UTS UNIVERSITY OF TECHNOLOGY SYDNEY

Path Establishment in Software Defined Optical Networks

We propose that initiation of an adaptation or middle-ware system is required to manifest the optical network as an SDN capable domain

by Sadia Qureshi

Thesis submitted in fulfilment of the requirements for the degree of

Doctor of Philosophy (Engineering)

under the supervision of Prof. Dr. Robin Braun

University of Technology Sydney
Faculty of Engineering and Technology.

February 2023

Certificate of Original Authorship

I, *Sadia Qureshi*, declare that this thesis is submitted in fulfilment of the requirements for the award of *Doctor of Philosophy*, in the *school of Electrical and Data Engineering* at the University of Technology Sydney.

This thesis is wholly my work unless otherwise referenced or acknowledged. In addition, I certify that all information sources and literature used are indicated in the thesis.

This document has not been submitted for qualifications at any other academic institution.
This research is supported by the Australian Government Research Training Program.

Signature: Production Note:
Signature removed prior to publication.

Date: 10/3/2023

This thesis is dedicated to my beloved parents (Mr and Mrs. Abdul Qadeer Qureshi)

Included publications:

1. "Dynamic Light Path Establishment In Switch Fabric Using OpenFlow", published in ICSEng, 2018.
2. "Mininet Topology: Mirror of the Optical Switch Fabric", published in ITNAC, 2019.
3. "Dynamic LightPath Allocation in WDM Networks Using an SDN Controller", published in IEEE Access open journal in 2021.

Permission for inclusion of above papers respectively:

1. Copyrights are with IEEE. Reprinted, with permission from Sadia Qureshi and Robin Braun, "Dynamic Light Path Establishment In switch Fabric Using OpenFlow", 2018.
2. Copyrights are with IEEE. Reprinted, with permission from Sadia Qureshi and Robin Braun, "Mininet Topology: Mirror of the Optical Switch Fabric", 2019.
3. Copyrights are with CreativesCommon.

   See `https://creativecommons.org/licenses/by/4.0/`.

Status of Publications:

All three publications have been already published.

Extent of Contribution of Authors:

All the research work, implementation and write-up has been done by the primary author and co-author has supervised this work.

Signatures:

Primary Author:

Co-Author:

Acknowledgments

Firstly, I would like to express my deepest gratitude and appreciation to my supervisor, Professor Robin Braun, whose support, consistent encouragement and guidance helped me to fulfill my ambitions. He has been my mentor, and without his kind help, availability, and accessibility, it would not have been possible for me to achieve this goal. He taught me to stay motivated, be a positive thinker, and be logical. Working with him was indeed a privilege and a unique learning experience. I am highly grateful for his valuable guidance, time and support. I am also thankful to my co-supervisor, Dr. Zenon Chaczko, for his help and sparing his priceless time for me whenever I needed it.

I want to express my sincere gratitude to my assessment panel (Dr. Mehran Abolhasan and Dr. Daniel Franklin) for their guidance and profound help. I am indebted to UTS and the Commonwealth of Australia for providing me international student academic research scholarship (IRS) to support my Ph.D. degree financially.

I want to acknowledge my friends' debt to me for their support over the last few years. I would like to thank Pakawat, Durga, Hina, Farhana, Tanzila, Ammara, Maria and Madeha for their patience and understanding, especially in my critical times. I have been fortunate enough to have their company and enjoy happy chats with them during my Ph.D. study.

Lastly, I am thankful to my parents, husband and siblings for their continuous moral support and help in various ways. Their encouragement always made me keep going and achieve that extra mile which led to this achievement. I am truly indebted to my parents, who sacrificed to give me the best and whose prayers protect me and keep me in peace. I am blessed to have three beautiful daughters whose presence I cherish; they are the happiness of my life and motivation to keep achieving. I am thankful to my daughters for the joy and cheers they have have given to me as their support.

Abstract

A newly emerged networking paradigm called Software Defined Networks (SDN) is an architecture to overcome the problems with conventional data networks. SDN renders a solid potential for flexible network control and management. In particular, the standard OpenFlow protocol is often referred to as a primary new notion in the networking domain. SDN presents the idea of separating the control plane decisions from the data forwarding plane and gives the concept of programmable networks by forming a standard programming interface. A centralized controller makes routing decisions, while the switches only perform forwarding actions. It allows testing new protocols and algorithms to control data packets. The SDN technology provides ease of control of the network's infrastructure, is cost-efficient and open and enables programmable components.

SDN is very efficient at providing flexible control in packet networks. However, in optical networks, circuit-switched paths are planned and set up much before the arrival of any data and any change of route or service providers will require a long, slow process and manual intervention. Previous solutions (GMPLS/ASON etc.) to automate optical networks have become very complex. Adopting SDN for optical networks requires initiating an intermediate system to manifest the network as an SDN-capable domain.

This research aims at developing an intermediate system that virtualizes layer one optical network to the SDN controller as a network of layer two OpenFlow switches. It allows the programmability of paths through optical switches in an on-demand fashion using OpenFlow protocol and eliminates the need for traditional remote and local decision-making algorithms. This contributes to the dynamic service provision in optical networks. Our model offers a translation mechanism between the controller and optical switches so that the controller can control path computation without being informed of underlying structure and wavelengths. Adding programmability to the network environment will bring scalability, adaptability and robustness. The translation mechanism is integrated into the middle of the SDN architecture to

provide an abstraction layer to manage the flow setups, allowing them to adapt dynamically according to real-time demands and needs. The component-based model provides room for adding more features in the future.

This dissertation is presented in three parts. In part 1 motivation for the research and the problem background are presented. The proposition is proved in the second part of the thesis (chapter 4) by implementing the mode. Finally, the project is concluded in part 3 of this dissertation, and future directions are provided (chapter5).

Contents

List of Figures

Nomenclature

API Application Programming Interface

ASON Automatic Switched Optical networks

BGP Broader Gateway Protocol

CLI Command Line Interface

EMS Element Management System

EON Elastic Optical Networks

GMPLS Generalised multi-protocol Label Switching

GUI Graphical User Interface

LLDP Link Layer Discovery Protocol

LSP Label Switched Path

MINs Multi-Stage Interconnections

OCS Optical Circuit Switches

OEO Optical-Electrical-Optical

ONF Open Networking Foundation

OOO Optical-Optical-Optical

OTN Optical Transport Network

OXC Optical Cross Connects

PCE Path Computation Element

RMS Resource Management System

RPC Remote Procedure Call

RTT Round Trip Time

RWA Routing and Wavelength Assignment

SNMP Simple Network Management Protocol

STP Spanning Tree Protocol

TDM Time division multiplexing

VMs Virtual Machines

WCC Wavelength Continuity Constraint

WRA Wavelength Routing and Assignment

WRN Wavelength Routed Networks

ROADMs Reconfigurable Optical Add Drop Multiplexers

SDON Software Define Optical Networks

Part I.

Giving weight to the proposition

1. The Introduction

1.1. Overview

This chapter describes the research topic, investigation methodology, scope and objectives. The main goal of this research is to develop a dynamic system for path establishment in core optical networks and analyze this system's performance. Furthermore, the research motivations are presented, and after identifying the gap in the literature, the research questions are defined and stated. Overall thesis structure is presented at the end of the chapter.

1.2. Background of Optical Networks

In the Open Systems Interconnections (OSI) model for a traditional data communication network's layered structure, the lowest layer is the physical layer which essentially provides a physical medium for the transmission of information. The physical transmission medium can be optical, air, or electrical (e.g., copper cables). Optical networks offer physical layer services and provide a solution to the problem of the network's capacities. Optical fiber offers large bandwidth and fewer losses. As a result, it is the chosen medium when data transmission is required at more than a few tens of megabits/sec speed or distance over a few kilometers.

The evolution of optical networks can be categorized into two generations. In the first-generation, optics was only used for data transmission purposes and to provide a pipe of bandwidth since it is efficient in terms of lower bit error rates and higher capacities than copper cables. Another technique to further increase the channel capacity is Wavelength Division Multiplexing (WDM) which resulted in the second generation of optical networks.

In first-generation optical networks, the intelligent network functions were handled in the electrical domain by the electronic modules inside network devices. Example of first generation optical network is Synchronous Optical Network (SONET) [1] or the similar Synchronous Digital Hierarchy (SDH). In SONET/SDH, the lower-speed data streams are multiplexed together into a higher-speed stream at the transmission bit rate to utilize the optical capacity. The multiplexing is done by employing electronic Time Division Multiplexing (TDM) at SONET/SDH switches may also drop some of the streams at the switch). The bit streams are then converted to an optical signal (single lambda) to be transported over the fiber [2][3].

Realization of the second-generation optical networks became possible because of the capacity provided by the multiplexing technique called wavelength division multiplexing (WDM). Using this technique, different wavelengths of the light with each carrying different data stream, are combined and transmitted over a fiber. These wavelengths are kept sufficiently far apart and hence do not interfere with each other. Thus WDM composites multiple signals and it appears that there are multiple fibers inside a single fiber and each fiber (virtual) carries a stream of data. In contrast to TDM, where signals are distributed across time slots, WDM signals arrive at the same time and also each wavelength can carry a different signal with its own speed and protocol independent of other wavelengths.

Optical fibers can carry massive bandwidth, while the first-generation optical networks had limited ability to benefit from this advantage. For this reason and also to exploit other benefits of the optical domain, second-generation optical networks (WDM) became popular. One of the significant advantages achieved with WDM is that it incorporates some of the switching and routing tasks initially performed by electronics. The network provides optical circuits called lightpaths to connect network nodes, such as SONET switches or IP routers. Optical circuits span over the source to the destination and cross over multiple intermediate nodes. The network exploits the WDM technique at intermediate nodes to route and switch signals in the optical domain. Sometimes wavelength of the lightpath may also need to be changed along the route, for which suitable equipment is required [4].

Optical networks with WDM systems are vastly deployed today in long-haul backbone and core networks, undersea networks in metropolitan networks [5]. Second-generation of optical networks added a new layer to the OSI layered hierarchy, which is WDM optical layer, also called layer 0, see Figure 1.1. However, it is also regarded

as part of layer one or sometimes layer one itself. As mentioned earlier, the WDM layer is a protocol and bit rate independent, so it can serve to transport ATM (Asynchronous Transfer Mode), SONET, Ethernet, Frame Relay, and IP packets simultaneously; see, Figure 1.2.

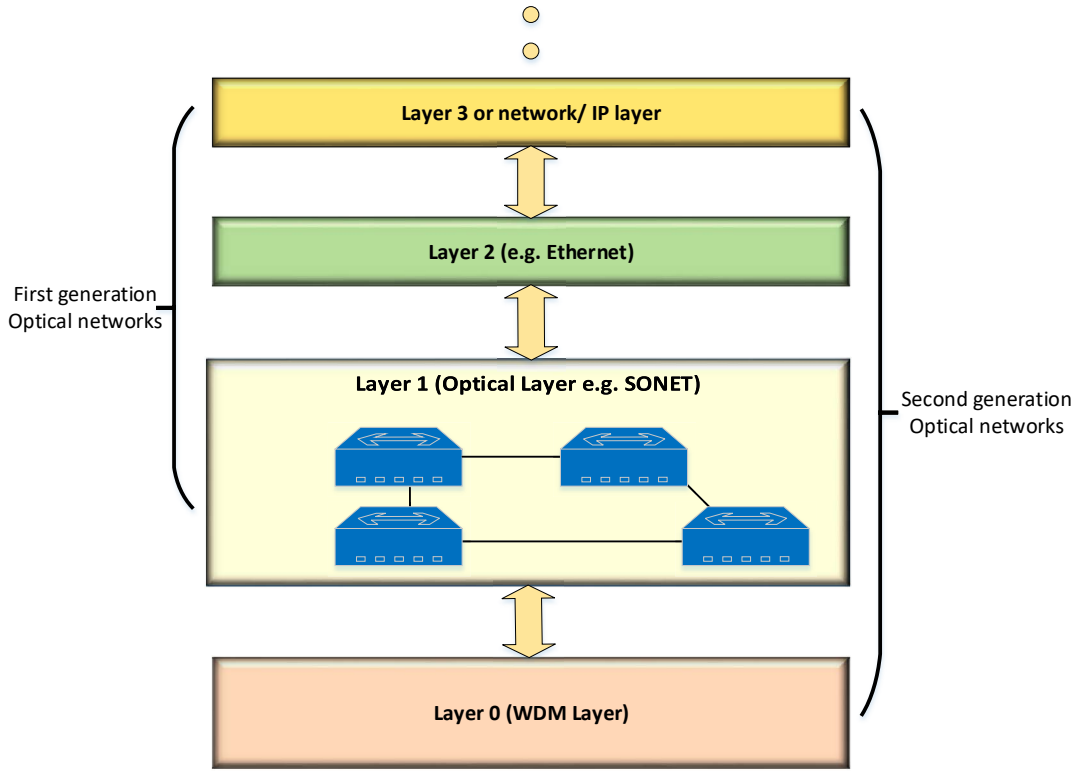


Figure 1.1.: First and second generation optical networks

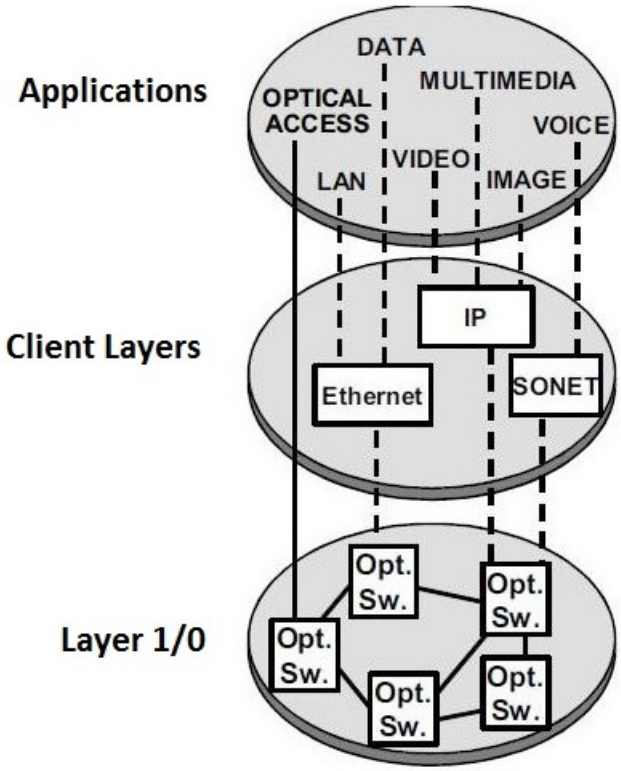


Figure 1.2.: Client layers[6]

1.2.1. Optical Networks Management

Network management is crucial for any service-providing data network. The main recurring cost of operating a network is its management cost, and in many cases leads to the deployment cost of the network. Therefore network management is given special attention to reducing cost while its complexity remains the bottleneck issue.

Figure 1.3 shows an idea of how network management methods are implemented in a typical optical network. Management is performed through a centralized Network Management System (NMS) involving multiple lower-layer systems. The primary central monitoring system is at the highest level and is connected to several lower-level management systems called Element Management Systems (EMS) [7]. Net-

work elements that need to be managed include optical line terminals (OLTs), optical add/drop multiplexers (OADMs), optical amplifiers, and optical cross-connects (OXCs). Each device is managed by its EMS. The devices are deployed with the built-in agent, which interfaces with its respective EMS. The agent is usually implemented in the firmware of the element. The EMS manages its associated elements by communicating with them through a Data Communication Network (DCN). This DCN can be any existing traditional network.

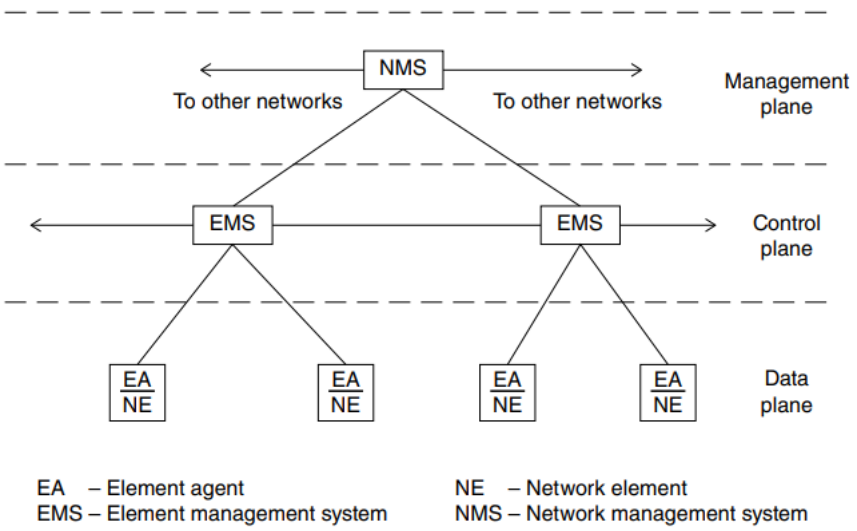


Figure 1.3.: Typical optical network management framework [8]

The EMS is generally coupled to one or more of the network devices. Apart from the DCN, a fast signaling mechanism is also required between network devices to exchange real-time urgent control information. This can be a control signal, e.g., the optical supervisory channel (OSC). A dedicated wavelength is used to carry out control functions. To manage the entire network, multiple EMSs are necessary. Mainly an individual vendor's network elements are supervised by a single EMS. Thus if the line systems and the cross-connects are from different vendors, then two EMS will be required to manage each vendor's devices. The EMS can view only its reporting network elements and does not have a complete network view. Hence, for the overall management, all of the EMSs communicate with and report to the NMS. The NMS then manages all types of network elements. Additionally, a simplified local management system is provided to the craftspeople and other network service personnel to configure and manage individual network elements. The local management system is generally available either as a pluggable terminal

that can be plugged into individual elements or on a local machine, such as a laptop. This type of management model represents a complex way of management which does not allow growth of the network.

1.3. Motivations for this research

1.3.1. Motivation from Complex Management of Optical Networks

The first research motivation came from the consideration of complex, centralized and hierarchical management of the optical core networks, which are required to support multiple super high-speed services. Also, the current static network service provision system cannot cope with today's dynamic needs of the enterprise sector and end users. The size and complexity of existing networks are increasing, thus leading to significant challenges in the network's management, operations and maintenance. One goal of this research is to study the growth of modern technologies in optical core networks that would be beneficial in the future. For example, these networks include software-defined optical networks and elastic optical networks. These networks are expected to change our everyday life in the future years. It is presumed that high-end devices (e.g., routers and switches) will become programmable and software-controlled. Typical management methods are often inefficient and require excessive manual intervention. The research results can be used to design a system to manage complex optical networking systems, specifically service provision and path management. This thesis studies and examines the relationship among managed elements, switch control algorithms and path establishment protocols, and applies an on-demand strategy to achieve an effective dynamic control plane for SDN-based optical networks.

1.3.2. Motivation from OpenFlow-Based SDN

Current network management must shift towards a more flexible network control and management. SDN has emerged recently and is being developed and promoted

by the Open Network Foundation (ONF). SDN is a newly emerged computer networking architecture in which the control plane is isolated from the data plane and implemented in a centralized software called the controller. This setup allows network administrators to have programmable and centralized control of network devices and traffic without requiring them to access them individually when network behavior needs to be changed. The separation between SDN's control and data plane offers new models and paradigms in managing networks, such as developing autonomous behaviors. With SDN deployment, networks can be more flexible and cost-effective by improving network resource utilization and reducing operational costs. This research is inspired by the development of an OpenFlow-based SDN network, which allows researchers to quickly deploy their experiments and run new and innovative layer two and three protocols in the local area and campus networks we use daily. OpenFlow is a standard open-source protocol, and recently SDN and OpenFlow have been adopted in many industries for WAN and LAN. Since L2-L4 are already migrating to be software-operated, carriers want optical networks SDN-operated too. This research is an effort to implement and test SDN controller-based path control for optical networks and exploit the advantages of the on-demand behavior of SDN and OpenFlow protocol.

1.4. Research Objectives and Scope

The goal of this research is to apply SDN principles to the optical network architecture to yield a logically centralized and dynamic control plane for Software Defined Optical Networks (SDON). The idea of SDN-based optical networks is proposed as a favorable solution to aid the optical core networks' increased complexity and bandwidth demand. The key is to design an intermediate system that will serve as an intermediate system between lower-level switches and higher-level software applications, allowing to translate of lower layer information and demand of network elements. This study will provide the methodology, structure and algorithm working to support dynamic path establishment. Once the model is developed and tested in this dissertation, the research results will create a room that enables automatic path establishment capability to be introduced in wavelength-based optical networks.

1.4.1. Research Goals

1. The primary objective of this research is to develop a foundation structure for dynamic path establishment in optical core networks, which can integrate with the standard OpenFlow-based SDN controller and optical switches, and to produce the theoretical model required for the implementation of such a system. The structure will enable the switches to proactively invoke path establishment requests to the controller.

2. To produce the methodology and algorithm for the flow of operations that will help implement a desirable path establishing the centralized system and to simplify complex and slow management of optical entities and networks. The model will serve as an abstraction layer to map the connectivity of layer one network. After designing and implementing a basic model in this thesis, it can be improved for variants of optical networks, e.g., with or without wavelength converters or establishing protection paths in cases of failures.

3. To investigate and test the developed model and the required resources for its implementation. Also, measure the performance in terms of latency for path establishment. To record the observations that come along this journey and are not directly related to the project objectives.

The expected result of this research is a successful implementation of the model which works for an SDN-controlled optical network. This will enable to control of paths in optical networks and cover them under the umbrella of SDN like other layers of the OSI model. The model is implemented using python scripts and a Mininet emulator with a local SDN controller. The model is to be tested with a layer one switch.

1.4.2. Scope

The main areas involved in this research are

1. Theory and background of SDN and OpenFlow protocol: Separation of control and forwarding planes in the SDN paradigm is investigated, and also how OpenFlow (OF) protocol implements the SDN paradigm is studied. The investigation involves the study of the implementation of SDN in layers 2-4, OpenFlow protocol details, and communication between control and data plane, e.g., Packet In, Packet Out messages, versions and flow of steps. Various controllers and OpenFlow protocol versions are studied, and choosing appropriate ones for implementation of this research proposal. Mininet has its reference controller, which it uses by default; however, an external controller like NOX/POX can also be passed to Mininet. Similarly, a remote controller can also be used, placed at some other place, e.g., in a LAN. This work will involve a deeper investigation of some well-known OF controllers for establishing paths in layer one.
2. Theory and Background of Optical Layer and SDN in optical layer: Study of management protocols and path establishment processes, e.g., control algorithms, types of optical switches and their internal fabrics. Study of architectures for the dynamic behavior of optical networks, history and overview of SDN-based optical networks.
3. Implementation in Mininet: Exploration and experimentation with Mininet, its use with controllers and observing OF protocol behavior. Measurement of resources used by Mininet. Integration of Mininet with other systems. It implements the proposed model in Mininet, i.e., as an intermediate system.
4. Computational performance measurement tools: Measuring performance of the model in terms of CPU, memory utilization and latency.

With the implementation of a proposed solution, some sub-areas will be investigated, e.g., software OF switches and other types of architectures offered for future optical networks, e.g., Elastic Optical Networks and their properties.

1.5. Problem Statement

In optical core networks, service provisioning and path establishment is a complex,

expensive and time-consuming process, particularly in vendor-specific environments. At a centralized Network Management System, algorithmic computations are performed and implemented through a lower layer of element management systems and a data communication network. Specially trained technical staff and network administrators are required to convert higher-layer computations and policies into manually configured commands across the lower-layer devices. Optical devices, e.g., ROADMs are configured physically by I.T. personnel to provide static paths for service provision. This is a challenging and complex job; consequently, network control and management, along with the other tasks associated with providing services, have always been complex and fallible.

A rapid change in traffic behavior over the past few years and the increasing demands due to the growth of the network traffic has caused bottlenecks in the conventional static optical networking paradigm. In such a situation, dynamically managing the growing traffic demands is challenging but is of great importance to the carriers and the researchers in the optical domain.

Moreover, a centralized network management system is also considered a single failure point. As a result, some specific functions (e.g., service monitoring) were initially implemented as the distributed control plane function; later, two distributed control plane architectures, i.e., ASON and GMPLS, were developed. However, the distributed architecture turned out to be more complex to implement.

Above limitations, a settlement between the centralized and distributed control and management architectures, along with the recent advancements in traffic demands, have steered up the concept of an innovative, dynamic control plane for optical networks, primarily in the context of SDN. The separation of data and control planes of the traditional networking paradigm characterizes software-defined networking. SDN allows the programming of forwarding elements through a centralized controller. SDN controller performs global tasks such as topology discovery and path computation; policies are implemented by pushing the rules for individual devices using standardized protocols. By deploying SDN, network operators can achieve network automation, and flexible network control, enabling them to build highly scalable and programmable networks that serve developing business needs.

SDN has already been adopted for packet technologies (L2-L4) and seems feasible for Layer one electronic switching. However, the implementation of SDN in optical wavelength switching networks is still a vital challenge. It is being investigated, but

it has already been studied as advantageous over other standards for multi-layer management of IP and optical networks.

Implementing SDN in optical networks might require the initiation of an adaptation or middle-ware system to manifest the network as an SDN capable domain while employing under lying control plane functionalities [9].

Although SDN is inherently able to contribute to on-demand behavior in terms of network service provision and aids in network control simplification, which leads to innovation and evolution, SDN also opens up a set of major challenges in the optical layer, such as service orchestration and performance. How to take in and operate SDN reliably using a middle-ware mechanism? Can SDN provide enough functionality required for network path allocation, and where should SDN be placed to implement path establishment based on different design architectures of optical devices? How can network monitoring be adapted to such static and pre-allocated resource networks? This research addresses some of the these and related questions in OpenFlow-based SDON.

1.5.1. Research Questions

This research aims to develop and implement a model that allows path computation and allocation in optical switches using an SDN controller with OpenFlow as the standardized protocol. From the research problem, the primary research question can be deduced as, " Is it possible to dynamically allocate physical layer resources on the arrival of data on optical device's input ports, with a control plane technique which can view the network, communicate, control and program it whenever required". Or other ways of stating the same question could be

1. Can an SDN mechanism be used for dynamic provisioning of resources in wavelength-switched optical networks?

2. Can physical layer switches be made intelligent enough to communicate with a central manager (SDN controller) and follow its commands to setup paths through fabric and providing end-to-end connectivity dynamically through optical layer without modifications in standard OpenFlow protocol?

3. Is it possible to design optical networks to work in compliance with an on-demand protocol (OF) for circuit provisioning without introducing intolerable latency?

4. What advantages SDN offers, if it is introduced for path establishment in optical

networks?

1.5.2. Propositions

This dissertation describes the author's research effort over the last few years to address the limitations of the current static path establishment methods. This research attempts to develop a mechanism for dynamic path allocation in an optical switch. Moreover, the research results are expected to contribute to the management of SDON and the optical networking research groups. This work proposes an innovative effort to develop a connectivity abstraction layer between the optical network and the SDN controller. The research is supposed to use an OpenFlow protocol as a standard communication protocol for Software Defined Networking and proposes:

1. That a Software Defined Networking paradigm for controlling the optical paths without extending OpenFlow protocol, can be implemented.

2. That the OpenFlow protocol can be used to program and set up layer one switch and its fabric, and that the optical switch can notionally act as a network of layer two switches to the SDN controller.

3. That SDN can allow dynamic optical path establishment on the first-time arrival of the data signal and on a request from the optical node itself. And that it can allow autonomic service provision in future optical networks and a single control mechanism for the multi-layered networks.

1.6. Approach and Methodology

This section describes the approach and methods adopted for identifying problems and analyzing gaps, followed by proposing a candidate solution to resolve the identified issue. The next step is implementing the designed model, running experiments and verifying the results. Finally, the system is tested to demonstrate that the implemented mechanism is operating functionally, and results are discussed. This thesis enumerates sets of activities and tasks to obtain results that will add significance and check (prove/disprove) the propositions listed in the following sections.

1.6.1. Literature Review

The purpose of reviewing the literature is to identify the problems of wavelength-

routed networks regarding an optical path establishment and analyze them to find research gaps. The focus of the literature study is to evaluate previous research and to propose a new suitable and beneficial system for development. Furthermore, it targets to analyze several theoretical and practical aspects of alternate approaches. Therefore, various other research works and previous implementations proposed to establish optical paths dynamically have been reviewed in detail. The research works in the past few years have been briefly reviewed and examined to produce new research proposals and compare variations and similarities in related works. The investigation is conducted in the domain of optical core networks, components and their functionalities, including their internal structure, algorithms and control protocols. This survey also studies state of the art in the programmable optical layer, which targets particularly OpenFlow-based solutions and discusses other's prospects from earlier proposals to the latest developments. The examination is done on the alternate solutions regarding the implementation, testing services and protocols used and their shortcomings. This research study used standard tools and resources, for example, various university library databases, online research articles, books and another relevant thesis. Hence the strategy to structure this literature review is to identify the requirements of a candidate solution, which needs the analysis of network services that can reform the problems in the current core optical networks. The review and study are examined to clarify the gaps, which provides research motivation.

1.6.2. System Design

After defining the research gaps, the next step is to facilitate designing the proposed solution. Developing a candidate solution is considering a plan for a research software solution that can solve the issue. Usually, the most challenging and visionary step of the research process is to design the system due to the reason that it is demanding and very important, which requires technical exercises, for example, defining the required output of the system, finding requirements to produce the output, specifying the software skills to be learned, documenting all the steps of the design process etc. Here the prime purpose of the design phase is to create a middle-ware model which is feasible to be implemented in some programming language. Here, two major steps are involved in the design activity. Initially, the architecture of the high-level design is decomposed into components for different functionalities; the components are then aggregated and bonded together in an application using a

programming language. The algorithms and data structure developed for the components are also discussed and documented. It is essential to specify well-defined requirements for the proposed model, which employs describing system lay out the functionalities that the system has to perform. The aim is to implement the model as a middle-ware translator between the SDN controller and optical switches.

1.6.3. Emulation

For emulation, a network of OpenFlow layer two switches mapping an optical switch and scaling up fabric topologies is specified in Mininet (virtual machine environment). Mininet is a Linux-based virtual networking environment that can handle IP test packets, e.g., FTP transfer or Ping etc. To allow scalability, instead of running multiple VMs; Mininet uses Linux processes in the network Namespaces. It allows you to make an emulated network of OpenFlow switches rapidly and provides a direct pathway to deployment on hardware. No changes are required for the programs that are evolved and tested on Mininet for an SDN controller, OpenFlow switch or a host when moving to a real network for real-world deployment, testing, or performance evaluation. After building up the Mininet network and abstracting the optical network, the intermediate system can be tested by receiving triggering data from the layer one switch. The intermediate system forwards the relevant information to the controller; the controller calculates the path for the layer two network and passes flow entries to OpenFlow switches. In this method, the controller manages the layer one switch without knowing the actual layer one network. To start testing the model, we will need a real hardware switch, either an optical switch or a switch with layer one characteristics, i.e., switching data without parsing it an OpenFlow controller. According to [10], Mininet is a network emulation platform that helps research, testing, prototyping, development, and debugging and supports other complex experimental system-related activities on a single device. Mininet gives a straightforward, uncomplicated, low-cost test bed for testing and deploying OpenFlow applications. Create a virtual OpenFlow network; it uses virtualization features at OS-level, involving network namespaces, Linux processes, and virtual Ethernet pairs. In a virtual machine environment, Mininet can quickly prototype, modify, and share SDN protocols and applications with up to hundreds of network nodes upon gigabits of bandwidth. Mininet is a great way to learn SDN and OpenFlow and test SDN controllers and applications on large topologies. Examples of components emulated by Mininet with lightweight virtualization are switches, hosts,

links, and controllers in the OpenFlow network.

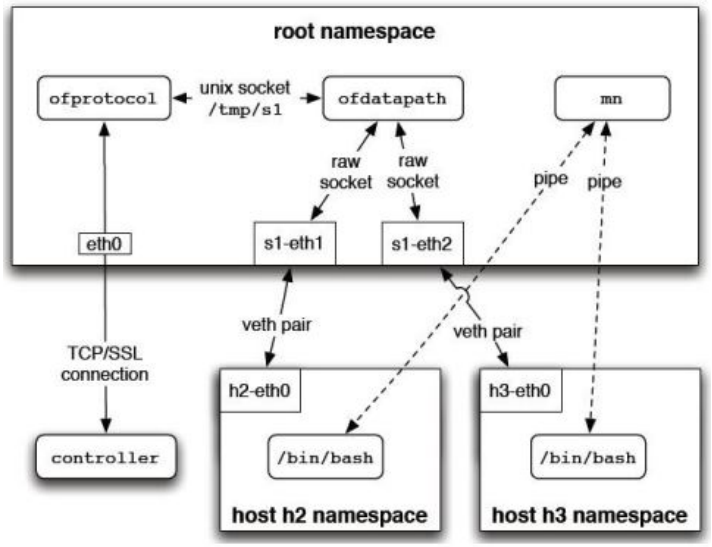


Figure 1.4.: Two host network created in Mininet [10]

Hosts: In Mininet, host is the namespace that has a network state. It processes packets through interfaces, ports, and routing tables like they would have processed through a real machine. In order to send and monitor outputs and commands every host has a pipe associated to a parent Mininet process.

Links: Mininet link serve as a wire connection between two virtual interfaces of emulated switches or hosts. This is virtual Ethernet pair which functions as a full Ethernet port, and can send and receive packets from all system and application software.

Switches: Mininet switches are software-based OpenFlow switches, which processes packet delivery same as a hardware switch. They can be either kernel-space and user-space switches. Examples are Open vSwitch and OpenFlow reference switch.

Controllers: For Mininet network, controllers can be internal (such as Mininet's default OF controller) or external (such as a controller running on LAN or another virtual Machine or cloud) and needs IP connectivity to the controller.

MiniEdit: This is a graphical user interface and editor to visually create and run network and open ups CLI for network components. Components properties can also be customized in this editor.

1.6.4. Implementation

The main goal of the implementation phase is to implement the proposed model based on the developed theory (described in Chapter 3). In this phase, the focus is to implement the potential solution in the best possible way. The first critical step to successfully implementing the solution is to put the system's specifications in order. The system's components and architecture must be well-identified and organized, including the controller. A detailed and profound study of the functionalities of the controller, OpenFlow protocol, path computation process and applications required to implement the model efficiently. This new model must mimic layer one connectivity and can process incoming triggers and notifications locally, forwarding to a centralized controller for switching decisions. This model is implemented using Using the RYU controller. RYU controller is chosen based on the fact that it is fully open source and dictates the behavior of the SDN controller. It is a component-based controller with example applications that can be easily customized to the network requirements.

The primary programming language used to implement the model in Python. This work aims to develop a centralized control plane for path establishment by integrating a middle-ware translation mechanism. This will enable the optical switches to work the way they are already; however, they are being controlled for path switching by an SDN centralized controller. The core modules that construct the middle-ware system comprise the Emulation and Translation module, the Information collector module, the Spoofing module, and the Path-Setup module, which are essential for the path established between the source and destination switches. Chapter 4 presents further details.

1.6.5. Validation

After implementation, a set of tasks needs to be conducted to check the overall model's compliance. These tasks involve testing and verifying that the model is working functionally and fulfills requirements and specifications that meet its expected objectives and measuring the model's performance. The method for testing is to check the model physically with the Cisco layer one switch and generate a network topology depicting its internal switch fabric as a Layer two network made up of OpenFlow switches in Mininet to demonstrate that the model can be deployed in real networks. Performance is evaluated using the Ping tool to measure the path

establishment time. CPU and Memory usage for the topology creation has also been considered a measure of the model's performance. The next phase is to collect the experimental results to evaluate the performance, which will support the fact that the system is feasible for real optical devices. The model's performance is assessed in terms of CPU usage, memory usage, network graph creation time, and latency. Validation aims to verify that the implemented solution can deliver all functionalities as expected in accordance with the defined requirements in theory. Verification and validation are two different but equally important phases. Verification is required to evaluate the code, documentation, requirements and specifications to find whether they fulfill the specified demands for that phase. On the other hand, validation evaluates the proposed solution at the end of its development process to check whether it meets specified business demands. The inputs for the verification phase are illustrated as checklists or reviews, while the input for the validation process is the actual testing of the proposed solution.

1.7. Thesis Outline

This thesis is composed of five chapters and is organized as follows:

Chapter 2: Background and Literature Review

Chapter two provides a detailed review of related literature sorted in terms of important aspects and the current state of art in Optical Networking. Additionally, it gives an overview of the issues of managing complex optical networks, and service provision problems of core networks using NMS, and the ability of the switches to operate autonomously. Previous related work to resolve the static network issues are described, and their connections for the motivation of this study are also stated.

Chapter 3: Proposed Model Theory

This chapter presents the theoretical design of the model based on the Mininet structure, OpenFlow protocol and SDN controller. Details of the proposed solution, components required, and operational flow of the model are discussed, and its practical feasibility is also evaluated. Furthermore, the basic principles of the model, such as design principles, architectural foundations and implementation fundamentals, have been presented. The difference between path-finding in conventional networks and using the proposed solution has been described in terms of theory and implementation. The reason and motivation behind this research are brought up, which

justifies the model's contribution. The model is expected to dynamically find and establish the path for current optical networks. Requirements and limitations for such a system are also stated. This chapter discussed the theoretical details of path initiation request, path-finding and path setup process.

Chapter 4: Model implementation and Performance

This chapter includes two research articles. Both articles cover the implementation details of the model. In the first phase of implementation, the abstraction of the layer one network is performed. The first included article explains all the details related to the abstraction and why this abstraction is required. The second included article also starts with building up the background for this research and then presents all the details for model implementation. The core modules that construct the model are discussed with some sample code review diagrams of the implemented algorithms. The developed modules include the information collection module, the spoofing component and the path setup component. To check the conformity of the implemented system with its fundamental purpose , the model needs to be validated with experimental tests. Hence, different topologies are experimented with to test the model's functionality.

The second article presents the technical details and testing procedure to experimentally evaluate the model. Furthermore, the type and scale (stages) of the topologies used for the performance evaluation are discussed. Performance of the model is assessed by four characteristics: Ping delay, CPU usage, Memory usage and Topology creation time. The ping command is executed to measure the average RTT (Round Trip Time) values, to quantify model's performance. Further, "free" command is used to measure the amount of memory utilized and "ping" tool for measuring latency.

Chapter 5: Conclusion and Future Directions

Outcomes and conclusions are summarized here and future work is also stated.

2. Background and Literature Review

2.1. Overview

Social needs of the people to communicate among themselves and demands for new applications have evolved the essential telephony network to the current fast and wide-area networks. Today's communication networks are rapidly changing owing to the user's requirement to connect to the world anywhere and anytime. Real-time communications, such as video-conferencing, interactive gaming and other multimedia and internet services, are all bandwidth-demanding. Besides bandwidth demand, the network beneath must be stable and cost-effective. Therefore, a communication network requires high-capacity transmission, which is fast, reliable, dynamic, and economical. Optical fiber is the technology of choice that suits most of the above demands.

Optical fiber has gigantic bandwidth and holds fewer losses and costs. It has strength, flexible, is not hefty and combats noise and interference. It is safe and has other characteristics as well, which makes it an ideal transmission line for high-speed data networks. Thus, optical fiber is the most acceptable solution to fulfill the necessities of a communication network. A vast quantity of optical fiber was buried everywhere on the earth in the twentieth century, which made the base for today's wide-bandwidth highway of information. To increase the capacity, coaxial and copper wires were replaced by optical fibers. Later, to improve further, several wavelengths carrying multiple data streams in a single fiber were combined, called wavelength division multiplexing (WDM). However, network switching and routing operations were still to be done in the electrical domain. Hence, it was not an all-optical network.

Each fiber communication device can provide routing, information transmission, grooming, control and multiple restoration functions. There are many benefits of using the operation of a network at the wavelength granularity or optical level. For instance, it efficiently utilizes energy; i.e., it consumes less power than its electronic counterparts. Moreover, it also provides fast restoration of connections in optical networking. The current optical technology can only cost-effectively achieve some of the functions mentioned above. Thus, the hybrid optical network is used that utilizes the devices of both electronic and optical networks [11].

A few decades ago, communications using telephone networks were considered only voice traffic; however, currently, it carries both voice and data traffic by supporting ultrafast, low-latency, data-driven services. Now, the components involved in optical WDM networks can support data rates of Tbps such that each fiber can support multiple wavelengths. Furthermore, optical networking is becoming more elastic and proactive due to the advancements in the intelligent algorithms and communication protocols defined for optical networks, which improves the data demands of user equipment.

The recent trend in networking is shifting towards Software Defined Optical Networking (SDON) [12, 13]. SDON has more power in controlling networking functions by promoting the programmability of networking operations. For instance, it easily accommodates new services and improves the quality of services. Furthermore, it helps in enhancing revenue. For the past few years, SDON has been evolving rapidly; thus, the conventional optical networking paradigm is gradually shifting toward SDON. The rest of the chapter briefly overviews the details of the optical network background and SDN.

2.2. Optical Networks Background

Conventionally, optical networks are classified based on geographical area as local area networks (LANs), metropolitan area networks (MANs), and wide area networks (WANs). Typically, access networks cover up to around a 20 km radius, and the traditionally known wired access network variants are passive optical networks (PONs), Ethernet PONs (EPONs), Gigabit PONs (GPONs), WDM PONs (WPONs), etc. Furthermore, networks without wired connections, e.g., wireless-fidelity (Wi-Fi), worldwide interoperability for microwave access (WiMax), digital subscriber lines (DSL), and higher speed lines like T1 and E1, are also referred to

as access networks [14].

The networks that cover a radius of around a few or hundreds of km are called WANs. WANs mostly follow the ring topology by considering either asynchronous transfer mode (ATM), optical transport network (OTN), or synchronous digital hierarchy (SDH)/synchronous optical network (SONET). The network nodes responsible for transferring/receiving between access networks and WANs are metro network nodes; see Figure 2.1 (repeated in chapter 4 for reader's convenience).

The core or backbone network is responsible for global coverage by providing connections between access networks of one part of the country/world and user equipment leveraging LANs in any other part of the world. Globally, mesh topology is deployed for the core networks. They are also referred to as long-haul core networks. Core networks can carry very high data rates due to high bandwidth pipes and have typical speeds above 100Gbps.

Historically, optical networks have two generations. The first generation primarily focused on WDM technology to improve transmission capacity by disrupting nodes electronically for routing and switching functions. Here, the signals were transformed from the optical to an electrical domain to perform switching and routing tasks and then re-transformed back to the optical domain for transmitting optical signals to another node. Since high speeds like Tbps were not handled by electronic nodes efficiently (because of electronic processing speeds), the electronic nodes were considered a bottleneck of high-speed networks. The node traffic has to deal with its own traffic; that is, the one intended for it and the one it has to process or transmit to another node for other destination nodes. Typical examples of electronic switches are fiber distributed data interface (FDDI), ATM switch, and SDH/SONET electronic switches.

The limitation of electronic switches is overcome by the optical networks of second-generation, where signals are switched in the optical domain. The second generation optical nodes are capable of routing and switching functionalities by avoiding or bypassing the signals that are not intended for the nodes. For instance, optical cross-connectors (OXCs) and optical add-drop multiplexers leverage the benefit of high bandwidth due to exploiting the advantages of optical fibers. Thus high bandwidth advantage of optical fiber could now be better exploited. For this reason and other benefits, e.g., cost and lesser heat dissipation of optical nodes, second-generation optical networks (WDM) became popular. Such a network is called a wavelength

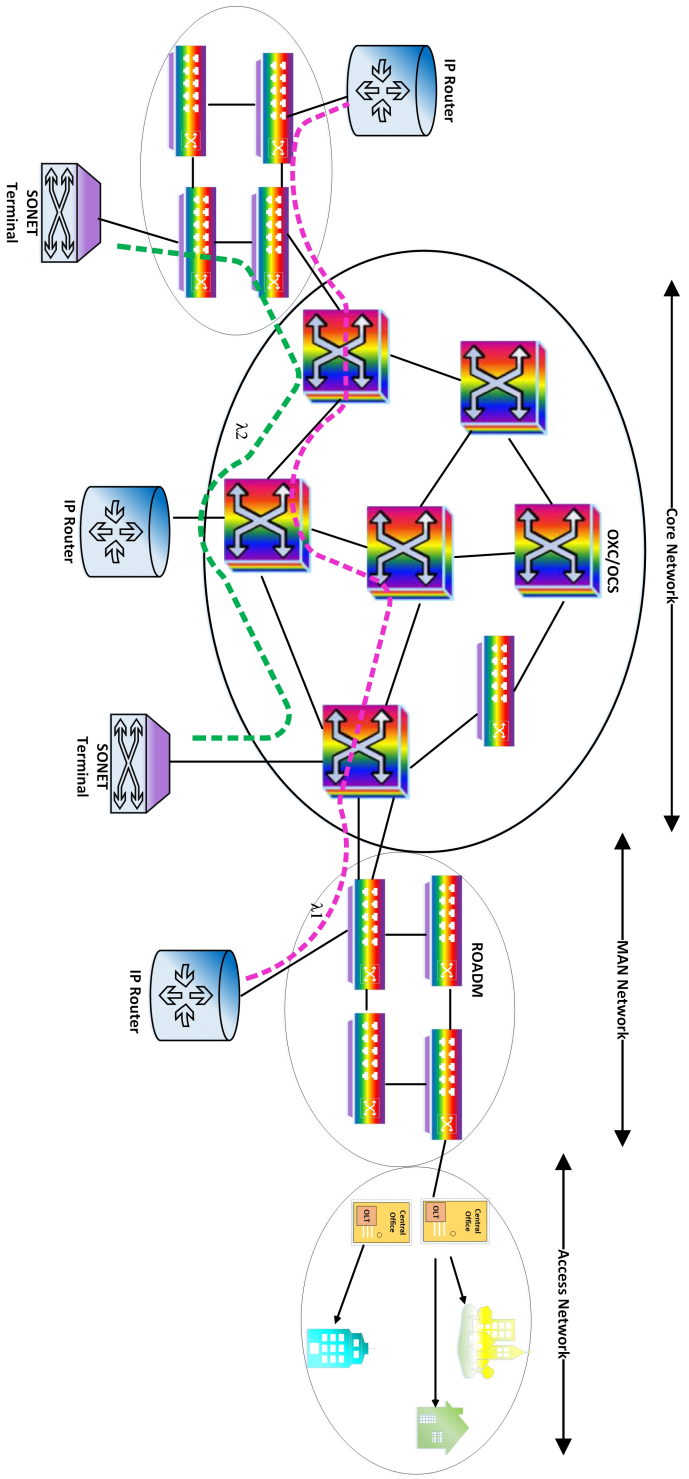


Figure 2.1.: Architecture of an optical network [15]

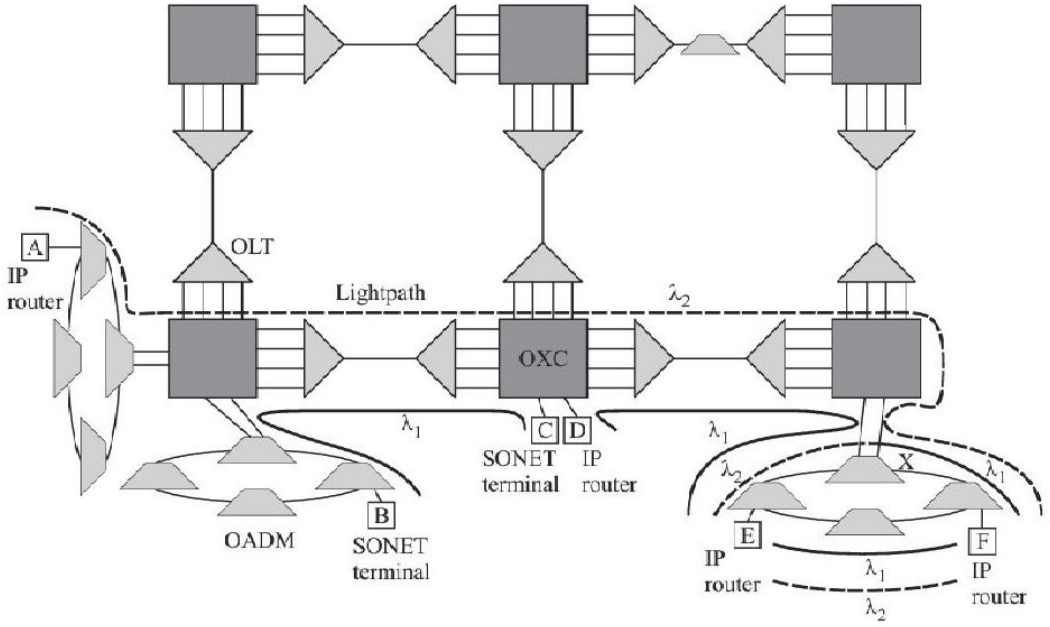


Figure 2.2.: Wavelength Routing Network [17]

routing network (WRN) [16]. Figure 2.2 shows architecture of a WRN.

