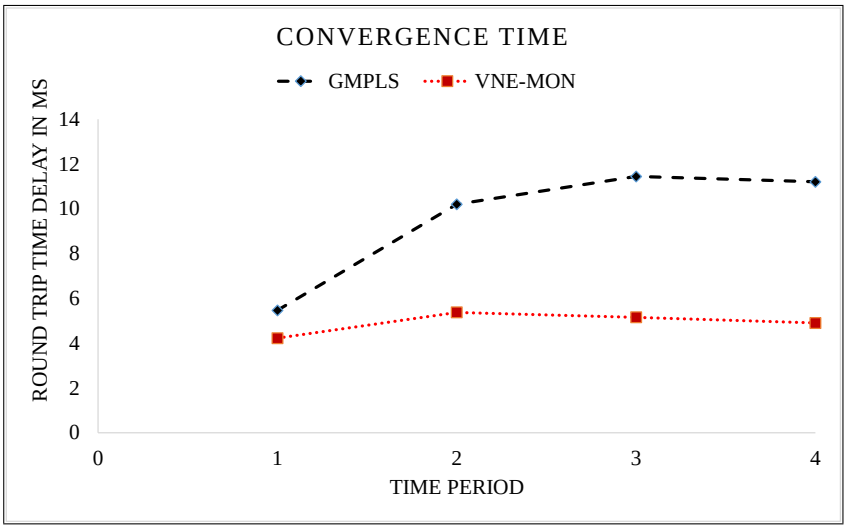

Figure 5.7: Comparison of Convergence Time for VNE-MON and GMPLS.

The time taken to converge in GMPLS is directly proportional to the total number of flows in an affected node or link, and the total number of hops in the path. VNE-MON is based on SDN architecture. When the controller senses failure notification, it updates the path with available network data, thus saving time in collecting the network information. The re-installation of flows is performed on the affected nodes, unlike in the case of GMPLS where a completely new path is established. These factors play a crucial role in improving link failure recovery time.

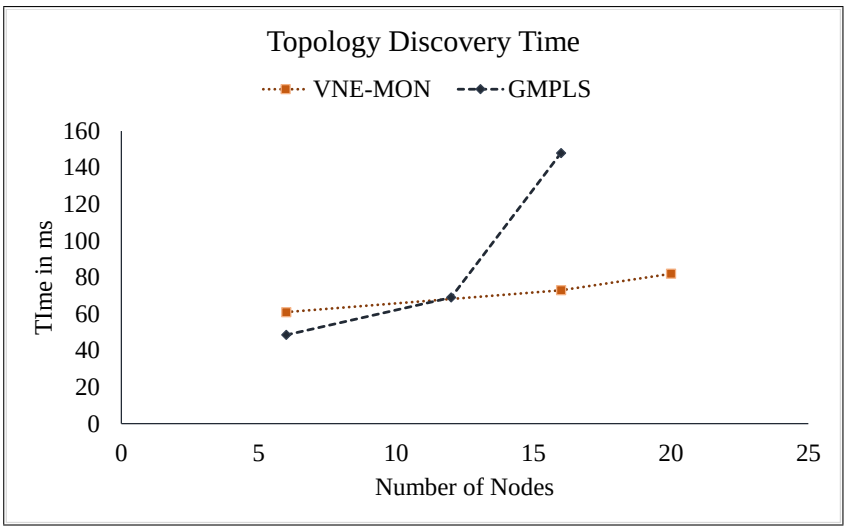

Figure 5.8: Bootstrap Time for Discovering the Nodes in VNE-MON and GMPLS.

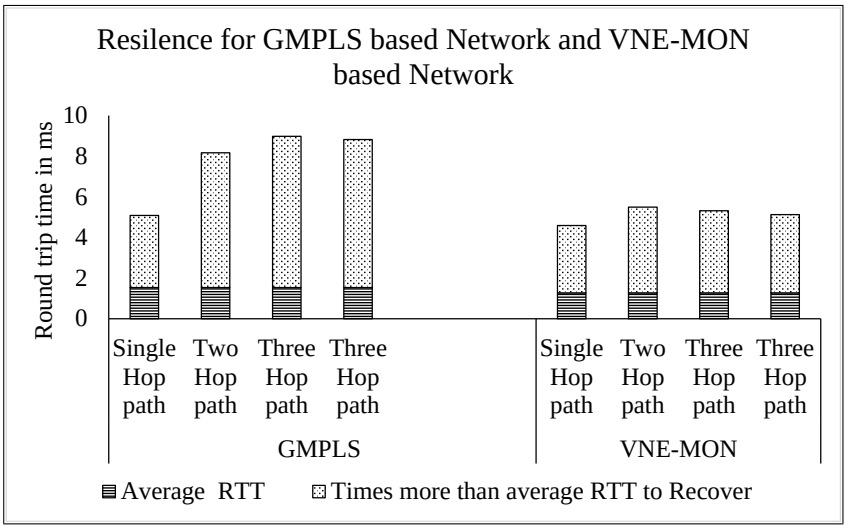

Figure 5.9: Path Re-Calculation Time after a Link Failure Event is Triggered.

The Topology discovery time is the time taken by the control plane to detect the network elements (switches and links) from the topology at the bootstrap. This is directly proportional to the scalability factor of the control plane. GMPLS was found to be stable for smaller networks, and as the network size increases, the time taken to form the initial topology was found to increase exponentially (when the number of nodes increases for more than 12 nodes in case of our simulation). This is evident from the Figure 5.8. It is due to the fact that OSPF used in GMPLS which works by following the principles of distance vector. Wherein the nodes share the information to build the topology. In contrast, for the VNE-MON model, topology discovery time remained almost constant, even after increasing the topology size. This is because of the fact that each node communicates to SDN controller rather than to its neighbor. This is an important parameter when considering a network which has many edge data centers to be managed.

Both the performance of SDN-based architecture, i.e., VNE-MON and GMPLS are dependent on the number of switches and flows they configure. GMPLS performance was found to be highly unstable and Fig 5.9 compares the result obtained from GMPLS and VNE-MON architecture for path setup time during a link failure event. The results are obtained for paths of different lengths and length is defined by the number of intermediate

O-E-O conversions in a path. It is clear from the obtained results that the GMPLS based network took nine times more its average round trip time to converge in a worst case scenario. Whereas, VNE-MON took little more than five times the average round-trip time to re-establish the path. In the following section we evaluate the VNE-MON for different Quality of Service (QoS) metrics

Evaluation of QoS Metrics

QoE and QoS are part of Service Level Agreement between the service provider and the client. QoS in networking includes the amount of guaranteed bandwidth and worst case Jitter in the signal that the service provider promised the client during the lease time. Simulating the TCP and UDP packets with the use of IPerf, enabled the determination of maximum achievable throughput and worst case jitter. Also, to measure the performance of the model the acceptance ratio is calculated on the number of O-E-O conversions. The evaluation of different QoS metrics is illustrated in the subsequent paragraphs.