The above network consists of optical channels called lightpaths to connect network nodes, such as SONET switches or IP routers. Optical circuits span from source to destination and cross multiple intermediate nodes. The network exploits WDM at intermediate nodes to route and switch signals in the optical domain using wavelengths. Sometimes, the wavelength of a lightpath is required to change along the route using a wavelength converter. Optical networks with WDM systems are widely deployed today in long-haul backbone and core, undersea, and metropolitan networks.

Some features of the WDM network are:

Wavelength reuse: The same wavelength can be used on multiple lightpaths, provided they do not overlap over the same link. Due to this reuse capability, many lightpaths are supported by the network with a few wavelengths.

Wavelength conversion: Wavelength is converted at the network's borders because the external signals have to have a suitable wavelength that can be used inside the WDM network. However, wavelength converters can also be utilized to change the signal wavelength along its route to improve the re-usability of wavelengths in

the network. If wavelength converters are not used, a lightpath must continue with the same wavelength over the whole physical fiber link until its destination. This is called the wavelength continuity constraint (WCC) [5].

Transparency: Optical layer can support several higher layers simultaneously owing to its transparency feature. Transparency means that light waves can carry data at different bit rates, and protocols and thus be made protocol insensitive.

Circuit switching: Making and releasing a lightpath in an optical network is analogous to provisioning and taking down a circuit in a circuit-switched network, with the difference in the rate at which it is done, for example, in telephone networks. These circuits remain in the network for months or years.

2.2.1. WDM network components and Operation

WDM networks provide circuit-switched lightpaths between network client terminals. A lightpath or wavelength crosses intermediate nodes where they are routed. At intermediate nodes switching and wavelength- conversion may take place. Lightpaths are made and released as required by the network clients. The main network components are optical line terminals (OLTs), optical cross-connects, and optical add/drop multiplexers (OADMs), connected by optical fiber links. Optical line amplifiers are also set up along the fiber link at periodic locations for signal amplification. The network elements (OLTs, OADMs, and OXCs) may have optical amplifiers to reduce losses.

The function of the OLTs is to combine multiple wavelengths into a single composite signal over the fiber and segregate this WDM signal into separate wavelengths. Thus OLTs may be employed at one of the ends of a point-to-point link. As evident from the name, OADMs are used when a few wavelengths may need to be dropped locally, some to be passed to their destinations, and others just to pass through on the way to the destination. OADMs are used in linear or ring topologies. OXCs with more ports and wavelengths involved perform the same functions as OADMs and are deployed in mesh topologies to perform at a larger scale or to connect other ring topologies [18]. OADMs and OXCs are often referred as optical switches. Figure 2.3. shows a high level diagram of an optical switch.

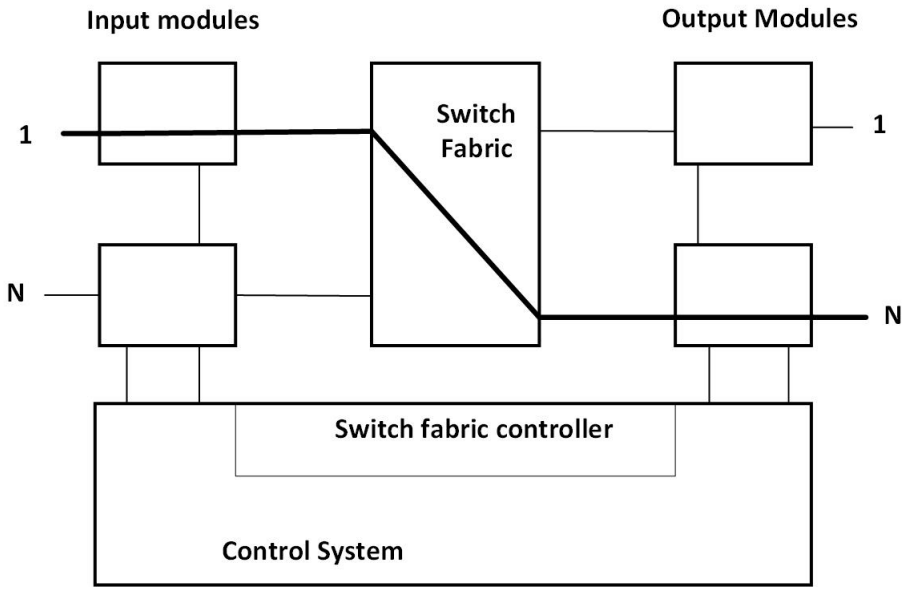


Figure 2.3.: Generic optical switch components[19]

Each optical switch contains input/output modules, a switch fabric, and a controller. Any suitable technology can be used for internal architecture of a switch. Techniques used for switching and routing are either Optical-Electrical-Optical (OEO) or all-optical (OOO) switching. A comparison of OEO and OOO architecture and working can be reviewed in [7].

The control system maintains information about the paths through the fabric and whether they are accessible or utilized. When a new request from the manager is received, it looks in the information database for a free path. It sends commands/electric signals to set up/release the switching elements associated with the paths. Depending on which design architecture/topology has been used to set up the fabric, there may or may not be available paths between any pair of input/output ports because of the internal blocking of the fabric. The controller issues respective command signals to the switching elements, and the information database is updated. The algorithms that are meant to do these tasks are called control algorithms [19].

From the operation of a generic optical switch, we can deduce the layer 1/0 switch behavior as follows.

1. Routing or switching signals is done on a wavelength granularity basis.

2. There are no MAC or other associated addresses with the signals to get the data stream routed through the switch.
3. There is no parsing or analysis of the incoming light wave to find a destination.
4. A local controller controls the internal switch fabric.
5. The control system receives requests for new connections from the management system of the whole optical network.
6. Switches do not make any discovery to find out about their neighbors.

2.2.1.1. OXC

OXCs are conventionally used at network locations requiring multiple bandwidths to be managed. For instance, OXCs are used in managing the coverage on highways and managing advanced devices for optical networks. They can switch lightpaths, add and drop traffic, and configure various topologies in optical networking. They can carry out tasks related to managing the wavelengths and improving the traffic in mesh topologies. The switching core exploits architectures from the rich collection of electronic MINs (Multi-Stage Interconnections) for telecommunication networks [20]. Figure 2.4 shows the general architecture of an OXC.

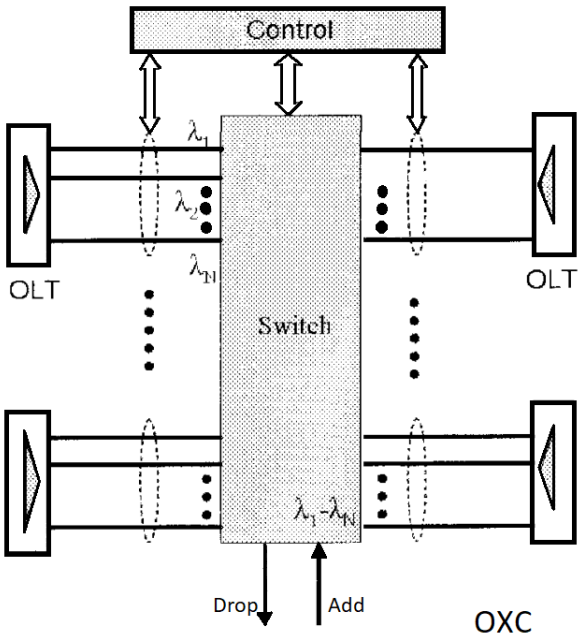


Figure 2.4.: Optical cross connect [19]

2.2.1.2. Re-configurable OADM (ROADM)

Re-configuration is essential for a wavelength routing network and needs dynamic traffic connectivity. Remind that once a route is designed for a static network, it is impossible to change it without physical intervention. Static wavelength routers can introduce reconfigurability by constructing a router that dynamically routes the wavelengths. This is done by incorporating a space-switching stage between a multiplexer and a demultiplexer. Lightpath routing refers to the circuit-switched traffic routing done after establishing the path.

Figure 2.5, shows static and re-configurable OADMS. Static OADMs do not make use of optical switching technology. Their configuration for add/drop wavelengths is fixed. In contrast, re-configurable OADM, has the capability to readily choose wavelengths, to be added/dropped or passed through without advanced planning. This is done on the fly, without physically intervening and interrupting any communications in operation. Thus designing and operation of ROADMs is flexible. As shown in Figure 2.5, re-configurability in OADMs is accomplished by using optical space switches. The ROADM shown in Figure 2.5 is similar to a small size OXC with fewer ports and adjustable capability of Add/Drop [21].

2.2.2. Optical Switching Fabrics

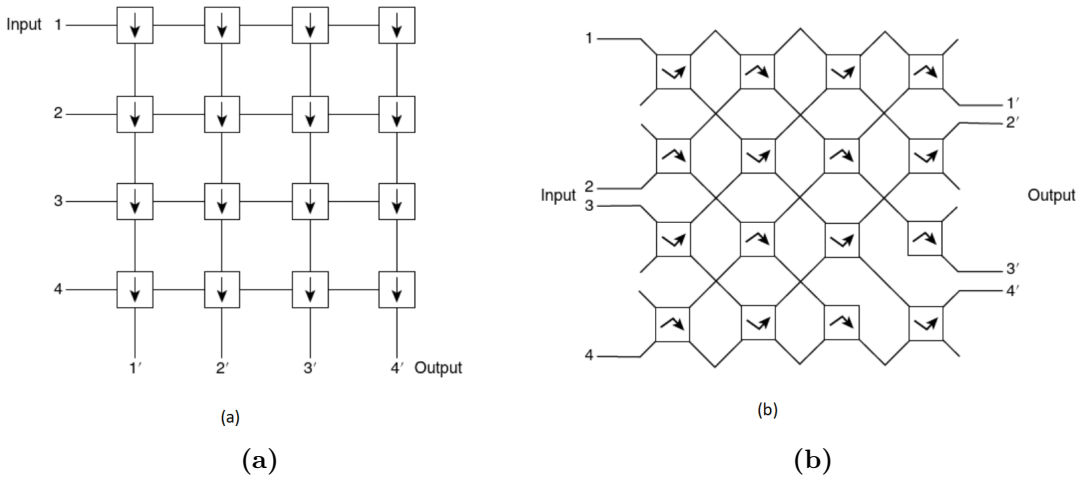
Switching in high-traffic networks considers around 100s of ports in the switch fabric, where each port handles many wavelength channels. Note that increasing the number of ports increases the complexity and cost of the switching fabric. Multiple performance criteria characterize switching fabrics. Typically, switch fabrics are illustrated by fabrication and material technology. Different types of switches are electronic, electro-mechanical, and photonic technology (micro-mirror, liquid crystals, integrated optics). The selection process for switch fabric considers blocking characteristics, size, loss uniformity in input and output, cost, and the number of ports.

The switch element incorporates insertion loss in the fabric; thus, the number of switch elements is responsible for input-to-output port optical losses. The difference in switches between the shortest and longest path connections characterizes the uniformity losses. Furthermore, cross-over between two links is another aspect responsible for losses where cross-talk is introduced, resulting in network performance loss. In addition, if cross-overs are present in the networks, the fabrication

process becomes more complex. PLC is used in directional couplers, while silica is used for fabricating switch fabrics of large sizes. Typically, in optical networks, Cross-bars, Benes, and Clos topology are used for switching fabrics [22][23], and a brief overview of these, is given below.

2.2.2.1. Cross-Bar Switching Fabric

The cross-bar fabric can be implemented as an NXN permutation-switch or RXN generalized-switch, see Figure 2.6. There are N input and N output lines in NXN permutation-switch fabric with N^2 cross-points. Typically, $2X2$ optical controllable element is used as a switch's cross-point. For the case of RXN generalized-switch, RXN cross-points will be required. Cross-bar fabric is implemented in various ways in optical domain. Optical $4X4$ cross-bar in both arrangements are shown in Figure 2.6.



(a) **(b)**

Figure 2.6.: Crossbar fabric [24] (a) Permutation form (b) Generalized form

In 2.6a, the light from an input-guided region is introduced horizontally towards its path using a cross-point if the cross-point switch is turned off. While if the cross-point switch is turned on, the light is introduced from the input towards the vertical path using the cross-point. Thus, when all the switches are turned off, no active connection is present. Due to unequal path lengths, losses are not uniform. However, insertion losses are a bit better. The introduced cross talk is also of unequal levels.

In 2.6b, the length of all the paths is the same, and from left to right, all the switches traverse the same number; thus, their insertion losses are independent of paths. If

the cross-point is turned off, the light remains in the guided medium; however, when the switch is turned on, the light is directed from the input-to-output guide, where each of its paths traverses four switches. In summary, it can be deduced from the abovementioned details that cross-bar switches have either no or lesser cross-talk, larger switch elements, and larger fabric size as N increases in the network.

2.2.2.2. Clos Switch Fabric

The Clos switch fabric [25] is a multi-stage non-blocking cross-bar, and its architecture differs from the single-stage cross-bar switch fabric. A large port-count switch is built from it and typically arranged as a strict-sense or wide sense non-blocking switch fabric. Figure 2.7 shows Clos switch with the three-stage NXN ports. Design metrics are M, K, and P, where $N = PK$ for NXN fabric. The middle stage has $M(KXK)$ number of switches, while first and third stage of the fabric has a K number of switches, equal to (PXM). Each of the switch in first and third stage are connected to each of the middle stage in Clos infrastructure. Multistage Clos fabric can be arranged in odd-numbered stages. The cross-bar switch requires more switch elements (crosspoints) than the Clos switch. Furthermore, the cross-bar switch has less loss uniformity between input-to-output combinations than the Clos switch [26].

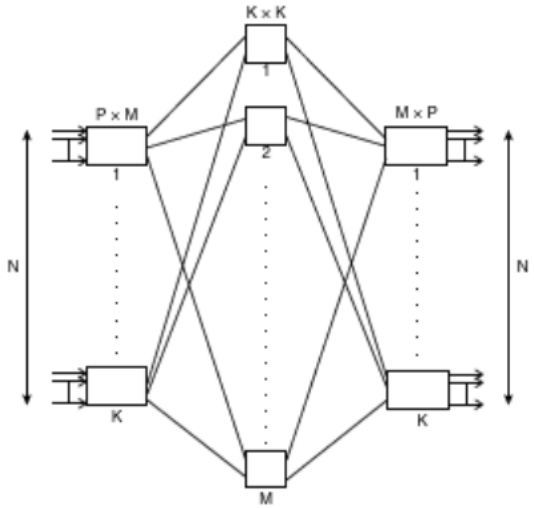


Figure 2.7.: Clos switch fabric [24]

2.2.2.3. Wavelength selective architecture

Wavelength selective architecture is shown in Figure 2.8. In this architecture there is

an individual switch for processing each wavelength. Thus only one same wavelength is coming from every OLT, gets routed at the switch to the right port which connects to the output OLT. Wavelength selective architectures have been discussed in [27, 28, 29].

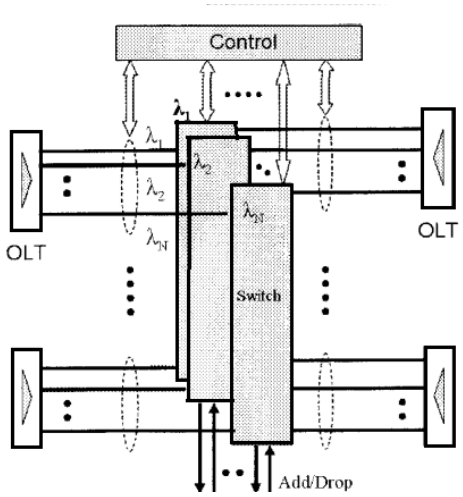


Figure 2.8.: Wavelength selective architecture of OXC [19]

2.2.2.4. Wavelength Converter

WRN has a constraint called wavelength-continuity constraint. In this constraint, nodes are not allowed to change the wavelength of the light wave from the source to the destination port. This restricts the network performance in terms of bandwidth re-usage. Bandwidth re-usability can be improved if the network introduces wavelength converters along their paths. Wavelength converters enhance the throughput of the wavelength-routed networks. Furthermore, they help improve the overall load-carrying capability, decrease the blocking of dynamic data, and make the restoration and protection process easier for the network.

2.2.2.5. Control Algorithms

As explained above, switch fabric comprises small switch elements arranged in one or several stages and has N input and output terminals. A path must be established from its input-to-output terminals during a new call connection. Whereas, if the services of the connection path are no longer required, the call is disconnected and the path is released for others to use. Circuit-switched lightpaths are established

between its end-users in optical transport networks through OXCs and optical fibers, as shown in Figure 2.2.

A separate input module, output module, switch fabric, and control system is embedded in each OXC. When the input module receives the signal, it transmits this information to the control system. The control system configures the switch fabric exploiting the desired paths for interconnecting input to the output. To route incoming calls, it maintains information about the routing table. When the signal leaves the switch fabric, it is passed to the output module, where controlling information about optical switching is re-attached. Input/output modules perform other functions like traffic grooming and monitoring.

Typically, a controller (i.e., a microprocessor with a separate embedded program) receives commands from the control systems to either release or set up new paths in the network. It depends upon the switch fabric whether there should be one, more than one, or even no path (i.e., due to blocking) between input-to-output terminals. The architecture of the switch fabric can be based on any typical multistage interconnected networks (MINs) topologies used in a telecommunication network. If the network has to establish a new connection for a call, a connecting path is located by the controller in the switch fabric. Then, it is checked if the path is free and is not in the use of any other ongoing calls. The steps mentioned above are performed by controlling the algorithm. The switch fabric stores its state in a table. This table maintains the record of all the connecting paths previously established and in use. This table removes the information it has once any of its paths is released and modifies the configuration of the switch and resets itself.

Besides switches, optical amplifiers can also be controlled by the controlling algorithms. Control algorithms in a switching fabric can establish different connections depending upon the requirements, e.g., unicast, multicast or broadcast. The connection between a single input and single output terminal is called a unicast. The connection between one input terminal and multiple output terminals is called a multicast. In comparison, the connection of one input terminal with all the output terminals of the set is referred to as broadcast. Various controlling algorithms are developed based on their connection types and switch fabric infrastructure. Typically, the algorithms designed for the topology considering a particular fabric cannot operate at other topologies. More complex and sophisticated algorithms might be necessary for a specific technology or purpose (e.g., to avoid cross-talk), which are

not required with other technologies [30].

2.2.3. WDM networks management

Management and control of the network are crucial parts of any functioning network. Does network management address how the network operates and uses its resources? How are the actual connections made for data signal transmission? How are they taken off and monitored, and how the network fulfills future demands? How is the network made more survivable? Energy-efficient and cost-effective?

Today optical networks serve a variety of client networks, e.g., IP, cellular, SONET, point-to-point, corporate and private networks etc. Equipment of the more extensive networks is from different companies, and their control mechanism is usually incompatible with devices from other vendors. These networks are controlled through a hierarchy of controllers. The service provision is very much automatic in the higher-level electronic management layer, but it also needs manual steps in the optical layer.

Figure 2.9 shows the NMS architecture of the optical network. The optical network based on wavelength division multiplexing is interconnected with the client network via ingress optical nodes. The ingress nodes switch to the correct ports whenever data needs to be sent to the optical network. This helps facilitate the signals to be carried to their destinations using suitable lightpaths. Therefore, it is crucial to manage the optical network components such as optical cross-connects, optical amplifiers, add and drop multiplexers, and optical line terminals. NMS is responsible for allowing the requests of setting up the networks as soon as the wavelength routing and assignment issue is resolved [8].

For the network provision operation, two approaches are typically followed and are given as following:

(i) Set up the network request for path computation on ingress nodes in a distributed way.

(ii) Set up the network requests for path computation by considering centralized way.

For the first approach, ingress nodes are given the provision of path computation. Thus, to perform wavelength routing and assignment (WRA), global network state is required on each network node. With this approach, the scalability of the network

as well as resilience to the network failures is obtained by considering distributed provisioning. This is because various nodes are responsible for path computation thus, even if a single node fails, it is not possible that this failed node will bring the remaining network down. Note that it is important for the smooth and effective functioning of the network, that the state information must be updated periodically. Moreover, the updates from each of the nodes must be disseminated as fast as possible so as to have efficient path computation decision. The increase in the rate of updating the status of the network also increases the overheads (OHs).

In centralized approach, only one entity is responsible for path computation; thus, there is no need of disseminating the state information in the whole network. The requirement of advanced as well as complex routing protocols is also eliminated. Consequently, the overheads are further reduced. In other words, the network resources are optimally utilized in the centralized way of path computation. Thus, centralized provision is less complex as well as cost-effective. However, with the scalability in the network, management tasks for NMS become complex and it would be more appropriate option if distributed control is considered in the network. For the distributed provision, a separate element management system (EMS) manages each optical device in the transport layer, see Figure 2.9 for details.

The parameters and attributes that are to be managed are made available in element agent (EA) (a software present in the microprocessor). NMS and EMS both have the EA information for signaling purposes and is referred as data communication network (DCN). This information also allows the network to be controlled as well as monitored automatically. The EA can be asked with queries about the network and EA can also respond back by sending messages in the form of alarms. Moreover, the interfaces are provided to NMS by EMS, conventionally, referred as northbound interface by exploiting simple network management protocol (SNMP), extensible markup language (XML), or common object request broker CORBA.

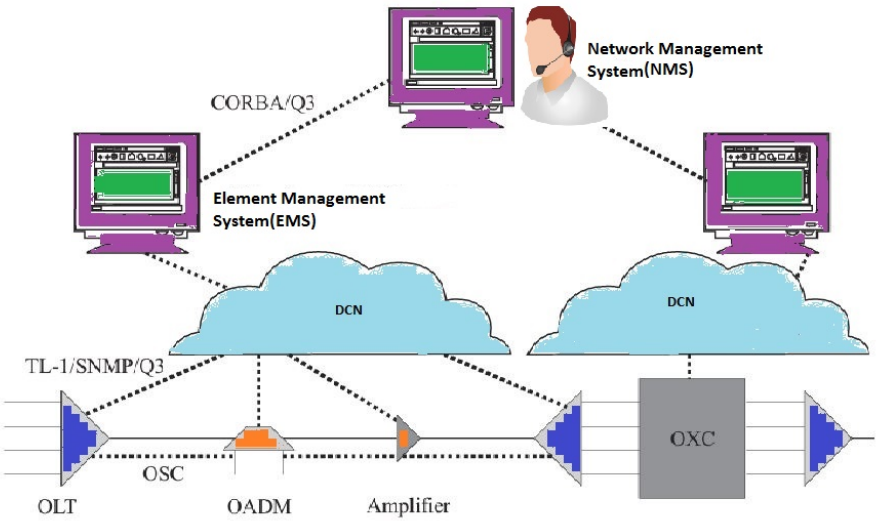


Figure 2.9.: Optical network management using NMS

2.2.3.1. Routing functions of the control plane:

In a network, routing is done to select paths to establish connections between the nodes using routing tables. Whenever the network receives connection requests, a route and the resources required to establish this connection are found. This can be obtained by using the routing algorithms on the topology database. Different constraints (imposed by the network) are considered by routing algorithms, e.g., available bandwidth at each connection and wavelength conversion ability.

The control protocol is responsible for the computation of the path. It determines the process of selecting the route requests by considering carrier's policies. If various routes are available, policies help determine the allocation of resources for a particular route request. Furthermore, the impairments during transmission in optical paths must also be exploited before calculating routes. Route information must be spread along the network, which helps the control agent find a route to the destination.

2.2.3.2. Signaling:

Transferring the control messages between different entities and layers is carried out using a signaling mechanism conducted by the network's control plane—the signaling protocols help manage connections and recover the faults. The conventional process

usually considers creating the connections, then maintaining the connection and releasing the link. If connections are lost for some reason, they are also responsible for restoring them.

The process of signaling, in circuit-switched networks, includes the resources to be reserved for setting up the connections. For instance, with the switched link, when the client layer or NMS issues a service request command, source or destination nodes along the path in the network send control messages to the intermediate nodes. This command triggers the signaling exchange, and a connection between the source port and destination port is provisioned through OXCs by configuring suitable settings. The signaling mechanism requires a data communication network, either the existing optical network's data transport network or an explicit control system. For further information regarding signaling network, please see [8].

2.3. Related Work

Today's optical network traffic is voluminous and highly dynamic. Therefore the control of optical networks needs to be improved to handle the requests efficiently and can quickly perform reconfiguration of the resources with demands. The topic of managing optical networks is rapidly evolving, and various organizations are working to develop control plane standards. The main international bodies are the International Telecommunications Union-Telecommunications sector (ITU-T), IETF (International Engineering Task Force) And The OIF (Optical Internetworking Forum).

- ASON

The ASON architecture [31] had been developed with dynamic configuration (self-organizing) power. This potential is controlled by the connection management system of the ASON control plane. ASON control plane divides the optical network into domains, and these domains interact with each other through standardized interfaces [32]. According to IETF, ASON can be considered a substitute for NMS based connection management. ASON is not a set of protocols but rather a three-layer architecture. ASON reduces the NMS load [33, 34]. ASON defines optical system specifications and its requirements; then, standard protocols that fulfill these re-

quirements can be adopted by ASON, e.g., ASON formalizes GMPLS overlay model into its own description and specifies GMPLS RSVP-TE as a signaling method in a distributed management system for setting up a call and administer the connection, (G.7713.2/Y.1704.2) [35]. End-to-end path establishment requirements are specified through G.807/Y.1302 and G.7715/Y.1706. ASON is being developed and investigated for standardization.

- GMPLS

In GMPLS [36], the idea of Multi-Protocol Label Switching (MPLS) is generalized to the physical layer. MPLS was developed for IP routing so that QoS can be provided for high-speed packet forwarding. MPLS labels packets and swaps these labels for forwarding data through the network. A label-switched path (LSP) appears as a virtual circuit-switched path and can be applied to any circuit-switched connection. GMPLS generally enables interoperability between electronic and optical switching systems and is widely accepted as optical-layer control. Three different interconnection GMPLS models have been defined for organizing the control plane to accommodate various network deployment scenarios, i.e., overlay, augmented, and peer model. The overlay model obtained significant popularity. The overlay model observes separate client and server layers with their own control systems and does not transfer any information. Addressing schemes are also different in both planes. GMPLS has not been widely used in optical networks, the complex control scheme being one of the main reasons [37]. GMPLS offers little help to the NMS in providing a circuit when it has been triggered manually by the control system [35].

- PCE with GMPLS

Previously, regular GMPLS is advanced to a Path Computation Element (PCE)/GMPLS-based architecture i.e., RFC6163 [38]. The PCE-based GMPLS moves the task of computing the path from regular controller to another powerful computational resource i.e., Path Computation Element (PCE) [39], which brings several advantages to the optical network control. Complete path computation using PCE has been proven to be a feasible solution in optical transport networks with multiple

domains. PCE is taken as a standard solution for path computation in GMPLS networks. GMPLS network adopts the constraint based optimal path computation. In PCE/GMPLS architecture, a Path Computation Client (PCC) e.g., a network element, requests PCE to find the path with certain constraints. Taking the constraints into account, the PCE uses its Traffic Engineering Database (TED) to calculate optimal path to the requested destination. The communication between the PCC and the PCE takes place through IETF's standardized Path Computation Element Protocol (PCEP) [40]. Hence GMPLS became complicated because of introducing a separate PCE and its associated protocols.

- SDN/OpenFlow based Hybrid approach

Software-based control seems to be a viable solution for the dynamic demands of the current complex and static optical networks. A review of benefits of SDN is provided in [41]. The aim of SDN and OpenFlow protocol is to design a programmable network that may also be presented as a unified control plane for hybrid networks. SDN has been studied as a control plane technique in optical networks e.g., [42, 33, 43, 44, 34], but is yet in early stages. Various studies are briefly discussed below.

- OpenFlow with GMPLS

[45] implements a software-based packet over optical network solution based on the integration of the OpenFlow protocol and GMPLS. This approach is also presented in [43]. The IP router sends the unknown packet to the OF controller (NOX), on receiving new packet the NOX will automatically interact with the GMPLS controllers for setting up a path in the optical layer using RSVP-TE protocol.

- OpenFlow and PCE

OpenFlow and Path Computation Element (PCE) based integrated architecture is proposed in [46], in which controller is off-loaded of the task of the path computation. An open and standard protocol, PCEP is used for communication between

controller and PCE while the PCE gets its network topology from a dynamic topology database, which is contacted when ever the request arrives for path computation. The architecture is tested experimentally and evaluated on a real network comprising of IP over WDM network. Furthermore, [47] explores role of PCEs in SDNs.

- Pure SDN/OpenFlow-based

– Ciena/Stanford Prototyping [48]

First pure OpenFlow based work demonstrated a merged control plane for IP/Ethernet and TDM networks. The work demonstrates the integrated packet and circuit control in which L1/L2 flows are dealt but layer zero's WDM switching is not addressed.

– Proof of Concept

In [42, 49], pure SDN architecture, a centralized control system, establish dynamic paths through an optical layer. SDN controller (NOX) is used as a central controller without the support of any other control mechanism. NOX obtains the logical view of the optical devices through virtual Ethernet Interfaces ("veths") introduced in the open flow switch. Veths correspond to the interfaces of the optical nodes; thus, mapping actual physical interfaces is done on OF switches. OpenFlow switches are linked with the optical nodes through the TCP interface. The PXC in which the OpenFlow agent is integrated is called OpenFlow-enabled PXC (OF-PXC). On obtaining the request from OF router/switch, the controller computes the path and inserts flow entries in relevant OF switches. OF switches, in turn, communicate with optical nodes to establish a path using Transaction Language-1. Thus the OF control plane efficiently manages all-optical nodes for path establishment purposes. NMS in the system can be used for other network monitoring purposes. Other than the above approaches, there are few efforts [50, 51],in which generic programmable OpenFlow-compliant switches are proposed. It has been claimed that they can be used in optical layer devices.

In a recent work [52], an SDN-controlled platform is designed to simulate an optical network with an integrated end-to-end packet and optical layer data plane. Analytical models are used to mimic the physical behavior of a WDM network. The data plane of the optical layer is emulated as data packets with appended wavelength information. The designed platform is used to customize network elements; however,

switching constraints are neglected. Below Figure 2.10 shows the categorisation of the major previous relevant works.

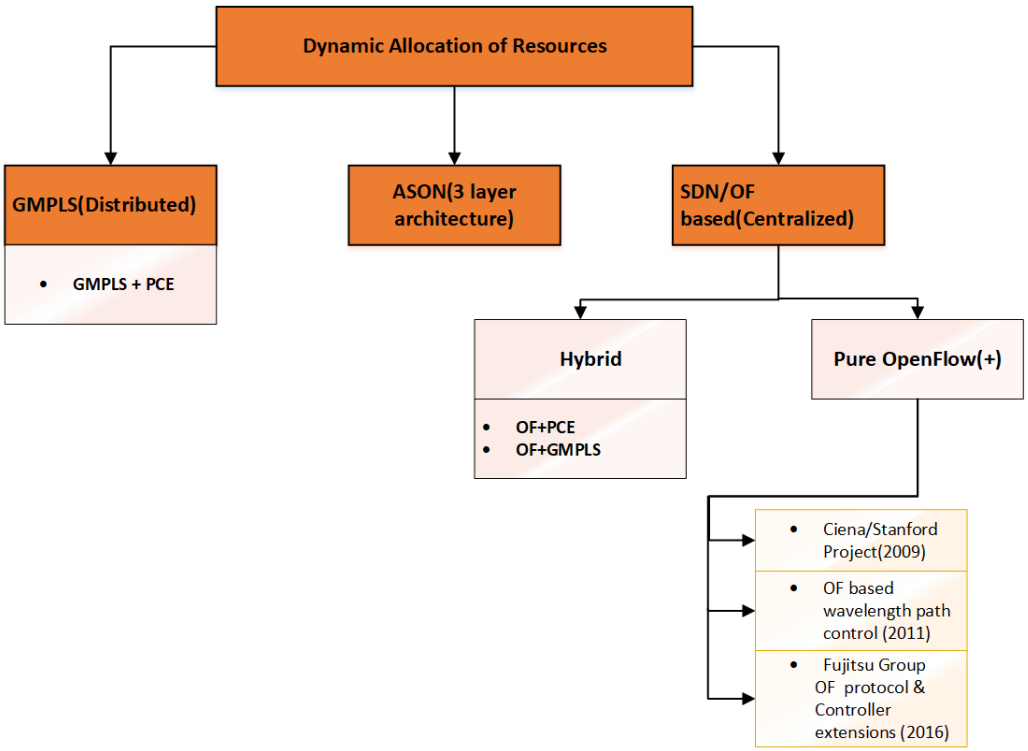


Figure 2.10.: Related work

2.4. Software Defined Networks (SDN)

In the future, the software will be the key to the networking domain. Software-defined networks (SDN) brought a revolution in the field of computer networks. Open networking foundation (a non-profit industry) that standardized SDN. SDN transforms static networks into programmable platforms. This is typically achieved by decoupling data planes from control planes and shifting from tightly bound static nodes to programmable and computing nodes. Moreover, the state as well as the intelligence of the network is considered logically centralized. This enables the infrastructure to efficiently modify network services and applications, resulting in virtual reality where networks are treated as logical. Multiple enterprises and carriers can

obtain separate control over the complete network's programmability and automation from a logical entity. This results in providing flexible and scalable networks to fulfill the business community's requirements.

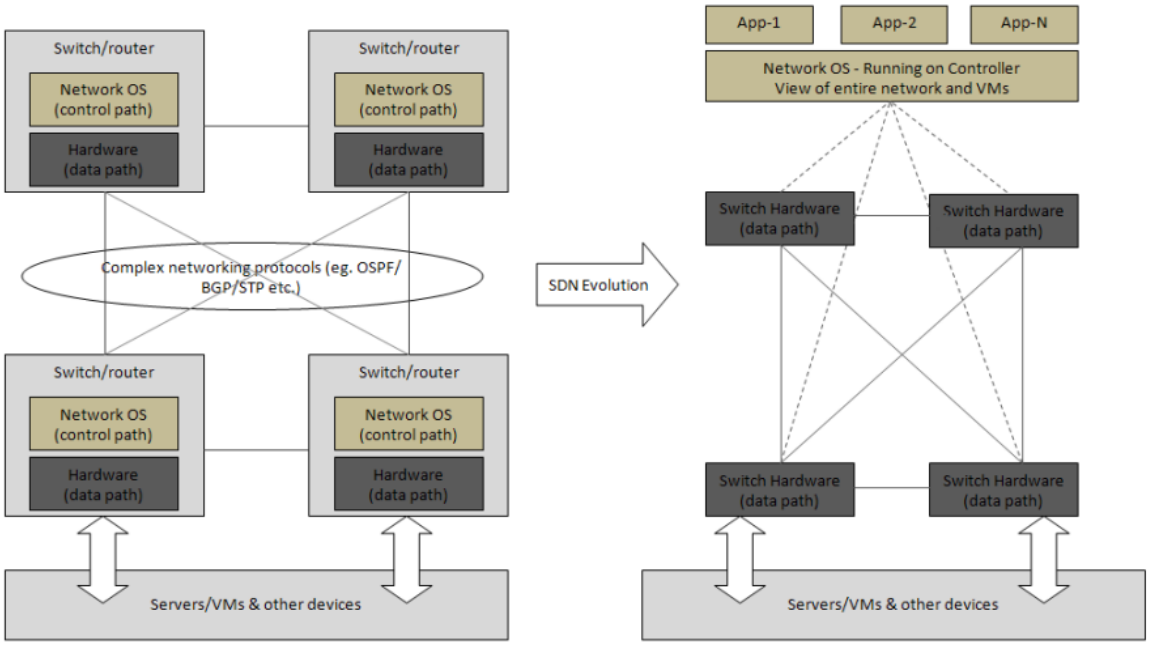


Figure 2.11.: Separation of control and data plane [53]

The traditional network infrastructure consisting of various routers and switches inter-connected with each other and other nodes, e.g., virtual machines and servers is shown on left-hand side in Figure 2.11. The current network and the control path are built using distributed protocols. When the control path is established, the device hardware installs the data path, and the user data traffic is transferred through the route programmed. Typically, multiple protocols, i.e., open shortest path first (OSPF), spanning tree protocol (STP), and broader gateway protocol (BGP), are used for routing purposes to make data path that is mainly Layer 2 and Layer 3 protocols.

Right-hand side of the Figure 2.11 represents the definition of SDN in the same network. Typically, an externally attached device is used for running the network operating systems (NOS) (i.e., a microcontroller) where most of the high-level instructions are managed. The microcontroller computes the traffic paths once the decision is finalized. Figure 2.11 shows a secure channel using a dotted line repre-

senting the communication of OpenFlow switch and micro-controller. A view of the full network showing multiple links is accessible to the micro-controller. Standard data flowing through switches and routers is shown using solid lines.

A 3-tier SDN logical layer is shown in Figure 2.12. Here, networking intelligence is centralized (logically) using software-based SDN controllers by exploiting the network's global view. Thus a single switch (logical) in a network appears to the applications and policy engines. In particular, SDN makes a single point (logical) that controls the entire network independently and can also program various desired interfaces for multiple applications by the designers [54, 55, 10]. SDNs can simplify the networks by themselves in terms of their designs and operations for different carriers and enterprises. This is because SDN does not need to follow and understand many of the standard protocols but rather follows SDN controller instructions. Particularly, network designers considering SDN are allowed to use a single platform for controlling access as well as routing.

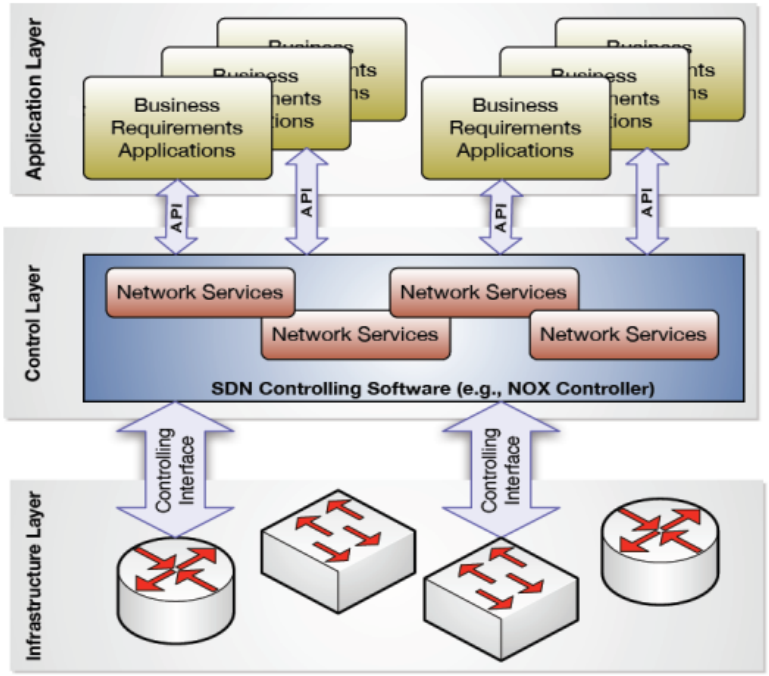


Figure 2.12.: Logical layers of SDN [56]

SDN paradigm considering OpenFlow mainly focuses on addressing the practical needs of the business community (that typically needs dynamic applications, services, and adapting to the changing environment) by decreasing the management

and operational complexity. There are many advantages of using SDN technologies considering OpenFlow for the business community, enterprises, and carriers. A few of them are given below [56].

- OpenFlow helps provide the network's flexibility in terms of operation, selling, and usage. The controlling program specifies the behavior of the network model. The network view is written by administrators or system operators configured by OpenFlow, disrupting joint programming paradigms.
- OpenFlow helps in the automation processes that captures the demands of the business community, enterprises, administrators, etc.
- OpenFlow helps in the innovation easily using customizing features. For instance, different administrators and software vendors are allowed to develop new features disrupting their controlling programs.
- OpenFlow helps business community, users, vendors, administrators, etc. to program easily. This also helps in storage, computation, network integration as well as virtualization. Moreover, in this way operations can be managed easily using a single logical node.
- OpenFlow helps in managing network devices automatically ensuring secure and reliable network.
- OpenFlow helped users by providing them better experience with the applications.
- OpenFlow-based SDN technology is capable of conveying flow-table information to the neighboring nodes of the network thus becoming a better paradigm for vendor market. The typical standard interface providing control functionalities in networking is the OpenFlow.

New opportunities can be realized for controlling experiments and standards using the OpenFlow platform. This allows the programming controllers on the network side to be flexible for the new researchers. High-level routing at the elements is separated at the OpenFlow platform. Thus, programs and software can be run over it [57]. Moreover, switches in OpenFlow can be easily managed as they are simple and flexible. The functions of Data-path in packet-switching are only implemented in OpenFlow. The controller is responsible for centralization and executing operations

such as installation, deletion, etc. The controller and OpenFlow switch use OpenFlow protocol for communication purposes. Several texts (messages), e.g., send data packets, receive data packets, modify data packets, get statistics of packets, etc., are defined in OpenFlow protocol. In summary, this results in adapting to various environments and enabling the networks to construct a flexible and scalable network capable of controlling and gaining automation and better programming capacity [58, 59].

2.4.1. OpenFlow based SDN

In 2008, OpenFlow was presented at Stanford University, where the first standardized communication interface between SDN forward layers and the control was introduced. In this network protocol, it is allowed for the research community to perform experimental protocols for routers and switches uniformly without affecting or exposing the internal working or infrastructure of various vendors' products. Since OpenFlow is based on an Ethernet switch; therefore, multiple vendors are encouraged to use different features of OpenFlow. This helps vendors develop a network as a backbone at universities and closed networks.

Project tools that are proposed for large-scale testing can be performed in OpenFlow. This allows the research community to analyze various scenarios in real-world environments. Since SDN supports OpenFlow, therefore, it is capable of manipulating the forward plane in a network. Due to its simple firmware and software update process, multiple vendors have already implemented OpenFlow. [60, 53, 61]. Network control at multiple switching devices is logically separated into centralized controlling software using OpenFlow protocols. The sets of instructions in a CPU are typically compared with OpenFlow. The protocol specifies external applications of the software just like the set of instructions in a CPU [62]. OpenFlow can be implemented for both interface sides, i.e., SDN control and network infrastructure.

The concept of flows has been used in OpenFlow. This helps in characterizing the traffic related to pre-set rules of match that are allowed to program statistically and dynamically by the software of SDN control [63]. Furthermore, based on performance metrics like cloud resources, usage patterns, etc.; OpenFlow allows traffic flow transmission through the network nodes.

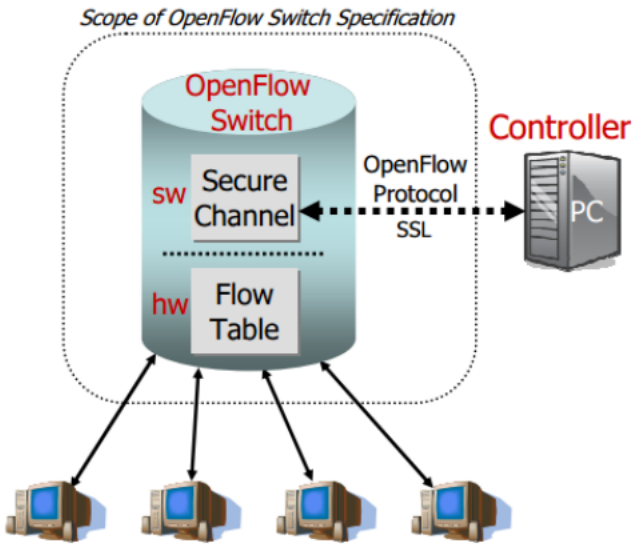


Figure 2.13.: Ideal OpenFlow Switch

In Figure 2.13 a remote controller manages the Flow-table via the secure channel. Flow refers to a set of packets from one end in a network to another end of a network, or it refers to a group of packets from the collection of end nodes in a network to another set of end nodes in a network (sharing the same header fields). For example, all the packets in a flow will have the IP source/destination address or VLAN ID. Typical endpoints in a network can be VLAN endpoints, TCP/UDP ports, IP addresses, input ports or L3 tunnel endpoints, etc. The main goal of the open protocol is programming the interface and flow table in various switch and routing devices to add and remove the devices.

The research community is allowed to manage and control the local network (experimental network). They are capable of selecting the packet transmission routes as well as their processing functionalities. Furthermore, the research community can perform new experiments, such as testing the security of a particular protocol and multiple addressing scenarios that do not disrupt other nodes in terms of the traffic transmission [64]. OpenFlow gives larger flexibility using flow-based routing and virtualization that helps in real-world network demands. Using the connections of Secure Socket Layer (SSL), a centralized but remote controller mainly controls the flow tables. This strategy raised serious concerns on the networks performance in terms of the scalability as well as in terms of the reliability of a controller device [55].

Traditionally, the decisions of high-level route computation in the control plane and the fast forwarding of packets in the data plane are all made in routers and switches. But the OpenFlow switch segregates these two tasks. This helps network nodes like switches and routers to be programmed easily using a standard interface. The decisions regarding packet forwarding are still carried out by the data plane. However, a standard server or a separate control device makes the high-level decisions of the network. This results in implementing new protocols, virtual networks, and mobile users by allowing the control device to change the forwarding principles in switches, see [54] for further details.

The OpenFlow protocol is responsible for communication with the control device by defining sent packets, received packets, modifying forwarding tables, and obtaining statistics. In response to the packets processed by the OpenFlow switch, the use of OpenFlow protocol allows the controlling device to actively participate in adding, removing, and updating the flow entries [65]. In Figure 2.14 functions of packets are handled by the OpenFlow switch. When an OpenFlow switch gets its first packets, which are not in its previous data records or do not match flow, that packet is sent to the controller. Flow request is referred to as the first packet. The control device computes a flow path, and then for the computed path, flow entries are installed for all of the switches. As a result, the control device sends the packets to the origin switch, which are then sent to the destination nodes.

An OpenFlow switch presents one or more flow tables in the forwarding plane. For matching purposes, various data packet fields are present in each entry in the flow table. For instance, various header fields, switching ingress port, an action (for dropping, sending out port, modifying field, etc.), and counters. Expendable matching fields containing multiple header field packets, as well as the ingress port of the switch, are identified in each of the flow table entries.

In the flow entries, the ingress port and the headers in the packets are tried for matching purposes as soon as an OpenFlow switch device receives the packets. Additional entries in the flow tables are compared with the less-defined or fully-defined entries. As a result, further processing in the form of metadata is stored for communicating among various tables. The priority field in the flow entries is selected in case of multiple flow entries. Multiple packet types are matched with the look-ups of the flow table entries in the packet header.

Conventionally, multiple matching fields, such as the IP destination address or the

MAC source address, are included in the packet header. There are two ways to match flow tables with the packets. (i) Specific matching and (ii) Wild-cards. Specific matching refers to the packets that reach the flow table with various exact identity fields, i.e., TCP, IP, Ethernet, MAC address, source address, destination address, etc. In comparison, the matching in wild cards refers to transmitting those packets to any addresses, destinations, or ports.

In Figure 2.14, an OpenFlow switch disrupting packet flow is shown. The associate counters should be updated in the switch. Then, the set of instructions that match the flow entries with the highest priorities should be applied first [66]. When the entry found in a flow table is matched with the set of specified instructions, then those instructions are executed. But, when the flow entries in the flow table are not matched with any flow entry, the packets are sent automatically to the controller device. Security policies in the flow entry of the packet are first verified by the controller. Based on security policies, it will either install a flow entry or it will drop the packet. If it installs the flow entry, then it informs the switch, how to handle similar packets in the future along its path.

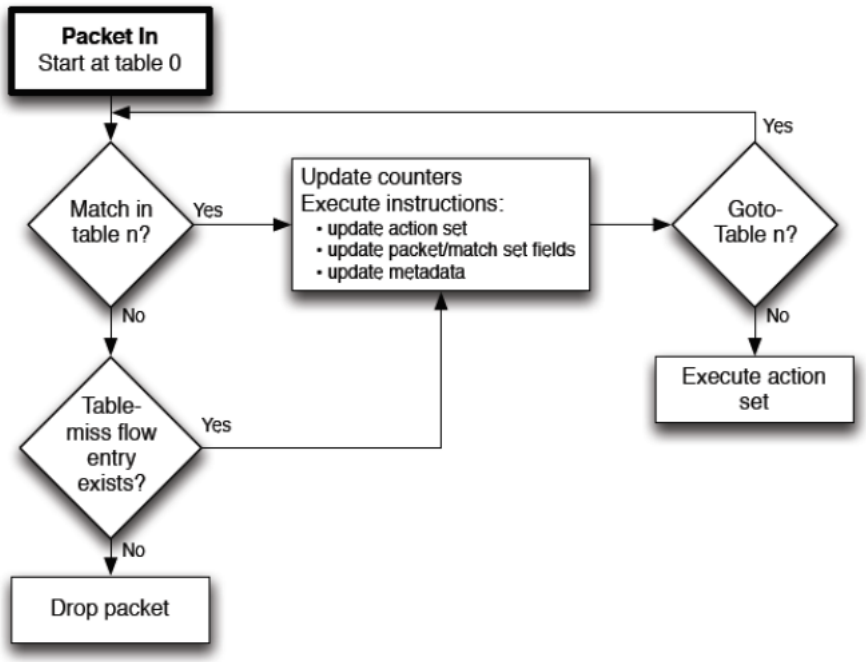


Figure 2.14.: Packet flow in an OpenFlow switch [65]

Typically, there are two classifications of OpenFlow switches in an OpenFlow network. (i) Hybrid switch (ii) Pure switch. Hybrid switches are also called OpenFlow-

enabled switches, and they can disrupt Ethernet protocols (L2/L3 switching) in addition to OpenFlow support. Pure switches are also referred to as OpenFlow-only and do not support any other features except OpenFlow. Thus, high-level decisions solely rely on the controller devices. It is crucial for the connections used in communication to be available and secure, as the open interface manages the OpenFlow switches. Since every switch needs to establish a connection with the flow table and the controller device, the module of OpenFlow must support them. OpenFlow protocol is a standard implementation for communication between the SDN controller and the SDN switch.

2.4.2. Mininet

Simulation and emulation software are essential when prototyping and testing are required without expensive physical devices. Mininet is a system that emulates a network using lightweight virtualization to test and evaluate OpenFlow protocols and SDN applications on a single machine. Mininet uses the virtualization concept, running actual kernel, software switch and applications code, thus imitating a real network. Mininet CLI or scripting can be used to configure the network and share it with others.

Mininet constructs virtual networks and runs its elements (end-hosts, OF switches, routers, links) on a Linux kernel using Linux processes and network namespaces. All virtual network components of Mininet, such as hosts, switches and links, are software-based and are images of real things which behave like individual hardware elements. Customized Mininet networks can also be experimented with by writing Python codes in which Mininet libraries can be imported. Mininet is a great way to learn SDN and OpenFlow and test SDN controllers and applications. Applications and controllers prototyped and tested in Mininet can easily be transferred to OpenFlow-enabled hardware with almost zero modification. Mininet web-page [http://mininet.org/] is an excellent source for learning and understanding Mininet.

MiniEdit: Mininet also has an example GUI editor named as MiniEdit. It can be used to quickly create and simulate a test topology. Different tests can be performed by changing element's specifications/characteristics. It is mostly used for observing packet flow/transfer in the network.

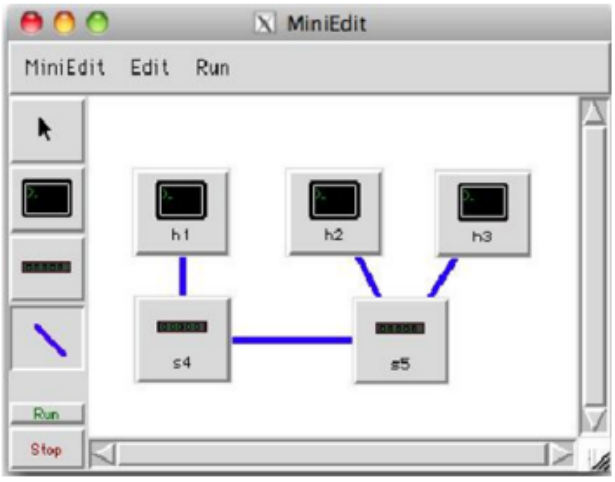
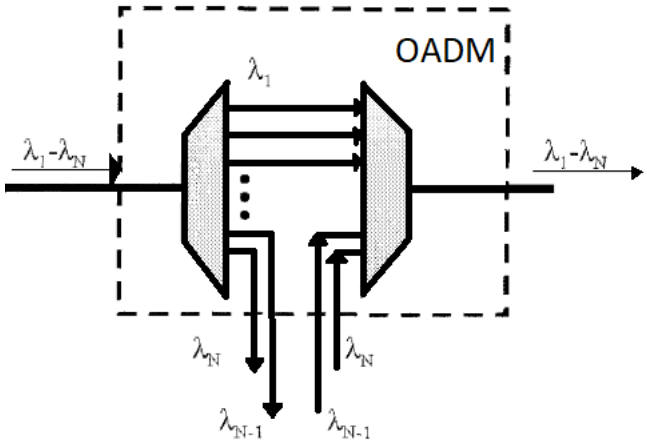


Figure 2.15.: MiniEdit editor

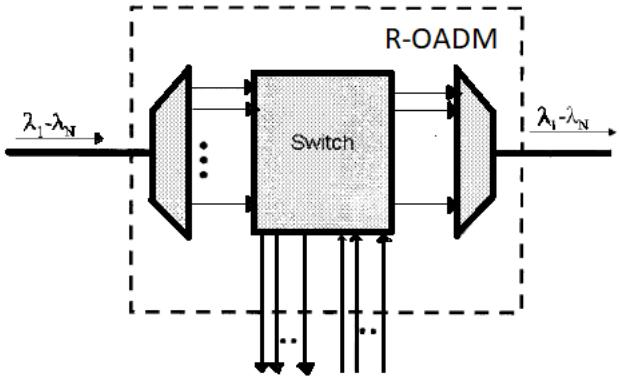
2.5. Summary

In this chapter wavelength routed networks' components and operation are explained in detail. The constraints, complexity and static way of controlling optical networks do not align well with the emerging demands and evolution of network technologies. International organisations and research community have been working towards flexible optical networks for more than a decade now, however due to complex nature of these networks, various struggles are still in progress and nothing could be standardised yet. Software Defined Networks offering this flexible control, attracted researchers and is under investigation. However, SDN due to its suitability for packet networks, has certain limitations for circuit-switched networks. Thus some hybrid solutions have been offered, however research is in early stage and until something gets standardised, a hybrid or intermediate system can be utilised.



(a)

(a)



(b)

(b)

Figure 2.5.: (a) Static OADM (b) Reconfigurable OADM [19]

3. Proposed Model Theory

3.1. Overview

Dynamically set up optical networks have become the necessity of the day due to the revolutionary changes in the world of information and technology. Previously suggested ASON and GMPLS architectures have yet to be able to impress the network operators due to their complexity and compatibility issues. SDON has been regarded as the future generation of optical networks [67]. However the approach is in preliminary stage and requires a considerable amount of research work.

SDN and OpenFlow protocol's packet-based operation restricts its use to the packet-switched layers and does not adapt well to the WDM-based circuit-switched networks. Research efforts are going on, though more needs to be standardized. Network operators today favor staying constant as they prefer to reduce the risk of disturbance or degradation of services. Future optical networks must perform better than ever to remain cost-effective and equally stable.

Various approaches have been proposed to extend SDN for optical networks. Some of them suggest to hybrid SDN with other architectures like GMPLS, while few of them offer to incorporate other components like PCE [68]. Since OpenFlow yet needs to be embraced by many vendors, a modular OpenFlow agent might be set down in the node to lay out hardware abstractions for OpenFlow interface. Utilizing the network equipment management interface, this agent can be used as an interim solution before a protocol is standardized for implementation at the level of network elements [69].

The vital background based on the pathfinding in switch fabrics and control algorithm is presented in this chapter. Moreover, a new intermediate model is proposed to manage WDM layer connections using OpenFlow to provide dynamically controlled service provision. The proposed model allows the incorporation of other features that may be required in the future that will contribute to covering different aspects of network management, for instance, network protection and restoration.

The proposed model is flexible, and the middle layer collects, maintains, and updates all the related information, replacing complex control algorithms locally processing within the network elements. The model provides dynamic service-provision capability depending on the current availability in the network. Details of the model and its service-provision process are described in the included research article in section 3.4.

3.2. Optical Cross-Connects and their Control

Functionally, three essential parts of an OXC are input/output units, a switching unit, and a controller. Some OXC ports are connected to ports of other optical network elements, and others are connected to client ports. The switching unit is the core of the OXC.

Optical switching technology of the switching fabric depends on the application and required specifications of the OXC. Switching elements in the fabrics are interconnected using various strategies from multistage electronic interconnections used in telecommunication networks to form the switch fabric. The fabric can take a simple form, e.g., Crossbar, to any other complicated standard pattern, e.g., Clos, Benes, and Banyan, comprised of minor ON/OFF interconnected switches.

The switching unit can be made as a wavelength-independent fabric or a wavelength-selective architecture where a small space switch can be used for each wavelength. These two options are shown in Figure 3 in [15]. In order to set up (and release) lightpaths, the local control unit of an OXC communicates with its peers in the network. It then uses that information to generate local commands to configure the switch fabric. Several standard and proprietary algorithms can be used to configure switch fabrics. Here we choose Clos type of switch fabric to explain an example of the control algorithm.

Clos network is usually denoted by $C(m, n, r)$; see Figure 3.1. Clos switch fabrics have multi-paths through them from input to the output ports. To set up a new call, a center stage switch needs to be found.

In Figure 3.1, the input and output terminals are assumed as numbers; 0- $N - 1$, where $N = n * r$. $'m'$ and $'r'$ are number of middle and output stage switches

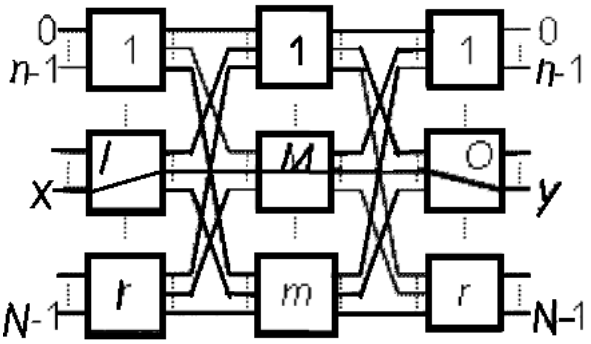


Figure 3.1.: 3-Stage Clos network $C(m, n, r)$ [70]

respectively while 'n' is the number of input signals per switch. The first and third stage switch blocks are numbered from 1- r. (x, y) is a call connection from input port to the output port which covers switches from stage I to stage O and inter-links between stages. Whenever a new connection is requested, the path through center stage must be found. There are different types of algorithms that can be used to find a path through the central stage. A sequential algorithm checks the center stage in sequence from the first switch block and selects the first one available regardless of its previous status. Another algorithm named Quasi-Random starts its search from the switch right next to the switch used for the previous connection and selects the first available switch. Benes algorithm chooses the switch which has routed the maximum connections from a pool of available switches.

The control algorithm's choice affects a connection's blocking probability if the switch fabric is not strictly non-blocking. Benes algorithm offers the lowest blocking probability. Choosing the right algorithm depends on factors, e.g., usage sensitivity technology, blocking probability and time complexity.

3.3. Theoretical framework

End-to-end path allocation in optical networks relies on a service provision, management model, and algorithms for path finding within switching devices. OXCs communicate with peers in optical and client layers and use a local control system to configure the path. Since switch fabrics are not directly managed by the NMS controller, service provision might fail due to different managing entities. Here, we propose the SDN controller, which is assumed to contain already the client layer

(layer two or layer three), to control path setup in the WDM layer while considering OXC's internal limitations simultaneously. The central concept is to abstract the optical network into small virtual networks (devices and links) and hide the hardware details from the SDN controller. The advantages of such virtualization are discussed in [69].

The published paper

Below is the paper published in 26th IEEE International Conference on Systems Engineering (ICSEng); December 2018, https://ieeexplore.ieee.org/document/8638220. Section 3.4 is the title of the paper [71] and all the subsections have same titles as in the original publication. Original pdf proof can be found in the Appendix.

3.4. Dynamic Light Path Establishment In Switch Fabric Using OpenFlow

3.4.1. Abstract

In today's optical networks, Light paths are established through the Network Management System, which is a manual and cumbersome process. Light Paths are computed and pre-setup according to the known traffic demands, and provisioning any new service takes time. Several efforts have recently been made to make the path establishment process dynamic, including the Software Defined Networking approach. However, all of the works have assumed fully interconnected fabrics of the network devices, which is generally not the case. The task of the end-to-end path establishment between network elements and through the fabrics are interrelated, and this task remains incomplete, without consideration of the switch's limitations. This work highlights for the first time the issue of the path establishment dynamically through a switch fabric. The paper briefly explains the problem and suggests an SDN based solution using the OpenFlow protocol. This work contributes a representation of the basic framework which will be required to make the complete path setup process dynamic. The paper also includes an operational flow description.

Index Terms:

Optical switch, SDN, Light Path

3.4.2. Introduction

Rapidly growing demands for the Internet bandwidth and applications have provided an impetus for the change in the current complex and inflexible way of controlling optical networks [72]. To cater today's dynamic demands, carrier operators need a mechanism for the fast, simple and dynamic provisioning of bandwidth resources. Currently in the Core Transport Networks, a central Network Management System (NMS) is employed which is manual and inefficient. Operators at the Network Operation Center do the path planning offline and the network is then laid down according to that plan. Two main components of the optical networks are Optical Cross Connects (OXCs) and re-configurable optical add drop multiplexers (ROADMs) which may manually be configured to provide circuit-switched end-to-end Light paths. The connecting paths in successive switches' fabrics, along with the fiber transmission links between them, form the complete Light path across the network. The process for setting up a service through the network may take several months and once done, the connections usually remain there for a few months or even years. In the cases where a new service demand has to be catered for, there may be a need to change the external fiber connections of the switch manually. In the current control methodology although the computations are done prior, the service may still not be provided at times if no path is available through the any of the switch's internal fabric. The switch has its own control mechanism which uses complex algorithms to find a path through the fabric and setup physical cross connects. The tasks of setting up a path through the fabric and between the switches are interrelated [19], thus at the network level a single control mechanism is necessary which can have a view of the entire network connections. For a new connection request, the controller should be able to find a free path dynamically and decide if the connection request has to be accepted or rejected. This method will avoid wasted NMS resources and time.

The Software Defined Networks (SDN) approach has been recently developed and is being researched for various networking fields. The idea of Software Defined Optical Networks is also not new and has been in discussion in the last few years [73, 74, 75, 72]. However, SDN has inherent characteristics for the packet networks while

optical networks are circuit switched, the task is therefore challenging. A few efforts to mention are [76, 44, 42]. They deal with controlling paths through layer 0 i.e., the photonic (WDM) layer; however, they all assume fully interconnected fabrics, and the switches' limitations are neglected. In above mentioned works, the OpenFlow protocol extensions have been proposed, but none of them have been tested in Mininet which is a popular SDN emulation platform. In our work we propose a way that a single SDN controller can have a view of the fabrics connections together with the links between the switches. We propose to use OpenFlow protocol and Mininet tool to find path through the network. The organization of the rest of the paper is as follows. Background is discussed in subsection 3.4.3 and it also briefly describes the current control mechanism of optical switches. Related work is stated in subsection 3.4.4 and subsection 3.4.5 explains our proposed solution. Finally the paper is concluded in subsection 3.4.6.

3.4.3. Optical Network And Management

Optical network elements that need to be managed include Optical Line Terminals (OLTs), ROADMs, Optical Amplifiers(OA) and Optical Cross Connects. ROADMs and OXCs are intermediate nodes where the switching or routing occurs. NMS deal with the setting, releasing and keeping track of the path connections through the intermediate nodes. End-to-end routes through the network are computed and wavelengths for those paths are also assigned. NMS is implemented through another lower layer of systems called Element Management Systems (EMSs). An EMS is connected to a domain of the network elements usually from the same vendor. It has a view of only its associated network elements. Thus for the overall management, all EMSs in turn communicate with and report to the NMS. The EMS communicates to NEs through an existing traditional Data Communication Network using Simple Network Management Protocol or legacy protocol e.g., Transaction Language 1. In addition to the EMSs, a simplified local management system is usually provided to enable the operator to configure individual network elements manually. Timing diagram in Figure 3.2 explains the connection setup process in a conventional management system [77].

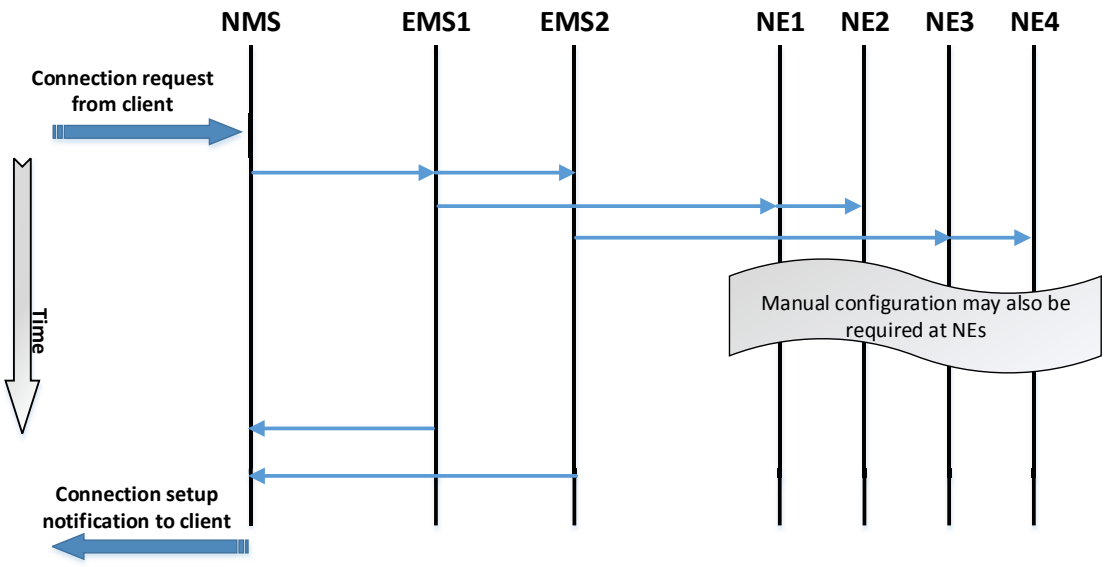


Figure 3.2.: Connection setup process through conventional NMS

3.4.3.1. Switch Fabric and Control Mechanism

Light paths in the Optical networks are switched at OXCs and ROADMs. Each switching device contains input modules/output modules, a switch fabric, and a control system; see Figure 3.3. Light signals are received by input modules where control/signaling information are passed to the control system. Signals leaving the switching fabric are passed to output modules where control information is again appended. A Light path originates at an electrical-to-optical (E/O) transmitter in the ingress node, and terminates at an optical-toelectrical (O/E) receiver in the egress node. Light signal may or may not be converted into electrical domain at intermediate nodes for the switching purpose depending on the switch type, whether Optical-Electrical-Optical (OEO) or all optical (OOO). The switch fabric is usually a collection of optical domain switches purposed to switch an optical connection (wavelength) from an input port to any idle output port. Most of the optical space switches have been re-used from the rich collection of electronic multistage interconnection network architectures (MINs). Examples of these MINs include the optical crossbar architecture, the Benes and three-stage Clos networks [78]. Some of the switches are equipped with the wavelength converters to reduce the blocking probability due to unavailability of the wavelengths. The control system inside

the switching device controls the switch fabric. It has routing tables which have information of the paths through the fabric and their statuses. When a new request from the manager is received, it sends commands to the controller to setup (release) the cross connects associated with the paths. Depending on the switch fabric design architecture, there may be multiple paths between any pair of the input and output ports. It is also possible that there is no available path due to the limited connectivity or paths being reserved for other requests. The switch local controller uses complex algorithms to find a free connecting path in the switch fabric. If a free path is found, the controller then issues the respective control signals to change the states of the physical switches, and the current state of the switch fabric is updated. The types and details of the control algorithms can be found in [19].

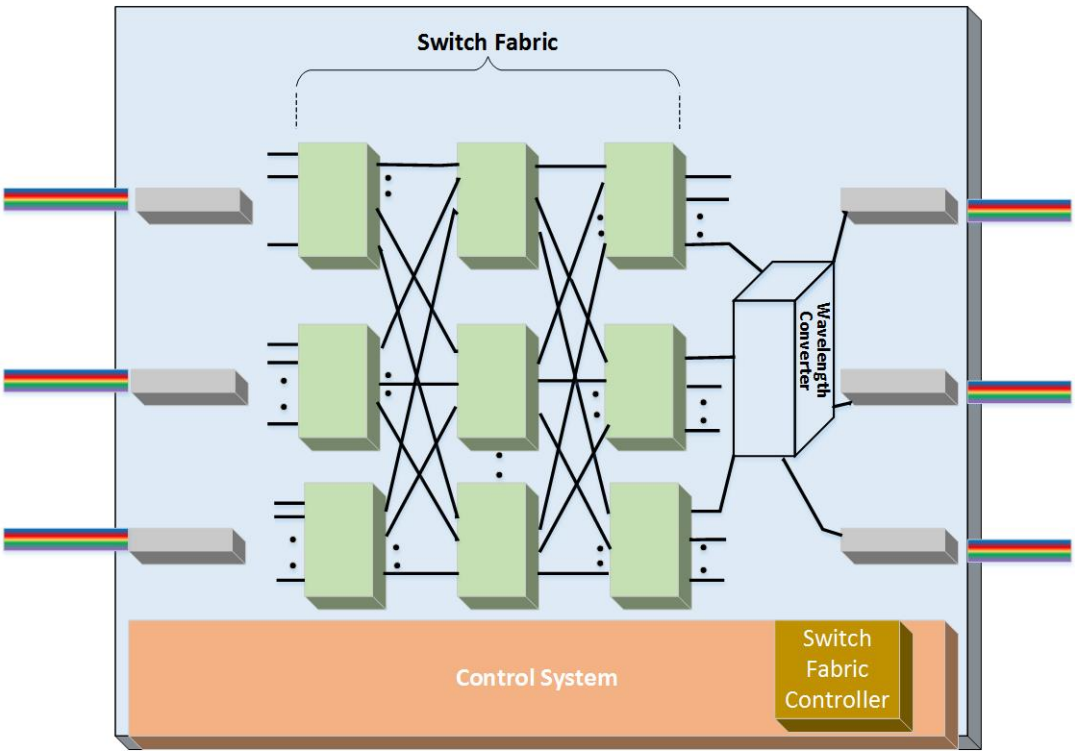


Figure 3.3.: General Optical Switch

3.4.4. Related Work

SDN has started to accommodate optical networks, and over the last decade there have been number of efforts to control Light paths using the OpenFlow protocol. A good survey of previous research work has been discussed in [79] and [80]. These

research works propose OF-Extensions for dynamic circuit switching. In [42] the authors have proposed OpenFlow enabled optical switches which can get commands from the NOX controller to establish paths. However, the request is generated from the IP router and the switch itself can not invoke requests for path setup. References [44, 34] develop an extended OF controller with an application to compute a Light path and implements it through an agent in the optical devices. It is not clear whether this work has considered the switch fabric's connectivity or not. Authors of [76] use almost same concept and make use of a Path Computation Engine (PCE) and Wavelength Assignment (WA) Algorithm in the controller for the Light path computations while an agent is being used for the communication between controller and the switch. In this work, a port emulation entity is also maintained which gets updates about the ports' statuses but no information about the fabric limitations. Open Networking Foundation's Open Transport Working Group (ONFTWG) is working towards the standardization of OpenFlow protocol for transport networks. The latest work [81] by a group of Fujitsu engineers proposed OF controller extensions which (with minor modifications) are accepted by the ONFTWG. They did port to port connections, however, the switch's connectivity constraints are not discussed. ON.Lab is working towards the control of converged packet and optical switches [82]. Recently in [83], they have presented control method using a dis-aggregated transport network.

3.4.5. Proposed Solution

Previous research works assumed that the fabric is fully connected which is not generally the case. Thus even if the path setup instructions are given to the optical switch there may not be an available path through fabric, and the purpose of dynamic provisioning will fail. Also these works have been represented through experiments and no testing has been done in Mininet. Here we devise a mechanism where optical switch internal configurations are represented as a combination of regular layer two OpenFlow switches to the SDN controller. It is also worth mentioning that although the OpenFlow version 1.4 has some provision for the optical ports; Mininet does not support it, so our proposed mechanism uses the current capabilities of Mininet for path establishment.

3.4.5.1. Structural Overview

Our proposed mechanism dynamically retrieves the switches' internal configuration and links and sets up the Light path in an on-demand manner. when the optical signal arrives. Figure 3.4 represents the overview of the proposed structure. The optical switches in the WDM network can be assumed to maintain their external connections through a topology discovery process, or manually populated lists (Neighbor Discovery is an issue, still to be addressed, as referenced by [82]). Our Resource System consists of two main components and may use SNMP/CLI/JSON to interact with the WDM network. The information collector and mapper component extracts the switches' resources and connectivity limitations and maps this information to create a mirror Mininet network. To the controller this network is a regular layer two OF switches' network. The bigger Mininet network can be considered as a collection of small clusters of the OF Switches where each cluster represents the fabric connectivity of a device, see Figure 3.4.

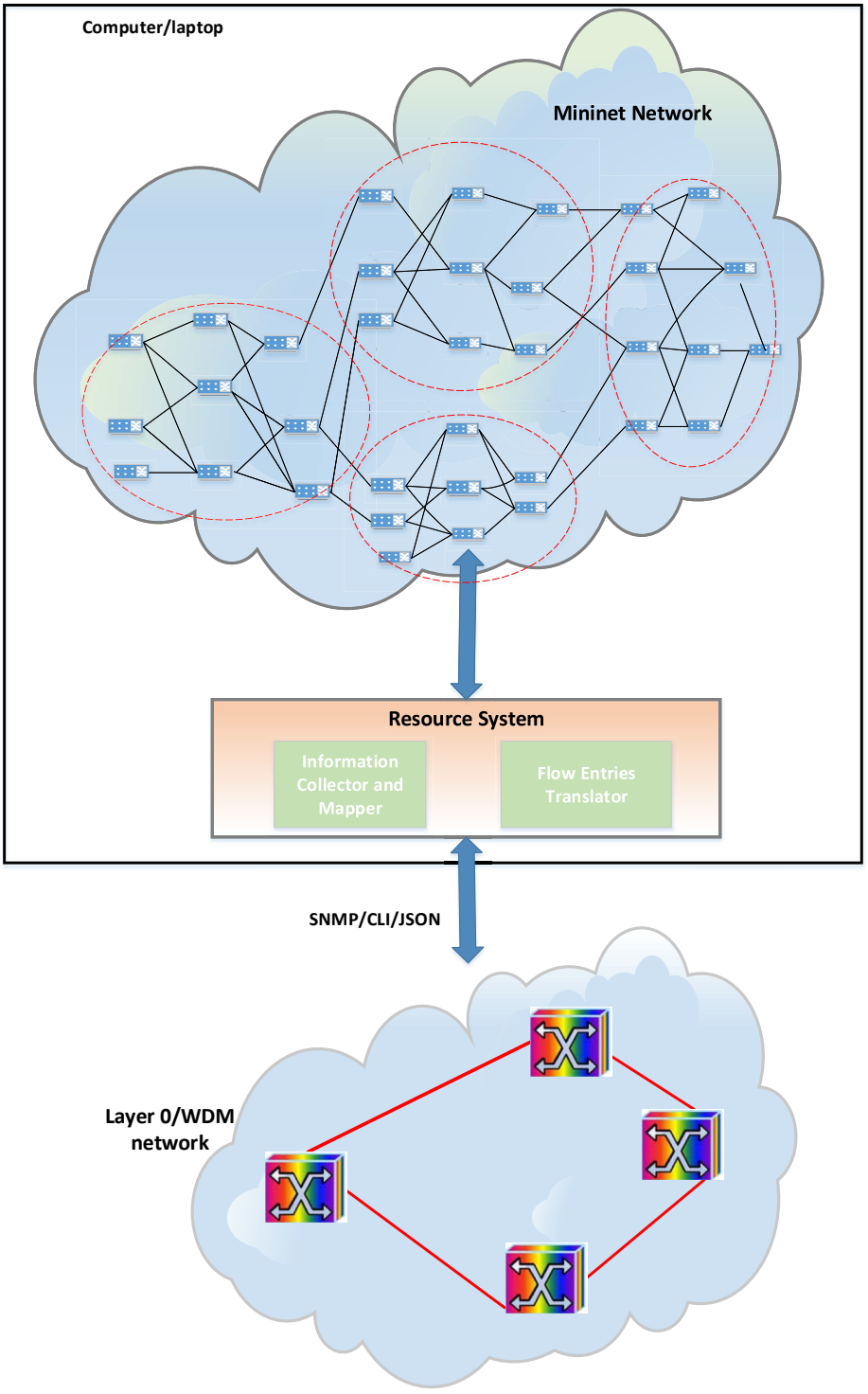


Figure 3.4.: Structural overview

3.4.5.2. Network Graph Creation

We have assumed that the Optical switch has tables/data sheets of its external connections, either manually populated or through a topology discovery mechanism. Our Information collector and mapper component can query the switch using SNMP/CLI/JSON to obtain the information about the switch's internal and external connections. After getting this information, it invokes a Mininet network which replicates the same connectivity which is obtained from the WDM network. The SDN controller can then start its conventional method of network discovery and normal OpenFlow process may start. Figure 3.5 represents the network graph creation process.

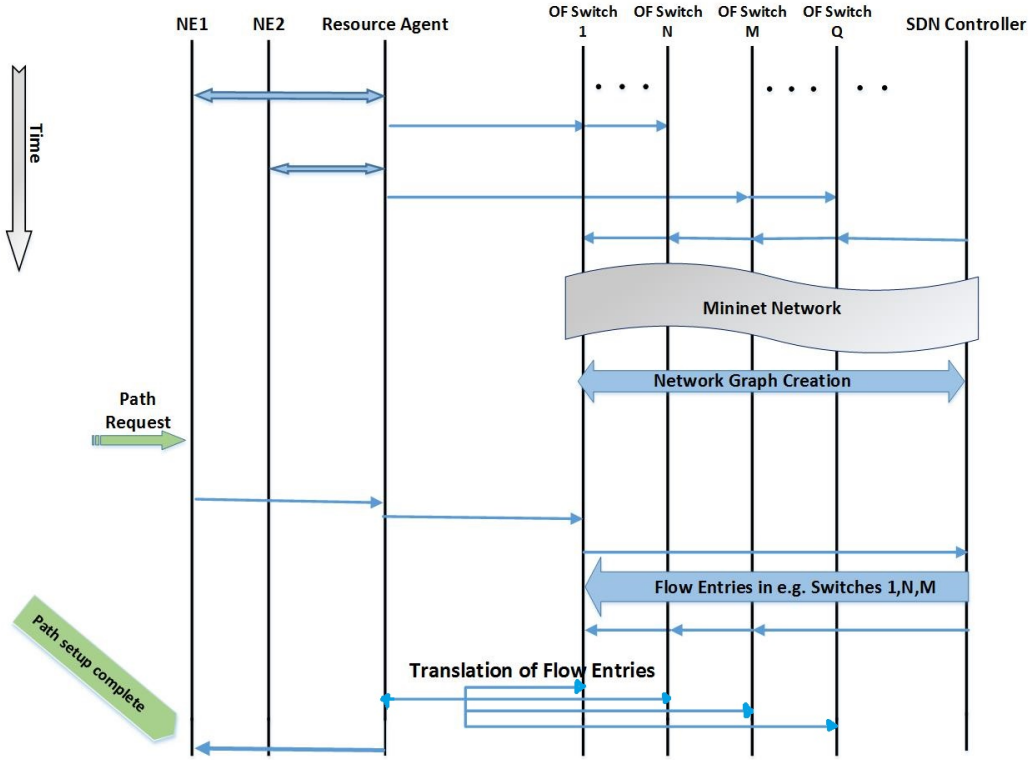


Figure 3.5.: Network Graph creation and connection setup

3.4.5.3. Operational Flow

Once the controller creates the network graph, the network is ready for path setup operation. Figure 3.6 explains the flow of steps. On the arrival of optical signal, the information about source and destination addresses is extracted and forwarded

to the resource system. Here the switch is assumed to have capability of header extraction and conversion to electrical form. The resource system spoofs as a host and sends a frame to the port of the Mininet switch that corresponds to the optical switch's port on which the signal has arrived. The switch checks its flow table entries for matching, if there is no matching then the switch sends the packet to the SDN controller. The controller computes the path and installs the flow entries in all switches. The resource system based on the installed flow entries, can translate these into the commands that can setup cross connects at the optical switch and establish Light paths. During the setup phase, the frames that arrive are lost. Obviously this lost needs to be minimized. However, as setups occur relatively seldom, this is not considered to be a problem. However, it needs investigation. See Figure 3.6 for the flow of steps.

Step by step explanation of the flow chart:

1. On the arrival of Light signal (wavelength) on any port of the optical switch, the system converts it into electrical signal if needed (the flag for conversion is up).
2. Source and destination addresses are extracted from this electrical signal and forwarded to the resource system (which is running inside for example a computer system). SNMP /CLI is used for this purpose.
3. The resource system makes a packet with the source and destination addresses obtained from the optical device and sends to that port of the openflow switch which corresponds to the input port on the optical switch.
4. OpenFlow switch checks for the match of incoming packet and sends to the controller if there is a table-miss.
5. Controller computes the path through the mininet network and install flows in all switches that are part of the path, using Packet In messages.
6. The translator component of the resource system notices any change in the flow entries and convert these flow entries into suitable commands for physical cross connections in optical switches.
7. Since the path is now setup, the conversion from optical to electrical signals is paused for sometime (conversion flag down) and is started again after a regular interval.

8. The data transmission can be started/continued now.

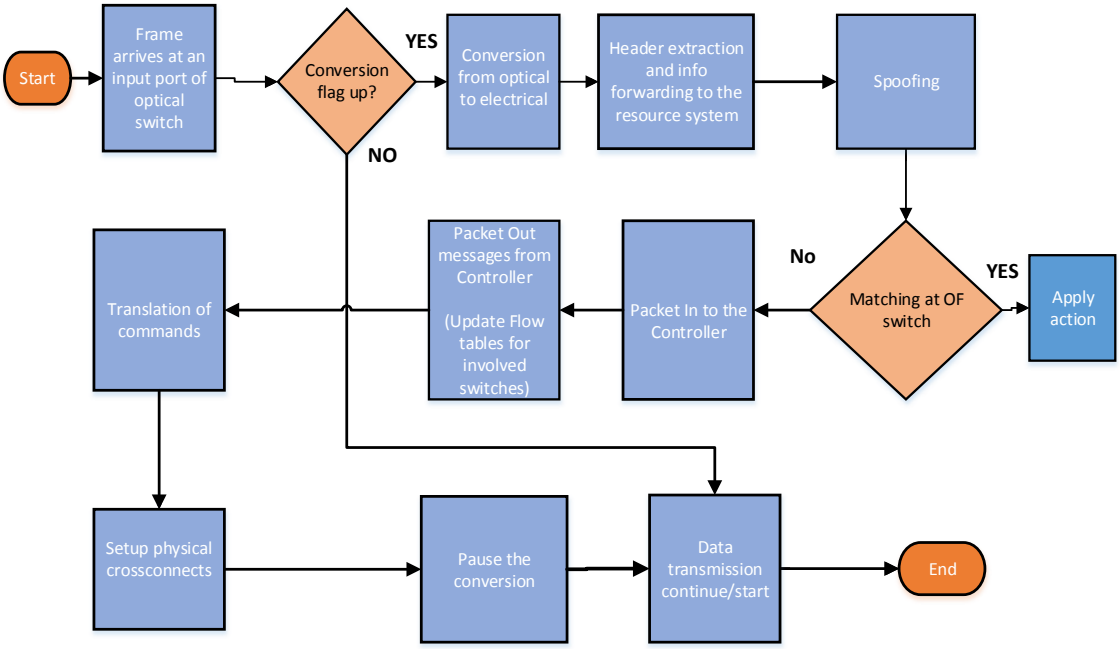


Figure 3.6.: Operational flow of steps

3.4.6. Conclusion

Dynamic Light path establishment in WDM layer of core transport networks is an an important topic at the present time. However, the aspect of a switches' limitations during path calculation has been neglected. In this paper we suggest a mechanism for a complete end to end path establishment while taking into account the switches' fabrics connectivity. The mechanism is SDN based and physical layer connectivity is abstracted as a Mininet network. Our solution is in the process of implementation and will be presented in future work with results. The purpose of this paper is to introduce our ongoing work and direct the attention to the issue that switch fabric should also be considered while path planning for the WDM layer.

References for ICSEng paper

[19] Tareq S. El-Bewab. *Optical Switching*. Ed. by Tarek S. El-Bawab. Boston, MA: Springer US, 2006. ISBN: 978-0-387-26141-6. DOI: 10.1007/0-387-29159-8. URL: http://link.springer.com/10.1007/0-387-29159-8.

[34] Dimitra E. Simeonidou, Reza Nejabati, and Mayur Channegowda. "Software Defined Optical Networks Technology and Infrastructure: Enabling Software-Defined Optical Network Operations". In: *Optical Fiber Communication Conference/National Fiber Optic Engineers Conference 2013* 5.10 (2013), OTh1H.3. ISSN: 1943-0620. DOI: 10.1364/OFC.2013.OTh1H.3. URL: https://www.osapublishing.org/abstract.cfm?uri=OFC-2013-OTh1H.3.

[42] Lei Liu et al. "Experimental validation and performance evaluation of OpenFlow-based wavelength path control in transparent optical networks". In: *Optics Express* 19.27 (2011), p. 26578. ISSN: 1094-4087. DOI: 10.1364/OE.19.026578. URL: https://www.osapublishing.org/oe/abstract.cfm?uri=oe-19-27-26578.

[44] Mayur Channegowda et al. "Experimental Evaluation of Extended OpenFlow Deployment for High-Performance Optical Networks". In: *European Conference and Exhibition on Optical Communication* (2012), Tu.1.D.2. DOI: 10.1364/ECEOC.2012.Tu.1.D.2. URL: https://www.osapublishing.org/abstract.cfm?uri=ECEOC-2012-Tu.1.D.2.

[72] Partha Bhaumik et al. "Software-defined optical networks (SDONs): a survey". In: *Photonic Network Communications* 28.1 (2014), pp. 4–18. ISSN: 1387-974X. DOI: 10.1007/s11107-014-0451-5. URL: http://link.springer.com/10.1007/s11107-014-0451-5.

[73] Saurav Das, Guru Parulkar, and Nick McKeown. "Unifying packet and circuit switched networks". In: *2009 IEEE Globecom Workshops, Gc Workshops 2009* (2009). DOI: 10.1109/GLOCOMW.2009.5360777.

[74] Steven Gringeri, Nabil Bitar, and Tiejun J. Xia. "Extending software defined network principles to include optical transport". In: *IEEE Communications Magazine* 51.3 (2013), pp. 32–40. ISSN: 0163-6804. DOI: 10.1109/MCOM.2013.6476863. URL: http://ieeexplore.ieee.org/document/6476863/.

[75] ONF Solution Brief. *OpenFlow-enabled Transport SDN*. 2014.

[76] Mahmoud Bahnasy, Karim Idoudi, and Halima Elbiaze. "OpenFlow and GMPLS Unified Control Planes: Testbed Implementation and Comparative Study". In: *Journal of Optical Communications and Networking* 7.4 (2015), p. 301. ISSN: 1943-0620. DOI: 10.1364/JOCN.7.000301. URL: http://www.opticsinfobase.org/abstract.cfm?URI=jocn-7-4-301.

[77] Dimitrios Pindarakis and Subir Biswas. "Management and control of optical networks". In: 2002, pp. 31–47.

[78] Haitham S. Hamza and Jitender S. Deogun. "Wavelength-exchanging cross connects (WEX) - A new class of photonic cross-connect architectures". In: *Journal of Lightwave Technology* 24.3 (2006), pp. 1101–1111. ISSN: 07338724. DOI: 10.1109/JLT.2005.863279.

[79] Rodolfo Alvizu et al. "Comprehensive Survey on T-SDN: Software-Defined Networking for Transport Networks". In: *IEEE Communications Surveys and Tutorials* 19.4 (2017), pp. 2232–2283. ISSN: 1553877X. DOI: 10.1109/COMST.2017.2715220.

[80] Daniel C Kilper. "Optical Physical Layer SDN [Invited]". In: 10.1 (2018), pp. 110–121. ISSN: 19430620. DOI: 10.1364/JOCN.10.00A110.

[81] Shinji Yamashita et al. "Extension of OpenFlow protocol to support optical transport network, and its implementation". In: *2015 IEEE Conference on Standards for Communications and Networking, CSCN 2015* (2016), pp. 263–268. DOI: 10.1109/CSCN.2015.7390455.

[82] Marc De Leenheer, Guru Parulkar, and Tom Tofigh. *SDN Control of Packet-over-Optical Networks*. 2015.

[83] Marc De Leenheer et al. "SDN Control of Optical Networks". In: *Ecoc* 1 (2017), pp. 1–3.

Part II.

Proving the proposition

4. Implementation and Performance

4.1. Overview

This chapter presents the implementation of (RMS) model based on the theory represented in [71].RMS consists of several components to implement the process of path establishment in optical switches/cross-connects. Path establishment steps are explained in three major phases. The first phase is the abstraction of layer one switch connectivity, the second is path computation, and the third is the path setup. The purpose of developing an abstraction layer is to model, layer one connection to an OpenFlow controller as a network of Layer two OpenFlow switches. This gives the controller a bird's-eye view of the network and the capability to compute a path while not having any information about the wavelengths.

Furthermore, details of the modules which construct the RMS are discussed in the attached published article (The link for actual codes is available in the Appendix). The model aims to address the SDN concept's current constraints with respect to the circuit switching capability. Moreover, building a component-based solution creates room for future inclusions that grants the development of a more comprehensive and autonomic solution to the current management strategy.

The Mininet emulator is used for abstraction purposes. Mininet is a tool for network emulation that provides an environment for developers and researchers to emulate the OpenFlow network using a single machine for testing and validation. Mininet uses lightweight process-based virtualization and can build a virtual network of hosts and switches on a kernel. Users can create user-space or kernel OpenFlow switches, controllers and hosts for communication purposes. Hosts and switches are connected using virtual Ethernet (veth) pairs in Mininet. Mininet can help simplify the development processes, e.g., testing, debugging and implementing in the real world. New SDN-based concepts can first be tested on an emulation of the network to be deployed. Once a developed application is working on Mininet, it can be imported to the actual network to be used in the real world, in contrast to a

simulation tool.

Furthermore, the implemented model is tested against an actual switch to demonstrate the tasks and scenario and validate the model's functionality. The computational resources required are also calculated through vigorous testing.

4.2. Layer one connectivity abstraction

The first step for dynamic path establishment is to create an OpenFlow network topology that virtualizes layer one network connections. While abstracting the topology, links between devices and the possible internal connections of the switches, i.e., constraints, wavelength conversion capability, and wavelengths to be added/dropped, are the factors that need to be considered.

Only two OXCs are considered in the below-included publication to simplify the demonstration of the translation mechanism. Since transponders, demultiplexers and transmission cables do not play any role in switching tasks, these are eliminated in the abstraction process. The key to virtualizing an OXC as an OpenFlow network is to use wavelength-selective architecture for any fabric topology, as the below-included article explains. Wavelength-selective architecture is necessary because it allows the separation of the wavelengths. The architecture ensures to have represented all optical paths in the same way as in the original fabric while maintaining separation between wavelengths. Thus controller finds the route without knowing the wavelengths. Later a translation mechanism can be used to map this path computed by the SDN controller for OXC.

For implementation of the first step, the most popular MIN architecture of Clos configuration (strictly non-blocking), [25] is chosen. A variety of network topologies (e.g., flat butterfly [84], B-cube [85], and [86]) exist in literature to connect tens of thousands of nodes (ON/OFF cells), however; topology choice often depends on the complexity of routing software, number of physical links and performance as well as scalability of the fabric [87].

To ensure that the wavelength-selective architecture employed for OpenFlow abstraction does not allow wavelengths to get mixed, a "Ping" connectivity-test tool is used. For generating ping test packets, a python's tool, namely "scapy," has been

utilized—coding details for which are provided in the appendix. The attached article initially provides some necessary background and preliminary information to explain why and how abstraction is done and how the translation of a switch can help find the path in core networks. Furthermore, it describes an algorithm for the generation of the abstracted topology.

The abstraction process can be integrated into the developed RMS system or can work as an individual component. The abstraction is done using python programming language, which calls Mininet libraries. At the time of implementation, since Mininet libraries are not supported by Python 3, thus we used Python 2. The script can be called a CLI script. It can also be passed with different arguments for providing custom input parameters of several fibers, wavelengths etc.

The published paper

Below is the paper published in IEEE 29th International Telecommunication Networks and Applications Conference; `https://ieeexplore.ieee.org/document/9078014`, November 2019. Section 4.3 is the title of the paper [15] and all the subsections have same titles as in the original publication. The original PDF proof of the publication can be found in the Appendix.

4.3. Mininet Topology: Mirror of the Optical Switch Fabric

4.3.1. Abstract

Software Defined Networks (SDN) is a new approach to change the conventional networking and is being researched in the various networking domains. To test and prototype SDN based concepts, a lightweight and closer to reality option is Mininet emulator. Mininet emulates SDN behavior by creating a virtual network of elements using Network Namespaces on a single Linux kernel machine. In this work, we have developed a Mininet topology that emulates the structure of a WDM Switch. The topology mirrors the paths that can be used by the wavelengths in a WDM switch

fabric. The SDN controller can find a path for communication between hosts through this network. We simulated our Mininet topology, which mirrors an architecture for three wavelengths. The Ping command results show that only a set of hosts can be reached out by a particular host; which is the requirement of a WDM switch, and this verifies that Mininet topology is mapping a WDM switch.

Index Terms:

Optical switch, Mininet, Software Defined Networks

4.3.2. Introduction

Current conventional IP networks are complex and pose many operational and management challenges to the network administrators. IP networks have distributed control plane which is vertically coupled with the data plane within network devices. To express any particular network behavior, devices are to be configured individually and to deliver any new functionality or policy; re-configuration of the devices is required. IP networks are not flexible, and due to their static nature could not evolve with time. Software Defined Networking (SDN); on the other hand, is a new approach to the architecture of computer networks that presents a change of conventional networking. It decouples the vertical integration of the network's control plane from the data-plane, putting the control in a central controller, which shall have the ability to program and manage the network dynamically.