The throughput was calculated by sending TCP packets between the end nodes. It is evident from the obtained results depicted in Figure 5.10, that the GMPLS was highly unstable compared to the VNE-MON. The reason for the instability of throughput in GMPLS network can be labeled to its protocols used for traffic engineering. Routing and Wavelength Assignment (RWA) uses best fit and OSPF algorithm, which aims at reducing the use of the number of wavelengths to as much as possible. The best fit leads to increased congestion of links, resulting in instability of the control plane for processing the incoming packets. The percentage utilization in the figure can be calculated as the percentage of total available link capacity. Let say x is the total capacity and y is the overhead due to use of different protocols and controller to node communication utilizing the link bandwidth. By industry standard one of the channel is reserved controller to node communication or in other words termed as control channel. By generalizing the above mentioned parameters we can represent the total available link utilization as

follows:

$$Percentage\ AvailableLink\ Bandwidth = ((x - y)/x) * 100 \tag{5.1}$$

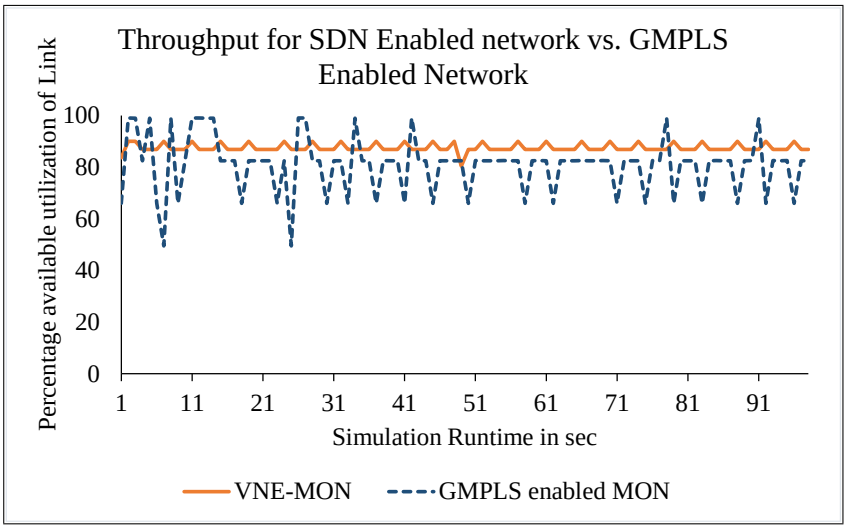

Figure 5.10: Throughput of TCP Packets in VNE-MON and GMPLS.

Hence, due to link congestion, link utilization in GMPLS is reduces drastically when all the labels in a link are active. In contrast, VNE-MON not only considers different factors like link cost and number of hops, but it also uses the first fit algorithm for wavelength assignment, along with ILP and Column generation (considering the specific requirement for Multimedia request). This assures that the links are not overly crowded and avoids link congestion. Hence, it is evident from the obtained results, for the VNE-MON based approach, that a constant throughput is achieved over the simulation runtime.

The obtained jitter values are depicted in Figure 5.11. From the graph, it is clear that the points are strongly correlated for both GMPLS and VNE-MON when there is no link or node failure in the network. The reason for maximum jitter in the case of the VNE-MON approach and GMPLS network at around period 100 secs, is the simulation of link failure. It is clear that the maximum jitter of VNE-MON is more than maximum jitter of GMPLS enabled MON. This is related to the use of single SDN controller and its placement.

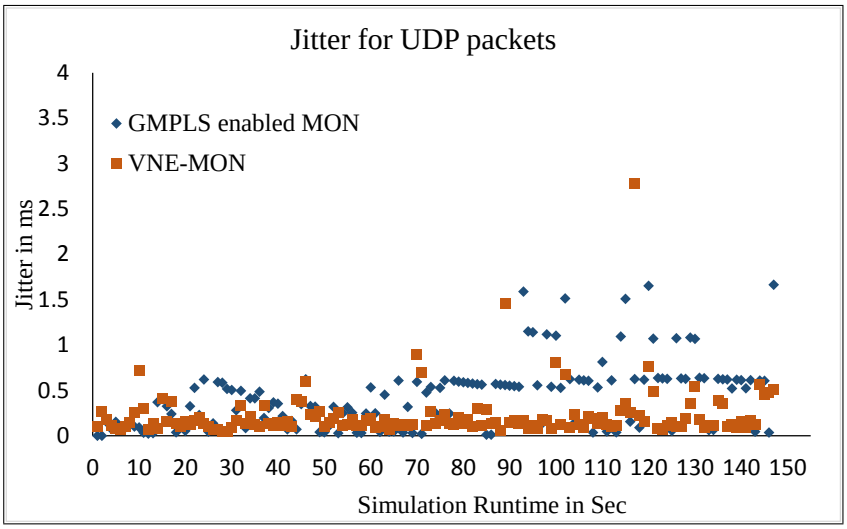

Figure 5.11: Jitter Variation in VNE-MON and GMPLS.

When a link failure event occurs, the packet_in message has to reach the SDN controller and then the SDN controller updates the required nodes. In VNE-MON, since a single SDN controller with random placement was used, the result led to a maximum jitter value. For GMPLS network, the jitter is not very high because the topology was small, therefore the system was able to converge faster. However, when link failure event is initiated, the overall jitter values change from tightly-coupled to moderately coupled, representing instability. This effect is a of GMPLS controller overloading the neighboring link for faster recovery.

Finally, the acceptance ratio of VN requests between GMPLS and VNE-MON is represented in Figure 5.12. At first, the acceptance ratio of both the GMPLS and the SDN-based systems looks similar, and are depicted in the Figure 5.12. However, due to the lack of global view, GMPLS performed some wavelength conversion for a given path compared to the SDN-based system. This not only increases the cost of communication but also degrades the performance. A request accepted in VNE-MON was dropped in a GMPLS testbed for mainly two reasons.

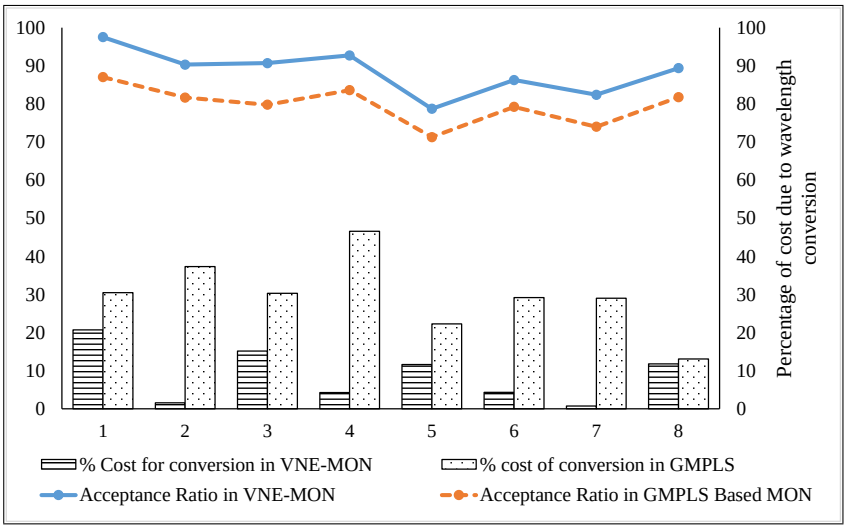

Figure 5.12: Acceptance ratio in GMPLS and VNE-MON.

Firstly, when an incoming request arrives, RSVP-TE protocol of GMPLS goes through the available resources in the selected path, if the request cannot be satisfied, RSVP-TE searches for the another path. As a result of brute force and lack of global view, the incoming VN request is kept on hold for a longer period. This lead to the situation where the VN Resource Manager determines whether it is best to drop the VN request or execute for the next in line request. Secondly, the dropping occurred because the GMPLS considered the path for a request with too many intermediate OEO conversions than the VN request's QoS requirement. This occurs because GMPLS is not completely aware of the application requirements. As a result, the VN resource manager drops the request, assuming that the network does not satisfy the QoS requirement of the VN request. From the cost of conversion, it is clear that as result of the lack of global view, the GMPLS becomes a very expensive and non-optimal solution for the MON.