SDN is an emerging paradigm that allows network administrators to deploy their policies in the controller in the form of Applications. The controller enforces those policies in the form of rules in the underlying switching hardware known as forwarding devices, using open interfaces. Depending on the principles set up by a controller application, the hardware device can behave like a router, Layer two switch or a Middlebox. An Open interface protocol, namely "OpenFlow," became a standard for communication between the controller and the switching devices. OpenFlow was introduced by N. McKeown et al. [55] as an academic activity at the Stanford University, USA. OpenFlow implements SDN and allows control and management of the network dynamically, which is a difficult task in traditional networks due to the various proprietary and closed interfaces.

SDN and OpenFlow are being researched in many domains, including academia and industry. Over the past few years, many industries have been considering SDN as a solution to their problems and are inclined towards SDN architectures. Some vendors of commercial switches now include OpenFlow support in their switching hardware [88]. Various groups are also conducting several standardization activities, e.g., Open Networking Foundation (ONF) aims for promoting and standardizing SDN using open standards and is funded by different enterprises e.g., Google and Facebook [57].

Testing and prototyping new SDN and OpenFlow ideas with real devices is challenging and costly. One can think about a network of virtual machines ; however, it has been experienced as a heavyweight solution [10]. These and other reasons, drive the researchers to use the emulator called Mininet. Mininet is a new rapid prototyping environment which offers lightweight virtualization with extensible CLI and API support. It is developed with attributes like flexibility and scalability and is also interactive and shareable, the characteristics which other prototyping environments are lacking. It provides a simple and easy way to the real systems because a working prototype in Mininet should require no changes to the code or layout when deploying on actual hardware [10].

Mininet has a GUI editor called Miniedit. It is an experimental tool which helps to quickly create and run topologies and test them by changing the component's specifications. In this work, we have designed a topology in Miniedit which mirrors a specific type of optical switch fabric. This topology and SDN controller can help in path finding, which is a complex problem in optical networks. We also simulated the topology with a CLI program with some custom input parameters. Our paper is organized as follows. Background of the work is given in subsection 4.3.3. It describes optical networks and the problem of static path finding. subsection 4.3.4 briefly introduces Mininet. Model is explained in section 4.4, that we have made for optical switch fabric. Simulation scenario and also the results are described in subsection 4.4.1. At the end, in subsection 4.4.2 conclusions are drawn.

4.3.3. Background

4.3.3.1. Optical Networks

Developments in the optical devices' technology led to the rise of the second generation of optical transport networks. It expanded the use of Wavelength Division Multiplexing (WDM) technique from transmission purposes to optical networking. One of the significant advantages achieved through optical networking is that some of the electronic switching and routing tasks are now incorporated into the optical part of the network. Thus the optical network layer serves a major role in a transport layer infrastructure.

Typically, Optical network structure spans over three main tiers, namely Access networks, Metropolitan Area Networks (MAN) and Core (backbone) networks as shown in the figure. Main differentiating attributes among tiers are the number of users, expected capacity, and geographic area. Core networks span over thousands of kilometers and are structured to interconnect various MANs on an international scale. They are mostly connected in a mesh topology and engage very high-speed optical equipment and transmission links [89]. Figure 4.1 is the depiction of a generic simplified optical WDM network.

The main components of WDM network are optical line terminals (OLTs), (reconfigurable) optical add/drop multiplexers ((R)OADMs), and optical cross-connects (OXCs)/Optical Circuit Switches (OCS). The function of the OLTs is to multiplex/demultiplex multiple wavelengths at the edges of network into a composite signal; and also to convert to a suitable wavelength to be transmitted over the fiber. ROADMs are required where some of the wavelengths need to be dropped locally, and others need to be switched to their destinations. OXCs are electronic devices and have a larger number of ports and wavelengths involved. They perform almost the same tasks, as ROADMs but on a much larger scale and are deployed in core optical networks.

Using WDM technique, data is transmitted at different carrier wavelengths of light over a fiber. Each wavelength of light can carry a signal with its own speed and protocol, and is independent of what's on the other wavelengths hence can serve to transport SONET, Ethernet and IP packets simultaneously [17], see Figure 4.1. Data enters the Core network via edge devices and is routed to its final destination by ROADMs and OXCs. Each wavelength traverses some hops before reaching its

final destination providing optical circuits called Lightpaths. Light paths are end-to-end optical connections which require fixed allocation of bandwidth resources [90]. Optical circuit switched networks need lightpaths to be established between its source and destination before actual data transmission [91]. Lightpaths are set up and broken down according to the requirements of the users.

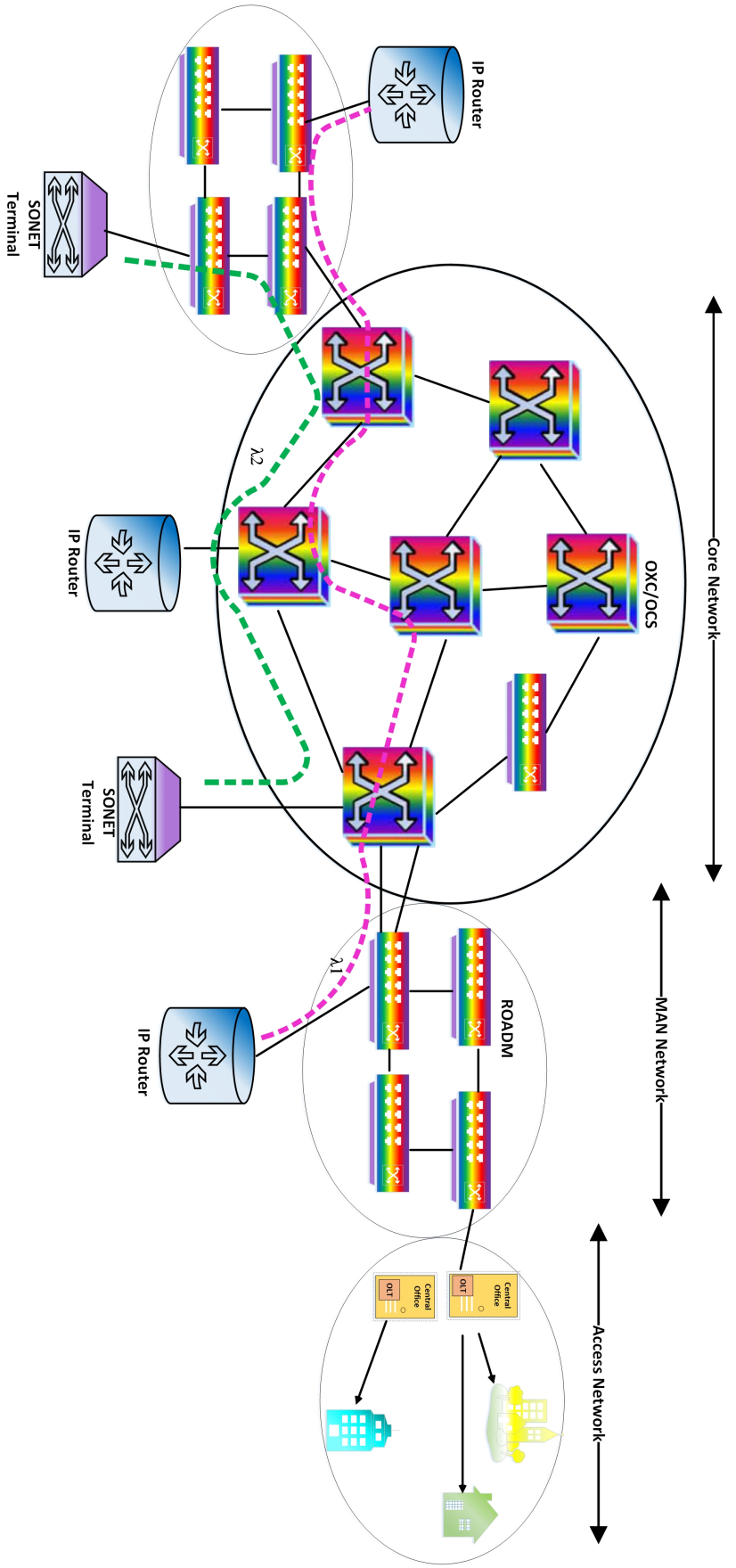


Figure 4.1.: General Optical Network

4.3.3.2. Optical Switch

Optical network switches exploits Wavelength Division Multiplexing technique to route and switch the light signals. Sometimes a light signal may also be required to change its wavelength along the route called "wavelength translation." This process takes place within the optical switch using wavelength converters. A generic architecture of an optical switch is shown in figure Figure 4.2. Each OXC is expected to integrate input & output modules, a switch fabric, and a control system. The internal architecture of the switch may employ Optical-Electrical-Optical (OEO) or all-optical (OOO) technique for switching and routing purposes [6]. Incoming light is received on input modules. Signals get routed through the switch fabric and collected at the output modules and are processed to be transmitted on fibers.

The control system has information about the routing and different paths through the fabric and their statuses. When a new connection request is received, the control system uses algorithms to find a free connecting path in the switch fabric. After finding a path, it sends commands to a controller to setup/release the switch cells (crosspoints) in the structure associated with that path. Various standard and proprietary algorithms exist that can be used to set up switch fabrics. The controller generates the respective control signals to update states of the crosspoints, and then the state table of the switch fabric is updated. Depending on the switch fabric design and topology, there may not always be a free path. In such cases, service cannot be provided, and the request is blocked. Algorithms that perform these tasks are called control algorithms [19].

4.3.3.3. Switch Fabric

Optical switches can employ either OEO architecture or OOO. Both architectures have their own pros and cons, see [92]. In OEO architecture, optical signals are converted to electrical. After getting switched through the electronic fabric it is converted back to optical form while in OOO architecture signals remain in optical form. Due to different design principle, two different architectures are usually used for the internal fabric of the switch e.g. Wavelength Selective architecture is typicaly employed for OOO structure, while OEO architecture employs structures which come from extensive collection of electronic multistage interconnection

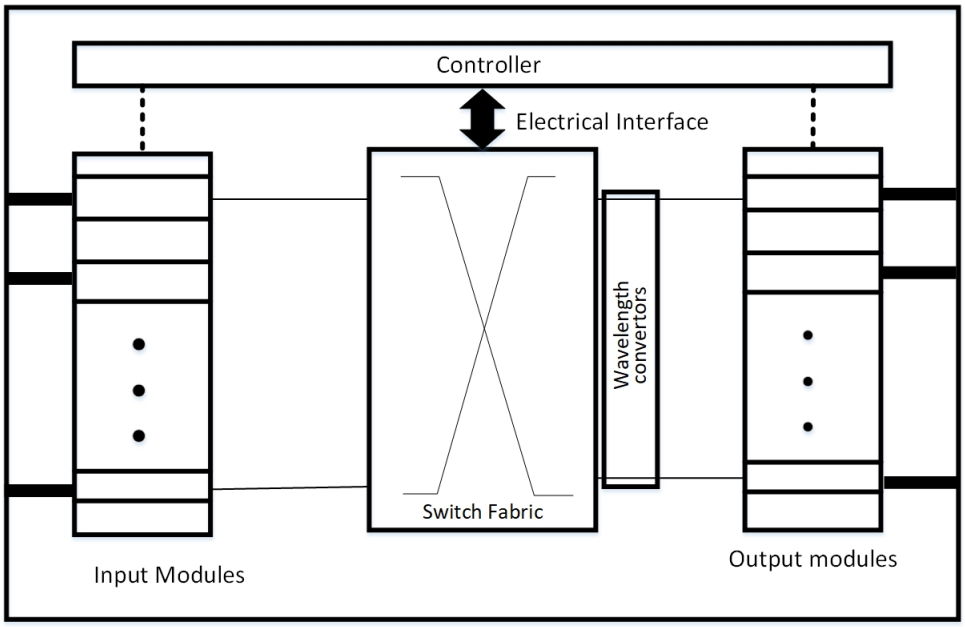
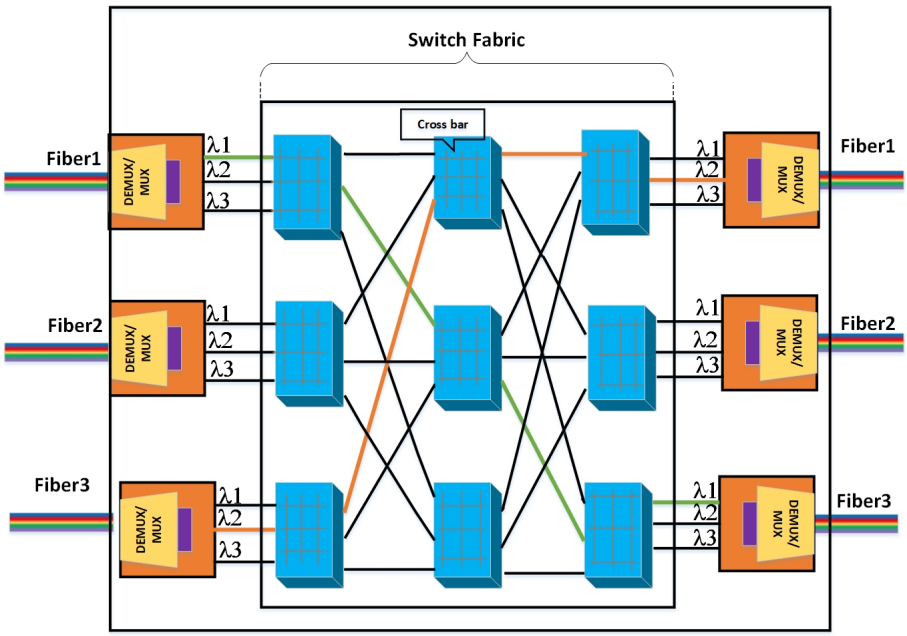
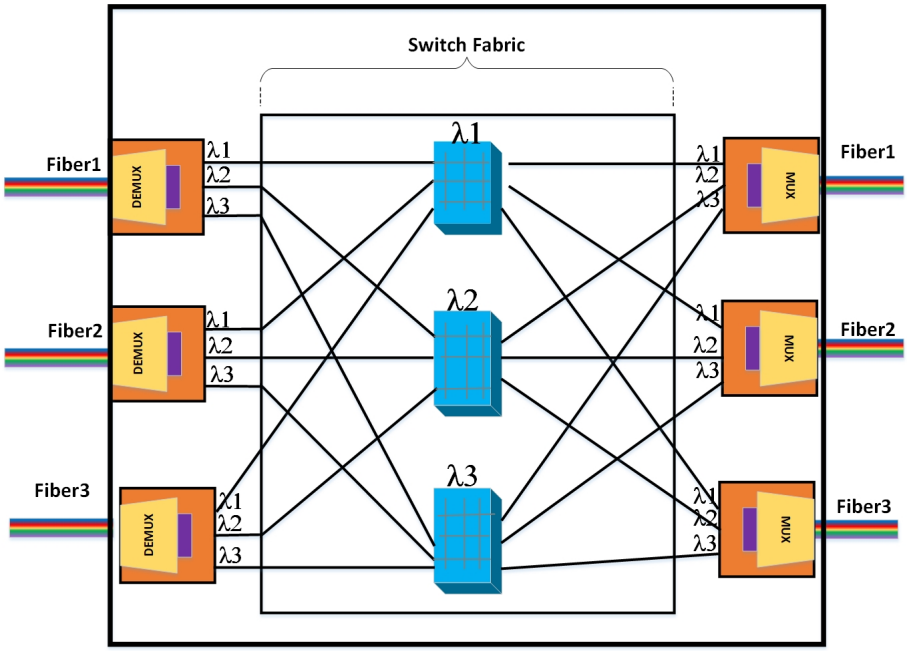


Figure 4.2.: Generic architecture of the optical switch

networks (MINs) [93]. MINs contain many switch cells that are interconnected in stages to form a topology. Most widely studied and used switching fabric is based on Clos network topology [94, 70, 95, 93, 96, 27]. Figure 4.3 shows two types of architectures used as optical cross connects, Wavelength-Selective architecture and Clos architecture. There is difference in the operation of two architectures [29]. Wavelength selective architecture has a specific block for each wavelength and its structure makes sure that no same wavelength is received by the same multiplexer thus is useful for all-optical switching. Clos configuration can satisfy this wavelength constraint in two ways. It can either employ mechanism to do switching in electronic domain and then at output ports convert the wavelength to a suitable one using transponder (that is why, useful for OEO) or using a graph coloring algorithm for correct configuration of the wavelengths e.g., [97].



(a) Clos architecture



(b) Wavelength-Selective architecture

Figure 4.3.: Two optical switch fabric architectures

4.3.3.4. Path establishment in Optical Networks

Optical core networks are circuit-switched networks which require setting up Light paths before transmission starts and then remain set up until the service is diminished. Thus wavelengths are reserved for the duration of the service. A LightPath has to have a common wavelength along all the links that it crosses, and also distinct wavelengths have to be assigned to the paths which share common connections. Therefore, planning for such wavelength-constrained networks is challenging. The algorithms and protocols that find paths and assign wavelengths should be efficient enough to optimize the utilization of network resources and minimizing the future blocking of lightpaths. This problem is called the routing and wavelength assignment (RWA) problem.

The RWA problem is generally resolved for two different traffic conditions. When the traffic pattern is known in advance, it is referred to as offline or static RWA, while when the requests arrive at random and unknown times, is referred to as online or dynamic RWA. The offline RWA is done when planning the WDM network, while the online problem occurs when the network is operational [98]. Offline RWA and setting up/releasing of LightPaths are done on network management level through "Network Management System". Switches are configured to allow circuit-switched paths using SNMP or any propriety protocol or CLI. Introducing a new service requires manual intervention and may take a few months. In online RWA, global or local network state information is necessary to compute paths. An edge router or a centralized node can work this out. Once paths are computed, efficient signaling and reservation protocols are required and invoked to establish paths, and if no resources are available, the request may be blocked. Signaling protocols are responsible for setting up lightpaths and exchange of control messages through the network [4]. Also the controller in the optical switch/OXC uses this control information to set-up the switch fabric [19]. The switch fabric is connected to an electrical control interface. The control interface allows electronic control signals to reach the switch fabric.

4.3.4. Mininet

Mininet is the network emulation system that provides lightweight virtualization to prototype and evaluate SDN protocols and applications on a single device. Mininet

essentially creates a virtual network, running real kernel, switch and application code thus emulating real network. Network can be customized using the Mininet CLI. Mininet creates virtual networks and runs its elements (routers, switches, hosts, links) on a single Linux kernel using process-based virtualization and network namespaces. Virtual network devices in Mininet (hosts, switches, controllers) and links are the real things and they behave as like discrete hardware elements which are created using software rather than hardware. Mininet can be experimented by writing Python scripts. Miniedit is a GUI for Mininet which is an easier and quick way to make and test a topology.

- Mininet hosts: These are processes which behave same as real machines, packets get processed through them as if they sent through real Ethernet interface and device, with certain link speed and latency.
- Switches: Mininet has software-switches e.g., Open vSwitch or OpenFlow reference switch.
- Links: These are virtual Ethernet pairs, which provide connection between emulated devices.
- Controllers: Mininet has a default OpenFlow controller that implements a simple Ethernet learning switch. User can create his own controller & pass it into Mininet. (NOX, POX, FloodLight, Ryu)
- External OpenFlow Controllers: Mininet can also be connected to an external controller which exists somewhere else, for example on LAN or in another Virtual Machine.
- Routing: Using the OpenFlow protocol, switches can be programmed to do almost anything user wants with the packets that enter them.

4.4. Mininet Model of Optical Switch Fabric

We have used Miniedit to model a topology which abstracts an optical switch fabric. We have selected a basic 3-stage Clos architecture with no wavelength converters for the demonstration purpose. Here we have assumed three fibers with each having three wavelengths as in Figure 4.3 (a). Clos architecture serves as a

switching fabric, and any input wavelength can be switched to any output; however, no same wavelengths can be received on the same Multiplexer as explained in subsubsection 4.3.3.3. This wavelength constraint is mapped through the architecture of our Mininet topology. A Wavelength-selective architecture is constructed in Miniedit to represent and function as a Clos structure.

In Miniedit structure; see, Figure 4.4, OpenFlow switches S1-S9 represent the Demultiplexers for three input fibers each carrying three wavelengths namely λ_1, λ_2, λ_3 and switches S19-S21, S31-S33, S43-S45 represent Multiplexers for output fibers. Every wavelength from the source fiber, can reach to a block of switches connected in the Clos configuration. For example, wavelength λ_1 from the source "Fiber1" and other fibers (Fiber2 and Fiber3) can only reach the block of switches represented in Figure 4.4 as λ_1 and cannot reach the blocks specified as λ_2 and λ_3. Similarly, a wavelength named as λ_2 from all source fibers can be switched through block λ_2 of Clos-configured switches represented in figure as λ_2. Same applies to the wavelength λ_3. Total number of ports being used by OpenFlow switches is the same as in the original optical switch.

due to this Wavelength-Selective architecture, only a particular input host can reach out to a particular output host. For example h1 can reach out to h4, h7, h10, h13 and h16. In the same way h2 can reach out to h5, h8, h11, h14 and h17. In this topology SDN controller finds path between the source and destination hosts. Input and output hosts can be considered as network devices connected to the optical switch, operating on a particular wavelength.

If middleware agent is developed which can translate layer one fabric structure to a Mininet topology and also provide a trigger (with destination address) on the arrival of light, the SDN controller can find the path through this topology for the destination address. The OpenFlow switches will receive the rules from the controller which can be translated back by the agent in the form of instructions for the optical switch to setup its fabric. To the SDN controller, the switches-network is an ordinary layer two network. The mechanism for such kind of dynamic path establishment in optical networks is explained in [71]. This mechanism will help in eliminating need for complex algorithms that are used to configure the switch fabrics. If all the devices get registered with the middle-ware system and their fabric pattern gets also translated into a Mininet topology, the SDN controller can find path through the complete optical network i.e., within switch fabrics as well as links

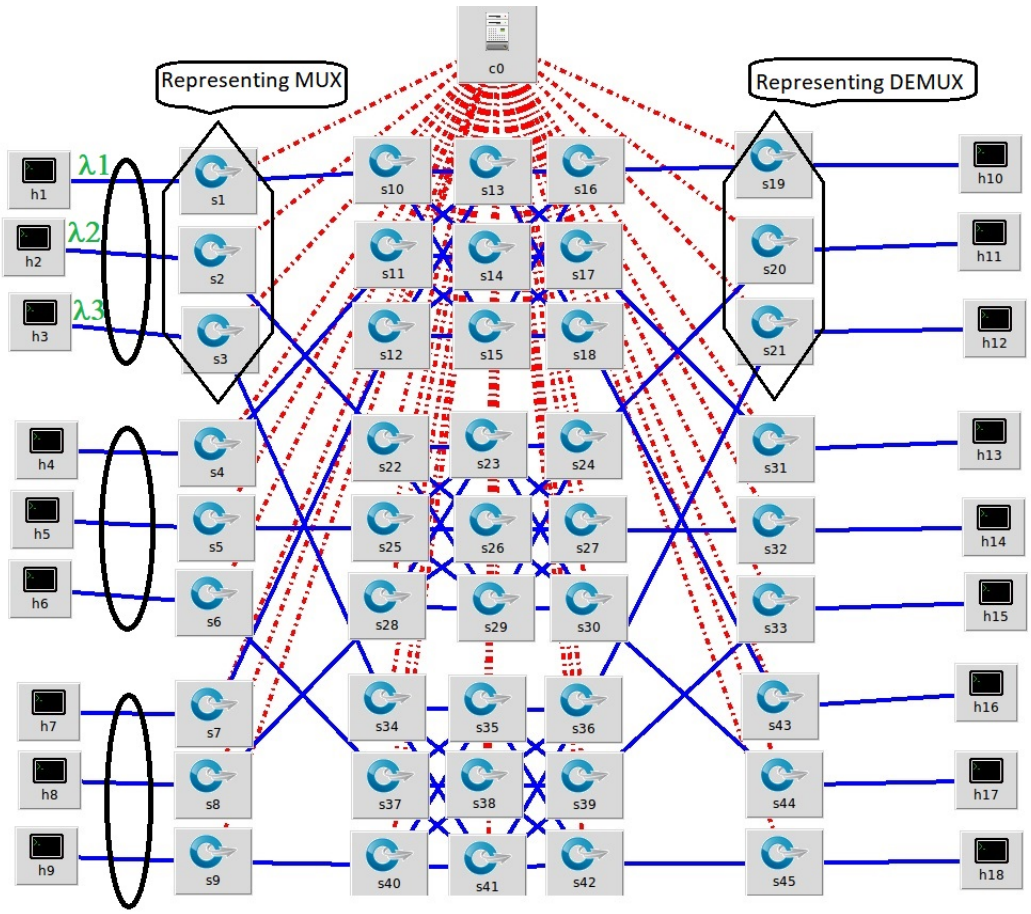


Figure 4.4.: Mininet topology mapped for a Clos-based Optical switch

between devices. This is due the fact that the controller will now have a complete network graph for the whole network. In order to deal with fabrics other than Clos and having wavelength converters, the architecture of our Mininet topology has to be changed.

4.4.1. Implementation and Results

Our sample model represents two optical switches (OXC/OCS) cascaded. Two connected switches and its mapped Mininet topology are shown in Figure 4.5 (a) and (b). Fibers connected at the output of every switch are collected in a transmission cable to span long distances between the devices deployed far apart. In Figure 4.5b the transmission cable is represented by OpenFlow cascaded switches within blocks

A, B, C. However, technically the switches in the blocks connected in series can be replaced by single switches. Thus to avoid these extra switches, we have used a single switch for simulation purpose to represent the connection for one wavelength between devices. See, Figure 4.6.

Our system specifications were "Intel Core i7-7600U CPU @2.8GHz, 64-bit operating system with 16.0 GB installed memory". Mininet version 2.3 was installed on Ubuntu version 16.04 LTS. OpenFlow version 1.3 is used and switches in the topology are all "Open vSwitch Kernel Mode."

We used RYU as an SDN controller and launched its Spanning Tree Protocol application for OpenFlow version 1.3 to avoid loops for path calculation. We launch the controller first and then run the topology. Since the topology has many switches and loops, so we needed to wait until all the switches are registered and paths are calculated. We then used ping command to check connectivity. Results were as expected. Host h1 (mapping $\lambda 1$ for Fiber1) can only reach h2, h3, h4, h5 and h6; see, Figure 4.7. Likewise, host h7 (mapping λ_2 for Fiber2) can reach h9, h11, h13, h15 and h17. This verifies that, a controller having no knowledge about wavelengths, is able to find the correct path, and will not send any wavelength to the wrong port.

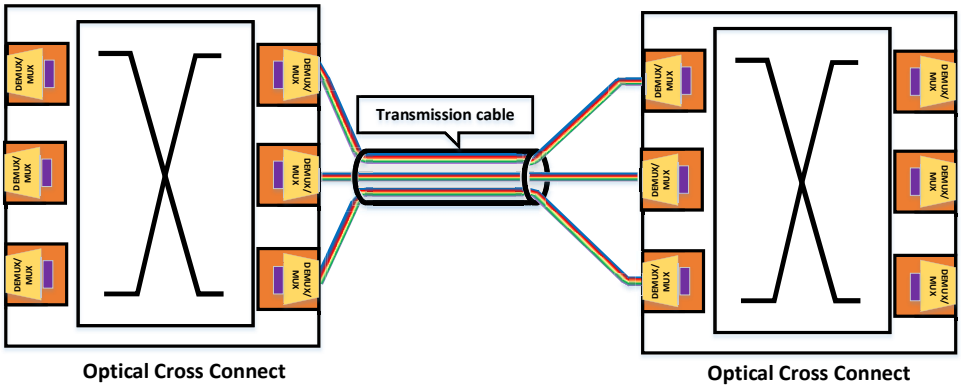
We also wrote a CLI program to generate this type of topology with custom no of wavelengths and number of stages for MIN structure. Writing a script with custom inputs is useful because the structure and input wavelengths may differ for real switches. General layout of our script is represented in Fragment 1. Simulated script also resulted with successful pings.

Algorithm 4.1 Generate Mininet Topology

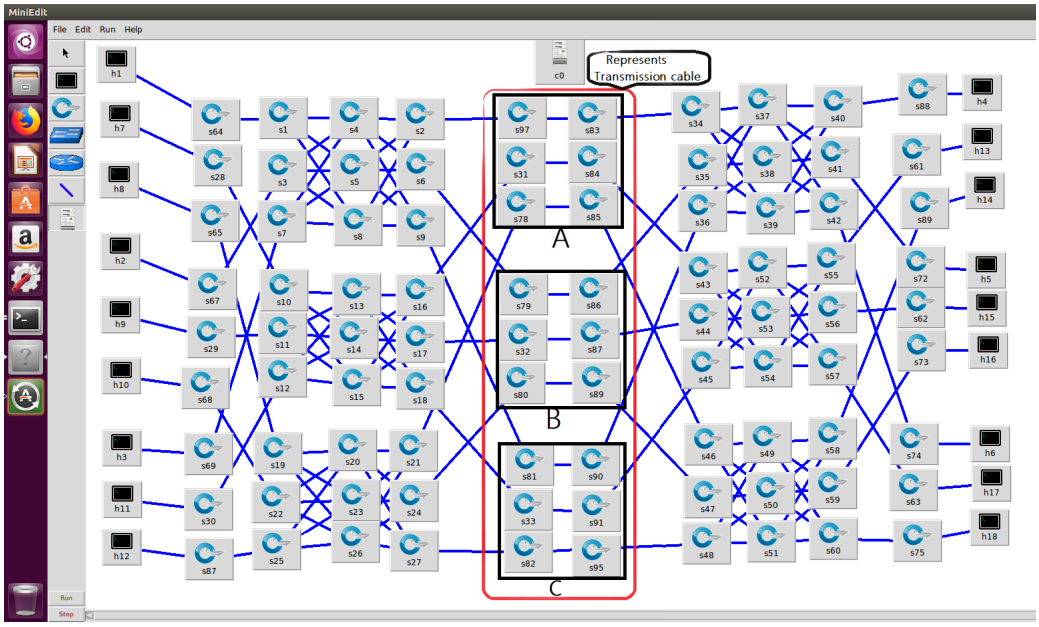
```
1: Import modules ▷ Import required modules e.g. mininet, Topo, controller
2: Define topology "class ClosTopo" ▷ Define the pattern of switches and links
3: Add switches and links for received arguments ▷ switches and links making clos
   topology
4: Add hosts ▷ To check connectivity
5: Add switches representing transmission cable ▷ connecting switches between
   OXCs/Roadms
6: procedure
7:     Pass Arguments (Input N,K,F,O) ▷ Number of MIN stages, wavelengths,
   fibers and OXCs respectively
8: end procedure
9: procedure MAIN FUNCTION
10:    Initialize Controller ▷ Start controller
11:    Initialize Topology ▷ Define topology elements
12:    Start and build network ▷ Start switches and communication
13:    Start CLI ▷ Open Mininet CLI
14: end procedure
```

4.4.2. Conclusions

Path finding in optical networks is a crucial and complex problem. In this paper we have presented continuation of our work [71] which devises a mechanism for dynamic path finding in optical networks. We have developed a topology of OpenFlow switches in Mininet which replicates functionality of an optical switch fabric. We have demonstrated that an SDN controller, without knowing about the wavelengths, can be used to find paths through Clos switch fabric. We used ping command to check and verify that the topology is working as a Clos based switch fabric. The work in this paper supports the idea that if internal configuration of all the network devices is translated to Mininet topology then SDN controller can be used for finding path through switch fabric as well as the whole optical network using Mininet.



(a) Actual cascaded switches



(b) Miniedit representation of cascaded switches

Figure 4.5.: Two optical switches cascaded

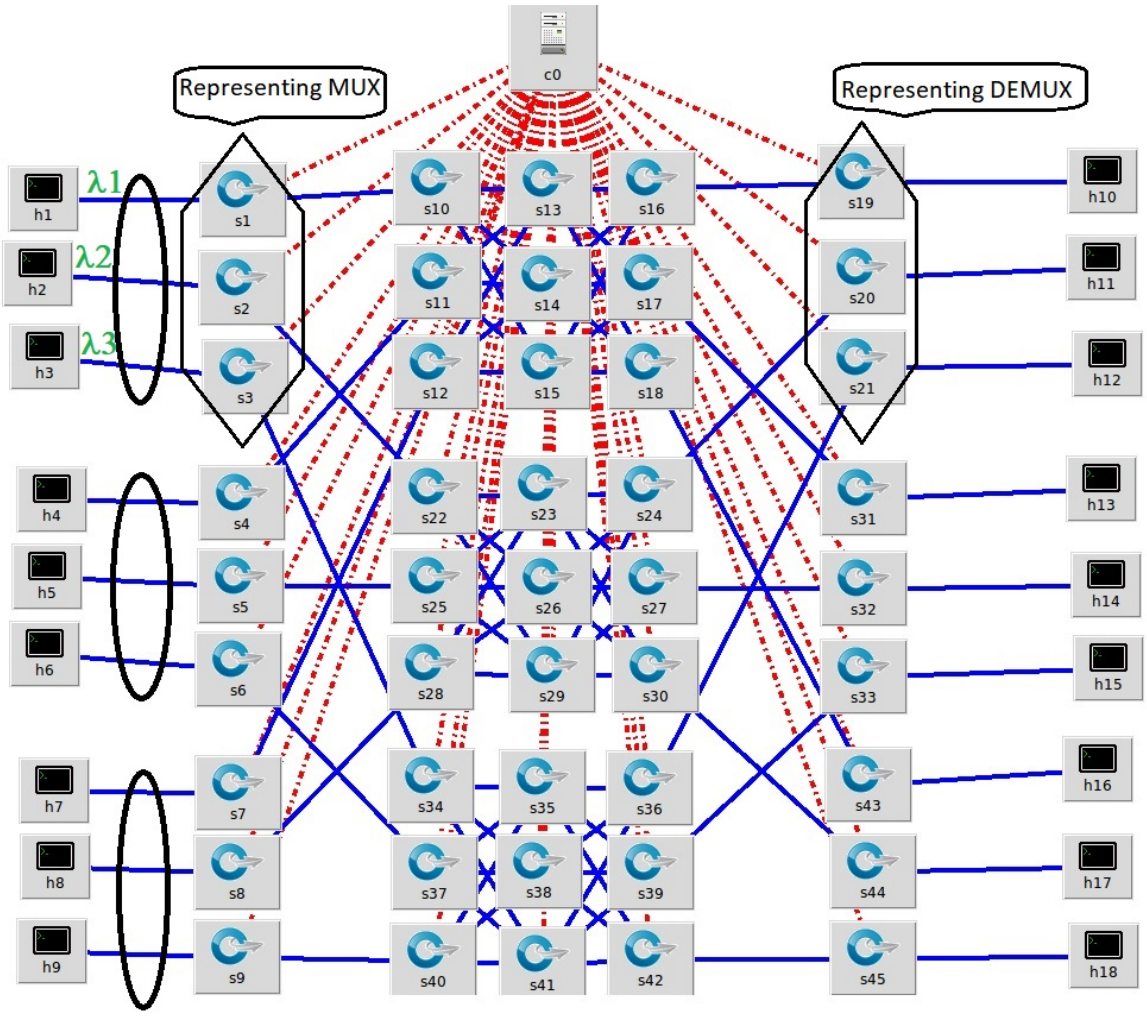


Figure 4.6.: Topology with mapped cable

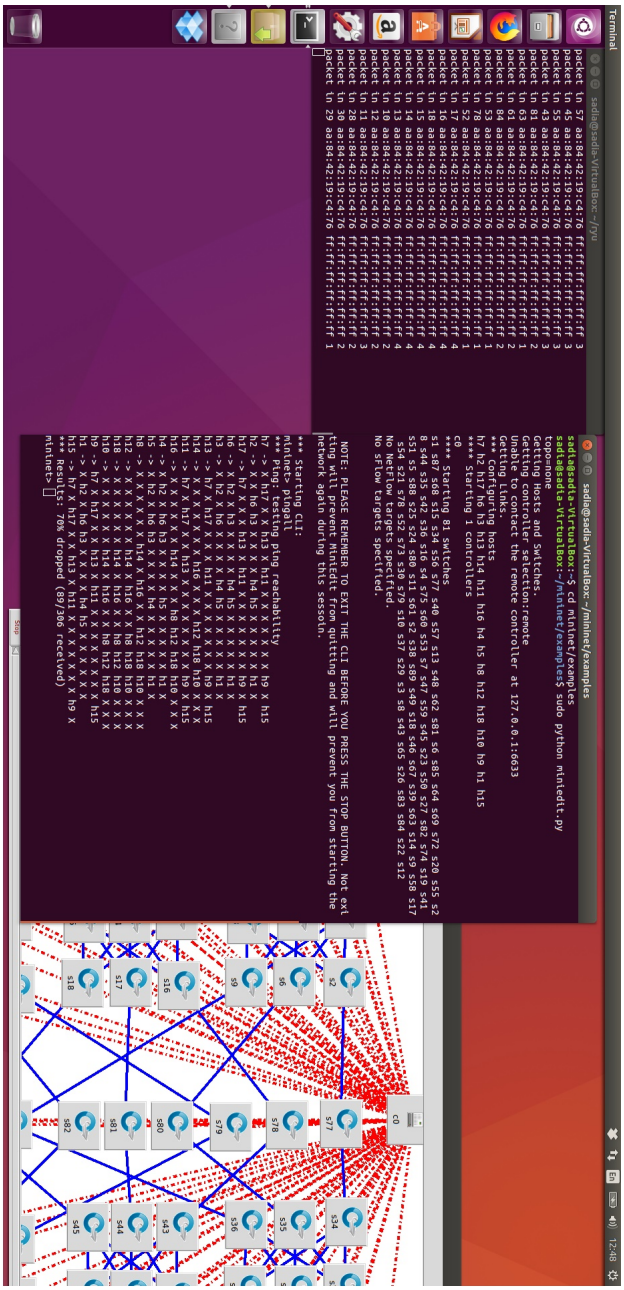


Figure 4.7.: Results for Ping command

References for ITNAC paper

[4] Hui Zang et al. “Dynamic lightpath establishment in wavelength-routed WDM networks”. In: *IEEE Communications Magazine* 39.9 (2001), pp. 100–108. ISSN: 01636804. DOI: 10.1109/35.948897.

[10] Bob Lantz, Brandon Heller, and Nick McKeown. “A network in a laptop:rapid prototyping for software-defined networks”. In: *Proceedings of the Ninth ACM SIGCOMM Workshop on Hot Topics in Networks - Hotnets '10*. New York, New York, USA: ACM Press, 2010, pp. 1–6. ISBN: 9781450304092. DOI: 10.1145/1868447.1868466. URL: http://dl.acm.org/citation.cfm?id=1868466http://portal.acm.org/citation.cfm?doid=1868447.1868466.

[15] Sadia Qureshi and Robin Braun. “Dynamic Light Path Establishment In switch Fabric Using OpenFlow”. In: *2018 26th International Conference on Systems Engineering (ICSEng)*. IEEE, 2018, pp. 1–4. ISBN: 978-1-5386-7834-3. DOI: 10.1109/ICSENG.2018.8638220. URL: https://ieeexplore.ieee.org/document/8638220/.

[27] Qixiang Cheng et al. “Scalable Microring-Based Silicon Clos Switch Fabric with Switch-and-Select Stages”. In: *IEEE Journal of Selected Topics in Quantum Electronics* 25.5 (2019), pp. 1–11. ISSN: 21910359. DOI: 10.1109/JSTQE.2019.2911421.

[29] Lena Wosinska, Lars Thylen, and Roger P. Holmstrom. “Large-capacity strictly nonblocking optical cross-connects based on microelectrooptomechanical systems (MEOMS) switch matrices: Reliability performance analysis”. In: *Journal of Lightwave Technology* 19.8 (2001), pp. 1065–1075. ISSN: 07338724. DOI: 10.1109/50.939785.

[56] Diego Kreutz et al. “Software-defined networking: A comprehensive survey”. In: *Proceedings of the IEEE* 103.1 (2015), pp. 14–76. ISSN: 00189219. DOI: 10.1109/JPROC.2014.2371999.

[87] Jacob H Cox et al. “Advancing Software-Defined Networks : A Survey”. In: *IEEE Access* 5 (2017), pp. 25487–25526.

[88] Akhilesh S. Thyagaturu et al. “Software Defined Optical Networks (SDONs): A Comprehensive Survey”. In: *IEEE Communications Surveys & Tutorials* 18.4 (2016), pp. 2738–2786. ISSN: 1553-877X. DOI: 10.1109/COMST.2016.2586999. arXiv: 1511.04376. URL: http://arxiv.org/abs/1511.04376http://ieeexplore.ieee.org/document/7503119/.

[89] Imrich Chlamtac and Gadi Karmi. "Lightpath Communications: An Approach to High Bandwidth Optical WAN's". In: *IEEE Transactions on Communications* 40.7 (1992), pp. 1171–1182. ISSN: 00906778. DOI: 10.1109/26.153361.

[90] Ajmal Muhammad. "Planning and Provisioning Strategies for Optical Core Networks". PhD thesis. 2015. ISBN: 9789175191157.

[91] Georgios I. Papadimitriou, Chrisoula Papazoglou, and Andreas S. Pomportsis. "Optical switching: Switch fabrics, techniques, and architectures". In: *Journal of Lightwave Technology* 21.2 (2003), pp. 384–405. ISSN: 07338724. DOI: 10.1109/JLT.2003.808766.

[92] Haitham S. Hamza and Jitender S. Deogun. "WDM optical interconnects: A balanced design approach". In: *IEEE/ACM Transactions on Networking* 15.6 (2007), pp. 1565–1578. ISSN: 10636692. DOI: 10.1109/TNET.2007.908151.

[93] Benjamin G. Lee and Nicolas Dupuis. "Silicon Photonic Switch Fabrics: Technology and Architecture". In: *Journal of Lightwave Technology* 37.1 (2019), pp. 6–20. ISSN: 07338724. DOI: 10.1109/JLT.2018.2876828.

[94] Wojciech Kabacinski. *NonBlocking Eectronic and photonic switching fabrics*. Springer, 2005. ISBN: 9780387254319.

[95] Francesco Testa and Lorenzo Pavesi. "Optical switching in next generation data centers". In: *Optical Switching in Next Generation Data Centers* (2017), pp. 1–336. DOI: 10.1007/978-3-319-61052-8.

[96] D. Fey et al. "Optical multiplexing techniques for photonic Clos networks in High Performance Computing Architectures". In: *Journal of Supercomputing* 62.2 (2012), pp. 620–632. ISSN: 09208542. DOI: 10.1007/s11227-010-0496-x.

[97] Konstantinos Christodoulopoulos, Konstantinos Manousakis, and Emmanouel Varvarigos. "Offline Routing and Wavelength Assignment in Transparent WDM Networks". In: *IEEE/ACM Transactions on Networking* 18.5 (2010), pp. 1557–1570. ISSN: 1063-6692. DOI: 10.1109/TNET.2010.2044585. URL: http://ieeexplore.ieee.org/document/5458043/.

4.5. Model Implementation and Performance

A below included paper [99] explains the steps involved in the dynamic path establishment using RMS in detail. The paper starts by introducing the problem and shortcomings of various approaches. The article briefly describes the traditional path setup method and also covers why path setup in optical networks using SDN is challenging. Some major advantages of using SDN in optical networks are explained. Related work is also covered in detail with their pros and cons. The paper first represents the overview of the proposed design and then explains individual components.

For this implementation, OpenFlow protocol versions 1.0 and 1.3 have been used; change in the protocol version does not impact the path establishment process. The Linux distribution used is Ubuntu 16.04 LTS with Mininet version 2.3.0d1.

The RYU controller has been used with the Spanning Tree Protocol (STP) application for SDN setup. STP is important because our network topology has many loops; without STP, packets can easily get lost in the network. In order to understand the working of STP, see; [100, 101]. RYU is an open-source component-based SDN controller which is particularly designed to have well defined APIs to easily create new applications to manage and control network traffic. In order to start RYU with STP application, following command has been used after getting into the "ryu"directory:

```
$PYTHONPATH= .  ./bin/ryu-manager ryu/app/simple_switch_stp_13.py
```

Ryu-manager is required to load and run Ryu applications and is the executable for Ryu applications.

For implementation of this model, following assumptions are made:

1. The optical switches know about their neighbors.
2. Extraction of destination address (MAC/IP address) is possible from the incoming optical signal.
3. The initial data loss till path is established is not above threshold and does not effect the user.

The paper also analyses the RMS in terms of its performance. Results are presented in the attached article. For more extensive and complex topologies, more resources are utilized. A "top" command is used to measure CPU utilization, while a "free" command is used to check memory. Because of more processing required for more extensive topologies, the time to establish a path, i.e., latency, increases. Topology creation time is also estimated using the "\time" command. Graphs and exact figures are shown in the article. Graphs are plotted with gnu plotting, and bash scripting is used to save the data in the DAT file format automatically, e.g.:

```
iteration=150

top -b -d -n $iteration grep | "cpu(s)" awk '{print $8}' |ts '%H:%M:%.S'
>> cpu.dat
```

The published paper

Below is the paper published in IEEE Access Journal; `https://ieeexplore.ieee.org/document/9` November 2021. Title of the paper is copied for section 4.6, [99] and all the subsections have same titles as in the original publication. The original PDF proof of the publication can be found in the Appendix.

4.6. Dynamic LightPath Allocation in WDM Networks Using an SDN Controller

4.6.1. Abstract

Core wavelength division multiplexed (WDM) networks are widely used to provide fixed physical connectivity and bandwidth to the logically connected upper electronic layer devices using optical signals. However, growing demands for bandwidth-intensive applications and cloud-based services push optical networks carriers' to provide scalable and flexible services dynamically. Software defined networking (SDN) has the potential to program electronic layers by dynamically controlling and managing network resources using SDN controller applications. SDN's on-demand

characteristics combined with the optical circuit switching can enable optical network service providers to customize their service provisioning dynamically to the user's requirements. They enable fast provision of new services, and minimize underutilization of resources. In this paper, a model is proposed to bring the dynamic allocation of resources which is a layer 2+ functionality, to the WDM layer using SDN. A middle-ware application based on SDN and OpenFlow for dynamic switching and provisioning of optical service is presented. The application abstracts the optical layer's connectivity, also accounting for the switching constraints. Details of the model's implementation are discussed considering classically used equipment and its performance in terms of CPU and memory utilization, topology emulation time, and latency is evaluated. Finally, the application is tested with a Cisco layer one switch. Performance results show that the latency doubles when increasing the number of fibers of an optical cross connect from 5 to 7 and keeping wavelengths equal to 8, with Clos fabric topology.

Index Terms

Wavelength division multiplexed network (WDM), Software Defined Networking (SDN), OpenFlow protocol, dynamic path allocation, optical cross connect (OXC).

4.6.2. Introduction

The traditional purpose of the circuit-switched long haul or core wavelength division multiplexed (WDM) networks is to provide high bandwidth physical network connectivity to the upper packet-switched electronic layers (e.g., Ethernet, IP). The end-to-end physical path between these packet-switched devices (routers/switches) is pre-planned to provide static wavelength routes through the optical devices. Preplanning is done using a tool called Network Management System (NMS), and the optical devices are then manually configured to reserve paths and resources according to that computed plan. Since resources are pre-allocated, it is almost impossible to change them dynamically to accommodate the instantaneous demands of the user. Such networks are thus difficult to automate. In WDM networks, although paths are pre-computed, a connection may still not be available at times due to all paths within the optical switch's internal fabric being busy. WDM devices have their own control system that uses complicated algorithms to find an available path

through the inter-connections in the fabric. Since, setting up of the path through the switch's fabric is related to the path establishment between end-to-end communication devices [70], a single path computation entity is required which has global information of network paths and all possible inter-connections. Moreover, the conventional NMS poses problems for path planning in multi-carrier and multivendor environments. Thus, need arises for a flexible control mechanism that may compute and provide the path resources dynamically when requested.

Recently emerged, software-defined networking (SDN) architecture and OpenFlow (OF) protocol [55, 56], work in an on-demand fashion, and hence has the inherent potential to resolve above mentioned issues. However, SDN has been designed for packet-switched networks while WDM networks are circuit-switched, further explained in subsection 4.6.3. The idea of extending SDN to optical networks has been discussed in previous works [41, 75, 72, 102, 73]. However, nothing could be standardized yet because of the complex nature of the physical layer issues, starting from accounting for the physical impairments to the vendor-specific configuration mechanisms for optical paths. A significant advantage of extending SDN to photonic networks is the combined dynamic service provisioning and optimization by the SDN controller because of its global knowledge of several network layers. A recent work [52], provides an SDN-controlled platform to simulate end-to-end integrated data plane of packet and optical network. The physical behavior of a WDM network is simulated using analytical models and an optical data plane is emulated as data packets with additional information about wavelength. The platform can be used to configure network elements in a customised manner; however, switching constraints are not considered in this model. More advantages of using an SDN controller in optical networks are explained in subsection 4.6.4.

In literature, there have been some efforts that deal with establishing the wavelength paths through the WDM layer using SDN, e.g., [44, 42, 76]. However, all of these have assumed fully interconnected switch fabrics and the switching constraints are over-sighted. If the switch's internal fabric is busy and there is no available path through the fabric, then user's service provision request cannot be completed dynamically. Furthermore, none of the works have exploited the core potential of the OF protocol, i.e., dynamic resource allocation in response of a request, by generating the path request initiation from the WDM device itself. Either an IP router in the upper electronic layer is used or an agent-base approach is applied for request initiation purpose, further details of which are discussed in subsection 4.6.5. Agent-

based approach requires both the controller and the OF protocol to be extended. Our work is an effort to design a middle-ware mechanism that can be used with any SDN controller and OF protocol versions without extensions, while taking into account the switch's internal limitations and which dynamically accomplishes the service request and allocation of the resources.

In this work, we propose a model in which a single SDN controller can have a global view of all the possible paths within the switch together with the end-to-end links between the devices. We use a popular emulator namely "Mininet", further discussed in subsection 4.6.6. The SDN controller can have knowledge of all the available paths and can dynamically provision a connection when a request arrives. Our work is based on emulating the network connectivity to the SDN controller as a network of layer two OpenFlow switches, using Mininet. The controller thus does not have any knowledge of the underlying optical switches nor the wavelengths. Therefore no extensions of the SDN controller as well as OpenFlow protocol are required. A middle-ware application is developed to do all the translations between the controller and the optical network. Details of the model's implementation and tests are provided in subsection 4.6.7 and subsection 4.6.8 respectively. creation and mapping of layer one connectivity to the controller is described in subsection 4.6.9. Performance results are reported in subsection 4.6.10 while subsection 4.6.11 provides discussion about the graphs. The paper summarizes with conclusions and future work in subsection 4.6.12.

4.6.3. Background

SDN's potential in the packet domain has already been proven, initially by the researchers and now by the fact that some major companies are providing services and products based on SDN architecture and technology. For optical networks, SDN has been suggested to be advantageous over existing control plane solutions in many aspects, see e.g., [103, 104, 105]. There have been efforts to extend SDN to optical networks; however, for multilayer convergence and control, layer one (electronic switching) as well as layer zero, i.e., wavelength switching layer should also be included in the SDN umbrella. Extending SDN to WDM layer has several challenges; though, from the physical characteristics of the communication signals to how the resources are allocated in vendors' proprietary equipment.

4.6.3.1. Packet and Circuit switched Networks

There is an essential and fundamental difference between circuit-switched and packet-switched networks, i.e., a packet-switched network do not pre-allocate resources before the first data flows. Instead, it allocates resources on-demand as datagrams arrive. On the other hand, circuit-switched systems pre-allocate resources before the first data flows. Those resources can then only be used for that instance of data flow. Optical transport networks are circuit-switched networks, which use light signals for faster delivery of packets from the source to the destination ports of layer two or layer 3 devices as shown in Figure 4.8. LightPaths are pre-computed and established by the management system and provisioning a new optical connection involves operator intervention. These mechanisms allocate the entire wavelength to a single connection, and the connections remain 'on' even when they are not utilized fully.

4.6.3.2. Path Establishment in Optical WDM networks

Todays' commercial long-haul or core optical transport networks are implemented in a mesh topology with path planning and management done through a centralized network management system (NMS). NMS employs multiple element management systems (EMSs) to manage the whole network [19]. Usually, a single vendor's network elements are managed by a separate EMS. The EMS can have a view of only its attached network devices and does not have the overall network view. Therefore, for the management of whole network, all EMSs, are supposed to connect and report to the NMS. The NMS then manages all types of network elements, see Figure 4.8. Additionally, a simple device management system is needed to enable the craftspeople and other technical staff to configure and manage individual network equipment.

Two main components of a WDM network are re-configurable optical add/drop multiplexers (ROADMs), and optical cross-connects (OXCs). ROADM is a wavelength switching device which is required when some of the wavelengths need to be dropped or added locally in the network while others need to pass through to their destinations. OXCs are designed to perform almost same tasks as ROADMs but they have larger number of ports and wavelengths involved and are deployed in core optical networks. The OXC's core is its switch fabric. The switch fabric is a

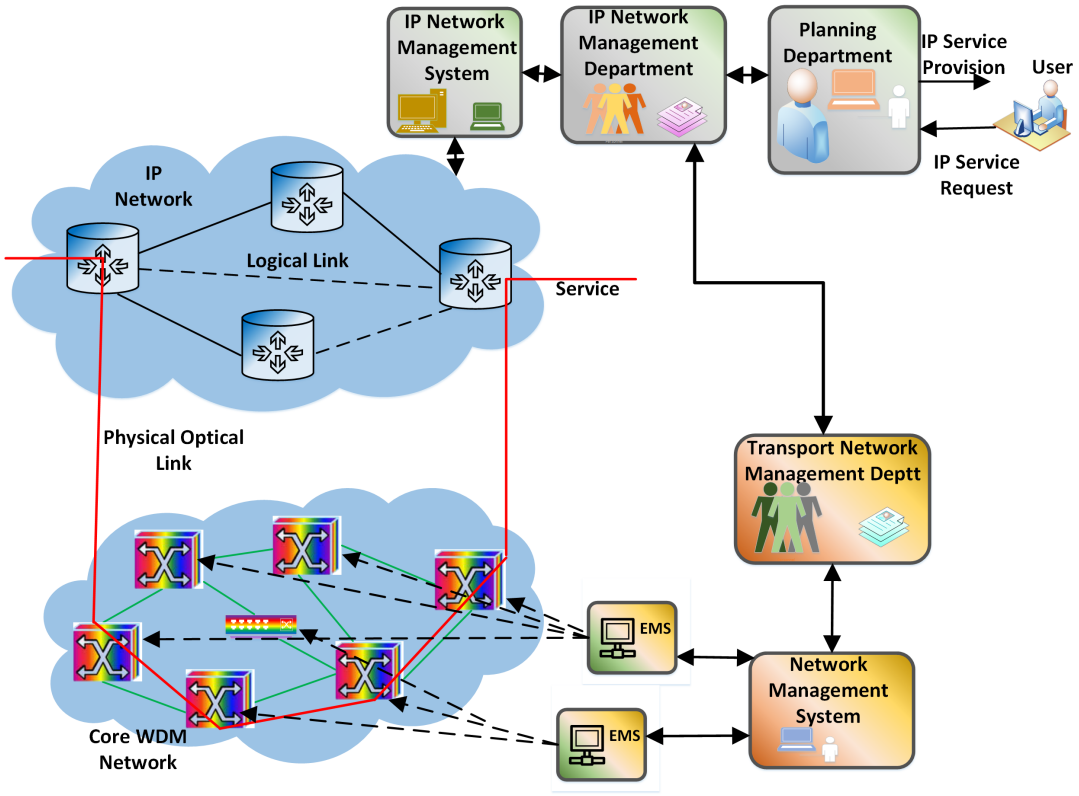


Figure 4.8.: Overview of IP and optical layer management system.

combination of switching cells which are arranged in the form of a either a crossbar matrix or multistage interconnections (MIN). The fabric usually exists in one of the popular form of MIN topologies, e.g., Clos, Benes, Banyan, or Tree. When a new connection is to be made, the local switch controller has to find an available connecting path in the switch fabric and issue respective electronic control signals to set up switching cells. Algorithms performing such tasks are reffered as control algorithms [19, 106]. The circuit-switched connection between WDM networks' client terminals is called LightPath and the process of finding the path and assigning a wavelength is designated as Routing and Wavelength Assignment (RWA) problem [107, 108]. Connection management tasks, e.g., setting/releasing and monitoring LightPath connections in a network is done by NMS and EMS.

For a new service provision, the user needs to request to the "Sales and Planning" department of the IP network. The Sales department puts up a request to the management department, which collaborates with the management of transport network and works out the feasibility of service provision, see Figure 4.8. As a result, it may take several months for a service provider to supply a new service

in response to a user's new connection request. The span of these connections is generally for years. With the evolution of optical networks, connection demands are becoming more dynamic and networks are turning more extensive and complex. Carriers prefer to provide connections to their customers quickly and not hold them for a long time. The current mechanism of LightPath management has proven to be very complex and slow. It involves human labor, time, and is error-prone [102]. To a certain extent, dynamic connection provisioning is achieved by the wavelength switching and routing at optical switches; however, more research, technology and new methods, are required in this domain.

4.6.3.3. SDN and Mininet

SDN paradigm separates the control plane from the forwarding plane of a traditional network and brings the intelligence in a logically centralized controller that keeps the global network information and does strategic changes in routing and switching devices dynamically, based on different requirements, e.g., quality of service (QoS) or traffic engineering. The OpenFlow protocol, first introduced by N. McKeown et al. implements SDN and is a communication standard between the controller and the forwarding devices. For testing SDN solutions, an emulator called "Mininet" is mostly used which is a rapid prototyping environment that offers lightweight virtualization and flexibility. For further understanding, the working, benefits, limitations, and features of SDN, OpenFlow and Mininet, the reader is referred to [63, 57, 109].

4.6.4. Advantages

Employing SDN controller for dynamic control of optical networks brings many advantages with it, some of which are discussed below.

4.6.4.1. Multilayer convergence

SDN has already shown its benefits in layers 2-4. A considerable benefit of encompassing optical networks with SDN will be the combined dynamic provision of

services as well as optimization of multiple network layers. A central SDN controller based on its global view of all the layers can play a role in optimal resource allocation. If a service needs on-demand connectivity, the SDN controller can calculate an optimum path through multiple layers and can dynamically establish the connection, see e.g., [43].

4.6.4.2. Dynamic customized provision of services

Provisioning of dynamic on-demand bandwidth pipes between end-to-end systems can benefit economically and improve network performance by dynamically giving preference to the customer's preferred service(s), based on instantaneous network status, e.g., see a use-case by [75].

4.6.4.3. Traffic engineering

In SDN, the controller has a bird's-eye view of the network's resources and can improve its performance by providing effective traffic engineering solutions. The SDN controller can choose paths in order to balance the load on various links and devices in the network where multiple alternate paths are available, see examples in [110, 111].

4.6.4.4. Optimum path calculation

Centralized path computation has always been known to be more effective when compared to the traditional distributed path computation. Combining this advantage of SDN with others, e.g. taking into account the physical layer impairments, can improve end-to-end optimum path calculation, see e.g., [112]. Layer one transport networks also face another major issue of RWA. RWA can be handled better by incorporating contemporary RWA algorithms within path computation engine of SDN controller, see e.g., [113].

4.6.4.5. Control of Elastic Optical Networks(EONs)

To meet the highly dynamic traffic demands of the future, an elastic optical network has been regarded as a promising paradigm, which can offer a flexible data rate/channel allocation and high resource efficiency. The main purpose of EONs is to replace the fixed spectrum spacing with a flexible grid where an optical channel can span a smaller grid size for low data rates, whereas multiple slots to accommodate higher bit rates, e.g., 400 Gbps, thus supporting various data rates dynamically in a spectrum-efficient manner [114]. Technology advances associated with EON nodes have been identified as flexible grid add/drop and switching (essentially Bandwidth Variable Transponders and flexible OXCs), as well as spectrum conversion. To harness such a flexible network, a programmable control plane technique is required to program network functions dynamically according to the applications. However, this isn't easy to implement, as the control and data planes are currently integrated within the network nodes. Newly emerging software-defined optical networking (SDON) paradigm has a separated programmable control and data plane (DP) in which network functions and protocols are manageable according to the user's demand and applications [115]. The first feasibility and validation check of an OpenFlow-based control plane for an EONs has been presented in [116]. Further studies regarding software-defined elastic optical networks are reported in [49, 115].

4.6.4.6. Protection and Restoration

A well-established key feature of fixed-grid optical core networks is their protection and agile restoration when link failures occur. The critical factor that needs to be considered while planning protection schemes is a trade-off between redundant capacity and restoration time [117]. Most popular protection schemes are based on dedicated path protection; however, research studies are exploring the ways to improve the performance of these techniques for fixed-grid and EONs. Network-coding-based protection [118] is one of such proposed techniques; see for example [117]. With the paradigm shift to SDON, the protection and restoration tasks are performed by a centralized controller, which can perform optimum computations based on the global map of the network. The reader can find references to various studies with a focus on SDN-based protection and recovery in [119].

4.6.4.7. Opportunity for new business models

SDN's programmability and flexibility have opened up opportunities for the network carriers' to try new business models. Network operators can investigate, test, and implement new models using the controller to optimize optical network's cost and performance, e.g., better bandwidth provision or energy effectiveness, see examples in [120, 121].

4.6.5. Relevant Work

SDONs are being investigated and studied at the industrial as well as academic level. However, these investigations have varied scales and aim for different objectives. A good survey of previous research efforts is available in [72, 102, 122, 79, 80]. In this section, we will discuss the works that are explicitly done at the WDM layer.

In [42], the authors propose to use IP router to invoke the request to the SDN controller for path establishment. The work proposes OpenFlow-enabled optical switches which can understand commands from the extended-NOX controller. The optical switch interfaces are mapped using virtual Ethernet interfaces 'veths' of an OF switch; however, internal switching constraints are invisible to the controller. Also, the optical switch is incapable of generating the request itself, and thus full potential of the dynamicity of the OpenFlow protocol is not exploited.

References [44, 104] have used an agent-based approach and developed an extended controller with an application to compute LightPaths. It is not clear how the optical devices' constraints are considered by the resource model and their way of abstracting the devices' hardware requires them to use an extended OpenFlow protocol and an extended controller. Reference [76] uses the same agent-based concept and utilises a Path Computation Engine (PCE) and a Wavelength Assignment (WA) algorithm in the controller for computing LightPaths. The authors use a port emulation entity which maintains and receives updates about port status and is used to calculate LightPath but no data is given for the switch fabric's limitations. They have also used an extended OF protocol.

Reference [123] uses a proxy application to control a WDM network device with the SDN controller. The authors have considered an end-to-end connection between

devices as an optical link and used Simple Network Management Protocol (SNMP) to get the features of the switch; hence, the internal fabric connectivity is hidden from the controller. The work is conducted for integrating ROADMS, much simpler than OXCs, which are the main operating devices in the optical core networks.

The Open Networking Foundation Transport Working Group (ONFTWG) is a team, working towards extending SDN and OpenFlow protocol for transport networks. Some initial efforts are being made by ONFTWG as well as other emerging organizations for standardization. In [83, 82], authors have presented an ONOS control method using a dis-aggregated transport network.

4.6.6. Proposed Design

The idea of parameterizing the WDM layer as a layer two system which allocates wavelength paths in an on-demand fashion, is presented in this paper. The proposed framework demonstrates a way to abstract layer one network connectivity so that it can be used in a layer two paradigm. The abstraction also includes the internal connectivity of the switch fabrics of OXCs and ROADMs and hence incorporating switching constraints as discussed in [102]. Switching constraints are imposed by the topology of the switch fabric, which is often chosen to reduce the cost and may result in congestion and hence non availability of the path through the fabric. This leads to declining a service provision.

Our model uses a central SDN controller and the emulated connectivity of layer one to control the end-to-end network connections. Incorporating the connectivity of switch fabrics helps the SDN controller to make "Network Graph". OXCs use different control algorithms to find paths through their switch fabrics. Algorithm for one fabric, may not work for others. Abstracting the connectivity eliminates the need for different equipment-specific algorithms, and one algorithm can be used to find end-to-end path in different classes of equipment.

In WDM optical networks, another limitation is "Wavelength Continuity Constraint" (WCC), which is catered in a component of our model called Resource Management System (RMS). The RMS component serves as a portal or application that maintains and manages virtual abstraction of paths of an optical network, so that customers are served dynamically as per their needs. With dynamic path establishment using

SDN controller, the requirement for the always-on connections can be eliminated thus avoiding wastage of resources, when not being used. Our way of abstraction does not need to use any extended controller nor any extensions to the OF protocol.

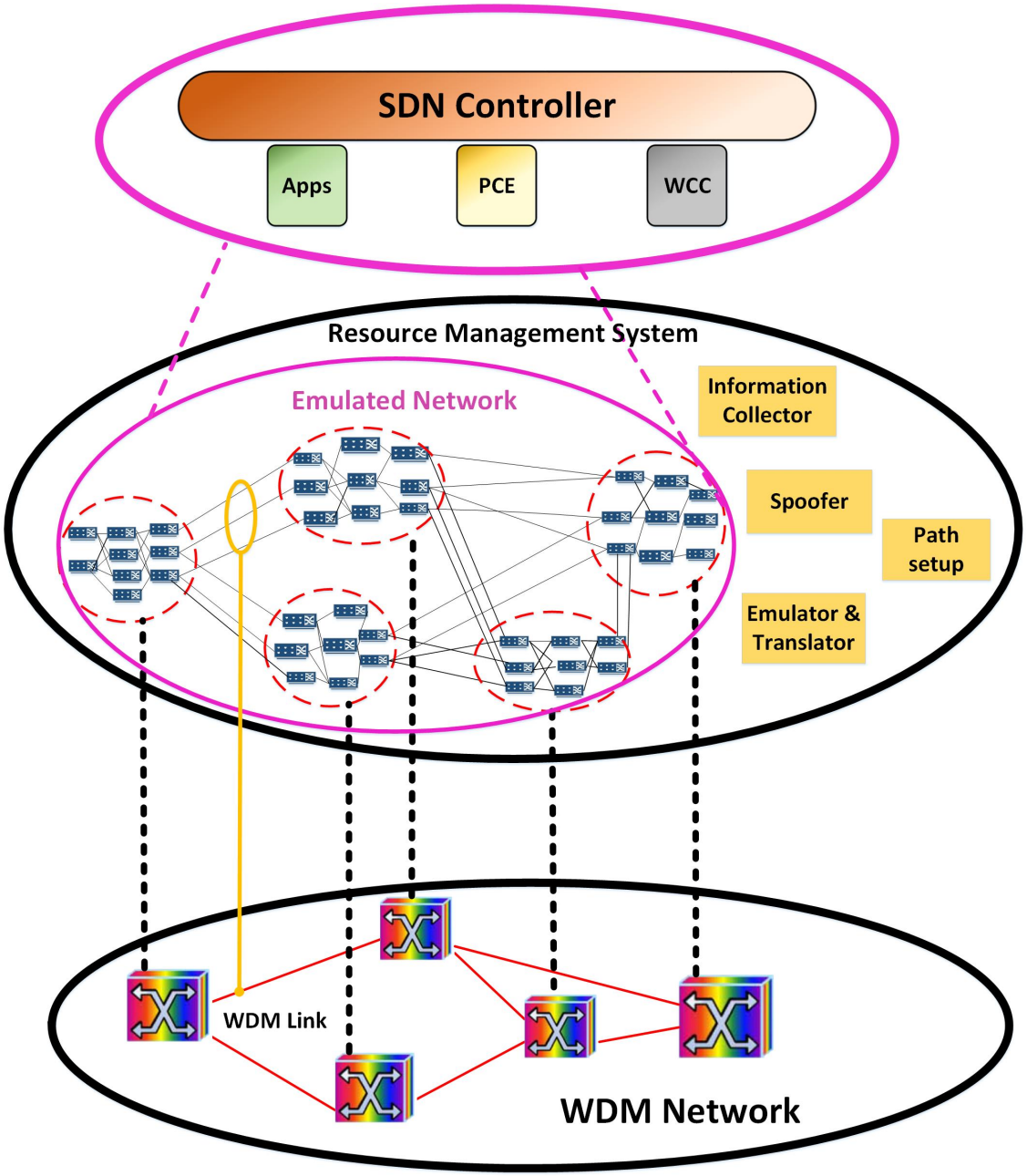


Figure 4.9.: Proposed model overview.

Figure 4.9 represents the overview of the proposed design. RMS is used to parameterize the layer one optical network to the SDN controller, as a network of layer

two OF switches. The WDM link is emulated as a combination of Ethernet-linked interfaces, each representing an individual wavelength. The controller is used to find a path from source to destination through this network. The RMS uses the path information in the emulated network and sends commands to Layer one devices to set up the physical path through the optical network. The proposed model assumes the invocation of path-requests from the optical switch itself. The model also assumes that the optical device can extract destination address from the incoming signals. The RMS can also include a "scheduler" component to save future requests. Details of the components of the proposed model are as below.

4.6.6.1. Resource Management System (RMS)

The RMS is responsible for performing multiple tasks and acts as a middle-ware between the SDN controller and optical network. It has many components and performs actions like collecting information from the optical switches, abstracting layer one connectivity, translating path found by the SDN controller, and wavelength availability check. It also maintains a table to store the status (idle/busy) of the ports and check their availability every time the path is requested.

4.6.6.2. SDN Controller

This is any conventional SDN controller with path computation ability. It can be loaded with customized applications for advanced control of this system, such as a few mentioned in the subsection 4.6.4.

The proposed design achieves the goal of dynamic path establishment in few steps. The first step involves "Network Graph" creation. This step can be challenging as there is no network connectivity initially (within switches), and resources will be allocated once the traffic flow starts. However, if all the possible paths are abstracted, see, e.g., [15], the SDN controller is able to make a network graph. Thus the SDN controller can have a global view of the layer one network, emulated as a layer two network. The second step follows when a request signal arrives at an optical switch. The relevant information of this request is sent to the RMS by the switch, which triggers the path calculation process in the RMS using the SDN

controller. The controller computes the path for the emulated network, and flows are installed in the OF switches. In the next step, this path is translated back by the RMS to a form that is understandable by the optical devices, and finally, it sends commands to physically set up the path through them. This model appears to be an agent-based approach; however, the way we implemented it (see subsection 4.6.7), is instead a middle-ware approach. Flow chart and timing diagram for this process are discussed and represented in [71].

4.6.7. Implementation

;Our RMS is in the form of a GUI application to implement the proposed design. We used the RYU controller and ran the application namely, "simpleswitch1.3" along with the "Spanning Tree Protocol" application to avoid any loops in the topology. Python and Bash scripting have been used to implement the components of the RMS. The components of this application are shown in Figure 4.10 and are explained as below.

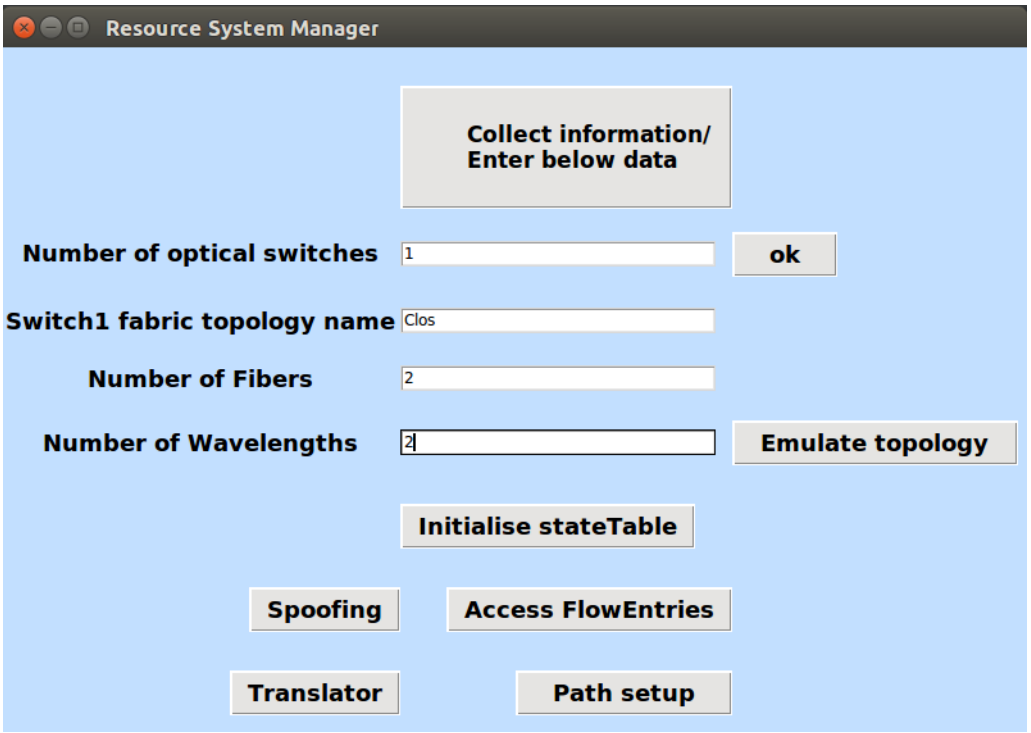


Figure 4.10.: GUI for resource management system.

4.6.7.1. Information collection component

The information collection component serves the purpose of the topology discovery. Layer one switches are required to have knowledge about their neighbors either through a protocol like Link Layer Discovery Protocol or installing static files. Every switch has an interconnection of switching cells to make input-output port connections through it, called switch fabric, common examples of which are Clos, Benes, and Banyan [19]. Information collection components can be made to use any protocol to extract switch fabric information from the switch, for example, SNMP. However; currently, switch fabrics are not managed in this way, so we have implemented it as a manual process to provide information such as, type of the switch fabric, number of fibers, number of allowed wavelengths per fiber, and a neighbors table.

4.6.7.2. Emulate network topology

In order to abstract the layer one network topology, we have used a popular emulator, called “Mininet”. The emulated network represents all layer one possible connections in the form of a layer two devices’ (OF switches) network to the SDN controller. For every possible wavelength route, a packet based Ethernet link between two OF switches is formed in an emulated topology and each port of the optical switch is represented by an individual OF switch. The component emulates Layer one connectivity using an "Emulation table" which has information for mapping of the optical paths of WDM layer in Mininet. For this work, we have chosen the “Clos” and “Crossbar” fabric topologies for demonstration; however, any other topology can be emulated. The Mininet topology is emulated as a wavelength selective architecture, further details of which are explained in [15]. Once the “Emulate topology” button is pressed, the topology is constructed in Mininet and OF switches will start registering themselves with the SDN controller.

4.6.7.3. State table initialization

The state table maintains the status of all the current, past and future connections and wavelengths assigned to them. The path found by the 'Translator' component is checked for its availability against the state table. The state table can be initialized at any stage of the simulation.

4.6.7.4. Spoofing

Once the Mininet topology is created, the SDN controller will have a global view of the network. The controller can now find a path between any source and destination port (OF switches). When the light signal arrives at any of the optical switches, the switch will send the request for path and also the destination address to the RMS. The RMS will initiate the process of spoofing a frame arrival at the OF switch, corresponding to the same optical port that received the signal. After this, a standard OF protocol procedure involving "PacketIn" and "PacketOut" messages will start.

To implement spoofing, we have used the "Ping" command. When the "Ping" process is invoked, the OF switch looks for a flow entry in its table. If no path is found, a 'Packet In' message is sent to the controller. The SDN controller finds a path between source and destination ports and installs flow entries in the OF switches. A successful ping means that there is a possible route between the source and destination ports of the optical switch (es).

4.6.7.5. Access Flow Entries

This component accesses flow entries of the OF switches after the 'Ping' process completes. It then verifies if the path is a valid path between the required source and destination ports. For this purpose it uses the topology information of the network.

4.6.7.6. Translator

Translator uses the "Emulation table" and translates the layer two network path into a layer one path. It then checks if this path is idle (using state table) and also availability of a single wavelength along this path to satisfy the WCC. If both conditions are satisfied, it can then trigger the "Path setup" component.

4.6.7.7. Path Setup

This component is responsible for sending commands to the optical switches that are understandable by them and establishing a LightPath. These commands tell the switch to make connections between specific switch cells and the ports' interfaces, thus establishing circuit switched LightPath.

Since the whole process initiates when the requests from the optical switches arrive, this is used to dynamically find the path with no need for pre-allocation of resources as in conventional WDM networks. Another worth considering aspect of the WDM networks, while moving from static to dynamic path allocation, is to take care of the protection strategy. With our framework, it is easy to add a "link failure" component in the RMS, which will store alternate paths computed by the SDN controller. Upon receiving a link failure alarm from the switch, this component can communicate the alternate route to the optical switch and restore the traffic. We left the implementation of such a feature for our future work. Here, the RMS GUI application has been designed to have buttons for performing the respective tasks. However, this is only done for demonstration purposes. The RMS only needs to receive a request from the switch and the rest of the path establishing process is automatic. If the light signal stops striking a port for a certain threshold time, the optical switch needs to inform the RMS and the path setup component will send commands to reset the switching cells of the optical switch to terminate the connection and will also update the state table.

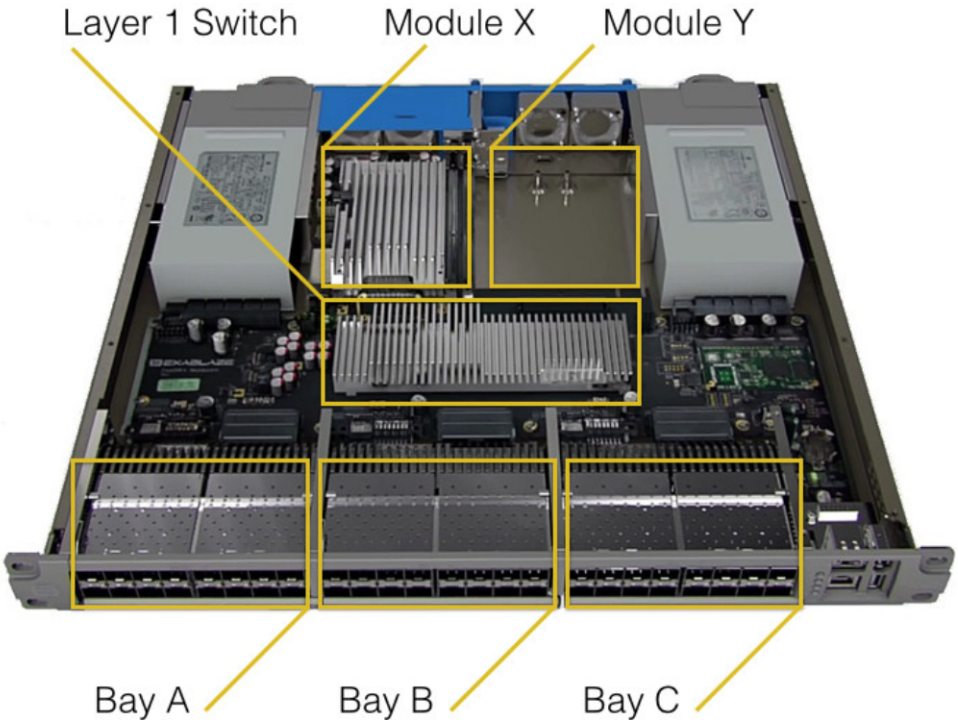


Figure 4.11.: Cisco Nexus 3550-F Fusion internal structure (Source: ExaLink Fusion documentation).

4.6.8. Testing

In order to prove the concept of the dynamic path establishment process using our model, we have tested our developed RMS with a layer one switch by Cisco namely Nexus 3550-F Fusion (formerly called ExaLINK Fusion), see Figure 4.11.

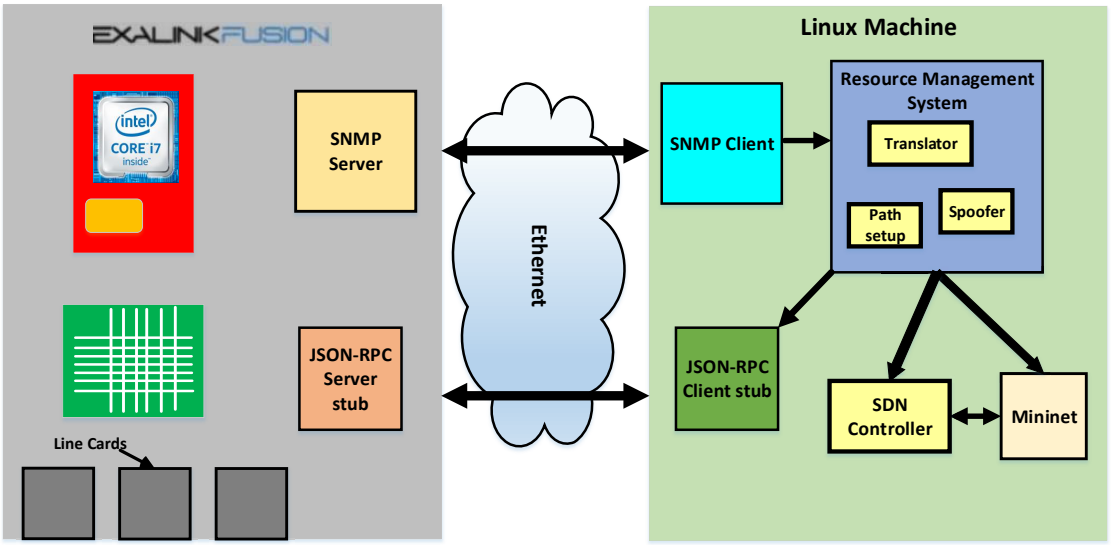


Figure 4.12.: Testing setup.

The ExaLINK Fusion has layer one switch characteristics and consists of a crossbar switch fabric that can be dynamically set up to provide a connection between any of its ports. Each of the three front panel line card bays designated as A, B, and C, have 16 inputs and outputs to the layer one switch fabric. Ports are named as A1-A16, B1-B16, and C1-C16. The switch offers SNMP and JSON remote procedure call services that we have used in our testing. Figure 4.12 represents the testing scenario. ExaLink Fusion is assumed to have one fiber having 16 wavelengths comprising of A1-A16 (input), and B1-B16 (output) crossbar fabric for this experiment. Steps followed for the testing are described below.

1. The RMS is provided with the information required for emulating a crossbar fabric topology, manually. Once the button “Emulate topology” is pressed, a cross bar topology is created in Mininet, see Figure 4.13.
2. When the signal arrives at one of the input ports of the ExaLink Fusion, it sends SNMP "trap" notification to the SNMP engine running in our Linux Machine which is integrated with the RMS, see Figure 4.14.
3. After analyzing this trap notification, the RMS uses spoofing mechanism (destination address is chosen random) to find path through the switch, see Figure 4.15.

4. When an available path is found and translated for the switch, the system uses remote procedure call (RPC) JSON interface to configure ExaLink Fusion to establish path between ports, e.g., path established between ports A1 and A3, as shown in Figure 4.16.

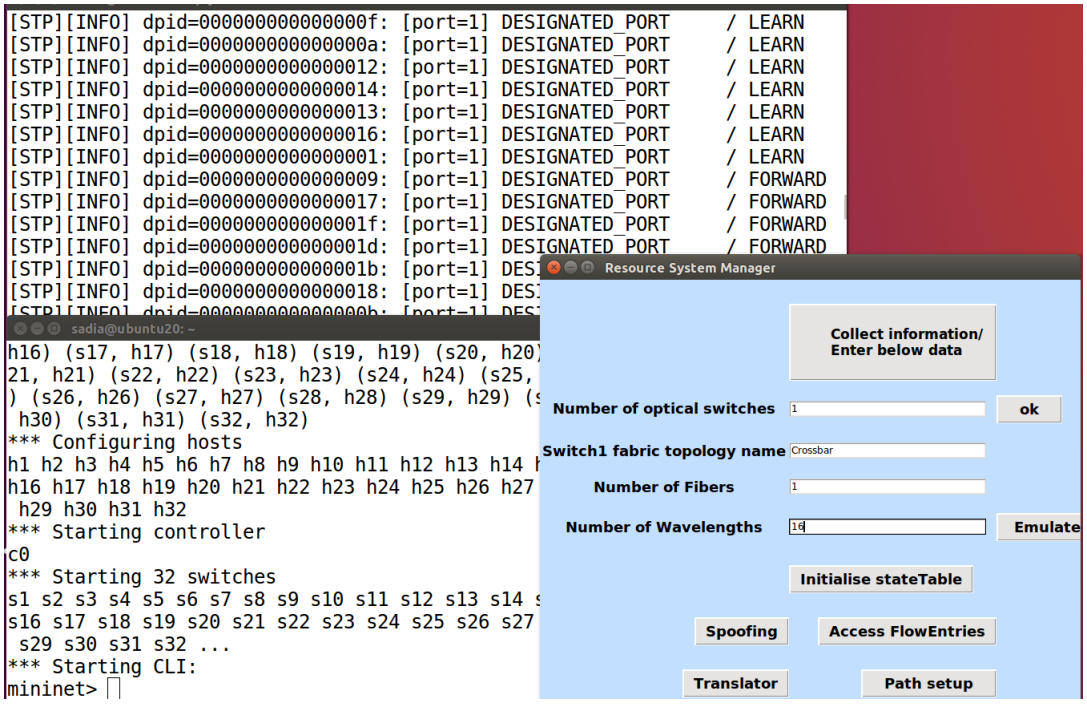


Figure 4.13.: Mininet topology generated and switches being registered with controller.

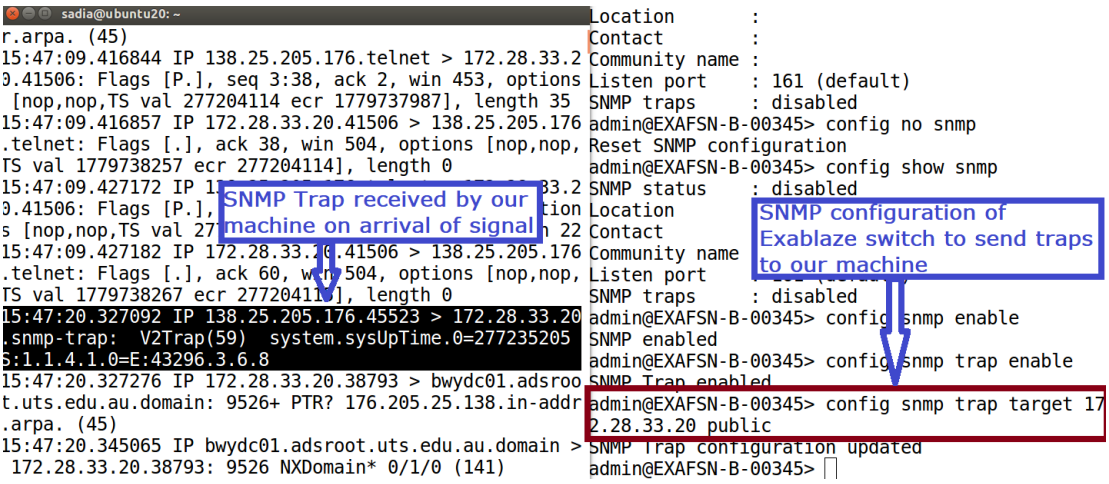


Figure 4.14.: SNMP trap received from ExaLINK Fusion on arrival of signal.

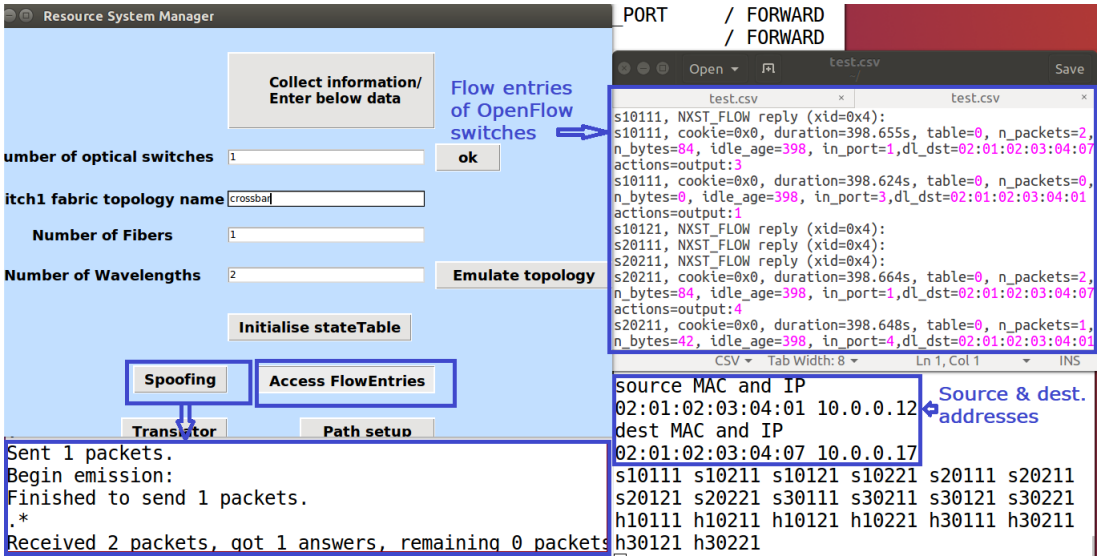


Figure 4.15.: Ping test invoked by spoofing step and flow entries in OpenFlow switches being accessed.

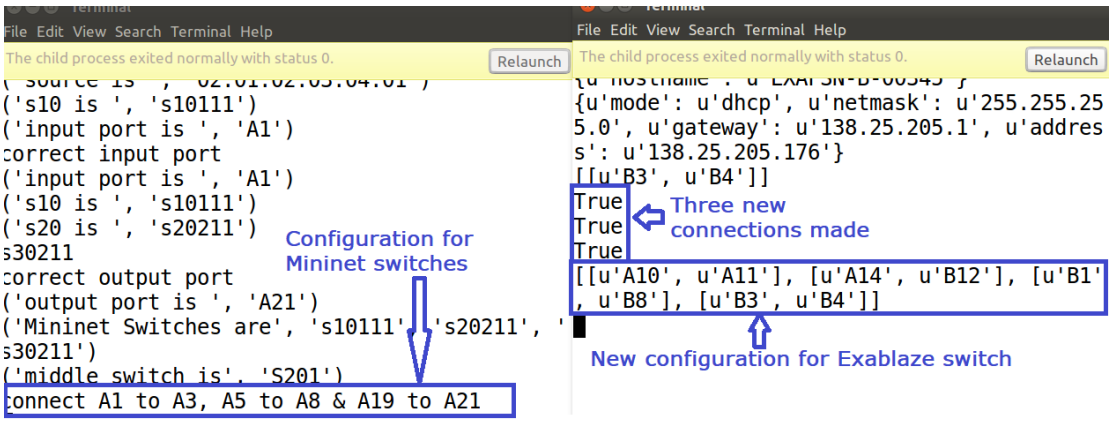


Figure 4.16.: Path found between source and host of Mininet topology and the switch is configured after the path translation.

4.6.9. Topology creation

Figure 4.10 represents the case where RMS GUI is given parameters of a "Clos" type of topology, and the number of switches/OXC in the network is one. At the same time, the number of fibers and wavelengths/fiber in that switch are also provided. Pressing the "Emulate topology" button of the RMS GUI runs a python script that takes these input parameters and uses an embedded topology creation algorithm to emulate the WDM network connectivity in Mininet. Figure 4.17 and Figure 4.18 show the cases for two fibers, two wavelengths (2F, 2W) and three fibers, two wavelengths (3F, 2W) for Clos type of OXC's switch fabric, respectively. To see the case for (3F, 3W) and understand how this topology maps the actual optical switch fabric, the reader is referred to [15].

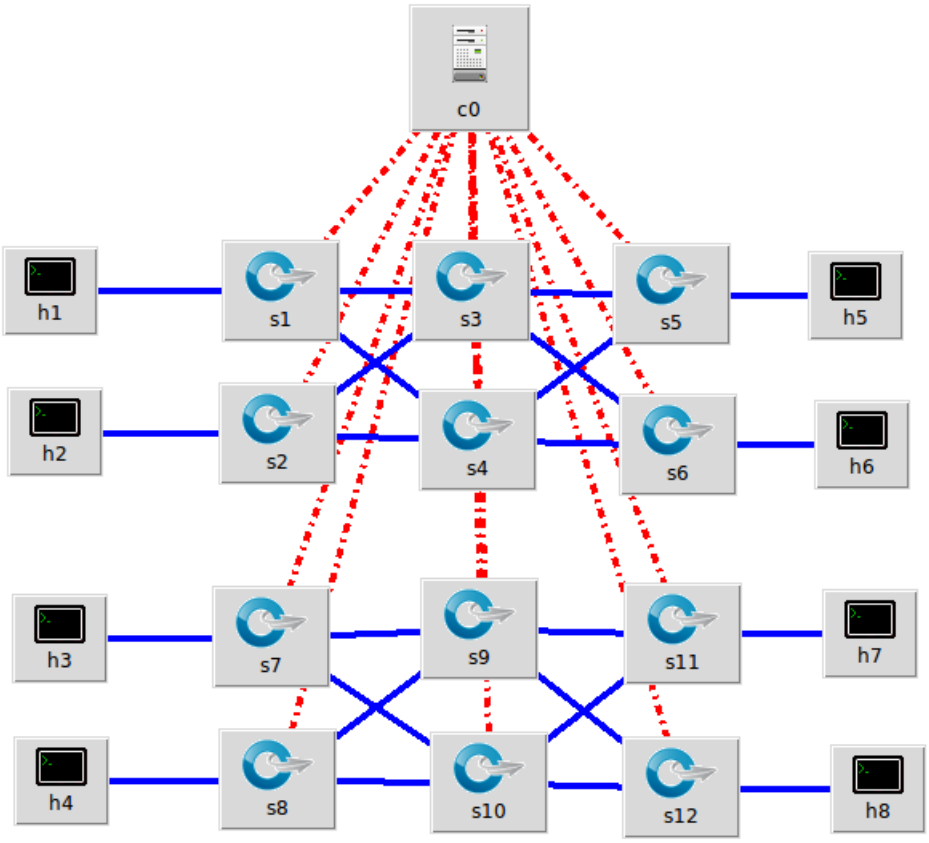


Figure 4.17.: Mapped Mininet topology for Clos switch fabric for two fibers and two wavelengths.

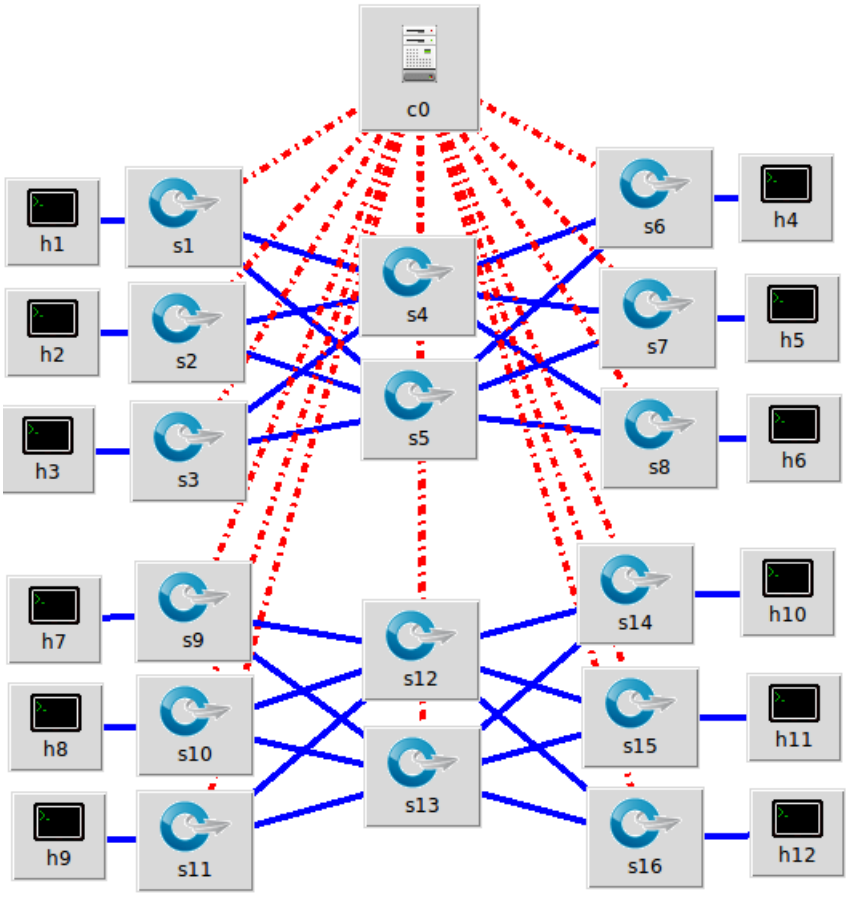


Figure 4.18.: Mapped Mininet topology for Clos switch fabric for three fibers and two wavelengths.