5.3.3 Effect of Controller Location on the Embedded VN requests

Many research proposals have addressed the issue that to satisfy the needs of a small scale network, a single controller is sufficient [51, 59]. Hence optimization is performed

to select only one controller location based on different performance parameters on the MON requirement.

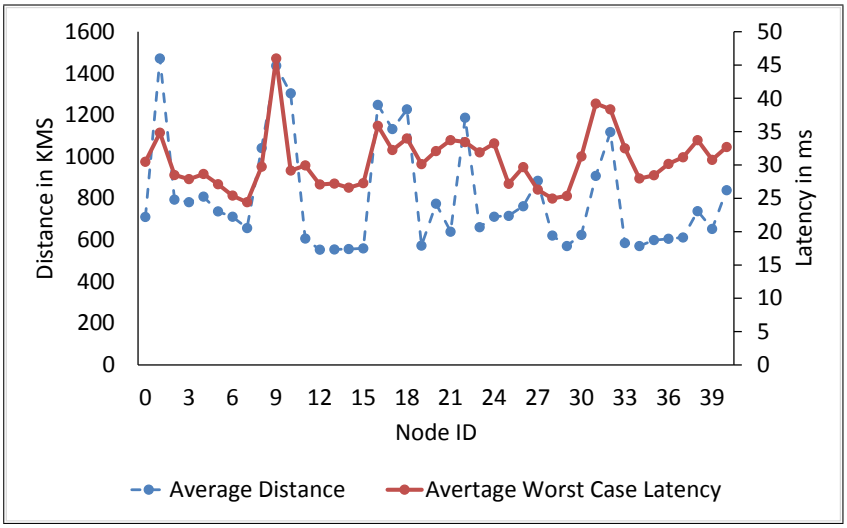

Figure 5.13: Variation of Average Distance vs. Latency.

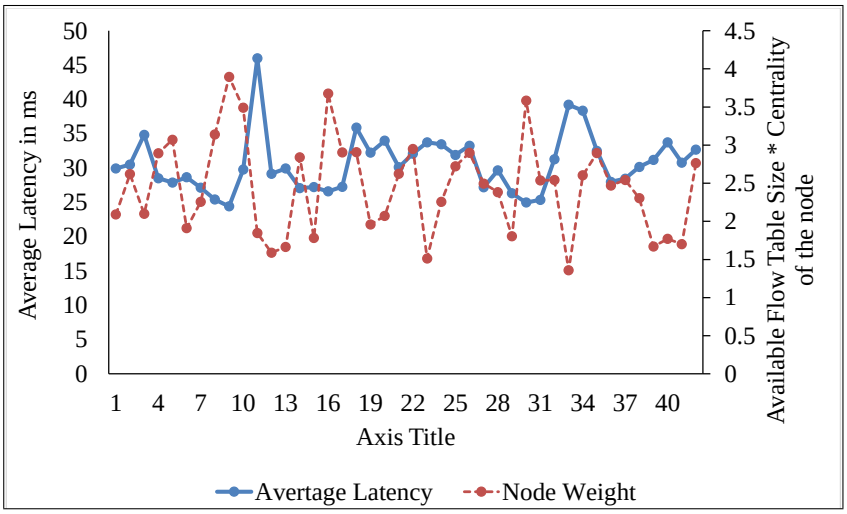

Figure 5.14: Variation of Response Time vs. Node Weight.

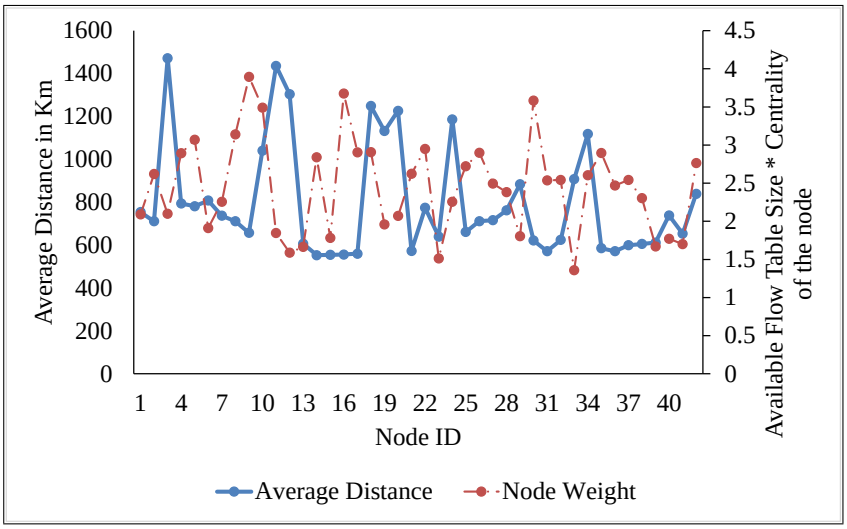

Figure 5.15: Variation of Average Distance vs. Node Weight.

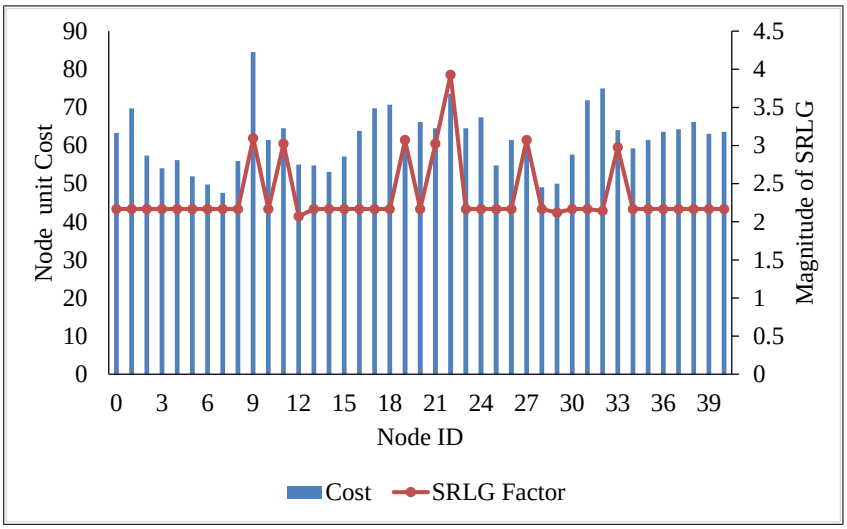

Figure 5.16: Cost vs. SRLG factor.

Evaluation at Phase - I

It is important that to determine the best controller location, a study on variation of different network parameters is vital. The magnitude of different metrics used for determining the location of the controller is identified for each node in the network. This determination of various performance metrics of each node aids is the study of the network. The following paragraphs manifest the variation of different performance parameters compared with other nodes in the network.

From Fig.5.13 - 5.16 provides a glimpse of the considered topology for study. Fig. 5.6 shows the interconnection of different nodes in the considered topology. This data related to the topology under the study is used to examine the variation of different metrics in the network from each node's point of view. Initially, all the nodes are assumed to be the potential locations for hosting the controller. The worst response time of each path is recorded during the peak hours. It is clear that some of the nodes, even though they are relatively distant from one another, their latency is less that those who are closer to the center of the graph. It is a common metric to use either the distance [54] or the number of hops to find the optimal location of the controller. Based on the distance, it is not possible to generalize that the network performance will be under the predicted behavior.