To find out the effect of the number of wavelengths and fibers on Mininet topology creation, we analyzed the "Generate Mininet topology" algorithm [15]. For example, we noticed that to map the wavelength-selective architecture of Clos fabric in Mininet; we need twice the number of hosts for a single increase in the number of (F) or (W), hence the equation Equation 4.6.8. We noticed that three entities of a Mininet topology, i.e., OpenFlow switches (S), hosts (H), and links (L) depend on two factors namely fibers (F) and wavelengths/fiber (W), see Equation 4.6.8, Equation 4.6.8, and Equation 4.6.8. Figure 4.19 represents the trend for these entities with the increase in the number of wavelengths (F is kept constant) for a Clos type of optical switch fabric. It is observed that the effect of varying F is not as prominent as with W, as apparent from the equations as well, thus, here we are only showing the trend with the number of wavelengths while keeping the number of F constant (e.g., five).

$$S = W^2 + 2F.W \tag{4.1}$$

$$L = 2F.W + F.W^2 \tag{4.2}$$

$$H = 2F.W \tag{4.3}$$

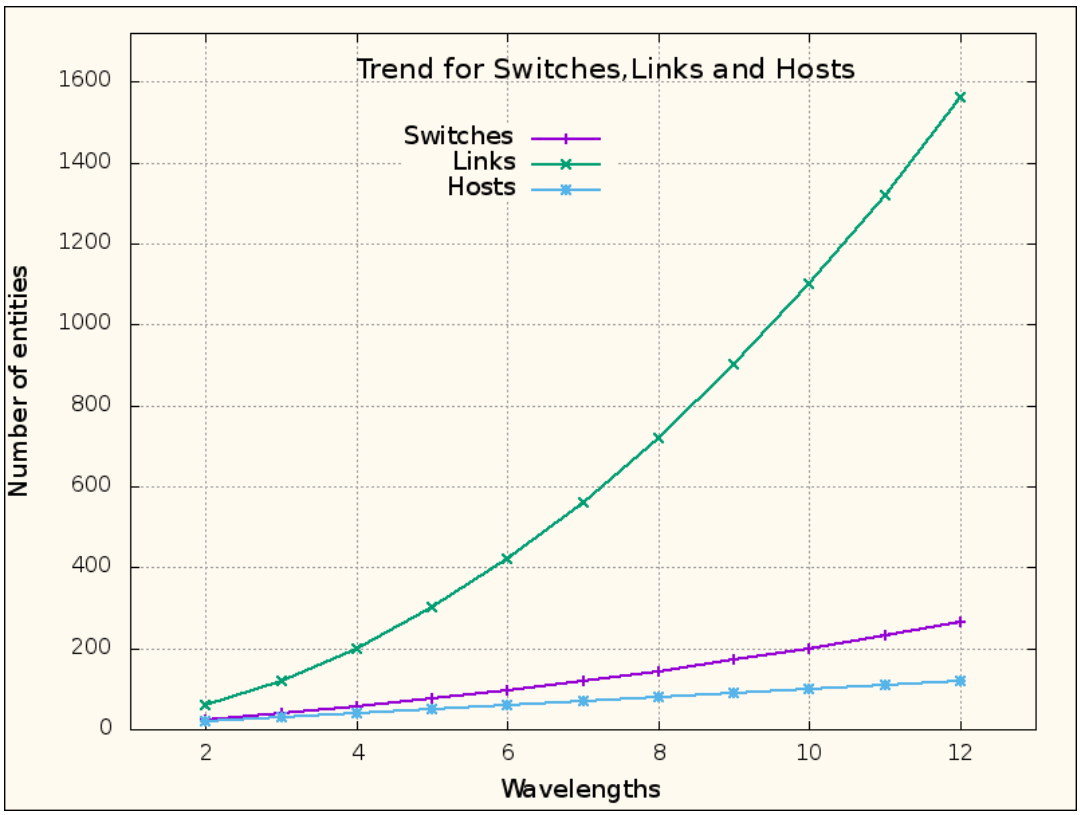


Figure 4.19.: Trend for number of switches, hosts and links for Clos fabric topology in Mininet with wavelengths.

It can be seen from the graph that the increase in the number of links is huge with the increasing number of wavelengths. For example, increasing wavelengths from 4 to 6, increases the number of links from 200 to 410, this is because every possible path for a wavelength in an optical network is represented as an individual Ethernet link in the Mininet emulation, see equation 4.2. Similar equations can be found for emulating other types of fabrics, and their effect can be observed. Here, it is important to consider the effects of W and F because they help understand the performance parameters' trends. A higher number of F and W means bigger and complex Mininet topologies.

4.6.10. Performance Results and Analysis

The emulated Mininet topology gets more complex with the complexity of the OXC's fabric topology and with the increase in the number of the wavelengths and fibers. We observed that more the fabric structure is complex, more the computational

resources are required by our designed system. We therefore did an experimental analysis of some main computational resource factors, e.g., memory, CPU usage, and time required for creating the Mininet topology. For this analysis we chose "Clos" switch fabric topology as an example.

Our system specifications are Intel Core i7-7600 CPU 2.80GHz & 2.90GHz with 16GB installed memory.

4.6.10.1. CPU Utilization

Figure 4.20 shows that the percentage of the idle CPU (mean values) decreases very quickly as soon as the topology starts getting created. As long as the topology is being created the CPU is utilized almost 100% and then comes back to become almost free. Smaller the number of F and W, quicker is the restored CPU usage, this is clearly because of less number of OpenFlow switches, links and hosts and thus less required processing. Same trend is followed when the topology is destroyed (not shown in the graph).

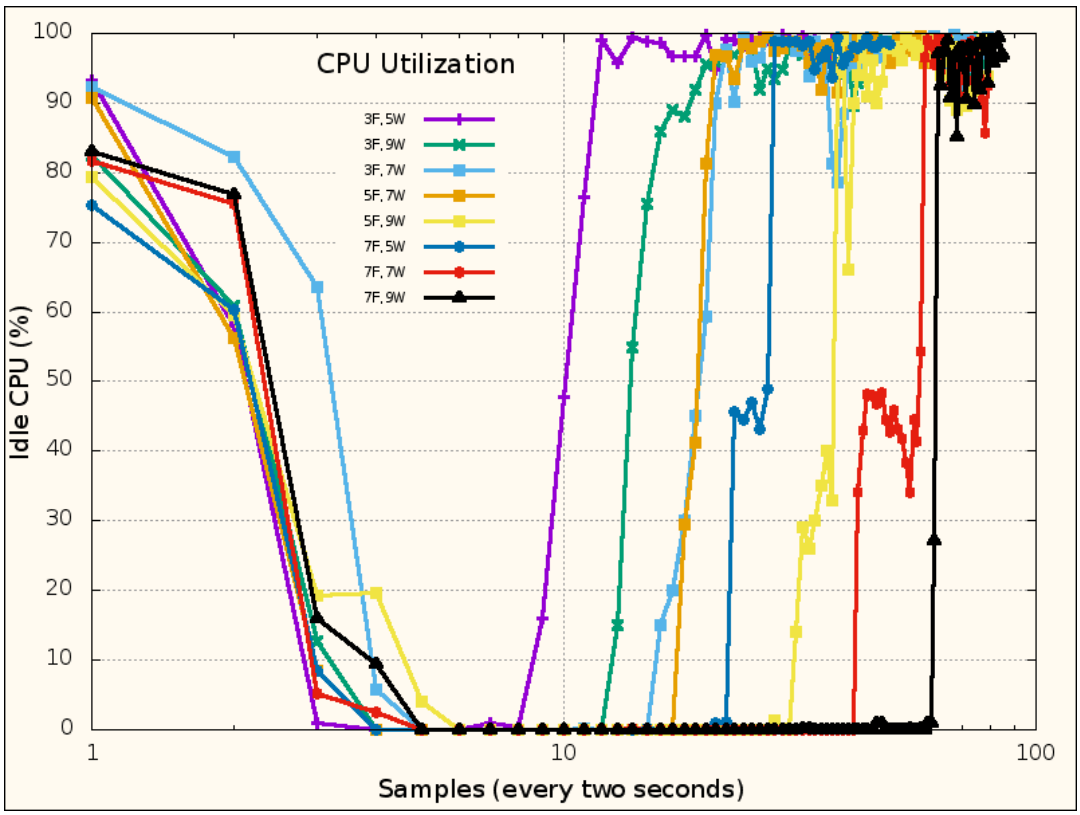


Figure 4.20.: CPU utilization for creating Mininet topologies with various number of fibers and wavelengths.

4.6.10.2. Memory Utilization

Figure 4.21 shows the graph for maximum Resident Set Size (RSS) memory allocated for creating Mininet Clos topologies with various number of fibers and wavelengths. Increasing the number of fibers and wavelengths increase the number of switches, links, and hosts and thus more memory is utilized. The graphs become steeper with higher values of W, since the number of links and switches increase as the square power of W, (see Equation 4.6.8, Equation 4.6.8, and Equation 4.6.8). Increasing the number of wavelengths in a sequence of one, increases the memory utilization by nearly more than 1.5MB for topologies with 3, 5, and 7 fibres, as seen in the graph. The memory utilization increases by 10.8% with increase in the number of fibers from 3 to 5, while the increase is 9.7% with increase in the number of fibers from 5 to 7, with the number of wavelengths equal to 8. However, this gain slightly increases with the number of wavelengths because of more memory required.

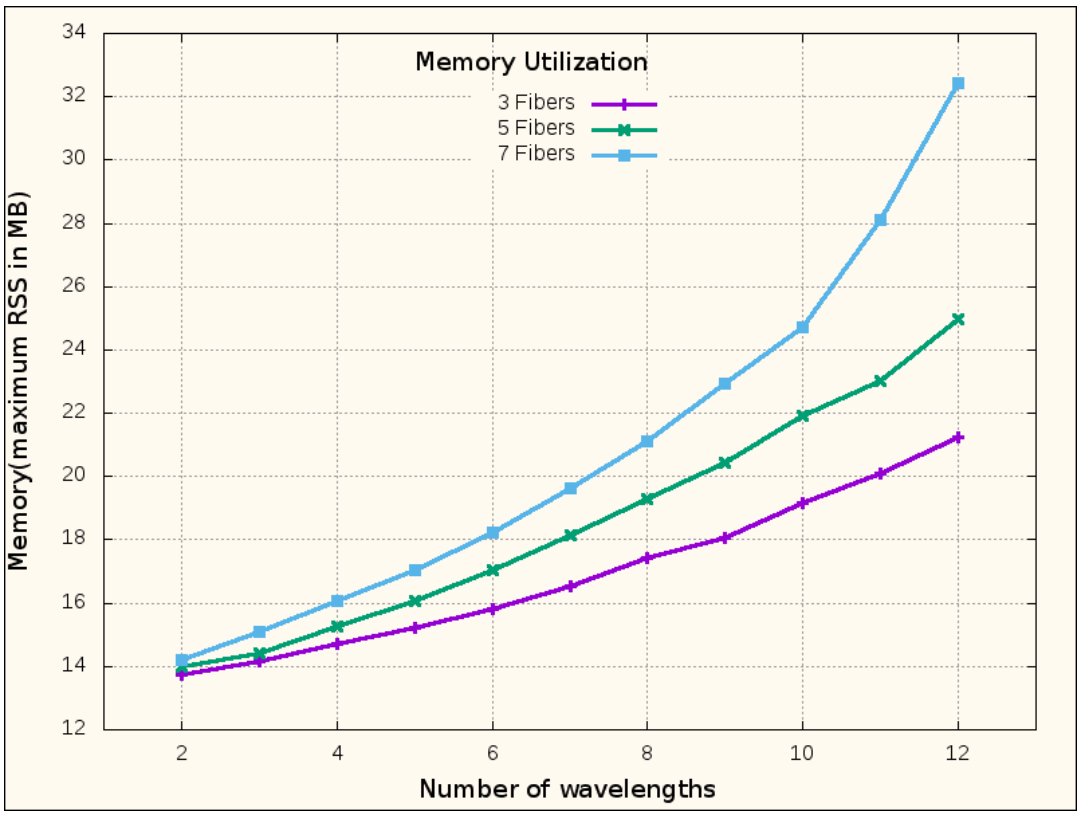


Figure 4.21.: Memory utilization for creating Mininet topologies with various number of fibers and wavelengths.

4.6.10.3. Topology emulation time

Figure 4.22 shows the increase in the time required for creating a topology in Mininet, with the increase in the number of fibers and wavelengths. It can be seen that the graph becomes steeper with higher values of F and W, this is compliant with our equations, (1-3) and Figure 4.19, where the number of switches and links increase as a square power of W. We observed that there is no specific approximate time for the creation of topology and the creation time is highly dependent on the state of the machine on which experiments are being conducted. For every fresh startup of the machine, less time is taken for the topology creation than if the machine is already running other processes or when cache memory is full. For our experiments, the topology creation time increases by 96% when increasing fibers from 3 to 5, and by 43% when increasing fibers from 5 to 7, while the number of the wavelengths is 8.

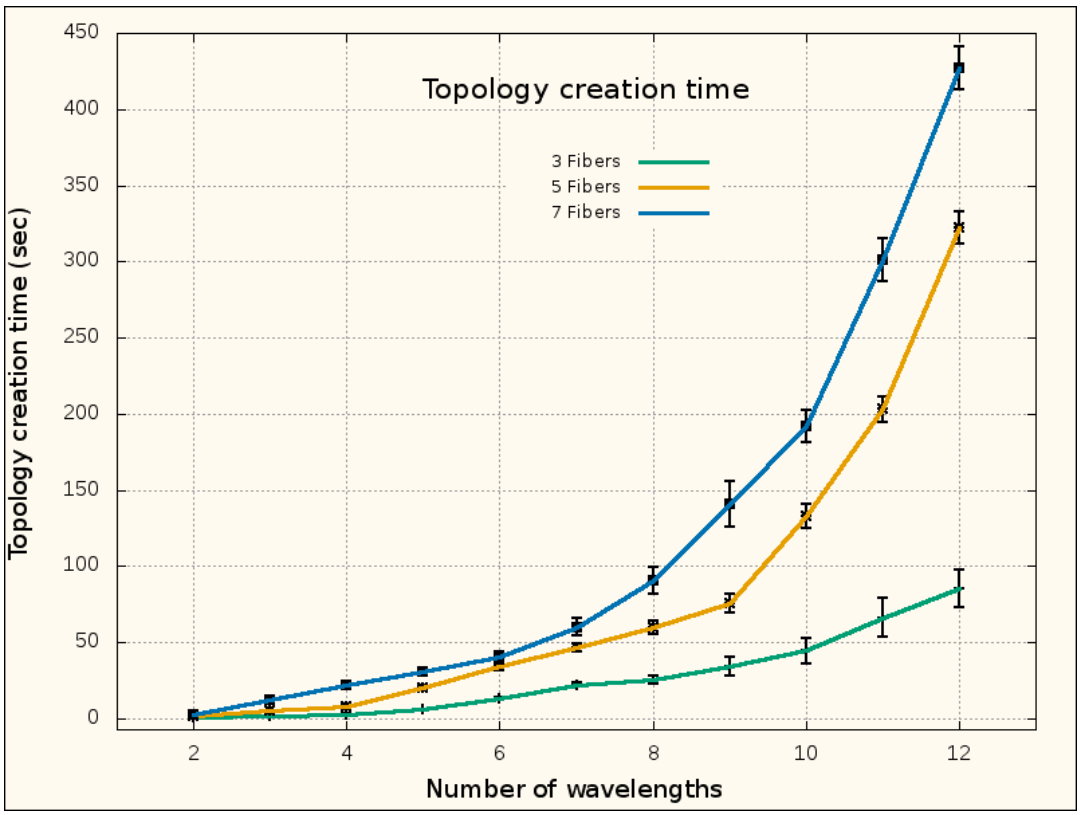


Figure 4.22.: Mininet topology creation time (mean and standard error) for various number of fibers and wavelengths.

4.6.10.4. Latency

The overall path setup delay depends on the delay in the links between OXCs, performance of the controller, time for translation of flows and configuration of optical switches. To measure the time required for the whole path setup process is difficult as it involves different systems; however, the most contributing and varying factor in the path setup delay is the path finding through the Mininet topology. Thus if we have an estimate of how much time is taken by the path finding process in the Mininet topology, then we can have a estimation of the overall path setup delay. In our model, RMS uses a spoofing mechanism to invoke path finding process which uses the 'Ping' command to find the path between the source and the destination hosts. Thus we measure the time taken by a successful 'Initial Ping' (Iping) [124, 125], by performing the Ping test between end hosts of the given network topologies for calculating round trip time (RTT) between hosts.

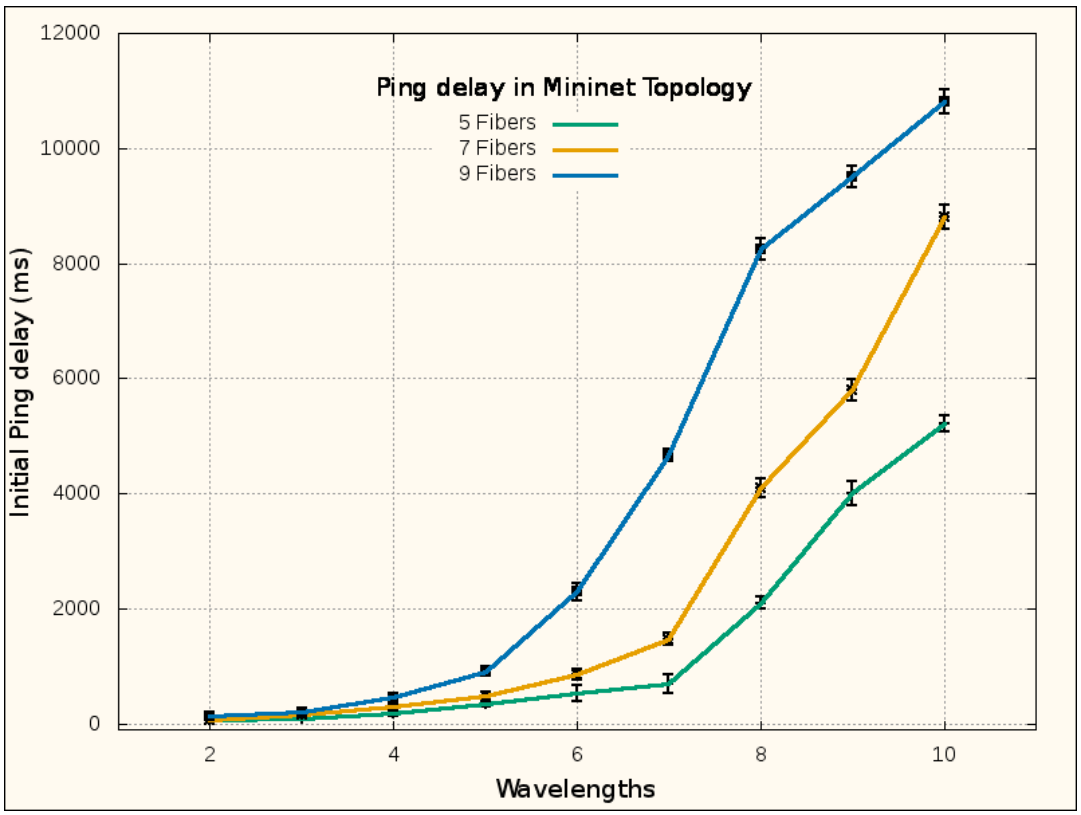


Figure 4.23.: Initial Ping delay for creating Mininet topologies for various number of fibers and wavelengths.

Figure 4.23 shows the Iping delays (mean values) between the end hosts for topologies with 5, 7, and 9 fibers. It is obvious from the graph that the latency has an increasing trend for bigger topologies; however, the increase is not smooth because of the high dependence on the state of the machine. For exact number of switches, links, and hosts in a topology, see equations (1-3). Standard error for bigger topologies is between 100-300 milliseconds approximately. Increase in the ping delay doubles when increasing the number of fibers from 5 to 7, and 7 to 9, for the number of wavelengths being equal to 8.

4.6.11. Discussion

The graphs shown here are only for a single switch having Clos topology switch fabric. For any other topology, re-analysis will be required. Moreover, there may exist different types of switches within the optical network, so overall network connectivity will be the combination of all the switch fabrics and hence will be the combined

required resources. Our results show compliance with the outcomes and discussions in some other performance analysis works, e.g., [126, 124, 127].

Figure 4.22 shows some large error bars especially for higher values of wavelengths, this is mainly because the speed of experiments depend on the background processes running in the machine (laptop) at the same time. This is an unavoidable situation in our case as the processor has to continue its back-end processes for its normal operation. Thus in cases, where automation of the process depends on the time of topology creation (as in our case), no certain time can be assumed and there is need of an alert system to tell that the topology creation is done and the system is ready for the next "path finding" step.

Dynamically establishing paths using SDN, seems to take time; however, considering core transport networks, the incoming requests do not arrive very often, and also once the path is established, it remains for longer times, e.g., months or years [128]. Although, there will be an initial loss of data until the path is found and established, but this is worth of the flexibility of the dynamic resources allocation, and offers a range of other benefits of SDN, as discussed previously in subsection 4.6.4.

4.6.12. Conclusion and future work

A model for path establishment in WDM network is presented in this paper which provides a flexible and configurable SDN solution for dynamic circuit-switching in WDM devices in core optical networks. It gives an advantage over other path establishment approaches presented in the literature in terms of considering switching constraints. None of the designs considered the congestion inside a switch and hence the non-availability of the path. This design is an effort towards a dynamic on-demand SDN based path establishment in WDM network without enhancing controller or OF protocol. However, the model considers the optical switch to be able to extract the destination addresses from the incoming requests. The model has been tested with Cisco layer one switch, and its performance is evaluated in terms of CPU and memory utilization, topology creation time, and latency. It is found that with more complicated switch fabric's topology, more computational resources are required and also path establishment time increases. Gain in the memory utilization, topology creation time, and latency is 9.7%, 43%, and 100% respectively, when increasing the number of fibers from 5 to 7 and the number of wavelengths is set to 8. Future work involves extension of the model for the optical switches with other

fabric topologies and any-to-any port connections. Furthermore, incorporating new modules for implementation of other optical network functionalities, e.g., protection and restoration.

References for IEEE ACCESS paper

[41] Steven Gringeri, Nabil Bitar, and Tiejun J. Xia. "Extending software defined network principles to include optical transport". In: *IEEE Communications Magazine* 51.3 (2013), pp. 32–40. DOI: 10.1109/MCOM.2013.6476863.

[48] Lei Liu et al. "Design and performance evaluation of an OpenFlow-based control plane for software-defined elastic optical networks with direct-detection optical OFDM (DDO-OFDM) transmission". In: *Optics Express* 22.1 (2014), p. 30. ISSN: 1094-4087. DOI: 10.1364/oe.22.000030.

[51] Bob Lantz et al. "Demonstration of software-defined packet-optical network emulation with mininet-optical and ONOS". In: *Optics InfoBase Conference Papers* Part F174- (2020), pp. 2020–2022. DOI: 10.1364/ofc.2020.m3z.9.

[54] Nick McKeown et al. "OpenFlow: enabling innovation in campus networks". In: *ACM SIGCOMM Computer Communication Review* 38 (2 Mar. 2008), p. 69. ISSN: 01464833. DOI: 10.1145/1355734.1355746. URL: http://portal.acm.org/citation.cfm?doid=1355734.1355746.

[55] Open Networking Foundation. "Software-Defined Networking: The New Norm for Networks [white paper]". In: *ONF White Paper* (2012), pp. 1–12.

[62] Paul Goransson and Chuck Black. *Software Defined Networks: A Comprehensive Approach*. second. Elsevier/Morgan Kaufmann, 2016, p. 436. ISBN: 9780128045558.

[67] Lei Liu et al. "Field trial of an openflow-based unified control plane for multilayer multigranularity optical switching networks". In: *Journal of Lightwave Technology* 31.4 (2013), pp. 506–514. ISSN: 07338724. DOI: 10.1109/JLT.2012.2212179.

[69] Tareq S. El-Bewab. *Optical Switching*. Ed. by Tarek S. El-Bawab. Boston, MA: Springer US, 2006. ISBN: 978-0-387-26141-6. DOI: 10.1007/0-387-29159-8. URL: http://link.springer.com/10.1007/0-387-29159-8.

[101] Mohit Chamania and Admela Jukan. *Springer Handbook of Optical Networks*. Ed. by Biswanath Mukherjee et al. Springer Handbooks. Cham: Springer International Publishing, 2020. Chap. Dynamic Co. ISBN: 978-3-030-16249-8. DOI: 10.1007/978-3-030-16250-4. URL: http://link.springer.com/10.1007/978-3-030-16250-4.

[102] Saurav Das, Guru Parulkar, and Nick McKeown. "Why OpenFlow/SDN can succeed where GMPLS failed". In: *European Conference on Optical Communication, ECOC* 1 (2012), pp. 31–33.

[103] D. Simeonidou, R. Nejabati, and M. P. Channegowda. "Software defined optical networks technology and infrastructure: Enabling software-defined optical network operations". In: *Optical Fiber Communication Conference, OFC 2013* 5.10 (2013), pp. 274–282. DOI: 10.1364/ofc.2013.oth1h.3.

[104] Saurav Das et al. "Packet and circuit network convergence with openflow". In: *Optics InfoBase Conference Papers* (2010), pp. 7–9. ISSN: 21622701. DOI: 10.1364/ofc.2010.otug1.

[105] R. Rejeb et al. "Management issues in transparent optical networks". In: *Proceedings of 2004 6th International Conference on Transparent Optical Networks* 1 (2004), pp. 248–254. DOI: 10.1109/icton.2004.1360286.

[106] Hui Yang et al. "Multipath protection for data center services in OpenFlow-based software defined elastic optical networks". In: *Optical Fiber Technology* 23 (2015), pp. 108–115. ISSN: 10685200. DOI: 10.1016/j.yofte.2015.03.002.

[107] H. Zang, J. P. Jue, and B. Mukherjee. "A review of routing and wavelength assignment approaches for wavelength- routed optical WDM networks". In: *Optical Networks Magazine* 1.January (2000), pp. 47–60. URL: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.127.5171{\&}rep=rep1{\&}type=pdf.

[108] Daniel F Macedo et al. "Programmable NetworksâFrom Software-Defined Radio to Software-Defined Networking". In: *IEEE Communications Surveys & Tutorials* 17.2 (2015), pp. 1102–1125. ISSN: 1553-877X. DOI: 10.1109/COMST.2015.2402617. URL: http://ieeexplore.ieee.org/document/7039225/.

[109] Zhaogang Shu et al. “Traffic Engineering in Software-Defined Networking: Measurement and Management”. In: *IEEE Access* 4 (2016), pp. 3246–3256. ISSN: 21693536. DOI: 10.1109/ACCESS.2016.2582748.

[110] Qian Dong et al. “A Path Allocation Method Based on Source Routing in SDN Traffic Engineering”. In: *Proceedings - 4th IEEE International Conference on Smart Cloud, SmartCloud 2019 and 3rd International Symposium on Reinforcement Learning, ISRL 2019* (2019), pp. 163–168. DOI: 10.1109/SmartCloud.2019.00037.

[111] Tianliang Zhang et al. “A WDM Network Controller with Real-Time Updates of the Physical Layer Abstraction”. In: *Journal of Lightwave Technology* 37.16 (2019), pp. 4073–4080. ISSN: 15582213. DOI: 10.1109/JLT.2019.2924397.

[112] Ignacio Martin et al. “Machine Learning-Based Routing and Wavelength Assignment in Software-Defined Optical Networks”. In: *IEEE Transactions on Network and Service Management* 16.3 (2019), pp. 871–883. ISSN: 1932-4537. DOI: 10.1109/TNSM.2019.2927867. URL: https://ieeexplore.ieee.org/document/8758853/.

[113] Dao Thanh Hai. “On the spectrum-efficiency of QoS-aware protection in elastic optical networks”. In: *Optik* 202.October 2019 (2020), p. 163563. ISSN: 00304026. DOI: 10.1016/j.ijleo.2019.163563. URL: https://doi.org/10.1016/j.ijleo.2019.163563.

[114] Devi Chadda. *Software â Defined Optical Networks*. 2019, pp. 331–358. DOI: 10.1002/9781119393399.

[115] Lei Liu et al. “OpenSlice: An OpenFlow-based control plane for spectrum sliced elastic optical path networks”. In: *European Conference on Optical Communication, ECOC* 21.4 (2012), pp. 4194–4204. ISSN: 1094-4087. DOI: 10.1364/oe.21.004194.

[116] Dao Thanh Hai, Le Hai Chau, and Nguyen Tan Hung. “A Priority-Based Multiobjective Design for Routing, Spectrum, and Network Coding Assignment Problem in Network-Coding-Enabled Elastic Optical Networks”. In: *IEEE Systems Journal* 14.2 (2020), pp. 2358–2369. ISSN: 19379234. DOI: 10.1109/JSYST.2019.2938590.

[117] Ahmed E Kamal. “Network Coding-Based Protection â”. In: (), pp. 1–20.

[118] Deepa B. Swarna and V. Muthumanikandan. “Nested Failure Detection and Recovery in Software Defined Networks”. In: *Proceedings of 2019 3rd IEEE International Conference on Electrical, Computer and Communication Technologies, ICECCT 2019* (2019). DOI: 10.1109/ICECCT.2019.8869085.

[119] Irena Seremet et al. “SDN as a Tool for Energy Saving”. In: *27th Telecommunications Forum, TELFOR 2019* (2019), pp. 2019–2022. DOI: 10.1109/TELFOR48224.2019.8971086.

[120] Adriana Fernández-Fernández, Cristina Cervelló-Pastor, and Leonardo Ochoa-Aday. “Achieving energy efficiency: An energy-aware approach in SDN”. In: *2016 IEEE Global Communications Conference, GLOBECOM 2016 - Proceedings* (2016). DOI: 10.1109/GLOCOM.2016.7841561.

[121] Mpho C. Nkosi et al. “Towards programmable on-demand lightpath services: Current state-of-the-art and open research areas”. In: *IET Networks* 8.6 (2019), pp. 347–355. ISSN: 20474962. DOI: 10.1049/iet-net.2018.5040.

[122] Imad Alawe et al. “Integration of legacy Non-SDN optical ROADMs in a software defined network”. In: *Proceedings - 2016 IEEE International Conference on Cloud Engineering Workshops, IC2EW 2016* (2016), pp. 60–64. DOI: 10.1109/IC2EW.2016.11.

[123] Philippos Isaia and Lin Guan. “Performance benchmarking of SDN experimental platforms”. In: *2016 IEEE NetSoft Conference and Workshops (NetSoft)*. IEEE, 2016, pp. 116–120. ISBN: 978-1-4673-9486-4. DOI: 10.1109/NETSOFT.2016.7502456. URL: http://ieeexplore.ieee.org/document/7502456/.

[124] Arun K. Arahunashi, S. Neethu, and H. V. Ravish Aradhya. “Performance Analysis of Various SDN Controllers in Mininet Emulator”. In: *2019 4th IEEE International Conference on Recent Trends on Electronics, Information, Communication and Technology, RTEICT 2019 - Proceedings* (2019), pp. 752–756. DOI: 10.1109/RTEICT46194.2019.9016693.

[125] Anh Nguyen-Ngoc et al. “Performance evaluation of selective flow monitoring in the ONOS controller”. In: *2017 13th International Conference on Network and Service Management, CNSM 2017* 2018-Janua (2017), pp. 1–6. DOI: 10.23919/CNSM.2017.8256058.

[126] David Muelas, Javier Ramos, and Jorge E.Lopez De Vergara. “Assessing the Limits of Mininet-Based Environments for Network Experimentation”. In: *IEEE Network* 32.6 (2018), pp. 168–176. ISSN: 1558156X. DOI: 10.1109/MNET.2018.1700277.

[127] Filippo Cugini and Piero Castoldi. “Software Defined Networking (SDN) in Optical Network”. In: *Elastic Optical Networks*. Ed. by Víctor López and Luis Velasco. Vol. 56. Optical Networks. Cham: Springer International Publishing, 2016. Chap. 9, pp. 101–193. ISBN: 978-3-319-30173-0. DOI: 10.1007/978-3-319-30309-3_3. URL: http://link.springer.com/10.1007/978-3-319-30174-7http://link.springer.com/10.1007/978-3-319-30309-3{_}3.

4.6.13. Additional testing details

Overview of the testing setup has been presented in the above included article. However there are some minor implementation details that are explained in this section. The ExaLINK-Fusion switch used for testing has a 48 cross-point structure(input/output). This is mapped in Mininet as following figure. For simplicity only 3 input and 3 outputs are shown in Figure 4.24. It is clear that every cross-point is represented by a single switch between two hosts i.e., input and output. There is only one path between any input and output port mirroring the crossbar architecture.

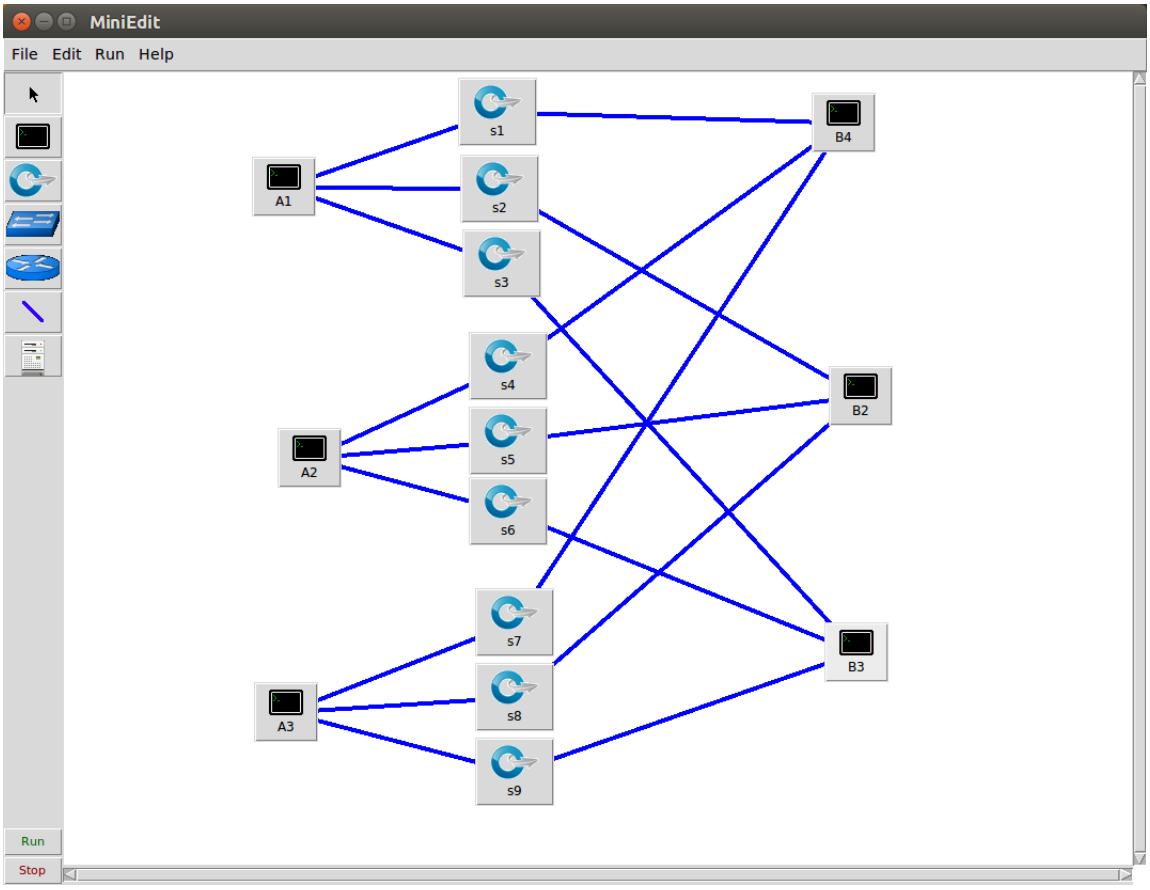


Figure 4.24.: Mininet topology for abstracted crossbar fabric with 3 input and output ports.

ExaLINK fusion is configured running following commands in Ubuntu terminal, to send the signal arrival trap notification to the RMS:

```
$telnet IP:Add:of:Switch
```

After entering logging in details (provided by manufacturer), the switch becomes accessible for configuration.

```
admin@EXFSN-XYZ> config show snmp
admin@EXFSN-XYZ> config snmp read community public
admin@EXFSN-XYZ> config snmp enable
admin@EXFSN-XYZ> config snmp trap enable
admin@EXFSN-XYZ>config snmp trap target my:IP:Add:ress public
```

On the receiving machine(SNMP manager), the snmptrap daemon/receiver needs to be turned on and put in listening state by running following command in an another Ubuntu terminal.

```
$sudo snmptrapd -c /etc/snmp/snmptrapd.conf
```

In above command, path to a file namely 'snmp.conf' is provided. The file "snmptrapd.conf" is a configuration file for trap receiver which is called to specify conditions and details of the snmptrap notifications. The file needs to be configured to receive connections on the required IP4 interfaces. Details for how to do this configuration are provided in the following link.

https://www.youtube.com/watch?v=S0ibDVFnUws

In order to stop snmptrap daemon type following command:

```
$systemctl stop snmptrapd.service
```

we can also check traffic received at the interface by typing following command:

```
$sudo tcpdump -i tun0
```

After receiving trap and path founded by the controller, Remote Procedure Call (RPC) is exploited to configure the switch ports. Here RPC is used (and not SNMP) because SNMP configuration only provides reading (and no writing) accessibility. RPC is a client-server based protocol which calls a process on a remote system without going into network details. Python script is used to create the RPC. Actual script is available in Appendix. One of the commands of this script to create a connection is as below:

```
fusion.create_patch(ports=["B1", "B2"]
```

Part III.

Drawing conclusions about the proposition

5. Conclusions and Future Work

This chapter presents the conclusions which are validated in previous chapters (3 and 4), and are stated in terms of the propositions made in chapter 1. This chapter discusses the research outcomes that have resulted from the investigation and implementation of this thesis. Furthermore, directions for future work and some of the gaps and shortcomings of this research are also presented. The main goal of this research work highlighted in this thesis is to design, model and develop a middleware mechanism as a new approach for controlling SDON networks. Due to the increased demand for several fast and bandwidth-hungry applications, large multi-service networks are increasingly becoming laborious and challenging to manage. The traditional management method needs help to handle on-demand bandwidth requirements in static and pre-planned network deployments.

SDN paradigm has emerged as a management framework that suggests defeating the bottlenecks of traditional packet networks by isolating the network's control plane from the data plane in network devices, aiding centralized network control and launching the programmable networks. SDN has boosted the concept of programmable networks, and since then, the idea has been gaining enormous attention. SDN promises to remarkably simplify network control and management through network programmability using open-source OpenFlow communication protocol. Using OpenFlow, the network elements are empowered to adapt to the developing environments and can fulfill the needs of network carriers, vendors and end-users.

An OpenFlow-based SDN network is designed and suited well to packet-based networks. In packet-based networks, each packet header is processed, and information is extracted, which is used to make a decision for further action managed from a central position. However, optical networks are circuit-switched, which means a complete bandwidth/channel is allocated for data transmission from one point to another; and to change the path for data transmission, the entire channel has to be switched. Thus implementing SDN is naturally challenging for optical/WDM networks. Various solutions have been researched and presented which offer extensions

of current standards using an agent-based approach. Since none of the extensions provided has been standardized, a well-designed system is required to be used with current standards to make optical networks more efficient and autonomous.

In order to use current SDN standards with optical hardware and network, we present a solution that is based on designing a middle layer system that can do translations in both directions, i.e., for the switch to controller and controller to switch. In particular, a resource management system has been proposed to allow dynamic path switching in optical network elements, and the on-the-flight decision-making process will essentially contribute to the freedom of complex management of optical networks.The latency of the path establishment process as well as the performance of RMS, has been presented. The research propositions in Chapter one are validated by the design and implementation of the proposed model. Furthermore, contributions are validated by the publication of work and experiments in this thesis.

This thesis exemplifies that a translation mechanism like the one in the proposed model can be applied between an OpenFlow-based SDN architecture and optical layer and can perform operational functions in terms of switching circuits. Further, the model abstracts layer one network to the SDN controller as a network of layer two switches. Hence the SDN controller does not see the wavelength information and associated constraints. The model is developed to solve the issue of static allocation of paths and eliminate manual interference when a change in service is required. In the case of dynamic path computation by the switching devices, this proposal shifts the computational load to the controller and effectively minimizes the communication overhead by having a central view of all networks. The on-demand nature offered By SDN and OpenFlow Allows Switches To switch paths and adapt dynamically to the developing and changing requirements. The model has much room for improvement, and many other features can be embedded, e.g., saving protection paths or preferred paths. Some other observations learned during experimentation are also presented in this dissertation.

5.1. Validating Research Propositions

The main objective of this project is to develop, implement and test a model that could switch optical circuits and setup switch fabric of optical devices in an on-

demand fashion. The primary objectives are met using OpenFlow based SDN controller and our developed RMS as explained in chapter 3 and chapter 4. The performance of the model with respect to latency, CPU and memory utilization have been thoroughly measured and analysed through a number of experiments and evaluations. The results obtained from the emulation tests and hardware testing indicate that the model can be used in optical core networks for path establishment, provided the conditions/assumptions are met as stated in section 4.5. The research propositions presented in chapter one are re-analyzed and validated here in this chapter. The initial research propositions outlined in the introduction chapter are justified and summed up in the same order here and their associated contributions towards software-defined optical networks are also stated.

Proposition 1: *That a Software Defined Networking paradigm for controlling the optical paths without extending OpenFlow protocol, can be implemented.*

Validation Summary: A model which is based on a translation mechanism is created and applied to the SDN architecture. The RMS translation system is deployed between the SDN controller and layer one network. The RMS first collects information on connectivity in the layer one network and internal fabric topology of switches and generates a layer two network of OpenFlow switches which maps the layer one network. On arrival of the signal, controllers are requested to find a path for this signal according to its destination. Once a course is found, the translation component of RMS translates back this information and sends suitable commands to the layer one switch to set up fabric and switch the signal to the appropriate port.

SDN and OpenFlow protocols appeared as an alternative to traditional data networks. The concept is well suited to the networks that transport data packet-by-packet basis. Optical networks, on the other hand, are circuit-switched, and to bring the programmability that SDN offers into the optical networks, an agent-based approach has been used. For the agent-based implementation, the OpenFlow protocol and SDN controller need to be extended to cater to the wavelengths and circuit information. The few agent-based deployments of OpenFlow (discussed earlier in chapter two) have to use their extended versions of OpenFlow protocol, which can give the impression of the proprietary protocol. In this thesis, these limitations are overcome by using the RMS model, which has no restriction to be physically or logically embedded in the optical device like an agent, and instead uses standard OpenFlow and a primary SDN controller.

The system that works like an agent is a translation system that does all the required translation between optical switch and the controller. This easy-to-operate and maintain system can reside in the management center where the controller may also be present. The theory of the model is developed in Chapter 3, whereby the connection setup process (timing diagram) shows the steps involved in the path setup using the SDN concept. The fundamental operational flow of the steps is also presented. In chapter 3, the problem background and the need for a dynamic path establishment system is also explained, hence the contribution of model is justified. In the first and second section of chapter 4 provides the justification for the contribution of the implemented model. The major contribution to the knowledge of this chapter is the implementation of the model and achieving the goal of path setup without extending any standards. the model is created with the goal to address the limitations of current agent-based approaches and future autonomic optical network systems and management. model implementation has been described in three major steps i.e., information collection, path computation and path setup. Path computation process starts when signal arrives at the optical port and an alert signal is received by the RMS along with the required information. All three steps are explained in details in the presented and published articles. The model has also been tested with a hardware switch which supports the characteristics of layer one processing of the incoming data. Hence model has been successfully applied in an appropriate environment. As a result, the proposition one is substantiated and the objective has been achieved. the model is further experimented for performance analysis in this dissertation.

Proposition 2: *That the OpenFlow protocol can be used to program and set up layer one switch and its fabric, and that the optical switch can notionally act as a network of layer two switches to the SDN controller.*

Validation Summary: We developed an algorithm that abstracts layer one network connectivity into a network of Layer two OpenFlow switches using python and Mininet libraries. SDN controller finds a path through this abstracted topology on a request. The translation mechanism translates the path for this abstracted topology by looking at the flows installed in OpenFlow switches. The path setup component in our developed model is designed to tell the optical switch to make connections between the input to output ports of the switch fabric. The model is tested with a real switch, and an experiment is performed in which connections are made between its input and output ports.

The agent-based solutions, which have been previously suggested to switch the optical paths, did not consider the constraints of the internal switch fabric, nor the path is established dynamically in its true sense as the request for the path setup is not generated when the data arrives but is pre-setup when on the request of optical layer's client. In our model, the switches' internal fabric connectivity is abstracted, as well as connections between the optical devices. This is explained in section 4.2. The path is installed in the OpenFlow network using OpenFlow protocol version 1.0. However, using version 1.3 does not make any impact on the path finding process. The main task for verification is check that the abstraction actually represents the mapped topology is to use ping command tool between various ports. For demonstration purposes, Clos and Crossbar fabric topologies are considered. Ping test was conducted and results were as expected. The other task is to verify if the path found for the translated topology can be used to setup path in fabric or not? Thus the model is tested with a real physical layer one switch where the switch and machine with our model are connected through an Ethernet network. In the experiment, the switch's input and output ports are made to connect using a Remote Procedure Call from our machine. Thus verifying our proposition number two which also employs the feasibility of applying model to real networks.

Proposition 3: *That SDN can allow dynamic optical path establishment on the first-time arrival of the data signal and on a request from the optical node itself. And that it can allow autonomic service provision in future optical networks and a single control mechanism for the multi-layered networks.*

Validation Summary: After successfully implementing a Resource Management System based proposed model in this thesis, the research results could be the first effort in bringing forward the approach of autonomic path establishment within switch fabric without pre-setup or pre-plan, i.e., on the arrival of data packets. Further, the created model supports the approach to a single control and management system when layer two and above are already managed by the SDN.

Software-defined control mechanism has already proved its potential and performance in packet-based networks, i.e., layers two to four in OSI model. SDN has been implemented by many corporate industries, e.g., Google and Facebook. It is reasonable to consider and deploy the same centralized management and control paradigm for layer one. Future networks will be even more complex than they are today, and a software-based single control mechanism for all layers in a multi-layered

network is a natural choice. Therefore, here, an SDN architecture is proposed, which is already designed with an approach of a single control entity for path control between devices and inside the switches. In this thesis, chapter 4 presents the system implementations and proposition validations as well as performance measurements and experiments.