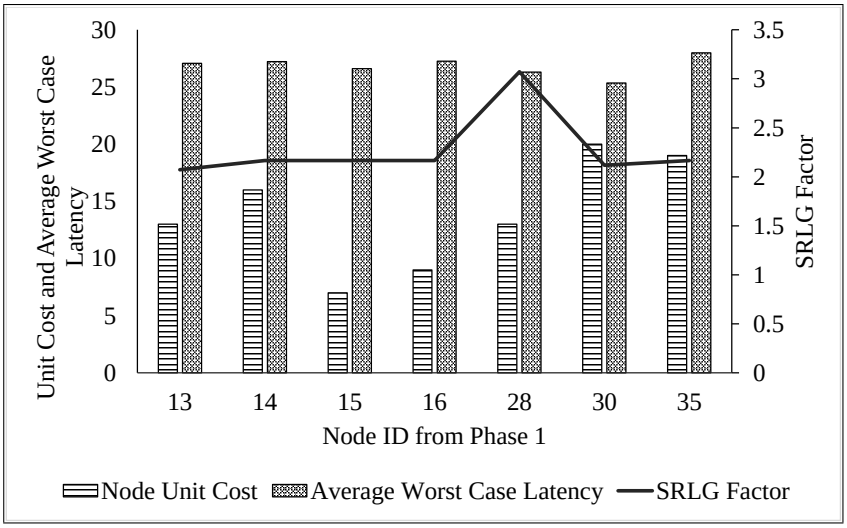

Figure 5.17: Cost vs. Latency vs. SRLG

Figure 5.16 depicts the cost vs. the SRLG factor value of different nodes. It is clear that the more the SRLG factor, the higher is the cost. Nevertheless, in a few cases, it shows that the cost is high whereas the SRLG is comparatively low. One such example is node number 10. Hence, optimizing a controller location based on a single metric may lead to a poor network performance, particularly in cases of MON where the traffic, as well as the type of access network, varies.

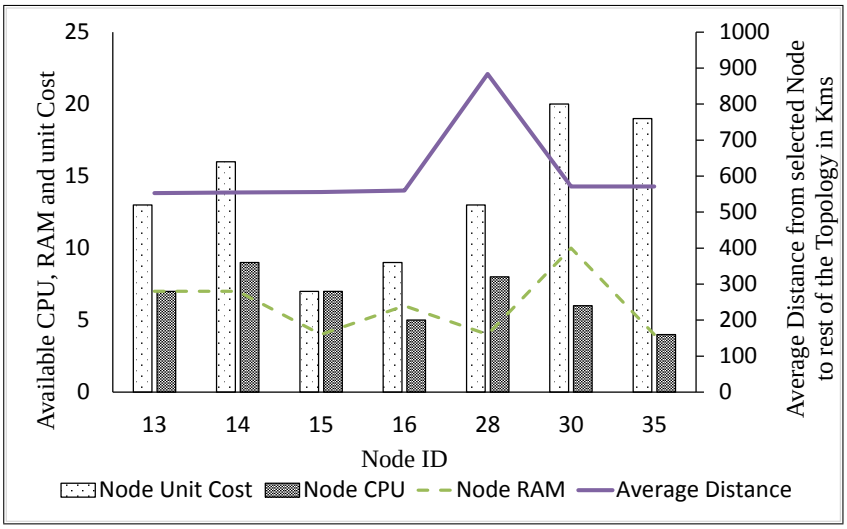

Figure 5.18: Resource vs. Average Distance.

The nodes obtained at Phase I of controller placement model are depicted in Figure 5.17 and Figure 5.18. Phase I aids in minimizing the search space, and facilitates in selecting controller locations which not only satisfy the network requirement but also ensure that the deployment cost is minimal. The cost is defined based on SDN controller's utilization of the network for management and control purposes. The cost does not generate any revenue to the service provider but is needed to maintain the stability of the network. The reliable location which can satisfy the network requirement is selected for the controller placement.

In the considered topology, phase I narrowed the potential candidates from 44 nodes to 7 nodes which is 85% of the total reduction in the search space. The results fluctuate depending on the metro network, service provider, and other switching infrastructure. It is clear that the latency of all the nodes from phase I are almost the same, whereas the distance is highly variable. The node resources play a crucial role. The node selected for controller placement has to host multiple virtual controllers for the incoming VN requests. For this, the controller should have enough substrate resources. It is interesting to note that the SRLG factor of node 28 is high (Figure 5.18) which means it is a highly reliable location but the average distance from node 28 is high compared to all other nodes. Thus it proves that optimization based on selecting the location only from the reliability factor

might lead to non-optimal results during peak hours of traffic.

Performance Evaluation of VN Requests for Different Controller Locations

Various performance parameters can lead to the same optimal controller location. Cases 2 and 5 have the same location as well as Cases 3 and 4 have the same location. Therefore, further elaboration in this article is focused only on three strategies, i.e., Cases 1, 2 and 3. Case 1 selects the controller location by minimizing the distance between control plane to the data plane. Case 2 is optimized for distance and worst-case response time. Case 3 selects controller location by minimizing latency and maximizing reliability. The obtained results show that it is not efficient to optimize based on distance. The QoS performance results for VN requests are highly variable at these locations. This proves that in a MON, selecting a location for the controller is a sensitive task.

Figure 5.19 and Figure 5.24 show different QoS performance metrics with respect to different controller locations. These results are obtained under the stressful conditions. The stressful conditions considered for the evaluation are link failures and sporadic traffic bursts.

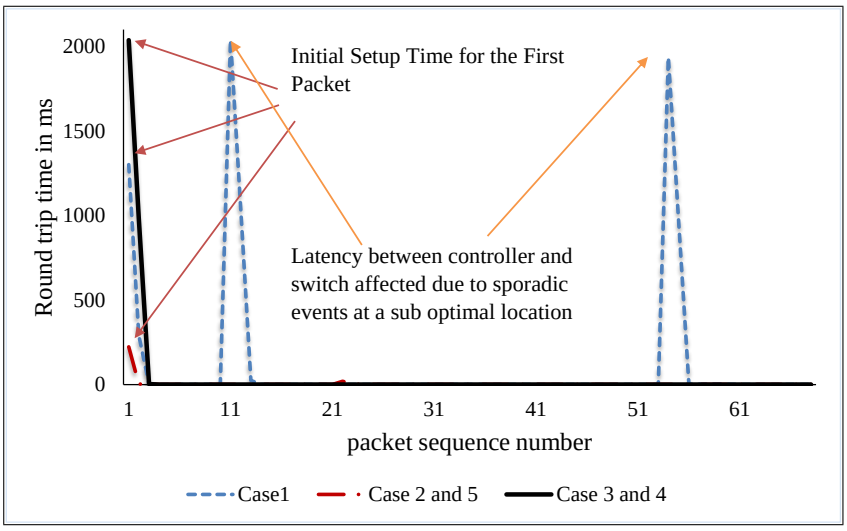

Figure 5.19: Reactive Forwarding of Packets.