As shown in Figure 1 of [99] which is included in section 4.5 of this dissertation, the traditional path establishment task is a long process and occurs after going through several systems. To cater to new requests, re-computations and re-configurations are required, and manual interference is also requisite; thus, the chances of error also increase. Using a single autonomic control mechanism will significantly simplify the process, especially when it is already used for managing other layers. Therefore SDN based mechanism is proposed, which can be used to perform different functions and replace traditional NMS. Thus implemented model can result from having a substantial impact on future single-system controlled multi-layered networks. The performance of the model is also therefore tested and evaluated.

SDN controllers and OpenFlow protocol have an inherent potential for the autonomic management of systems. SDN controller only starts computing for decisions when it is invoked. "PacketIn" messages with the required information and the applications in the controller work combine to decide according to an already defined behavior leading to autonomous behavior of the network. Applications that are hooked to the controller can be easily modified on demand to change the behavior of the network. Our model demonstrates an example of automatic path establishment when the node requests it. Thus verifying our proposition 3. Other advantages are also discussed in the section 4.5 in the published article, hence answering our research question number 4.

5.2. Research Findings & Contributions

The proposed model, theory, implementation, ideas and experiments presented in this thesis are considered contributions and findings to current research knowledge of the networks. The outcome of the proposed model, implementation and experiments are documented and published and added as part of this thesis. The findings and contributions include the following:

1. This thesis presents a review of the literature on the optical network, its components, management and their issues. This contribution provides the background of the management problem and service provision in optical core networks. Thesis chapter two section 2.2 has contributed to the knowledge.

2. Review of literature on the Software-Defined Networking (SDN) paradigm. Application and feasibility of SDN in optical networks, and challenges of Software Defined Optical Networks. This contribution presents state of the art in dynamic optical networks. It also provides a review of the ongoing research efforts. (Section 2.3 and 2.4)

3. The first effort considers optical switch fabric constraints and includes them in the path allocation step. Contribution is made in mapping the fabric into a Mininet layer two network. A conference paper is published as a contribution to the knowledge and added in section 4.2.

4. A new model is proposed as an alternative to the previous agent-based efforts, which required an extension of the protocol standards. The theory of the model is presented in chapter three. A conference paper based on the theory of this proposed model has been published as a contribution to the knowledge in section 3.3.

5. This model is implemented based on its theory and architecture using Mininet and Python programming tools. A new resource management system is created to facilitate translation and communications between the controller and optical switches, which is implemented using a GUI application, SNMP protocol and RPC. A journal paper included in section 4.6 describing all the implementation details contributes to the knowledge.

6. Testing of the model with a real switch and performance measurement in terms of memory and CPU utilization and also latency of path establishment process is contribution to knowledge.

7. The implementation code of the model is made available on GitHub to help further research, trials and improvement of the model. It is considered a contribution to the SDN and the optical networking research community, see Appendix A.

5.3. Research Limitations

Although the implementation of our model is accomplished and the outcomes are justified and published in articles, below are some limitations of this research:

- The OpenFlow mapped Mininet network is constructed on a limited scale, i.e., for only one optical switch and three stages of the Clos fabric topology, because of the limited research resources, e.g., computational power and time. With better available resources, a large-scale network can be mapped in Mininet to present and experiment with more real research scenarios. The performance analysis of the model is discussed in section 4.3. It has been observed that employing more computing power allows better handling of large Mininet networks and communication between the controller and OpenFlow switches. Thus more experiments from real large-scale networks can be performed to obtain practical results.
- In the implementation of the model, the information about the switch ports and cells is obtained using static files because of the unavailability of a suitable open source MIB database. However, using a real MIB, more sophisticated and generic translation mechanism can be constructed as RMS component and model can be applicable to more real scenarios.
- The model is tested with a layer one switch which is not an optical switch because of the limited resources. Although model's validity is justified with the used switch, the justification can be made more strong if in future a real optical switch or ROADM is used.
- In the current implementation of the model, a mechanism for link failure or saving protection paths is not incorporated. This is because the protection path setup is not in the scope of this project; however, there is room in the model to incorporate this feature in future without much effort.
- Because of the limited computational power resources, performance of the model can not be measured on a large scale mapped topology. The memory and CPU utilization could only be measured upto a certain processing limit. However, with more powerful resources, rigorous experiments and measurements can be performed to get more credible results.

5.4. Future Directions

As mentioned in the previous section, this research work has limitations which can be optimized to make it more realistic and practical. Furthermore, other features can be added and experiments can be done to improve the model. Below, further research directions are suggested which aim at improvement in the model:

- A more realistic network topology can be mapped in Mininet if high computational power resources are available. Thus in the future, with such high power resources, the model can be used to set up a path for a network of multiple optical switches connected in, e.g., a mesh topology. With more power, the performance of the model can also be improved in terms of time taken for topology creation, path computation and ultimately latency.
- In the current implementation, the Crossbar and Clos topology are mapped. However, future algorithms for other topologies and associated translation dictionaries can be developed. Furthermore, a generic translation system can also be developed through the analysis of MIB files for various vendors' equipment.
- In the current implementation of the model, the SDN controller applies STP application which employs the following two things.

1. Best path is calculated on the basis of the highest bandwidth. A Ryu application can be built to choose the best path based on any other metric, according to the user's requirement.
2. The best cost link is used as chosen path (preferred) unless it fails, in which case a non-preferred redundant link is enabled or STP recalculation is performed. This feature of STP can be used to establish a protection path in an optical network. The optical switch observing the link failure needs to send an alarm RMS. A component can be added in the RMS, which would mimic respective link failure and hence trigger STP recalculation. Other than this, a separate database can also be maintained to save protection paths in advance to save time when a failure occurs.

- In the implemented model, the sample optical switches are assumed to have no wavelength converters. However, it is very likely that the real optical

switch will have a few wavelength converters. Thus mirrored Mininet topology structure will change, and re-analysis and tests are required.

- Although SDN has been compared earlier with other path establishment techniques, e.g., GMPLS and has proved to outperform, this research can also be tested against different approaches in a lab setup to get somewhat comparative results. However, this comparison is not in the scope of this research and hence left for future work.
- Finally, research outcomes can be used in other related research, such as traffic engineering in SDON or multi-layer network convergence. The designed model, implementation and code details are provided in this dissertation and can be used with other related research experiments.

Bibliography

[1] John R. Vacca. "Optical Formats: Synchronous Optical Network (SONET)/ Synchronous Digital Hierarchy (SDH), and Gigabit Ethernet". In: *Optical Networking Best Practices Handbook*. John Wiley & Sons, Inc., 2006, pp. 179–232. DOI: 10.1002/0470075066.ch8.

[2] H.G. Perros. *Connection-Oriented Networks: SONET/SDH, ATM, MPLS and Optical Networks*. Wiley, 2005. ISBN: 9780470021644. URL: https://books.google.com.au/books?id=DjxrezJsVnsC.

[3] H. van Helvoort. *SDH / SONET Explained in Functional Models: Modeling the Optical Transport Network*. Wiley, 2005. ISBN: 9780470091241. URL: https://books.google.com.au/books?id=fEJqR_jug38C.

[4] Hui Zang et al. "Dynamic lightpath establishment in wavelength-routed WDM networks". In: *IEEE Communications Magazine* 39.9 (2001), pp. 100–108. ISSN: 01636804. DOI: 10.1109/35.948897.

[5] Eric Bouillet et al. "Optical Networking". In: *Path Routing in Mesh Optical Networks*. 2007, pp. 1–23. DOI: 10.1002/9780470032985.ch1.

[6] Jane M. Simmons. *Optical Network Design and Planning*. Springer International Publishing, 2014. DOI: 10.1007/978-3-319-05227-4.

[7] John R. Vacca. "Optical Network Configurations". In: *Optical Networking Best Practices Handbook*. John Wiley & Sons, Inc., 2006, pp. 326–336. DOI: 10.1002/0470075066.ch12.

[8] Devi Chadha. "Network Control and Management". In: *Optical WDM Networks: From Static to Elastic Networks*. 2019. Chap. 7, pp. 229–259. DOI: 10.1002/9781119393399.ch7.

[9] Bijoy Chand Chatterjee, Takehiro Sato, and Eiji Oki. "Recent research progress on spectrum management approaches in software-defined elastic optical networks". In: vol. 30. 4. Elsevier BV, 2018, pp. 93–104. DOI: 10.1016/j.osn.2018.07.001.

[10] Bob Lantz, Brandon Heller, and Nick McKeown. "A network in a laptop:rapid prototyping for software-defined networks". In: *Proceedings of the Ninth ACM SIGCOMM Workshop on Hot Topics in Networks - Hotnets '10*. New York, New York, USA: ACM Press, 2010, pp. 1–6. ISBN: 9781450304092. DOI: 10.1145/1868447.1868466. URL: http://dl.acm.org/citation.cfm?id=1868466http://portal.acm.org/citation.cfm?doid=1868447.1868466.

[11] Subir Bandyopadhyay. *Dissemination of information in optical networks. from technology to algorithms*. Springer, 2008, p. 308. ISBN: 9783540728740.

[12] Filippo Cugini et al. "Software Defi ned Networking (SDN) in Optical Networks". In: (2016), pp. 217–244. DOI: 10.1007/978-3-319-30174-7.

[13] Mohit Chamania and Admela Jukan. "Dynamic Control of Optical Networks". In: *Springer Handbook of Optical Networks*. Ed. by Biswanath Mukherjee et al. Cham: Springer International Publishing, 2020, pp. 535–552. ISBN: 978-3-030-16250-4. DOI: 10.1007/978-3-030-16250-4_15. URL: https://doi.org/10.1007/978-3-030-16250-4_15.

[14] Nirwan Ansari. *Media Access Control and Resource Allocation. For Next Generation Passive Optical Networks*. Springer New York, 2013, p. 111. ISBN: 9781461439394.

[15] Sadia Qureshi and Robin Braun. "Dynamic Light Path Establishment In switch Fabric Using OpenFlow". In: *2018 26th International Conference on Systems Engineering (ICSEng)*. IEEE, 2018, pp. 1–4. ISBN: 978-1-5386-7834-3. DOI: 10.1109/ICSENG.2018.8638220. URL: https://ieeexplore.ieee.org/document/8638220/.

[16] Jason P. Jue. "Lightpath Establishment in Wavelength-Routed WDM Optical Networks". In: *Optical Networks*. Springer US, 2001, pp. 99–122. DOI: 10.1007/978-1-4613-0291-9_5.

[17] Rajiv Ramaswami. *Optical networks. a practical perspective*. Elsevier/Morgan Kaufmann, 2010, p. 893. ISBN: 9780123740922.

[18] C. T. Politi, C. Matrakidis, and A. Stavdas. "A Tutorial on Physical-Layer Impairments in Optical Networks". In: *Optical Networks*. Springer US, 2013, pp. 5–29. DOI: 10.1007/978-1-4614-5671-1_2.

[19] Tareq S. El-Bewab. *Optical Switching*. Ed. by Tarek S. El-Bawab. Boston, MA: Springer US, 2006. ISBN: 978-0-387-26141-6. DOI: 10.1007/0-387-29159-8. URL: http://link.springer.com/10.1007/0-387-29159-8.

[20] Rinkle Rani, Lakhwinder Kaur, and Aggarwal. "Multistage Interconnection Networks: A Transition from Electronic to Optical". In: *Journal of Emerging Technologies in Web Intelligence* 2 (May 2010). DOI: 10.4304/jetwi.2.2.142-147.

[21] Rod C. Alferness. "The Evolution of Optical Transport Networks". In: *Springer Handbook of Optical Networks*. Springer International Publishing, 2020, pp. 1–19. DOI: 10.1007/978-3-030-16250-4_1.

[22] Iannone Eugenio. "Switching Systems: Architecture and Performances 7". In: 2017, pp. 437–538.

[23] S.C. Kothari. “Multistage Interconnection Networks for Multiprocessor Systems”. In: *Advances in Computers*. Ed. by Marshall C. Yovits. Vol. 26. Advances in Computers. Elsevier, 1987, pp. 155–199. DOI: https://doi.org/10.1016/S0065-2458(08)60007-8. URL: https://www.sciencedirect.com/science/article/pii/S0065245808600078.

[24] Devi Chadha. “Network Elements”. In: *Optical WDM Networks: From Static to Elastic Networks*. 2019, pp. 27–90. DOI: 10.1002/9781119393399.ch2.

[25] Charles Clos. “A Study of Non-Blocking Switching Networks”. In: *Bell System Technical Journal* 32.February (1953), pp. 406–424. DOI: 10.1002/j.1538-7305.1953.tb01433.x.

[26] Olav Lysne and Frank Olaf Sem-Jacobsen. “Networks, Multistage”. In: *Encyclopedia of Parallel Computing*. Ed. by David Padua. Boston, MA: Springer US, 2011, pp. 1316–1321. ISBN: 978-0-387-09766-4. DOI: 10.1007/978-0-387-09766-4_317. URL: https://doi.org/10.1007/978-0-387-09766-4_317.

[27] Qixiang Cheng et al. “Scalable Microring-Based Silicon Clos Switch Fabric with Switch-and-Select Stages”. In: *IEEE Journal of Selected Topics in Quantum Electronics* 25.5 (2019), pp. 1–11. ISSN: 21910359. DOI: 10.1109/JSTQE.2019.2911421.

[28] Akhilesh S. P. Khope. *Wavelength-selective optical switches help scale the datacenter | Laser Focus World*. 2021. URL: https://www.laserfocusworld.com/fiber-optics/article/14210712/wavelengthselective-optical-switches-help-scale-the-datacenter.

[29] Lena Wosinska, Lars Thylen, and Roger P. Holmstrom. “Large-capacity strictly nonblocking optical cross-connects based on microelectrooptomechanical systems (MEOMS) switch matrices: Reliability performance analysis”. In: *Journal of Lightwave Technology* 19.8 (2001), pp. 1065–1075. ISSN: 07338724. DOI: 10.1109/50.939785.

[30] G. Maier and A. Pattavina. “Design of photonic rearrangeable networks with zero first-order switching-element-crosstalk”. In: *IEEE Transactions on Communications* 49.7 (2001), pp. 1268–1279. DOI: 10.1109/26.935167.

[31] ITU-T G.8080: “Recommendation G.8080/Y.1304: Architecture for the automatic switched optical networks (ASON)”. In: (2012).

[32] Daniel King et al. “Path computation architectures overview in multidomain optical networks based on ITU-T ASON and IETF PCE”. In: *2008 IEEE Network Operations and Management Symposium Workshops - NOMS 08* (2008), pp. 219–226. DOI: 10.1109/NOMSW.2007.37.

[33] Lei Liu et al. “Optical Network Control and Management Technology Using OpenFlow”. In: 2013, pp. 3–4.

[34] Dimitra E. Simeonidou, Reza Nejabati, and Mayur Channegowda. "Software Defined Optical Networks Technology and Infrastructure: Enabling Software-Defined Optical Network Operations". In: *Optical Fiber Communication Conference/National Fiber Optic Engineers Conference 2013* 5.10 (2013), OTh1H.3. ISSN: 1943-0620. DOI: 10.1364/OFC.2013.OTh1H.3. URL: https://www.osapublishing.org/abstract.cfm?uri=OFC-2013-OTh1H.3.

[35] Sead Dizdarevic et al. "A survey on transition from GMPLS control plane for optical multilayer networks to SDN control plane". In: IEEE, May 2016, pp. 537–544. ISBN: 978-953-233-086-1. DOI: 10.1109/MIPRO.2016.7522202. URL: http://ieeexplore.ieee.org/document/7522202/.

[36] Ed. E. Mannie. "RFC 3945:Generalized Multi-Protocol Label Switching (GMPLS) Architecture Status". In: (October 2004), pp. 1–69.

[37] Fei Hu, Qi Hao, and Ke Bao. "A Survey on Software-Defined Network and OpenFlow: From Concept to Implementation". In: *IEEE Communications Surveys Tutorials* 16 (4 Jan. 2014), pp. 2181–2206. ISSN: 1553-877X. DOI: 10.1109/COMST.2014.2326417. URL: http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=6819788\nhttp://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=6819788http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=6819788.

[38] RFC 6163. "Framework for GMPLS and Path Computation Element (PCE) Control of Wavelength Switched Optical Networks (WSONs)". In: (2011), pp. 1–51.

[39] J. A. Farrel et al. "A Path Computation Element-Based Architecture, RFC 4655, IETF," in: (2006).

[40] J. L. Le Roux J. P. Vasseur. "A. Path Computation Element Communication Protocol, RFC 5440, IETF," in: (2009).

[41] Steven Gringeri, Nabil Bitar, and Tiejun J. Xia. "Extending software defined network principles to include optical transport". In: *IEEE Communications Magazine* 51.3 (2013), pp. 32–40. DOI: 10.1109/MCOM.2013.6476863.

[42] Lei Liu et al. "Experimental validation and performance evaluation of OpenFlow-based wavelength path control in transparent optical networks". In: *Optics Express* 19.27 (2011), p. 26578. ISSN: 1094-4087. DOI: 10.1364/OE.19.026578. URL: https://www.osapublishing.org/oe/abstract.cfm?uri=oe-19-27-26578.

[43] Lei Liu et al. "Field trial of an openflow-based unified control plane for multi-layer multigranularity optical switching networks". In: *Journal of Lightwave Technology* 31 (4 2013), pp. 506–514. ISSN: 07338724. DOI: 10.1109/JLT.2012.2212179.

[44] Mayur Channegowda et al. “Experimental Evaluation of Extended OpenFlow Deployment for High-Performance Optical Networks”. In: *European Conference and Exhibition on Optical Communication* (2012), Tu.1.D.2. DOI: 10.1364/ECEOC.2012.Tu.1.D.2. URL: https://www.osapublishing.org/abstract.cfm?uri=ECEOC-2012-Tu.1.D.2.

[45] Siamak Azodolmolky et al. “Integrated OpenFlow–GMPLS control plane: an overlay model for software defined packet over optical networks”. In: *Optics Express* 19.26 (2011), B421. DOI: 10.1364/oe.19.00b421.

[46] Lei Liu et al. “Experimental Demonstration of an OpenFlow/PCE Integrated Control Plane for IP over Translucent WSON with the Assistance of a Per-request-based Dynamic Topology Server”. In: *European Conference and Exhibition on Optical Communication*. OSA, 2012. DOI: 10.1364/eceoc.2012.tu.1.d.3.

[47] Ramon Casellas, Ricardo Mart, and Ricard Vilalta. “Applications and Status of Path Computation Elements(Invited)”. In: *J. Opt. Commun. Netw.* 5.10 (2013), A192–A203. DOI: 10.1364/JOCN.5.00A192. URL: http://opg.optica.org/jocn/abstract.cfm?URI=jocn-5-10-A192.

[48] Saurav Das et al. “Packet and circuit network convergence with openflow”. In: *Optics InfoBase Conference Papers* (2010), pp. 7–9. ISSN: 21622701. DOI: 10.1364/ofc.2010.otug1.

[49] Lei Liu et al. “Design and performance evaluation of an OpenFlow-based control plane for software-defined elastic optical networks with direct-detection optical OFDM (DDO-OFDM) transmission”. In: *Optics Express* 22.1 (2014), p. 30. ISSN: 1094-4087. DOI: 10.1364/oe.22.000030.

[50] Andrew Lake. “Open Transport Switch - A Software Defined Networking Architecture for Transport Networks”. In: *ACM SIGCOMM workshop on Hot topics in software defined networks* (2013). DOI: 10.1145/2491185.2491192.

[51] Reza Nejabati et al. “Toward a Completely Softwareized Optical Network [Invited]”. In: *Journal of Optical Communications and Networking* 7 (12 Dec. 2015), B222. ISSN: 1943-0620. DOI: 10.1364/JOCN.7.00B222. URL: https://www.osapublishing.org/abstract.cfm?URI=jocn-7-12-B222.

[52] Bob Lantz et al. “Demonstration of software-defined packet-optical network emulation with mininet-optical and ONOS”. In: *Optics InfoBase Conference Papers* Part F174- (2020), pp. 2020–2022. DOI: 10.1364/ofc.2020.m3z.9.

[53] Vishal. Shukla. *Introduction to Software Defined Networking - OpenFlow VxLAN*. 2013. ISBN: 978-1-48267-813-0.

[54] S. Oechsner et al. “Modeling and performance evaluation of an openflow architecture”. In: pp. 1–7.

[55] Nick McKeown et al. “OpenFlow: enabling innovation in campus networks”. In: *ACM SIGCOMM Computer Communication Review* 38 (2 Mar. 2008), p. 69. ISSN: 01464833. DOI: 10.1145/1355734.1355746. URL: http://portal.acm.org/citation.cfm?doid=1355734.1355746.

[56] Open Networking Foundation. “Software-Defined Networking: The New Norm for Networks [white paper]”. In: *ONF White Paper* (2012), pp. 1–12.

[57] Diego Kreutz et al. “Software-defined networking: A comprehensive survey”. In: *Proceedings of the IEEE* 103.1 (2015), pp. 14–76. ISSN: 00189219. DOI: 10.1109/JPROC.2014.2371999.

[58] Pakawat Pupatwibul et al. “Optimization of the openflow controller in wireless environments for enhancing mobility”. In: 2013.

[59] Ameen Banjar et al. “Developing an application based on openflow to enhance mobile IP networks.” In: 2013, pp. 936–940.

[60] Marc Mendonca et al. “A survey of software-defined networking: Past, present, and future of programmable networks. 2014.” In: (2014).

[61] Thomas D Nadeau and Ken Gray. *SDN: Software Defined Networks.* OâReilly Media, Inc, 2013.

[62] Siamak Azodolmolky. *Software Defined Networking with OpenFlow.* Packt Publishing, Limited, 2017. ISBN: 9781783984299.

[63] Paul Goransson and Chuck Black. *Software Defined Networks: A Comprehensive Approach.* second. Elsevier/Morgan Kaufmann, 2016, p. 436. ISBN: 9780128045558.

[64] Ameen Banjar et al. “Using an ICN Approach to Support Multiple Controllers in OpenFlow”. In: (2014).

[65] Brandon Heller. “OpenFlow Switch Specification 1.0.0”. In: *Current* 0 (2009), pp. 1–36. ISSN: 09226389. DOI: 10.1002/2014GB005021.

[66] Ameen Banjar et al. “Analysing the performance of the OpenFlow standard for software-defined networking using the OMNeT++ network simulator”. In: *2014 Asia-Pacific Conference on Computer Aided System Engineering (APCASE)*. IEEE, 2014. DOI: 10.1109/apcase.2014.6924467.

[67] Nicola Andriolli et al. “Optical networks management and control: A review and recent challenges”. In: *Optical Switching and Networking* 44 (2022), p. 100652. ISSN: 1573-4277. DOI: https://doi.org/10.1016/j.osn.2021.100652. URL: https://www.sciencedirect.com/science/article/pii/S1573427721000497.

[68] Lei Liu et al. “Field trial of an openflow-based unified control plane for multilayer multigranularity optical switching networks”. In: *Journal of Lightwave Technology* 31.4 (2013), pp. 506–514. ISSN: 07338724. DOI: 10.1109/JLT.2012.2212179.

[69] Jie Zhang et al. “Optical Network Virtualization”. In: *Springer Handbook of Optical Networks*. Ed. by Biswanath Mukherjee et al. Cham: Springer International Publishing, 2020, pp. 583–607. ISBN: 978-3-030-16250-4. DOI: 10.1007/978-3-030-16250-4_17. URL: https://doi.org/10.1007/978-3-030-16250-4_17.

[70] Tareq S. El-Bewab. *Optical Switching*. Ed. by Tarek S. El-Bawab. Boston, MA: Springer US, 2006. ISBN: 978-0-387-26141-6. DOI: 10.1007/0-387-29159-8. URL: http://link.springer.com/10.1007/0-387-29159-8.

[71] Sadia Qureshi and Robin Braun. “Dynamic Light Path Establishment In switch Fabric Using OpenFlow”. In: *2018 26th International Conference on Systems Engineering (ICSEng)*. 2018, pp. 1–4. DOI: 10.1109/ICSENG.2018.8638220.

[72] Partha Bhaumik et al. “Software-defined optical networks (SDONs): a survey”. In: *Photonic Network Communications* 28.1 (2014), pp. 4–18. ISSN: 1387-974X. DOI: 10.1007/s11107-014-0451-5. URL: http://link.springer.com/10.1007/s11107-014-0451-5.

[73] Saurav Das, Guru Parulkar, and Nick McKeown. “Unifying packet and circuit switched networks”. In: *2009 IEEE Globecom Workshops, Gc Workshops 2009* (2009). DOI: 10.1109/GLOCOMW.2009.5360777.

[74] Steven Gringeri, Nabil Bitar, and Tiejun J. Xia. “Extending software defined network principles to include optical transport”. In: *IEEE Communications Magazine* 51.3 (2013), pp. 32–40. ISSN: 0163-6804. DOI: 10.1109/MCOM.2013.6476863. URL: http://ieeexplore.ieee.org/document/6476863/.

[75] ONF Solution Brief. *OpenFlow-enabled Transport SDN*. 2014.

[76] Mahmoud Bahnasy, Karim Idoudi, and Halima Elbiaze. “OpenFlow and GMPLS Unified Control Planes: Testbed Implementation and Comparative Study”. In: *Journal of Optical Communications and Networking* 7.4 (2015), p. 301. ISSN: 1943-0620. DOI: 10.1364/JOCN.7.000301. URL: http://www.opticsinfobase.org/abstract.cfm?URI=jocn-7-4-301.

[77] Dimitrios Pindarakis and Subir Biswas. “Management and control of optical networks”. In: 2002, pp. 31–47.

[78] Haitham S. Hamza and Jitender S. Deogun. “Wavelength-exchanging cross connects (WEX) - A new class of photonic cross-connect architectures”. In: *Journal of Lightwave Technology* 24.3 (2006), pp. 1101–1111. ISSN: 07338724. DOI: 10.1109/JLT.2005.863279.

[79] Rodolfo Alvizu et al. “Comprehensive Survey on T-SDN: Software-Defined Networking for Transport Networks”. In: *IEEE Communications Surveys and Tutorials* 19.4 (2017), pp. 2232–2283. ISSN: 1553877X. DOI: 10.1109/COMST.2017.2715220.

[80] Daniel C Kilper. "Optical Physical Layer SDN [Invited]". In: 10.1 (2018), pp. 110–121. ISSN: 19430620. DOI: 10.1364/JOCN.10.00A110.

[81] Shinji Yamashita et al. "Extension of OpenFlow protocol to support optical transport network, and its implementation". In: *2015 IEEE Conference on Standards for Communications and Networking, CSCN 2015* (2016), pp. 263–268. DOI: 10.1109/CSCN.2015.7390455.

[82] Marc De Leenheer, Guru Parulkar, and Tom Tofigh. *SDN Control of Packet-over-Optical Networks*. 2015.

[83] Marc De Leenheer et al. "SDN Control of Optical Networks". In: *Ecoc* 1 (2017), pp. 1–3.

[84] John Kim, William J Dally, and Dennis Abts. "Flattened butterfly: a cost-efficient topology for high-radix networks". In: *Proceedings of the 34th annual international symposium on Computer architecture*. 2007, pp. 126–137.

[85] Chuanxiong Guo et al. "BCube: a high performance, server-centric network architecture for modular data centers". In: *Proceedings of the ACM SIGCOMM 2009 conference on Data communication*. 2009, pp. 63–74.

[86] William James Dally and Brian Patrick Towles. *Principles and practices of interconnection networks*. Elsevier, 2004.

[87] Hong Liu et al. "Evolving Requirements and Trends of Datacenters Networks". In: *Springer Handbook of Optical Networks*. Ed. by Biswanath Mukherjee et al. Cham: Springer International Publishing, 2020, pp. 707–724. ISBN: 978-3-030-16250-4. DOI: 10.1007/978-3-030-16250-4_21. URL: https://doi.org/10.1007/978-3-030-16250-4_21.

[88] Jacob H Cox et al. "Advancing Software-Defined Networks : A Survey". In: *IEEE Access* 5 (2017), pp. 25487–25526.

[89] Akhilesh S. Thyagaturu et al. "Software Defined Optical Networks (SDONs): A Comprehensive Survey". In: *IEEE Communications Surveys & Tutorials* 18.4 (2016), pp. 2738–2786. ISSN: 1553-877X. DOI: 10.1109/COMST.2016.2586999. arXiv: 1511.04376. URL: http://arxiv.org/abs/1511.04376http://ieeexplore.ieee.org/document/7503119/.

[90] Imrich Chlamtac and Gadi Karmi. "Lightpath Communications: An Approach to High Bandwidth Optical WAN's". In: *IEEE Transactions on Communications* 40.7 (1992), pp. 1171–1182. ISSN: 00906778. DOI: 10.1109/26.153361.

[91] Ajmal Muhammad. "Planning and Provisioning Strategies for Optical Core Networks". PhD thesis. 2015. ISBN: 9789175191157.

[92] Georgios I. Papadimitriou, Chrisoula Papazoglou, and Andreas S. Pomportsis. "Optical switching: Switch fabrics, techniques, and architectures". In: *Journal of Lightwave Technology* 21.2 (2003), pp. 384–405. ISSN: 07338724. DOI: 10.1109/JLT.2003.808766.

[93] Haitham S. Hamza and Jitender S. Deogun. "WDM optical interconnects: A balanced design approach". In: *IEEE/ACM Transactions on Networking* 15.6 (2007), pp. 1565–1578. ISSN: 10636692. DOI: 10.1109/TNET.2007.908151.

[94] Benjamin G. Lee and Nicolas Dupuis. "Silicon Photonic Switch Fabrics: Technology and Architecture". In: *Journal of Lightwave Technology* 37.1 (2019), pp. 6–20. ISSN: 07338724. DOI: 10.1109/JLT.2018.2876828.

[95] Wojciech Kabacinski. *NonBlocking Eectronic and photonic switching fabrics*. Springer, 2005. ISBN: 9780387254319.

[96] Francesco Testa and Lorenzo Pavesi. "Optical switching in next generation data centers". In: *Optical Switching in Next Generation Data Centers* (2017), pp. 1–336. DOI: 10.1007/978-3-319-61052-8.

[97] D. Fey et al. "Optical multiplexing techniques for photonic Clos networks in High Performance Computing Architectures". In: *Journal of Supercomputing* 62.2 (2012), pp. 620–632. ISSN: 09208542. DOI: 10.1007/s11227-010-0496-x.

[98] Konstantinos Christodoulopoulos, Konstantinos Manousakis, and Emmanouel Varvarigos. "Offline Routing and Wavelength Assignment in Transparent WDM Networks". In: *IEEE/ACM Transactions on Networking* 18.5 (2010), pp. 1557–1570. ISSN: 1063-6692. DOI: 10.1109/TNET.2010.2044585. URL: http://ieeexplore.ieee.org/document/5458043/.

[99] Sadia Qureshi and Robin M. Braun. "Dynamic LightPath Allocation in WDM Networks Using an SDN Controller". In: *IEEE Access* 9 (2021), pp. 148546–148557. DOI: 10.1109/access.2021.3124522.

[100] Radia Perlman. "An algorithm for distributed computation of a spanningtree in an extended LAN". In: *ACM SIGCOMM Computer Communication Review* 15.April (1985), pp. 44–53. DOI: 10.1145/318951.319004.

[101] "IEEE Standard for Local and metropolitan area networks: Media Access Control (MAC) Bridges". In: *IEEE Std 802.1D-2004 (Revision of IEEE Std 802.1D-1998)* (2004), pp. 1–281. DOI: 10.1109/IEEESTD.2004.94569.

[102] Mohit Chamania and Admela Jukan. *Springer Handbook of Optical Networks*. Ed. by Biswanath Mukherjee et al. Springer Handbooks. Cham: Springer International Publishing, 2020. Chap. Dynamic Co. ISBN: 978-3-030-16249-8. DOI: 10.1007/978-3-030-16250-4. URL: http://link.springer.com/10.1007/978-3-030-16250-4.

[103] Saurav Das, Guru Parulkar, and Nick McKeown. "Why OpenFlow/SDN can succeed where GMPLS failed". In: *European Conference on Optical Communication, ECOC* 1 (2012), pp. 31–33.

[104] D. Simeonidou, R. Nejabati, and M. P. Channegowda. "Software defined optical networks technology and infrastructure: Enabling software-defined optical network operations". In: *Optical Fiber Communication Conference, OFC 2013* 5.10 (2013), pp. 274–282. DOI: 10.1364/ofc.2013.oth1h.3.

[105] Saurav Das et al. "Packet and circuit network convergence with openflow". In: *Optics InfoBase Conference Papers* (2010), pp. 7–9. ISSN: 21622701. DOI: 10.1364/ofc.2010.otug1.

[106] R. Rejeb et al. "Management issues in transparent optical networks". In: *Proceedings of 2004 6th International Conference on Transparent Optical Networks* 1 (2004), pp. 248–254. DOI: 10.1109/icton.2004.1360286.

[107] Hui Yang et al. "Multipath protection for data center services in OpenFlow-based software defined elastic optical networks". In: *Optical Fiber Technology* 23 (2015), pp. 108–115. ISSN: 10685200. DOI: 10.1016/j.yofte.2015.03.002.

[108] H. Zang, J. P. Jue, and B. Mukherjee. "A review of routing and wavelength assignment approaches for wavelength- routed optical WDM networks". In: *Optical Networks Magazine* 1.January (2000), pp. 47–60. URL: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.127.5171{\&}rep=rep1{\&}type=pdf.

[109] Daniel F Macedo et al. "Programmable NetworksâFrom Software-Defined Radio to Software-Defined Networking". In: *IEEE Communications Surveys & Tutorials* 17.2 (2015), pp. 1102–1125. ISSN: 1553-877X. DOI: 10.1109/COMST.2015.2402617. URL: http://ieeexplore.ieee.org/document/7039225/.

[110] Zhaogang Shu et al. "Traffic Engineering in Software-Defined Networking: Measurement and Management". In: *IEEE Access* 4 (2016), pp. 3246–3256. ISSN: 21693536. DOI: 10.1109/ACCESS.2016.2582748.

[111] Qian Dong et al. "A Path Allocation Method Based on Source Routing in SDN Traffic Engineering". In: *Proceedings - 4th IEEE International Conference on Smart Cloud, SmartCloud 2019 and 3rd International Symposium on Reinforcement Learning, ISRL 2019* (2019), pp. 163–168. DOI: 10.1109/SmartCloud.2019.00037.

[112] Tianliang Zhang et al. "A WDM Network Controller with Real-Time Updates of the Physical Layer Abstraction". In: *Journal of Lightwave Technology* 37.16 (2019), pp. 4073–4080. ISSN: 15582213. DOI: 10.1109/JLT.2019.2924397.

[113] Ignacio Martin et al. "Machine Learning-Based Routing and Wavelength Assignment in Software-Defined Optical Networks". In: *IEEE Transactions on Network and Service Management* 16.3 (2019), pp. 871–883. ISSN: 1932-4537. DOI: 10.1109/TNSM.2019.2927867. URL: https://ieeexplore.ieee.org/document/8758853/.

[114] Dao Thanh Hai. "On the spectrum-efficiency of QoS-aware protection in elastic optical networks". In: *Optik* 202.October 2019 (2020), p. 163563. ISSN: 00304026. DOI: 10.1016/j.ijleo.2019.163563. URL: https://doi.org/10.1016/j.ijleo.2019.163563.

[115] Devi Chadda. *Software â Defined Optical Networks*. 2019, pp. 331–358. DOI: 10.1002/9781119393399.

[116] Lei Liu et al. "OpenSlice: An OpenFlow-based control plane for spectrum sliced elastic optical path networks". In: *European Conference on Optical Communication, ECOC* 21.4 (2012), pp. 4194–4204. ISSN: 1094-4087. DOI: 10.1364/oe.21.004194.

[117] Dao Thanh Hai, Le Hai Chau, and Nguyen Tan Hung. "A Priority-Based Multiobjective Design for Routing, Spectrum, and Network Coding Assignment Problem in Network-Coding-Enabled Elastic Optical Networks". In: *IEEE Systems Journal* 14.2 (2020), pp. 2358–2369. ISSN: 19379234. DOI: 10.1109/JSYST.2019.2938590.

[118] Ahmed E. Kamal and Mirzad Mohandespour. "Network coding-based protection". In: *Optical Switching and Networking* 11 (2014), pp. 189–201. ISSN: 1573-4277. DOI: https://doi.org/10.1016/j.osn.2013.06.006. URL: https://www.sciencedirect.com/science/article/pii/S1573427713000453.

[119] Deepa B. Swarna and V. Muthumanikandan. "Nested Failure Detection and Recovery in Software Defined Networks". In: *Proceedings of 2019 3rd IEEE International Conference on Electrical, Computer and Communication Technologies, ICECCT 2019* (2019). DOI: 10.1109/ICECCT.2019.8869085.

[120] Irena Seremet et al. "SDN as a Tool for Energy Saving". In: *27th Telecommunications Forum, TELFOR 2019* (2019), pp. 2019–2022. DOI: 10.1109/TELFOR48224.2019.8971086.

[121] Adriana Fernández-Fernández, Cristina Cervelló-Pastor, and Leonardo Ochoa-Aday. "Achieving energy efficiency: An energy-aware approach in SDN". In: *2016 IEEE Global Communications Conference, GLOBECOM 2016 - Proceedings* (2016). DOI: 10.1109/GLOCOM.2016.7841561.

[122] Mpho C. Nkosi et al. "Towards programmable on-demand lightpath services: Current state-of-the-art and open research areas". In: *IET Networks* 8.6 (2019), pp. 347–355. ISSN: 20474962. DOI: 10.1049/iet-net.2018.5040.

[123] Imad Alawe et al. "Integration of legacy Non-SDN optical ROADMs in a software defined network". In: *Proceedings - 2016 IEEE International Conference on Cloud Engineering Workshops, IC2EW 2016* (2016), pp. 60–64. DOI: 10.1109/IC2EW.2016.11.

[124] Philippos Isaia and Lin Guan. "Performance benchmarking of SDN experimental platforms". In: *2016 IEEE NetSoft Conference and Workshops (NetSoft)*. IEEE, 2016, pp. 116–120. ISBN: 978-1-4673-9486-4. DOI: 10.1109/NETSOFT.2016.7502456. URL: http://ieeexplore.ieee.org/document/7502456/.

[125] Arun K. Arahunashi, S. Neethu, and H. V. Ravish Aradhya. "Performance Analysis of Various SDN Controllers in Mininet Emulator". In: *2019 4th IEEE International Conference on Recent Trends on Electronics, Information, Communication and Technology, RTEICT 2019 - Proceedings* (2019), pp. 752–756. DOI: 10.1109/RTEICT46194.2019.9016693.

[126] Anh Nguyen-Ngoc et al. "Performance evaluation of selective flow monitoring in the ONOS controller". In: *2017 13th International Conference on Network and Service Management, CNSM 2017* 2018-Janua (2017), pp. 1–6. DOI: 10.23919/CNSM.2017.8256058.

[127] David Muelas, Javier Ramos, and Jorge E.Lopez De Vergara. "Assessing the Limits of Mininet-Based Environments for Network Experimentation". In: *IEEE Network* 32.6 (2018), pp. 168–176. ISSN: 1558156X. DOI: 10.1109/MNET.2018.1700277.

[128] Filippo Cugini and Piero Castoldi. "Software Defined Networking (SDN) in Optical Network". In: *Elastic Optical Networks*. Ed. by Víctor López and Luis Velasco. Vol. 56. Optical Networks. Cham: Springer International Publishing, 2016. Chap. 9, pp. 101–193. ISBN: 978-3-319-30173-0. DOI: 10.1007/978-3-319-30309-3_3. URL: http://link.springer.com/10.1007/978-3-319-30174-7http://link.springer.com/10.1007/978-3-319-30309-3{_}3.

Part IV.

Appendix

A. Code Overview

Source code for implementation of this project can be found at https://github.com/mininet-user/Resource-system.git, (username: mininet-user, password: asadia&123). Below is the overview of the main project file.

"app_code.py" is the main file that creates the GUI application. All other files/functions have been called within this code file. The package used for creating the GUI is "tkinter". Various buttons (e.g., "Enter topology name", "Access flow entries") are created on this application to implement the process of path finding through optical switch(es). Each button is binded to a command that will be executed on pressing respective button. The code also requires some input information from the user like "number of switches" and "number of wavelengths". Application needs to be provided information regarding the topology of the switches in the network. This has been provided in the form of csv. files e.g., "mac table" for the network and "initial status" of the ports/connections.

In order to demonstrate the behavior of an alarm received from the optical switch about a signal arrival and fetching source and destination hosts' information from this signal, we generate random source and destination port addresses. Further to make controller find path between these two hosts, we have used "scapy" tool to generate ping. The host switch can only be made to generate ping request if we use the code written in the bash file namely "m" to get connected to the host. This bash code has been taken from the example codes of Mininet.

If ping is successful, then by pressing the button "Access Flow entries" will run the function namely " runBash" which works to get to the path data inside OpenFlow switches. In order to understand and translate this path, we need the function called "ofctl_ parser".

After the path is found and translated, the actual network switches need to make port connections to establish that path, hence the button "configure switch". The switch that we used for real testing has JSON RPC to configure it. We have used the code "jsonrpc_request.py" to generate this request and configure the switch.

B. Pdf proof of included papers

B.1. ICSEng Publication

Dynamic Light Path Establishment In switch Fabric Using OpenFlow

Sadia Qureshi and Robin Braun
School of Electrical and Data Engineering
Faculty of Engineering and I.T, University of Technology Sydney

Abstract—

In today's optical networks, Light paths are established through Network Management System and Element Management System, which is a manual and cumbersome process. Paths are computed and pre-setup according to the known traffic demands, and provisioning any new service takes time. Several efforts have recently been made to make the path establishment process dynamic, including the Software Defined Networking approach. However all of this work has assumed fully interconnected fabrics of the network devices which is generally not the case. The task of end to end path establishment between network elements and through fabric are interrelated, and this task remains incomplete, without consideration of the switch's limitations. This work highlights for the first time the issue of path establishment dynamically through a switch fabric. The paper briefly explains the problem and suggests an SDN based solution using the OpenFlow protocol. This work contributes a represention of the basic framework which will be required to make the complete path setup process dynamic. The paper also includes an operational description.

Index Terms—

Optical switch, Software Defined Networks, Light path

I. Introduction

Rapidly growing demands for Internet bandwidth and applications have provided an impetus for the change in the current complex and inflexible way of controlling optical networks[1]. To cater today's dynamic demands, carrier operators seek a mechanism for the fast, simple and dynamic provisioning of bandwidth resources. Currently in the Core transport networks, a central Network Management System (NMS) is employed which is manual and inefficient. Operators at the Network Operation Center do the path planning offline and the network is then laid down according to that plan. Two main components of the optical networks are Optical Cross Connects (OXCs) and re-configurable optical add drop multiplexers (ROADMs) which may manually be configured to provide circuit-switched end-to-end Light paths. The connecting paths in successive switch's fabric, along with fiber transmission links between them, form the complete Light path across the network. The process for setting up a service may take several months and once done the connections may remain there for months or years. In cases where a new service demand has to be catered for, there may be a need to change the external fiber connections of the switch manually. Although computations are done prior, the service may still not be provided at times if no path is available through the switch's internal fabric. The switch has its own control mechanism which uses complex algorithms to find a path through the fabric and a setup of the physical cross connects. The tasks of setting up a path through the fabric, and transmission links are interrelated [2]. Thus at the network level a single control mechanism is necessary which can have a view of the entire network connections. For a new connection request, the controller should be able to find path dynamically and decide if the connection request has to be accepted or rejected. This method will avoid wasted NMS resources and time.

The Software Defined Networks (SDN) approach has been recently developed and is being researched for various networking fields. The idea of Software Defined Optical Networks is also not new and has been in discussion in the last few years. [3], [4], [5], [1]. However SDN has inherent characteristics for the packet networks while optical networks are circuit switched, the task is therefore challenging. A few efforts to mention are [6], [7], [8]. They deal with controlling paths through layer 0 i.e. the photonic (WDM) layer however they all assume fully interconnected fabrics, and the switches' limitations are neglected. OpenFlow protocol extensions have been proposed in these works but none of them have been tested in Mininet which is a popular SDN emulation platform. In our work we propose a way that a single SDN controller can have a view of the fabric connections together with the links between the switches using Mininet. The organization of the rest of the paper is as follows. Section II gives the background and briefly describes the current control mechanism of optical switches. Section III on the next page is about related work and section IV on page 3 explains our proposed solution. Finally the paper is concluded in section V on page 4.

II. Optical Network and Management

Optical network elements that need to be managed include optical line terminals (OLTs), ROADMs, optical amplifiers and optical cross connects. ROADMs and OXCs are intermediate nodes where the switching or routing occurs. NMS deal with

978-1-5386-7834-3/18/$31.00 ©2018 IEEE

the setting, releasing and keeping track of the path connections through the intermediate nodes. End-to-end routes through the network are computed and wavelengths for those paths are also assigned. NMS's are implemented through another lower layer of systems called Element Management Systems (EMSs). An EMS is connected to a domain of the network elements usually from the same vendor. It has a view of only its associated network elements. For the overall management, all EMSs in turn communicate with and report to the NMS. The EMS communicates to NEs through an existing traditional Data Communication Network using Simple Network Management Protocol or legacy protocol e.g. Transaction Language 1. In addition to the EMSs, a simplified local management system is usually provided to enable the operator to configure individual network elements manually. Timing diagram 1 explains the connection setup process in a conventional management system [9].

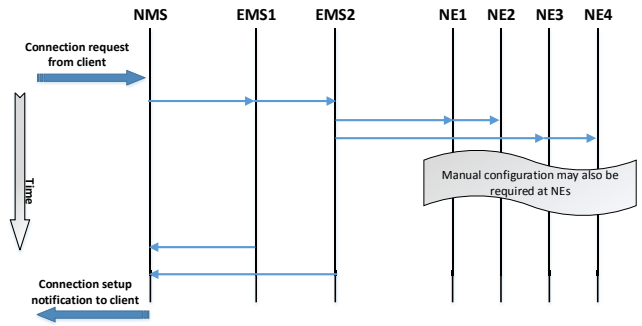


Fig. 1. Connection setup process through conventional NMS

A. Switch Fabric and Control Mechanism

Light paths in the Optical networks are switched at OXCs and ROADMs. Each switching device contains input modules/output modules, a switch fabric, and a control system. See figure 2. Light signals are received by input modules where control/signaling information are passed to the control system. Signals leaving the switching fabric are passed to output modules where control information is again appended. A lightpath originates at an electrical-to-optical (E/O) transmitter in the ingress node, and terminates at an optical-to-electrical (O/E) receiver in the egress node. Light signal may or may not be converted into electrical domain at intermediate nodes for switching purpose depending on the switch type, whether Optical-Electrical-Optical (OEO) or all optical (OOO). The switch fabric is usually a collection of optical domain switches purposed to switch an optical connection (wavelength) from an input port to any idle output port. Most of the optical space switches have been re-used from the rich collection of electronic multistage interconnection network architectures (MINs). Examples of these MINs include the optical crossbar architecture, the Benes and three-stage Clos networks [10]. Some of the switches are equipped with wavelength converters to reduce the blocking probability due to unavailability of wavelengths in wavelength routed networks.

The control system inside the switching device controls the

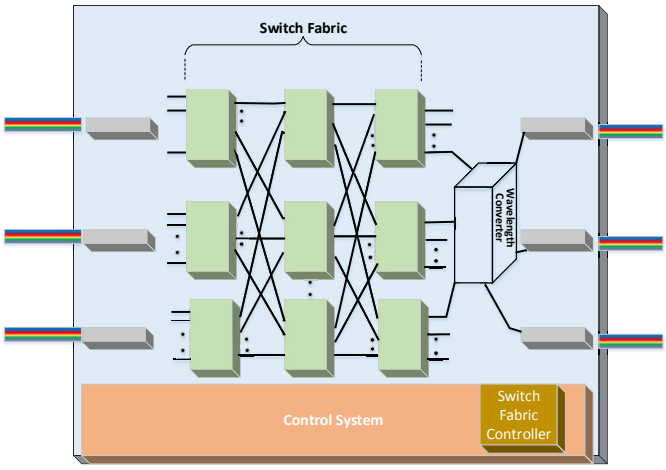


Fig. 2. General Optical Switch

switch fabric. It has routing tables which have information of the paths through the fabric and their statuses. When a new request from the manger is received, it sends commands to the controller to setup (release) the cross connects associated with the paths. Depending on the switch fabric design architecture, there may be multiple paths between any pair of input/output ports. Since there may or may not be any available path due to limited connectivity and paths being reserved for other requests, the controller uses complex control algorithms to find a connecting path in the switch fabric. The controller then issues the respective control signals to change the states of the physical switches, and the current state of the switch fabric is updated. The types and details of control algorithms can be found in [2].