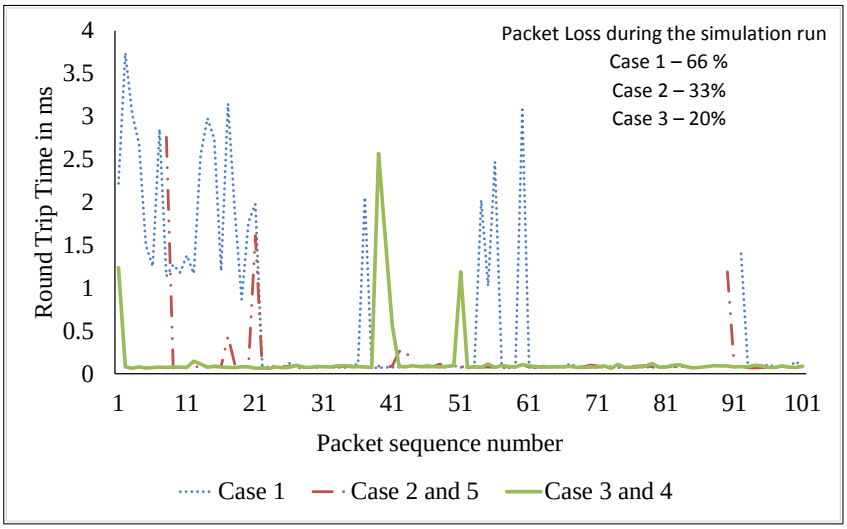

Figure 5.20: Reactive Recovery Time.

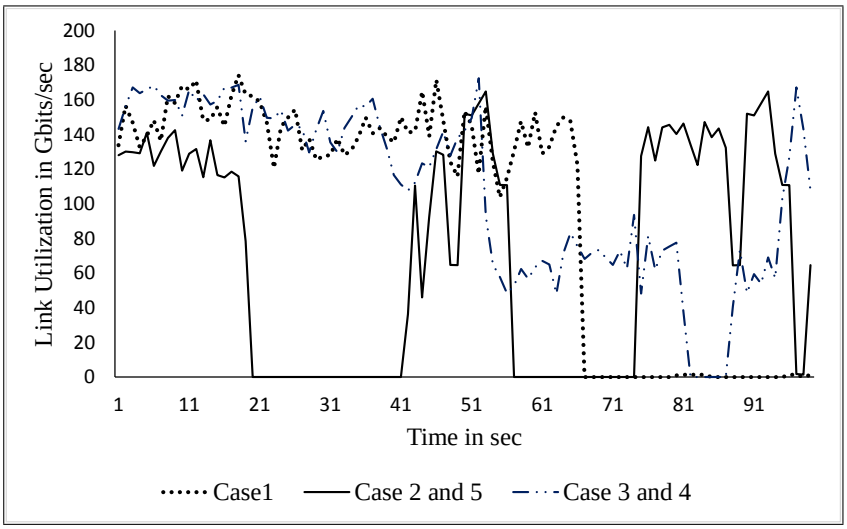

Figure 5.21: Bandwidth for TCP Traffic.

Fig. 5.20 shows the recovery time with respect to reactive routing. Reactive is based on the principle that when a link failure occurs, it performs the re-routing task. Reactive rerouting is a highly CPU intensive task which leads to unnecessary overhead for a network. Fig. 5.19 illustrates the performance of reactive forwarding of the accepted VN requests. The initial high value shows the time taken by the node to controller communication plus the controller response time. In case of optimizing based only on distance (case 1) has an adverse effect if the intermediate nodes' buffer is overloaded

during the traffic.

Fig. 5.21 and Fig. 5.23 evaluates the performance of TCP bandwidth and UDP Jitter respectively. TCP is used for reliable communication whereas UDP is used for multimedia communication. It is important that both types of application perform well in the network. Both these metrics are evaluated under stressful condition.

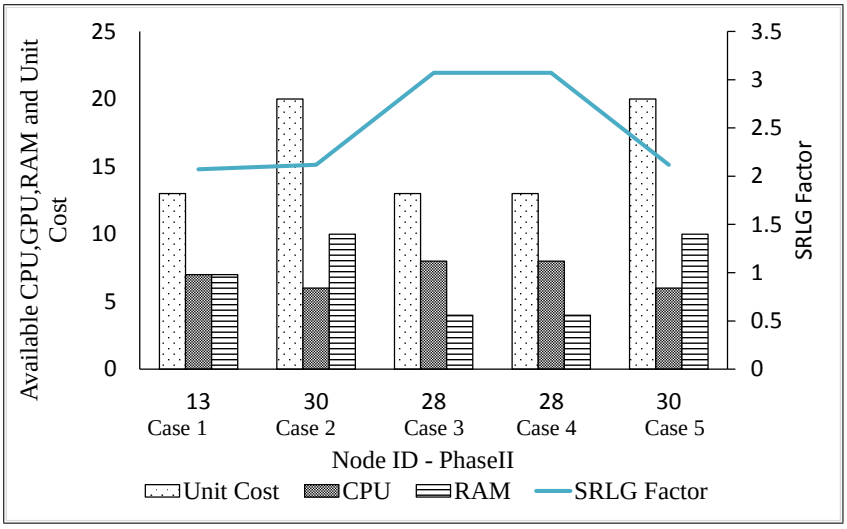

Figure 5.22: Computing Resource Availability vs. SRLG.

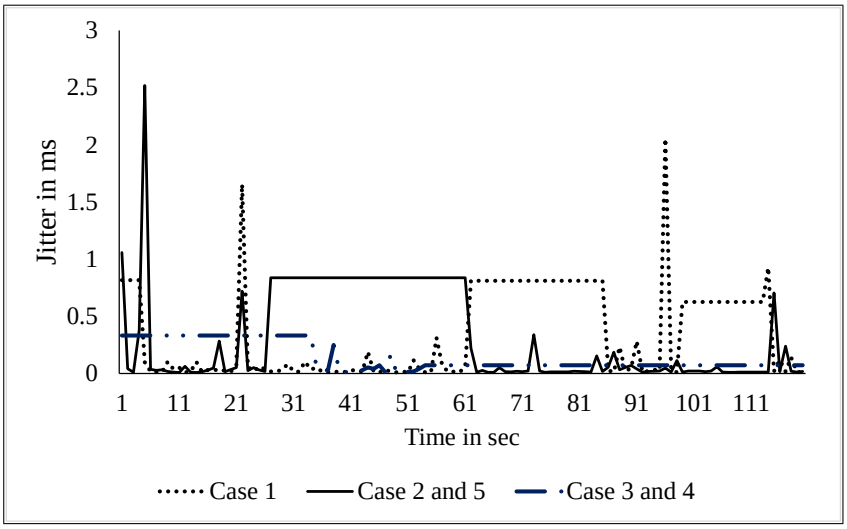

Figure 5.23: Jitter for UDP Traffic.

Case 3 had the least downtime proving it to be the most reliable controller location. Whereas, Case 2 had low downtime but was able to converge faster. However, the path

between the virtual nodes experience a constant drop of bandwidth. Coming to the distance based optimization case, i.e., Case 1 had the highest downtime. Initially, it was stable during the sporadic events, but when the network was simulated with link failures and sporadic events, it failed to converge within the acceptable limits. Jitter was stable for all cases except during the sporadic bursts. Case 3 had the least Jitter. From Fig. 5.23, it is clear that Case 3 was most suitable for controller location for multimedia communication on MON.

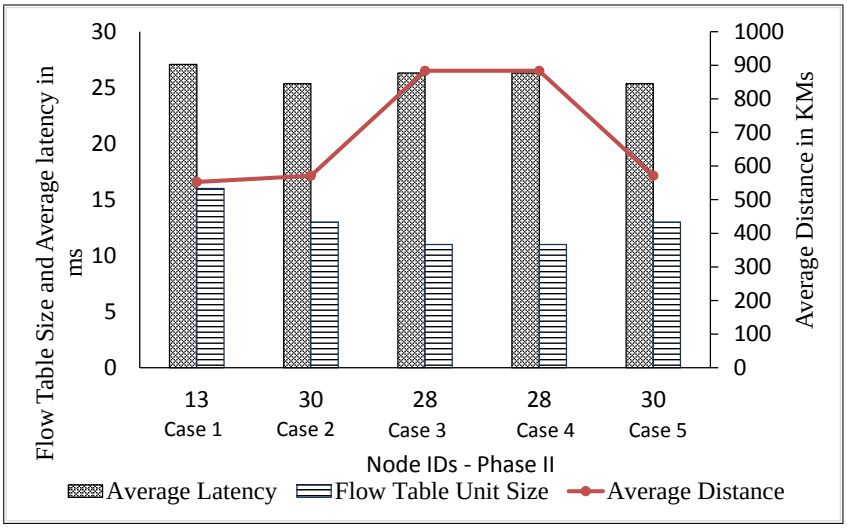

Figure 5.24: Variation of Average Latency, Flow Table vs. Average Distance.

Figures 5.22 and 5.24 compare different substrate resources available at the node location to host the controller, along with the average worst case metrics. Both the figures illustrate that case 3 has the higher available resources and the highest shared risk factor. Thus, it is evident that these factors play a vital role in the performance of the controller.

5.4 Summary

The obtained results prove that SDN based VN embedding for MON network i.e., VNE-MON, is more stable and achieved better acceptance ratio of VN requests than its counterpart (GMPLS). With the help of global view provided by SDN controller, a better

load-balancing of wavelength was achieved. This facilitated in reducing the cost of wavelength conversion. In short, although GMPLS can be used for MON network, due to lack of global view, it's efficacy becomes highly unpredictable with increases in network size. Also from the evaluation, it is clear that the performance of the controller can vary capriciously based on the location considered for hosting it. Comprehensive evaluation of controller placement strategies needs to be performed before installing the controller at a certain location. This holds particularly in the case of MON where a single controller can suffice the network requirement, but placing it in a non-optimal location can lead to poor performance of the network. Non-optimal controller location can also create an inference among the Service Providers that more controllers are needed for the efficient performance of network rather than a single optimized location.

6 Conclusion And Future Work

6.1 Conclusion

The influx of Edge Cloud computing to Metro network presented many opportunities for the application providers. However, due to the rigidity in the network infrastructure service providers were not able to completely realize Edge cloud computing. Also, the lack of interaction between the application provider and the carrier service provider, resulted in degraded QoS.

As a consequence of minimal interaction among the various service providers (i.e., application to application providers, application to carrier providers and carrier to carrier providers) led to under-utilization of the network occurred. To enable interaction and increase utilization of a network the concept of multi-tenancy was introduced. This concept of multi-tenancy involves the sharing of available resources with multiple users. As a result, Network Virtualization was introduced, NV was proposed to overcome the rigidity and to increase the network utilization. NV created the network as an overlay model. This overlay model enabled the service providers to be placed at different levels. The right of access to the service provider is given based on the type of service offered or requested by them. The layer model proposed by NV covered the intricacies of network and only provided the information necessary for the application provider.

Required agility and flexibility to realize NV can not be catered with the use of traditional control plane. To overcome the limitations GMPLS was introduced. GMPLS underwent several standardization procedures leading to increased complexities. Another

reason for failure of GMPLS to capture the carrier service providers attention was its proprietary and distributed nature. To overcome these challenges and to bring simplicity in controlling a network, a centralized control architecture was envisioned. Software Defined Network facilitated in the easy integration of user-built applications, enhancing the NV framework and promoting an open network concept.

The use of SDN comes with its own challenges. Failure to consider the challenges associated with the SDN framework can lead to under utilization of the network.

In this Thesis a novel architecture for VN embedding in a MON with the multimedia requirements was proposed. The proposed architecture leverages the SDN nature of externalizing the controller. In order to have a global view of the network while at the same time carefully considering the controller's CPU and other resources availability. VN embedding was also performed on the GMPLS-enabled testbed. The results obtained reveal that the performance of GMPLS was better when the topology had less than ten network elements. However, as the network size grew, the distributive nature of GMPLS resulted in not being feasible to use as a control plane. In particular this situation is for Metro network where multimedia communication forms the major part of the network traffic.

Additionally, the controller faced many challenges with the network. Careful consideration needs to taken when designing the implementation of the controller to a physical location. Therefore, numerous optimization techniques were proposed to determine the controller locations. Each technique confronts among themselves. It is difficult to generalize one methodology for determining the controller location for all the Metro networks. The comparison of different techniques shows that there is a fine trade off between the various QoS metrics, thus proving that each metro network should be treated separately based on their traffic pattern.

6.2 Future Work

6.2.1 Proactive Determination of Controller Location Based on Traffic Predictability.

The traffic behavior associated with the metro network is not stable and experiences constant traffic bursts. A reduction in the effect incurred by a controller during the traffic bursts can be alleviated by proactively determining the future location of the controller. Although several solutions have been already proposed for dynamic cloning of controller [52, 54], those proposals are made for a distributed controller. It is feasible to implement and clone multiple controllers if a system follows distributed architecture. Also, these proposals are not proactive in nature but rather dynamically determine the controller location. Once the controller senses that the load is beyond the controller's threshold capacity, it clones itself to create multiple copies at different locations. Enabling dynamic cloning is not feasible in the case of the single controller, whereas deciding the location of the controller in a proactive fashion is entirely feasible.

6.2.2 Using Simulated Annealing to Embed Virtual Controller of the VN Requests.

Simulated Annealing is the technique employed in an heuristic approach to reach the feasible solution in practical time limits. It considers numerous objectives to provide a global optimal. This technique can be used to determine the optimal location of the virtual controller at the different virtual locations. Since this is an heuristic approach with the acceptable error, this technique can be employed for dynamically solving large problems. Thus this approach can be used for implementing the proposed CG-ILP model along with providing the optimal location for the virtual controllers of a given network.

6.2.3 A Study on the Performance of Single Centralized Controller vs. Multi Controller for Metro Optical Network

In this Thesis consideration of a single controller is made and the proposed formulation is in regard to locating the optimal location for a single controller. Although an assumption is made based on the existing fact that a single controller is sufficient for a metro network, a comparative study has not been performed. A possible future research direction would be to formulate a model for determining the controllers' positions in a metro network while ensuring sustenance during the traffic bursts. This research could be followed up by studying of multiple controllers on the metro network compared to a single controller.

Final Comments

In conclusion, it is important to consider the design aspects of MON for providing seamless experience to the end users. There are many research gaps which need attention and in this Thesis an attempt was made to solving one such challenge associated with MON. In addition, use of recent technologies such as SDN and ROADMs can enhance the network utilization leading to the betterment of both the service providers and the users.

Bibliography

[1] Aguiar, E. et al., "A real-time video quality estimator for emerging wireless multimedia systems," in *Wireless Network* , 2014, pp. 1759-1776.

[2] P. Fraternali et.al "Rich Internet Applications," in *IEEE Internet Computing*, vol. 14, May-June 2010, no. 3, pp. 9-12.

[3] G. Ghoda et.al, "A survey on data center network virtualization," in *3rd International Conference on Computing for Sustainable Global Development (INDIACom)*, New Delhi, 2016, pp. 3464-3470.

[4] A. Jarray and A. Karmouch,"Cost-Efficient Mapping for Fault-Tolerant Virtual Networks," in *IEEE Transactions on Computers*, vol. 64, no. 3, March 2015, pp. 668-681.

[5] A. Jarray and A. Karmouch,"Decomposition Approaches for Virtual Network Embedding With One-Shot Node and Link Mapping," in *IEEE/ACM Transactions on Networking*, vol. 23, no. 3, June 2015, pp. 1012-1025.

[6] A. Jarray and A. Karmouch,"Periodical auctioning for QoS aware virtual network embedding," in *IEEE 20th International Workshop on Quality of Service*, Coimbra, 2012, pp. 1-4.

[7] N.M. Mosharaf Kabir Chowdhury, Raouf Boutaba "A survey of network virtualization,in *Computer Networks*, " Volume 54, Issue 5, 8 April 2010, Pages 862-876.

[8] W. Zhu et.al., "Multimedia Cloud Computing," in *IEEE Signal Processing Magazine*, vol.28, no.3, 2011, pp. 59-69.

[9] H. Xu and B. Li, "Joint request mapping and response routing for geo-distributed cloud services," in *Proceedings IEEE INFOCOM*, Turin, 2013, pp. 854-862.

[10] Gomes, R. L. et al., "Management of virtual network resources for multimedia applications", in*Multimedia Systems*, 2015, pp. 1-15.

[11] S. Gringeri et.al.,"Extending software defined network principles to include optical transport," in *IEEE Communications Magazine*, March 2013, vol. 51, no. 3, pp. 32-40.

[12] R. Nejabati et.al., "Toward a completely softwareized optical network [Invited]," in *IEEE/OSA Journal of Optical Communications and Networking*, vol. 7, no. 12, Dec. 2015, pp. B222-B231.

[13] M. Nooruzzaman and E. Halima,"Low-cost hybrid ROADM architectures for scalable C/DWDM metro networks," in *IEEE Communications Magazine*, August 2016 vol. 54, no. 8, pp. 153-161.

[14] J. Wu and T. Wang, "Research and Application of SOA and Cloud Computing Model," in *Enterprise Systems Conference, Shanghai*, 2014, pp. 294-299.

[15] Pedro Garcia Lopez et.al., "Edge-centric Computing: Vision and Challenges," in *SIGCOMM Comput. Commun. Rev. 45*, 5 (September 2015), pp. 37-42.

[16] H. Chang et.al., "Bringing the cloud to the edge," in *IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS),* Toronto, ON, 2014, pp. 346-351.

[17] B. Varghese et.al., "Challenges and Opportunities in Edge Computing," in *IEEE International Conference on Smart Cloud (SmartCloud),* New York, NY, USA, 2016, pp. 20-26.

[18] L. Ramaswamy, A. Iyengar and J. Chen, "Cooperative Data Placement and Replication in Edge Cache Networks,"in *International Conference on Collaborative Computing: Networking, Applications and Worksharing*, Atlanta, GA, 2006, pp. 1-9.

[19] D. S. Dias and L. H. M. K. Costa,"Online traffic-aware virtual machine placement in data center networks,"in *Global Information Infrastructure and Networking Symposium (GIIS)*, Choroni, 2012, pp. 1-8

[20] S. Harnal and R. K. Chauhan, "Multimedia support from cloud computing: A review," *International Conference on Microelectronics, Computing and Communications (MicroCom)*, Durgapur, 2016, pp. 1-6.

[21] N. Niebert et al., "The way 4WARD to the creation of a future internet," in *IEEE 19th International Symposium on Personal, Indoor and Mobile Radio Communications*, Cannes, 2008, pp. 1-5.

[22] N.M. Mosharaf Kabir Chowdhury, Raouf Boutaba,"A survey of network virtualization, Computer Networks", Volume 54, Issue 5, 8 April 2010, Pages 862-876.

[23] N. M. Mosharaf Kabir Chowdhury and R. Boutaba,"Network virtufalization: state of the art and research challenges," in *IEEE Communications Magazine*, July 2009, vol. 47, no. 7, pp. 20-26.

[24] Nick Feamster, Lixin Gao, and Jennifer Rexford, "How to lease the internet in your spare time" in *SIGCOMM Comput. Commun. Rev.* 37, 1 (January 2007), pp.61-64

[25] B. A. A. Nunes et.al.,"A Survey of Software-Defined Networking: Past, Present, and Future of Programmable Networks," in *IEEE Communications Surveys & Tutorials*, Third Quarter 2014, vol. 16, no. 3, pp. 1617-1634.

[26] D. Kreutz et.al., "Software-Defined Networking: A Comprehensive Survey," in *Proceedings of the IEEE*, Jan. 2015, vol. 103, no. 1, pp. 14-76.

[27] Mannie, E. et.al., *"Generalized Multiprotocol Label Switching Architecture(GMPLS),"* IETF RFC 3945, 2004.

[28] McKeown, Nick et.al., "OpenFlow: Enabling Innovation in Campus Networks," in *SIGCOMM Comput. Commun. Rev*, vol. 38, no. 2, April 2008, pp. 69-74.

[29] M. Shirazipour et.al., "Realizing packet-optical integration with SDN and OpenFlow 1.1 extensions," in *IEEE International Conference on Communications (ICC)*, Ottawa, ON, 2012, pp. 6633-6637.

[30] D. Simeonidou, R. Nejabati and S. Azodolmolky, " Enabling the future optical Internet with OpenFlow: A paradigm shift in providing intelligent optical network services,"in *13th International Conference on Transparent Optical Networks*, Stockholm, 2011, pp. 1-4.

[31] Open Network Foundation, *"OpenFlow Switch Specification,"* Version 1.3.0 (Wire Protocol 0x04), June 25, 2012.

[32] A. Sgambelluri, F. Paolucci, F. Cugini, L. Valcarenghi and P. Castoldi, "Generalized SDN control for access metro core integration in the framework of the interface to the Routing System (I2RS)," in *IEEE Globecom Workshops (GC Wkshps)*, Atlanta, GA, 2013, pp. 1216-1220.

[33] S. Seetharaman, "OpenFlow/SDN: How it works, and What it means for networks beyond the campus," in *National Fiber Optic Engineers Conference*, OSA Technical Digest (Optical Society of America, 2012), paper NTu1E.4.

[34] S. Yamashita et.al.,"Extension of OpenFlow protocol to support optical transport network, and its implementation," in *IEEE Conference on Standards for Communications and Networking (CSCN)*, Tokyo, 2015, pp. 263-268.

[35] S. Das, G. Parulkar and N. McKeown,"Unifying Packet and Circuit Switched Networks," in *IEEE Globecom Workshops*, Honolulu, HI, 2009, pp. 1-6.

[36] D. Kreutz, F. M. V. Ramos et.al., "Software-Defined Networking: A Comprehensive Survey," in *Proceedings of the IEEE*, Jan. 2015, vol. 103, no. 1, pp. 14-76.

[37] A. S. Thyagaturu et.al., "Software Defined Optical Networks (SDONs): A Comprehensive Survey," in *IEEE Communications Surveys and Tutorials*, vol. 18, no. 4, pp. 2738-2786, Fourthquarter 2016.

[38] A. Mendiola; J. Astorga; E. Jacob; M. Higuero,"A survey on the contributions of Software-Defined Networking to Traffic Engineering," in *IEEE Communications Surveys and Tutorials*, vol.PP, no.99, pp.1-1

[39] J. Reich et.al., "Leveraging SDN to streamline metro network operations," in *IEEE Communications Magazine*, vol. 54, no. 10, October 2016, pp. 109-115.

[40] Y. Zhu and M. Ammar, "Algorithms for Assigning Substrate Network Resources to Virtual Network Components," in *Proceedings IEEE INFOCOM 2006. 25TH IEEE International Conference on Computer Communications*, Barcelona, Spain, 2006, pp. 1-12.

[41] S. Kareche et.al., "A New Robust Heuristic for Assigning Substrate Network Resources to Virtual Networks," in *International Conference on Advanced Networking Distributed Systems and Applications*, Bejaia, 2014, pp. 47-52.

[42] Chapman et.al., "Software architecture definition for on-demand cloud provisioning," in *Cluster Computing*, 2012 , V 15, pp. 79 -100.

[43] F. Gu et.al., "Virtual network reconfiguration in optical substrate networks," in *Optical Fiber Communication Conference and Exposition and the National Fiber Optic Engineers Conference (OFC/NFOEC)*, 2013, Anaheim, CA, 2013, pp. 1-3.

[44] Jiachen Ma et.al,"Virtual optical network embedding with sdn architecture based on memetic algorithm," in *4th International Conference on Computer Science and Network Technology (ICCSNT)*, Harbin, pp. 1050-1053.

[45] M. Capelle et.al, "Online virtual links resource allocation in Software-Defined Networks," in *IFIP Networking Conference*, 2015, Toulouse, pp. 1-9.

[46] L. R. Bays et.al,"Virtual network embedding in software-defined networks," *NOMS 2016 - 2016 IEEE/IFIP Network Operations and Management Symposium*, Istanbul, 2016, pp. 10-18.

[47] P. Han, L. Guo and Y. Liu, "Virtual Network Embedding in SDN/NFV based Fiber-Wireless Access Network," in *International Conference on Software Networking*, Jeju, 2016, pp. 1-5.

[48] X. Wen et.al., "An Efficient Resource Embedding Algorithm in Software Defined Virtualized Data Center," in *2015 IEEE Global Communications Conference*, San Diego, CA, pp. 1-7.

[49] R. Guerzoni et al., "A novel approach to virtual networks embedding for SDN management and orchestration," in *IEEE Network Operations and Management Symposium (NOMS)*, Krakow, pp. 1-7.

[50] Y. Zhang et.al., "On Resilience of Split-Architecture Networks," in *IEEE Global Telecommunications Conference - GLOBECOM*, 2011, Houston, TX, USA, 2011, pp. 1-6.

[51] Brandon Heller, et.al.,"The controller placement problem," in *Proceedings of the first workshop on Hot topics in software defined networks* (HotSDN '12), ACM, New York, NY, USA, 7-12.

[52] L. Yao, et.al., "Controller placement and flow based dynamic management problem towards SDN," in *IEEE International Conference on Communication Workshop (ICCW)*, London, 2015, pp. 363-368.

[53] M. T. I. ul Huque et.al., "Revisiting the controller placement problem," in *IEEE 40th Conference on Local Computer Networks (LCN)*, Clearwater Beach, FL, 2015, pp. 450-453.

[54] K. S. Sahoo et.al., "Optimal controller selection in Software Defined Network using a greedy-SA algorithm," in *3rd International Conference on Computing for Sustainable Global Development (INDIACom)*, New Delhi, 2016, pp. 2342-2346.

[55] A. Sallahi; M. St-Hilaire, "Expansion Model for the Controller Placement Problem in Software Defined Networks," in *IEEE Communications Letters*, vol.PP, no.99, pp.1-1

[56] T. Y. Cheng et.al., "QoS-Guaranteed Controller Placement in SDN," in *IEEE Global Communications Conference (GLOBECOM)*, San Diego, CA, 2015, pp. 1-6.

[57] P. Vizarreta et.al., "Controller placement strategies for a resilient SDN control plane," in *8th International Workshop on Resilient Networks Design and Modeling (RNDM)*, 2016, Halmstad, pp. 253-259.

[58] Sheng Guo et.al., "Towards Controller Placement for robust Software-Defined Networks," in *IEEE 34th International Performance Computing and Communications Conference (IPCCC)*, Nanjing, pp. 1-8.

[59] S. Lange et al., "Heuristic Approaches to the Controller Placement Problem in Large Scale SDN Networks," in *IEEE Transactions on Network and Service Management*, vol. 12, no. 1, March 2015, pp. 4-17.

[60] A. Giorgetti et.al., "Dynamic restoration with GMPLS and SDN control plane in elastic optical networks [Invited]," in *IEEE/OSA Journal of Optical Communications and Networking*, vol.7, no.2, Feb. 2015, pp. A174-A182.

[61] S. Das et.al., "Why OpenFlow/SDN Can Succeed Where GMPLS Failed," in *European Conference and Exhibition on Optical Communication*, OSA Technical Digest (online) (Optical Society of America, 2012), paper Tu.1.D.1.

[62] Katz, L. *"A New Status Index Derived from Sociometric Index"*. Psychometrika, 1953, pp. 39-43.

[63] P. Berde et al., "ONOS: Towards an Open, Distributed SDN OS," in *Proceedings of the Third Workshop on Hot Topics in Software Defined Networking*, HotSDN '14, Chicago, Illinois, USA, ACM, pp.1-6.

[64] "FlowForwarding/LINC-Switch", GitHub, 2017. [Online]. Available: https://github.com/FlowForwarding/LINC-Switch. [Accessed: 23- Jan- 2017].

[65] S. Knight et.al., "The Internet Topology Zoo," in *IEEE Journal on Selected Areas in Communications*, vol. 29, no. 9, October 2011, pp. 1765-1775.

[66] Youngtak Kim et.al., "GLASS (GMPLS Lightwave Agile Switching Simulator): A Scalable Discrete Event Network Simulator for GMPLS-based Optical Internet," *White-paper*, NIST-Gaithersburg, USA, 2002.

[67] A. Jarray and A. Karmouch "Periodical auctioning for QoS aware virtual network embedding," in *Quality of Service (IWQoS)*, 2012 IEEE 20th International Workshop, Coimbra, pp. 1-4.

[68] A. Jarray et.al, "Column generation based-approach for VN aware Networked Edge Data-Centers" , in *IEEE Globecom Workshops, Austin*, TX, 2014, pp. 93-98.

[69] D.Eppstein, "Finding the K Shortest Paths", in *SIAM J. Comput.*, Philadelphia, PA, USA, 1999, pp. 652-673.

[70] Stacy Patterson et.al., "Serializability, not serial: concurrency control and availability in multi-datacenter datastores, " in *Proceedings of the VLDB Endowment (PVLDB)*, 2012, Vol. 5, No. 11, pp. 1459-1470

[71] "Scalable Simulation Framework", Ssfnet.org, 2017. [Online]. Available: http://www.ssfnet.org/homePage.html. [Accessed: 23- Jan- 2017].