III. Related Work

SDN has started to accommodate optical networks, and over the last decade there have been number of efforts to control Light paths using the OpenFlow protocol. A good survey of previous research work has been discussed in [11] and [12]. These research works propose OF-Extensions for dynamic circuit switching. In [8] the authors have proposed OpenFlow enabled optical switches which can get commands from the NOX controller to establish paths. However the request is generated from the IP router and the switch itself can not invoke requests for path setup. References [7], [13] develop an extended OF controller with an application to compute a Light path and implements it through an agent in the optical devices. Its not clear whether this work has considered the switch fabric's connectivity or not. [6] uses almost same concept and uses a Path Computation Engine (PCE) and a

Wavelength Assignment (WA) Algorithm in the controller for Light path computations while an agent is being used for communication between controller and switch. In this work, a port emulation entity is maintained which gets updates about port statuses but no information about fabric limitations. ONF Transport Working Group (ONFTWG) are working towards standardization of OpenFlow protocol for transport networks. The latest work [14] by a group of Fujitsu engineers proposed OF controller extensions which (with minor modifications) are accepted by ONFTWG in which port to port connections are done. However, the switch connectivity constraints are not discussed. ON.Lab is working towards control of converged packet and optical switches [15]. Recently in [16] they have presented control method using a dis-aggregated transport network.

IV. Proposed Solution

Previous research works assumed that the fabric is fully connected which is not generally the case. Thus even if the path setup instructions are given to the optical switch there may not be an available path through fabric, and the purpose of dynamic provisioning will fail. Also these works have been represented through experiments and no testing has been done in Mininet. Here we devise a mechanism where optical switch internal configurations are represented as a combination of regular layer two OpenFlow switches to the SDN controller. Here its also worth mentioning that although Open Flow Version 1.4 has some provision for optical ports, Mininet does not support them, so our proposed mechanism uses current the capabilities of Mininet for path establishment.

A. Structural Overview

Our proposed mechanism dynamically retrieves the switch's internal configuration and sets up the Light path in an on-demand manner when the optical signal arrives. Figure 3represents the overview of the proposed structure. The optical switch can be assumed to maintain its external connections through a topology discovery process, or a manually populated list. Neighbor Discovery is an issue still to be addressed as referenced by [15]. The resource agent may use SNMP/CLI/JSON to extract the switches' resources from the optical network, and all connectivity limitations. The information is then used to invoke a Mininet configuration which is the mirror of the underlying optical network and switch fabrics. To the controller this network is a regular layer two OF switches' network. The bigger Mininet network can be considered as a collection of small Mininet networks or OF Switches where each collection represents fabric connectivity of a device.

B. Network Graph Creation

We have assumed that the Optical switch has tables/data sheets of its external connections, either manually populated or through a topology discovery mechanism. Our resource agent mechanism can query the switch using SNMP/CLI/JSON to obtain the information about the switch's internal and external connections. After getting this information it invokes a Mininet network. The SDN controller can then start its conventional method of network discovery and normal OpenFlow process may start. Figure 4 on the following pagerepresents the network graph creation process.

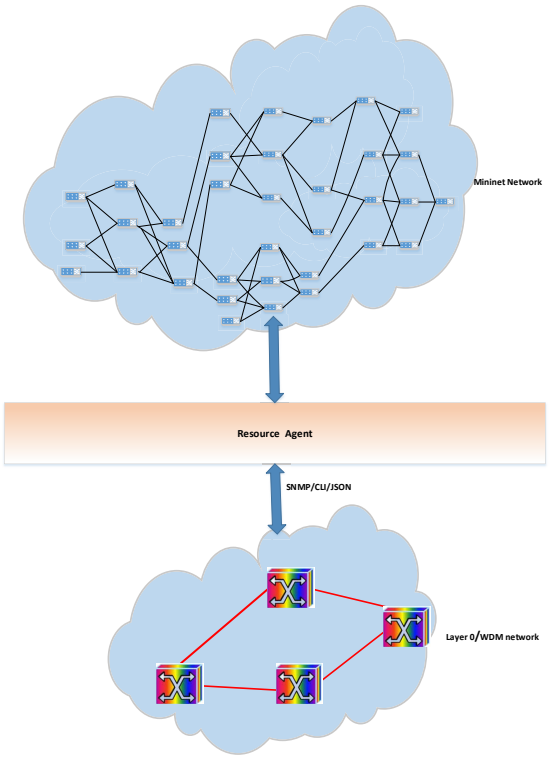


Fig. 3. Structural overview

C. Operational Flow

Once the controller creates the network graph, the network is ready for path setup operation. Figure 5 on the next pageexplains the flow of steps. On the arrival of optical signal, the information about source and destination addresses is extracted and forwarded to resource agent. Here the switch is assumed to have capability of header extraction and conversion to electrical form. The resource agent spoofs as a host and sends a frame to the port of the Mininet switch that corresponds to the optical switch's port on which signal has arrived. The switch checks its flow table entries for matching, if there is no matching then the switch sends the packet to the SDN controller. The controller computes the path and installs the flow entries in all switches. The resource agent based on the installed flow entries, can translate these into the commands

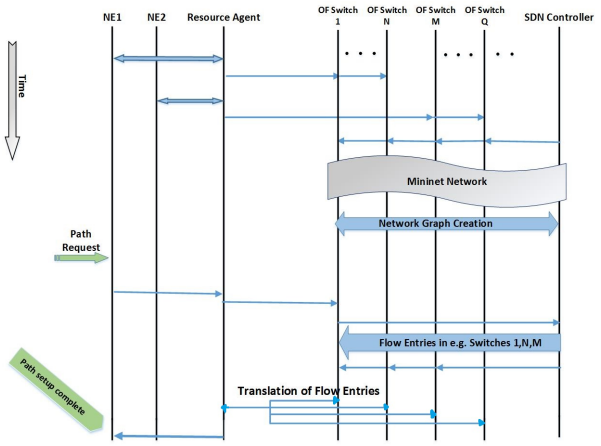


Fig. 4. Network Graph creation and connection setup

that can setup cross connects at the optical switch and establish Light paths. See figure 5 for the flow of steps.

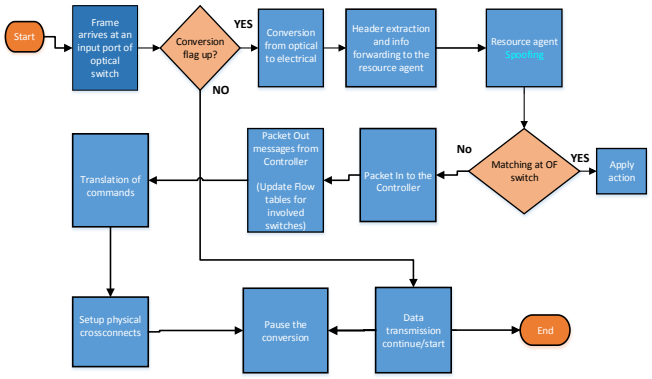


Fig. 5. Operational flow of steps

During the setup phase, the frames that arrive are lost. Obvious this lost needs to be minimized. However, as setups occur relatively seldom, this is not considered to be a problem. However it needs investigation.

V. Conclusions

Dynamic Light path establishment in WDM layer of core transport networks is an an important topic at the present time. However the aspect of a switch's limitation during path calculation has been neglected. In this paper we suggest a mechanism for complete end to end path establishment while taking into account the switch's fabric's connectivity. The mechanism is SDN based and physical layer connectivity is abstracted as a Mininet network. Our solution is in the process of implementation. The purpose of this paper is to introduce our ongoing work and direct the attention to the issue that switch fabric should also be considered while path planning for the WDM layer.

References

[1] P. Bhaumik, S. Zhang, P. Chowdhury, S.-S. Lee, J. H. Lee, and B. Mukherjee, "Software-defined optical networks (SDONs): a survey," *Photonic Network Communications*, vol. 28, no. 1, pp. 4–18, aug 2014. [Online]. Available: http://link.springer.com/10.1007/s11107-014-0451-5

[2] T. S. El-Bewab, *Optical Switching*, T. S. El-Bawab, Ed. Boston, MA: Springer US, 2006. [Online]. Available: http://link.springer.com/10.1007/0-387-29159-8

[3] S. Das, G. Parulkar, and N. McKeown, "Unifying packet and circuit switched networks," *2009 IEEE Globecom Workshops, Gc Workshops 2009*, 2009.

[4] S. Gringeri, N. Bitar, and T. J. Xia, "Extending software defined network principles to include optical transport," *IEEE Communications Magazine*, vol. 51, no. 3, pp. 32–40, mar 2013. [Online]. Available: http://ieeexplore.ieee.org/document/6476863/

[5] ONF Solution Brief, "OpenFlow-enabled Transport SDN," 2014.

[6] M. Bahnasy, K. Idoudi, and H. Elbiaze, "OpenFlow and GMPLS Unified Control Planes: Testbed Implementation and Comparative Study," *Journal of Optical Communications and Networking*, vol. 7, no. 4, p. 301, 2015. [Online]. Available: http://www.opticsinfobase.org/abstract.cfm?URI=jocn-7-4-301

[7] M. Channegowda, P. Kostecki, N. Efstathiou, S. Azodolmolky, R. Nejabati, P. Kaczmarek, A. Autenrieth, J.-P. Elbers, and D. Simeonidou, "Experimental Evaluation of Extended OpenFlow Deployment for High-Performance Optical Networks," *European Conference and Exhibition on Optical Communication*, p. Tu.1.D.2, 2012. [Online]. Available: https://www.osapublishing.org/abstract.cfm?uri=ECEOC-2012-Tu.1.D.2

[8] L. Liu, T. Tsuritani, I. Morita, H. Guo, and J. Wu, "Experimental validation and performance evaluation of OpenFlow-based wavelength path control in transparent optical networks," *Optics Express*, vol. 19, no. 27, p. 26578, dec 2011. [Online]. Available: https://www.osapublishing.org/oe/abstract.cfm?uri=oe-19-27-26578

[9] D. Pindarakis and S. Biswas, "Management and control of optical networks," 2002, pp. 31–47.

[10] H. S. Hamza and J. S. Deogun, "Wavelength-exchanging cross connects (WEX) - A new class of photonic cross-connect architectures," *Journal of Lightwave Technology*, vol. 24, no. 3, pp. 1101–1111, 2006.

[11] R. Alvizu, G. Maier, N. Kukreja, A. Pattavina, R. Morro, A. Capello, and C. Cavazzoni, "Comprehensive Survey on T-SDN: Software-Defined Networking for Transport Networks," *IEEE Communications Surveys and Tutorials*, vol. 19, no. 4, pp. 2232–2283, 2017.

[12] D. C. Kilper, "Optical Physical Layer SDN [Invited]," vol. 10, no. 1, pp. 110–121, 2018.

[13] D. E. Simeonidou, R. Nejabati, and M. Channegowda, "Software Defined Optical Networks Technology and Infrastructure: Enabling Software-Defined Optical Network Operations," *Optical Fiber Communication Conference/National Fiber Optic Engineers Conference 2013*, vol. 5, no. 10, p. OTh1H.3, 2013. [Online]. Available: https://www.osapublishing.org/abstract.cfm?uri=OFC-2013-OTh1H.3

[14] S. Yamashita, A. Yamada, K. Nakatsugawa, T. Soumiya, M. Miyabe, and T. Katagiri, "Extension of OpenFlow protocol to support optical transport network, and its implementation," *2015 IEEE Conference on Standards for Communications and Networking, CSCN 2015*, pp. 263–268, 2016.

[15] M. D. Leenheer, G. Parulkar, and T. Tofigh, "SDN Control of Packet-over-Optical Networks," 2015.

[16] M. D. Leenheer, Y. Higuchi, T. Furusawa, and G. Parulkar, "SDN Control of Optical Networks," *Ecoc*, no. 1, pp. 1–3, 2017.

B.2. ITNAC publication

Mininet Topology: Mirror of the Optical Switch Fabric

Sadia Qureshi and Robin Braun
School of Data and Electrical Engineering
University of Technology Sydney

***Abstract*—Software Defined Networks (SDN) is a new approach to change the conventional networking and is being researched in the various networking domains. To test and prototype SDN based concepts, a lightweight and closer to reality option is Mininet emulator. Mininet emulates SDN behavior by creating a virtual network of elements using Network Namespaces on a single Linux kernel machine. In this work, we have developed a Mininet topology that emulates the structure of a WDM Switch. The topology mirrors the paths that can be used by the wavelengths in a WDM switch fabric. The SDN controller can find a path for communication between hosts through this network. We simulated our Mininet topology, which mirrors an architecture for three wavelengths. The Ping command results show that only a set of hosts can be reached out by a particular host; which is the requirement of a WDM switch, and this verifies that Mininet topology is mapping a WDM switch.**

***Index Terms*—Optical switch, Mininet, Software defined Networks**

I. Introduction

Current conventional IP networks are complex and pose many operational and management challenges to the network administrators. IP networks have distributed control plane which is vertically coupled with the data plane within network devices. To express any particular network behavior, devices are to be configured individually and to deliver any new functionality or policy; re-configuration of the devices is required. IP networks are not flexible, and due to their static nature could not evolve with time. Software-defined networking (SDN), on the other hand, is a new approach to the architecture of computer networks that presents a change of conventional networking. It decouples the vertical integration of the network's control plane from the data-plane, putting the control in a central controller, which shall have the ability to program and manage the network dynamically.

SDN is an emerging paradigm that allows network administrators to deploy their policies in the controller in the form of Applications. The controller enforces those policies in the form of rules in the underlying switching hardware known as forwarding devices, using open interfaces. Depending on the principles set up by a controller application, the hardware device can behave like a router, Layer two switch or a Middlebox. An Open interface protocol, namely "OpenFlow," became a standard for communication between the controller and the switching devices. OpenFlow was introduced by N. McKeown et al.[1] as an academic activity at the Stanford University, USA. OpenFlow implements SDN and allows control and management of the network dynamically, which is a difficult task in traditional networks due to the various proprietary and closed interfaces.

SDN and OpenFlow are being researched in many domains, including academia and industry. Over the past few years, many industries have been considering SDN as a solution to their problems and are inclined towards SDN architectures. Some vendors of commercial switches now include OpenFlow API support in their switching hardware [2]. Various groups are also conducting several standardization activities, e.g., Open Networking Foundation (ONF) aims for promoting and standardizing SDN using open standards and is funded by different enterprises e.g., Google and Facebook [3].

Testing and prototyping new SDN and Openflow ideas with real devices is challenging and costly. One can think about a network of virtual machines (VMs); however, it has been experienced as a heavyweight solution [4]. These and other reasons, drive the researchers to use the emulator called Mininet. Mininet is a new rapid prototyping environment which offers lightweight virtualization with extensible CLI and API support. It is developed with attributes like flexibility and scalability and is also interactive and shareable, the characteristics which other prototyping environments are lacking. It provides a simple and easy way to the real systems because a working prototype in Mininet should require no changes to the code or layout when deploying on actual hardware [4].

Mininet has a GUI editor called miniedit. It is an experimental tool which helps to quickly create and run topologies and test them by changing the component's specifications. In this work, we have designed a topology in Miniedit which mirrors a specific type of optical switch fabric. This topology and SDN controller can help in path finding, which is a complex problem in Optical networks. We also simulated the topology with a CLI program with some custom input parameters. Our paper is organized as follows. Section II gives background of this work. It describes optical networks and the problem of static path finding. Section III briefly introduces Mininet. Section IV explains our model that we have made for optical switch fabric. Section V describes simulation scenario and also the result. At the end, in section VI conclusions are drawn.

II. Background

A. Optical Networks

Developments in the optical devices' technology led to the rise of the second generation of optical transport networks.

978-1-7281-3673-8/19/$31.00 ©2019 IEEE

It expanded the use of Wavelength Division Multiplexing (WDM) technique from transmission purposes to optical networking. One of the significant advantages achieved through optical networking is that some of the electronic switching and routing tasks are now incorporated into the optical part of the network. Thus the optical network layer serves a major role in a transport layer infrastructure.

Typically, Optical network structure spans over three main tiers, namely Access networks, Metropolitan Area Networks (MAN) and Core (backbone) networks as shown in the figure. Main differentiating attributes among tiers are the number of users, expected capacity, and geographic area. Core networks span over thousands of kilometers and are structured to interconnect various MANs on an international scale. They are mostly connected in a mesh topology and engage very high-speed optical equipment and transmission links [5]. Figure 1 is the depiction of a generic simplified optical WDM network.

The main components of WDM network are optical line terminals (OLTs), (reconfigurable) optical add/drop multiplexers ((R)OADMs), and optical cross-connects (OXCs)/Optical Circuit Switches (OCS). The function of the OLTs is to multiplex/demultiplex multiple wavelengths at the edges of network into a composite signal; and also to convert to a suitable wavelength to be transmitted over the fiber. ROADMs are required where some of the wavelengths need to be dropped locally, and others need to be switched to their destinations. OXCs are electronic devices and have a larger number of ports and wavelengths involved. They perform almost the same tasks, as ROADMs but on a much larger scale and are deployed in Core optical networks.

Using WDM technique, data is transmitted at different carrier wavelengths of light over a fiber. Each wavelength of light can carry a signal with its own speed and protocol, and is independent of what's on the other wavelengths hence can serve to transport SONET, Ethernet and IP packets simultaneously [6], see figure 1. Data enters the Core network via edge devices and is routed to its final destination by ROADMs and OXCs. Each wavelength traverses some hops before reaching its final destination providing optical circuits called Lightpaths. Light paths are end-to-end optical connections which require fixed allocation of bandwidth resources [7]. Optical circuit switched networks need lightpaths to be established between its source and destination before actual data transmission [8]. Lightpaths are set up and broken down according to the requirements of the users.

B. Optical Switch

Optical network switches exploits Wavelength Division Multiplexing technique to route and switch the Light signals. Sometimes a Light signal may also be required to change its wavelength along the route called "wavelength translation." This process takes place within the optical switch using wavelength converters. A generic architecture of an optical switch is shown in figure 2. Each OXC is expected to integrate input & output modules, a switch fabric, and a control system. The internal architecture of the switch may employ Optical-Electrical-Optical (OEO) or all-optical (OOO) technique for switching and routing purposes [9]. Incoming light is received on input modules. Signals get routed through the switch fabric and collected at the output modules and are processed to be transmitted on fibers.

The control system has information about the routing and different paths through the fabric and their statuses. When a new connection request is received, the control system uses algorithms to find a free connecting path in the switch fabric. After finding a path, it sends commands to a controller to setup/release the switch cells (crosspoints) in the structure associated with that path. Various standard and proprietary algorithms exist that can be used to set up switch fabrics. The controller generates the respective control signals to update states of the crosspoints, and then the state table of the switch fabric is updated. Depending on the switch fabric design and topology, there may not always be a free path. In such cases, service cannot be provided, and the request is blocked. Algorithms that perform these tasks are called control algorithms [10].

C. Switch Fabric

Optical switches can employ either OEO architecture or OOO. Both architectures have their own pros and cons, see [11]. In OEO architecture, optical signals are converted to electrical. After getting switched through the electronic fabric it is converted back to optical form while in OOO architecture signals remain in optical form. Due to different design principle, two different architectures are usually used for the internal fabric of the switch e.g. Wavelength Selective architecture is typicaly employed for OOO structure, while OEO archtictecture employs structures which come from extensive collection of electronic multistage interconnection networks (MINs)[12]. MINs contain many switch cells that are interconnected in stages to form a topology. Most widely studied and used switching fabric is based on Clos network topology [13] [10][14][15][12][16]. Figure 3 shows two types of architectures used as optical cross connects, Wavelength-Selective architecture and Clos architecture. There is difference in the operation of two architectures [17]. Wavelength selective architecture has a specific block for each wavelength and its structure makes sure that no same wavelength is received by the same multiplexer thus is useful for all-optical switching. Clos configuration can satisfy this wavelength constraint in two ways. It can either employ mechanism to do switching in electronic domain and then at output ports convert the wavelength to a suitable one using transponder (that is why, useful for OEO) or using a graph coloring algorithm for correct configuration of the wavelengths e.g. [18].

D. Path establishment in optical networks

Optical core networks are circuit-switched networks which require setting up Light paths before transmission starts and then remain set up until the service is diminished. Thus wavelengths are reserved for the duration of the service. A Lightpath has to have a common wavelength

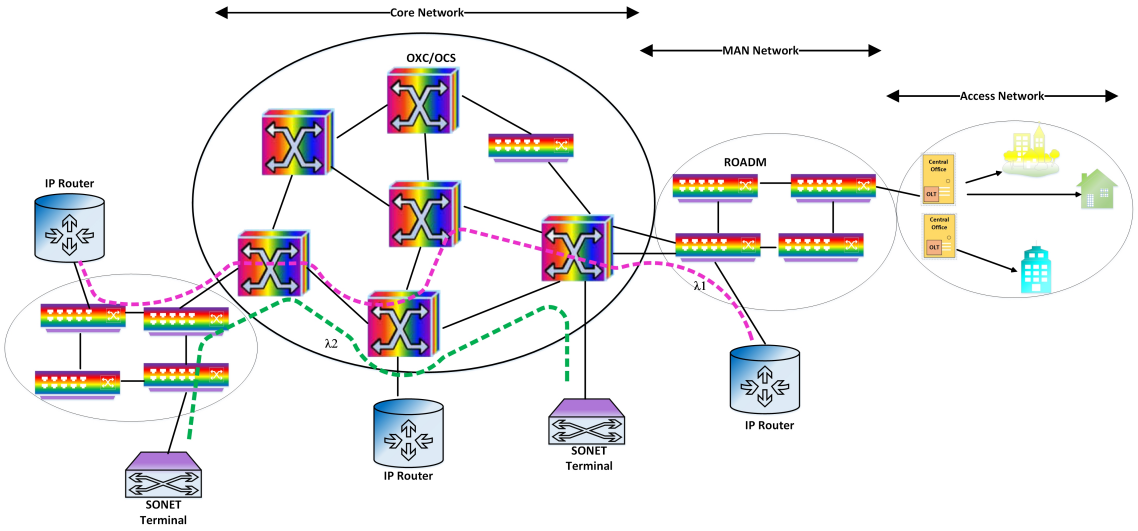


Fig. 1. General Optical Network

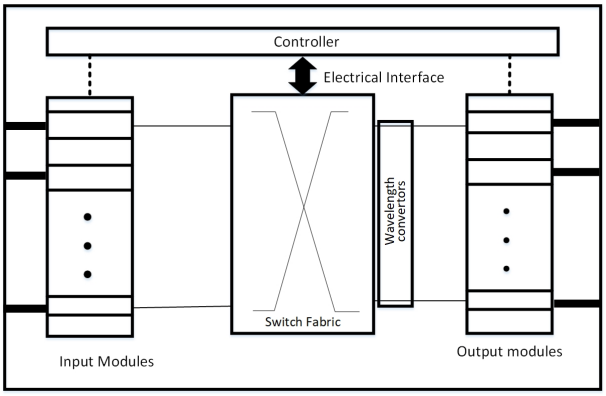


Fig. 2. Generic architecture of optical switch

along all the links it crosses, and also distinct wavelengths have to be assigned to the paths which share common connections. Therefore, planning for such wavelength-constrained networks is challenging. The algorithms and protocols that find paths and assign wavelengths should be efficient enough to optimize the utilization of network resources and minimizing the future blocking of light-paths. This problem is called the routing and wavelength assignment (RWA) problem.

The RWA problem is generally resolved for two different traffic conditions. When the traffic pattern is known in advance, it is referred to as offline or static RWA, while when the requests arrive at random and unknown times, is referred to as online or dynamic RWA. The offline RWA is done when planning the WDM network, while the online problem occurs when the network is operational [19]. Offline RWA and setting up/releasing of Lightpaths are done on network management level through "Network Management System." Switches are configured to allow circuit-switched paths using SNMP or any propriety protocol or CLI. Introducing a new service requires manual intervention and may take a few months. In online RWA, global or local network state information is necessary to compute paths. An edge router or a centralized node can work this out. Once paths are computed, efficient signaling and reservation protocols are required and invoked to establish paths, and if no resources are available, the request may be blocked. Signaling protocols are responsible for setting up lightpaths and exchange of control messages through the network [20]. Also the controller in the optical switch/OXC uses this control information to set-up the switch fabric [10]. The switch fabric is connected to an electrical control interface. The control interface allows electronic control signals to reach the switch fabric.

III. Mininet

Mininet is the network emulation system that provides lightweight virtualization to prototype and evaluate SDN protocols and applications on a single device. Mininet essentially creates a virtual network, running real kernel, switch and application code thus emulating real network. Network can be customized using the Mininet CLI. Mininet creates virtual networks and runs its elements (routers, switches, hosts, links) on a single Linux kernel using process-based virtualization and network namespaces. Virtual network devices in Mininet (hosts, switches, controllers) and links are the real things and they behave as like discrete hardware elements which are created using software rather than hardware. Mininet can be experimented by writing Python scripts. Miniedit is a GUI for Mininet which is an easier and quick way to make and test a topology.

- Mininet hosts: These are processes which behave same as real machines, packets get processed through them as if they sent through real Ethernet interface and device, with certain link speed and latency.
- Switches: Mininet has software-switches e.g. Open vSwitch or OpenFlow reference switch.
- Links: These are virtual Ethernet pairs, which provide connection between emulated devices.
- Controllers: Mininet has a default OpenFlow controller that implements a simple Ethernet learning

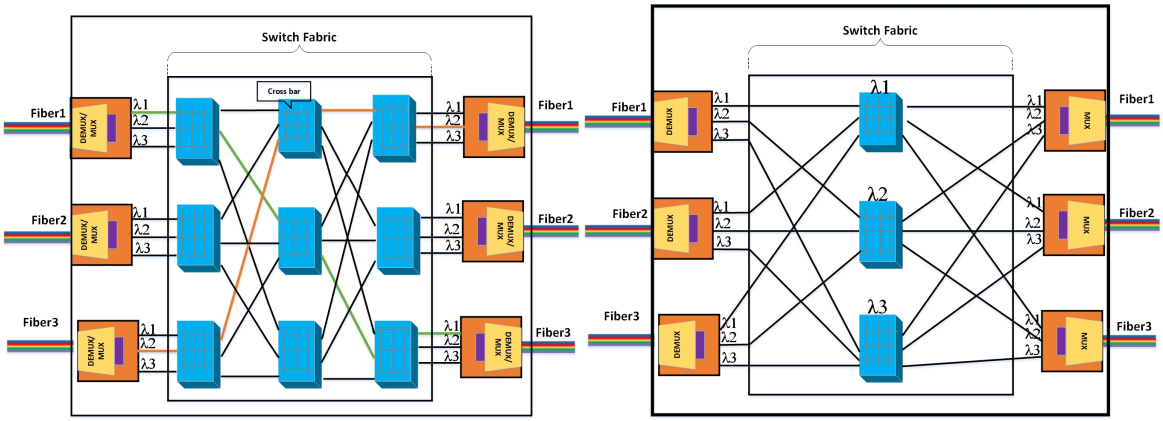


Fig. 3. Two optical switch fabric architectures (a) Clos architecture (b) Wavelength-Selective

switch. User can create his own controller & pass it into Mininet. (NOX, POX, FloodLight, Ryu)

- External OpenFlow Controllers: Mininet can also be connected to an external controller which exists somewhere else, for example on LAN or in another Virtual Machine.
- Routing: Using the OpenFlow protocol, switches can be programmed to do almost anything user wants with the packets that enter them.

IV. Mininet Model of Optical Switch Fabric

We have used Miniedit to model a topology which abstracts an optical switch fabric. We have selected a basic 3-stage Clos architecture with no wavelength convertors for the demonstration purpose. Here we have assumed three fibers with each having three wavelengths as in figure 3 (a). Clos architecture serves as a switching fabric, and any input wavelength can be switched to any output; however no same wavelengths can be received on the same Multiplexer as explained in section II-C. This wavelength constraint is mapped through the architecture of our Mininet topology. A Wavelength-selective architecture is constructed in Minedit to represent and function as a Clos structure.

In Miniedit structure, figure 4, OpenFlow switches S1-S9 represent demultiplexers for three input fibers each carrying three wavelengths λ_1, λ_2, λ_3 and switches S19-S21, S31-S33, S43-S45 represent Multiplexers for output fibers. Every wavelength from the source fiber, can reach to a block of switches connected in Clos configuration. For example, wavelength, λ_1 from the source Fiber1 and other fibers (Fiber2 and Fiber3) can only reach the block of switches represented in figure 4 as λ_1 and cannot reach the blocks specified as λ_2 and λ_3. Similarly wavelength, λ_2 from all source fibers can be switched through block λ_2 of Clos-configured switches represented in figure as λ_2. The same applies to wavelength λ_3. Total no of ports being used by OpenFlow switches is same as in original optical switch. due to this Wavelength-Selective architecture, only a particular input host can reach out to a particular output host. For example h1 can reach out to h4, h7, h10, h13 and h16. In the same way h2 can reach out to h5, h8, h11, h14 and h17. In this topology SDN controller finds path

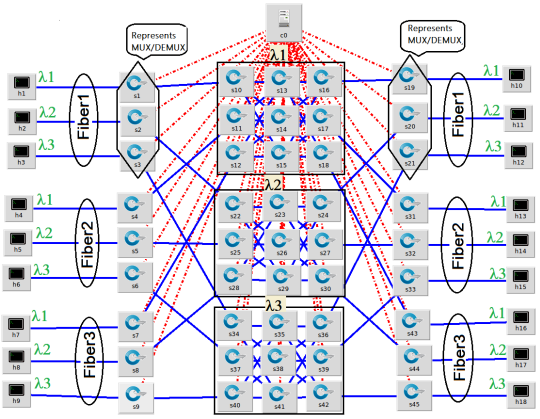


Fig. 4. Mininet topology mapped for a Clos-based Optical switch

between the source and destination hosts. Input and output hosts can be considerd as network devices connected to the optical switch, operating on a particular wavelength.

If middleware agent is developed which can translate layer one fabric structure to a Mininet topology and also provide a trigger (with destination address) on the arrival of light, the SDN controller can find the path through this topology for the destination address. The OpenFlow switches will receive the rules from the controller which can be translated back by the agent in the form of instructions for the optical switch to setup its fabric. To the SDN controller, the switches-network is an ordinary layer two network. The mechanism for such kind of dynamic path establishment in optical networks is explained in [21]. This mechanism will help in eliminating need for complex algorithms that are used to configure the switch fabrics. If all the devices get registered with the middleware system and their fabric pattern gets also translated into a mininet topology, the SDN controller can find path through the complete optical network i.e. within switch fabrics as well as links between devices. This is due the fact that the controller will now have a complete network graph for the whole network. In order to deal with fabrics other than Clos and having wavelength converters, the architecture of our mininet topology has to be changed.

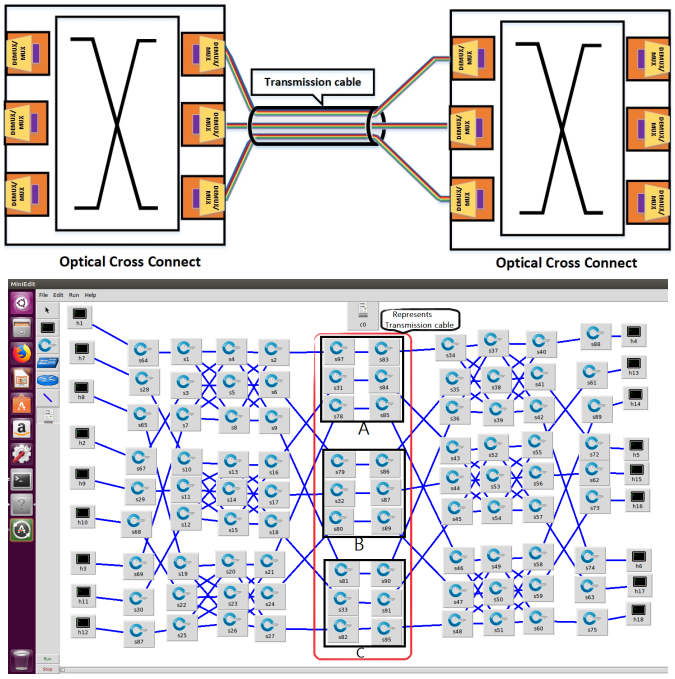


Fig. 5. Two optical switches cascaded (a) Actual cascaded switches (b) Miniedit representation of cascaded switches

V. Implementation & Results

Our sample model represents two optical switches (OXC/OCS) cascaded. Figure 5(a) shows two connected switches and its mapped mininet topology 5(b). Fibers connected at the output of every switch are collected in a transmission cable to span long distances between the devices deployed far apart. In figure 5 (b) the transmission cable is represented by OpenFlow cascaded switches within blocks A, B, C. However, technically the switches in the blocks connected in series can be replaced by single switches. Thus to avoid these extra switches, we have used a single switch for simulation purpose to represent the connection for one wavelength between devices. See figure 6.

Our system specifications were "Intel Core i7-7600U CPU @2.8GHz, 64-bit operating system with 16.0 GB installed memory". Mininet version 2.3 was installed on Ubuntu version 16.04 LTS. OpenFlow version 1.3 is used and switches in the topology are all "Open vSwitch Kernel Mode."

We used RYU as an SDN controller and launched its Spanning Tree Protocol application for OpenFlow version 1.3 to avoid loops for path calculation. We launch the controller first and then run the topology. Since the topology has many switches and loops, we needed to wait until all the switches are registered and paths are calculated. We then used ping command to check connectivity. Results were as expected. Host h1 (mapping $\lambda 1$ for Fiber1) can only reach h2, h3, h4, h5 and h6 see figure 7. Likewise host h7 (mapping λ_2 for Fiber2) can reach h9, h11, h13, h15 and h17. This verifies that, a controller having no knowledge about wavelengths, is able to find the correct path, and will not send any wavelength to the wrong port.

We also wrote a CLI program to generate this type of topology with custom no of wavelengths and number of stages for MIN structure. Writing a script with custom inputs is useful because the structure and input wavelengths may differ for real switches. General layout of our script is represented in Fragment 1. Simulated script also resulted with successful pings.

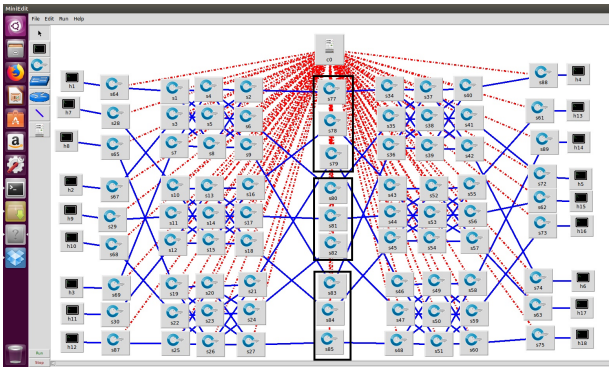


Fig. 6. Topology with mapped transmission cable

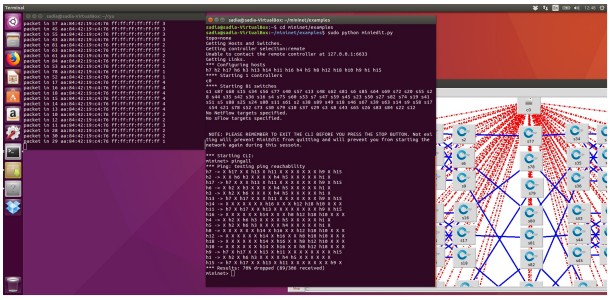


Fig. 7. Results for Ping command

Fragment 1 Generate Mininet Topology

```
Import modules                                  ▷ Import required modules e.g. mininet, Topo, controller
Define topology "class ClosTopo"                ▷ Define the pattern of switches and links
Add switches and links for received arguments   ▷ switches and links making clos topology
Add hosts                                       ▷ To check connectivity
Add switches representing transmission cable    ▷ connecting switches between OXCs/Roadms
procedure
    Pass Arguments (Input N,K,F,O)              ▷ Number of MIN stages, wavelengths, fibers and OXCs respectively
end procedure
procedure MAIN FUNCTION
    Initialize Controller                       ▷ Start controller
    Initialize Topology                         ▷ Define topology elements
    Start and build network                     ▷ Start switches and communication
    Start CLI                                   ▷ Open Mininet CLI
end procedure
```

VI. Conclusions

Path finding in optical networks is a crucial and complex problem. In this paper we have presented continuation of our work [21] which devises a mechanism for dynamic path finding in optical networks. We have developed a topology of OpenFlow switches in Mininet which replicates functionality of an optical switch fabric. We have demonstrated that an SDN controller, without knowing about the wavelengths, can be used to find paths through Clos switch fabric. We used ping command to check and verify that the topology is working as a Clos based switch fabric. The work in this paper supports the idea that if internal configuration of all the network devices is translated to mininet topology then SDN controller can be used for finding path through switch fabric as well as whole optical network using Mininet.

References

[1] N. McKeown, T. Anderson, H. Balakrishnan, G. Parulkar, L. Peterson, J. Rexford, S. Shenker, and J. Turner, "OpenFlow: enabling innovation in campus networks," *ACM SIGCOMM Computer Communication Review*, vol. 38, no. 2, p. 69, mar 2008. [Online]. Available: http://portal.acm.org/citation.cfm?doid=1355734.1355746

[2] J. H. Cox, J. Chung, S. Donovan, J. Ivey, R. J. Clark, G. Riley, H. L. Owen, and S. Member, "Advancing Software-Defined Networks : A Survey," *IEEE Access*, vol. 5, pp. 25 487–25 526, 2017.

[3] D. Kreutz, F. M. Ramos, P. E. Verissimo, C. E. Rothenberg, S. Azodolmolky, and S. Uhlig, "Software-defined networking: A comprehensive survey," *Proceedings of the IEEE*, vol. 103, no. 1, pp. 14–76, 2015.

[4] B. Lantz, B. Heller, and N. McKeown, "A network in a laptop:rapid prototyping for software-defined networks," in *Proceedings of the Ninth ACM SIGCOMM Workshop on Hot Topics in Networks - Hotnets '10*. New York, New York, USA: ACM Press, 2010, pp. 1–6. [Online]. Available: http://dl.acm.org/citation.cfm?id=1868466 http://portal.acm.org/citation.cfm?doid=1868447.1868466

[5] A. S. Thyagaturu, A. Mercian, M. P. McGarry, M. Reisslein, and W. Kellerer, "Software Defined Optical Networks (SDONs): A Comprehensive Survey," *IEEE Communications Surveys & Tutorials*, vol. 18, no. 4, pp. 2738–2786, jan 2016. [Online]. Available: http://arxiv.org/abs/1511.04376 http://ieeexplore.ieee.org/document/7503119/

[6] R. Ramaswami, K. Sivarajan, and K. Sasaki, *Optical Networks: A Practical Perspective*, 3rd ed. Elsevier, 2009. [Online]. Available: https://www.elsevier.com/.../optical-networks/.../978-0-12-374092-2

[7] I. Chlamtac and G. Karmi, "Lightpath Communications: An Approach to High Bandwidth Optical WAN's," *IEEE Transactions on Communications*, vol. 40, no. 7, pp. 1171–1182, 1992.

[8] A. Muhammad, "Planning and Provisioning Strategies for Optical Core Networks," Ph.D. dissertation, 2015.

[9] J. Simmons, *Optical Network Design and Planning*, ser. Optical Networks. Boston, MA: Springer US, 2008.

[10] T. S. El-Bewab, *Optical Switching*, T. S. El-Bawab, Ed. Boston, MA: Springer US, 2006. [Online]. Available: http://link.springer.com/10.1007/0-387-29159-8

[11] G. I. Papadimitriou, C. Papazoglou, and A. S. Pomportsis, "Optical switching: Switch fabrics, techniques, and architectures," *Journal of Lightwave Technology*, vol. 21, no. 2, pp. 384–405, 2003.

[12] H. S. Hamza and J. S. Deogun, "WDM optical interconnects: A balanced design approach," *IEEE/ACM Transactions on Networking*, vol. 15, no. 6, pp. 1565–1578, 2007.

[13] B. G. Lee and N. Dupuis, "Silicon Photonic Switch Fabrics: Technology and Architecture," *Journal of Lightwave Technology*, vol. 37, no. 1, pp. 6–20, 2019.

[14] W. Kabacinski, *NonBlocking Eectronic and photonic switching fabrics*. Springer, 2005.

[15] F. Testa and L. Pavesi, "Optical switching in next generation data centers," *Optical Switching in Next Generation Data Centers*, pp. 1–336, 2017.

[16] Q. Cheng, M. Bahadori, Y. H. Hung, Y. Huang, N. Abrams, and K. Bergman, "Scalable Microring-Based Silicon Clos Switch Fabric with Switch-and-Select Stages," *IEEE Journal of Selected Topics in Quantum Electronics*, vol. 25, no. 5, pp. 1–11, 2019.

[17] L. Wosinska, L. Thylen, and R. P. Holmstrom, "Large-capacity strictly nonblocking optical cross-connects based on microelectrooptomechanical systems (MEOMS) switch matrices: Reliability performance analysis," *Journal of Lightwave Technology*, vol. 19, no. 8, pp. 1065–1075, 2001.

[18] D. Fey, M. Schneider, J. Jahns, and H. Knuppertz, "Optical multiplexing techniques for photonic Clos networks in High Performance Computing Architectures," *Journal of Supercomputing*, vol. 62, no. 2, pp. 620–632, 2012.

[19] K. Christodoulopoulos, K. Manousakis, and E. Varvarigos, "Offline Routing and Wavelength Assignment in Transparent WDM Networks," *IEEE/ACM Transactions on Networking*, vol. 18, no. 5, pp. 1557–1570, oct 2010. [Online]. Available: http://ieeexplore.ieee.org/document/5458043/

[20] H. Zang, J. P. Jue, L. Sahasrabuddhe, R. Ramamurthy, and B. Mukherjee, "Dynamic lightpath establishment in wavelength-routed WDM networks," *IEEE Communications Magazine*, vol. 39, no. 9, pp. 100–108, 2001.

[21] S. Qureshi and R. Braun, "Dynamic Light Path Establishment In switch Fabric Using OpenFlow," in *2018 26th International Conference on Systems Engineering (ICSEng)*. IEEE, dec 2018, pp. 1–4. [Online]. Available: https://ieeexplore.ieee.org/document/8638220/

B.3. IEEE Access Publication

IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received October 5, 2021, accepted October 21, 2021, date of publication November 1, 2021, date of current version November 8, 2021.

Digital Object Identifier 10.1109/ACCESS.2021.3124522

Dynamic LightPath Allocation in WDM Networks Using an SDN Controller

SADIA QURESHI AND ROBIN M. BRAUN, (Life Senior Member, IEEE)
School of Electrical and Data Engineering, University of Technology Sydney, Ultimo, NSW 2007, Australia

Corresponding author: Sadia Qureshi (sadia.qureshi@student.uts.edu.au)

This work was supported by the Commonwealth of Australia.

ABSTRACT Core wavelength division multiplexed (WDM) networks are widely used to provide fixed physical connectivity and bandwidth to the logically connected upper electronic layer devices using optical signals. However, growing demands for bandwidth-intensive applications and cloud-based services push optical networks carriers' to provide scalable and flexible services dynamically. Software defined networking (SDN) has the potential to program electronic layers by dynamically controlling and managing network resources using SDN controller applications. SDN's on-demand characteristics combined with the optical circuit-switching can enable optical network service providers to customize their service provisioning dynamically to the user's requirements. They enable fast provision of new services, and minimize under-utilization of resources. In this paper, a model is proposed to bring the dynamic allocation of resources which is a layer 2+ functionality, to the WDM layer using SDN. A middle-ware application based on SDN and OpenFlow for dynamic switching and provisioning of optical service is presented. The application abstracts the optical layer's connectivity, also accounting for the switching constraints. Details of the model's implementation are discussed considering classically used equipment and its performance in terms of CPU and memory utilization, topology emulation time, and latency is evaluated. Finally, the application is tested with a Cisco layer one switch. Performance results show that the latency doubles when increasing the number of fibers of an optical cross connect from 5 to 7 and keeping wavelengths equal to 8, with Clos fabric topology.

INDEX TERMS Wavelength division multiplexed network (WDM), software defined networking (SDN), OpenFlow protocol, dynamic path allocation, optical cross connect (OXC).

I. INTRODUCTION

The traditional purpose of the circuit-switched long haul or core wavelength division multiplexed (WDM) networks is to provide high bandwidth physical network connectivity to the upper packet-switched electronic layers (e.g., Ethernet, IP). The end-to-end physical path between these packet-switched devices (routers/switches) is pre-planned to provide static wavelength routes through the optical devices. Pre-planning is done using a tool called Network Management System (NMS), and the optical devices are then manually configured to reserve paths and resources according to that computed plan. Since resources are pre-allocated, it is almost impossible to change them dynamically to accommodate the instantaneous demands of the user. Such networks are thus difficult to automate.

The associate editor coordinating the review of this manuscript and approving it for publication was Resul Das.

In WDM networks, although paths are pre-computed, a connection may still not be available at times due to all paths within the optical switch's internal fabric being busy. WDM devices have their own control system that uses complicated algorithms to find an available path through the inter-connections in the fabric. Since, setting up of the path through the switch's fabric is related to the path establishment between end-to-end communication devices [1], a single path computation entity is required which has global information of network paths and all possible inter-connections. Moreover, the conventional NMS poses problems for path planning in multi-carrier and multi-vendor environments. Thus, need arises for a flexible control mechanism that may compute and provide the path resources dynamically when requested.

Recently emerged, software-defined networking (SDN) architecture and OpenFlow (OF) protocol [2], [3], work in an on-demand fashion, and hence has the inherent potential

to resolve above mentioned issues. However, SDN has been designed for packet-switched networks while WDM networks are circuit-switched, further explained in Section II. The idea of extending SDN to optical networks has been discussed in previous works [4]–[8]. However, nothing could be standardized yet because of the complex nature of the physical layer issues, starting from accounting for the physical impairments to the vendor-specific configuration mechanisms for optical paths. A significant advantage of extending SDN to photonic networks is the combined dynamic service provisioning and optimization by the SDN controller because of its global knowledge of several network layers. A recent work [9], provides an SDN-controlled platform to simulate end-to-end integrated data plane of packet and optical network. The physical behaviour of a WDM network is simulated using analytical models and an optical data plane is emulated as data packets with additional information about wavelength. The platform can be used to configure network elements in a customised manner; however, switching constraints are not considered in this model. More advantages of using an SDN controller in optical networks are explained in Section III.

In literature, there have been some efforts that deal with establishing the wavelength paths through the WDM layer using SDN, e.g., [10]–[12]. However, all of these have assumed fully interconnected switch fabrics and the switching constraints are over-sighted. If the switch's internal fabric is busy and there is no available path through the fabric, then user's service provision request cannot be completed dynamically. Furthermore, none of the works have exploited the core potential of the OF protocol, i.e., dynamic resource allocation in response of a request, by generating the path request initiation from the WDM device itself. Either an IP router in the upper electronic layer is used or an agent-based approach is applied for request initiation purpose, further details of which are discussed in Section IV. Agent-based approach requires both the controller and the OF protocol to be extended. Our work is an effort to design a middle-ware mechanism that can be used with any SDN controller and OF protocol versions without extensions, while taking into account the switch's internal limitations and which dynamically accomplishes the service request and allocation of the resources.

In this work, we propose a model in which a single SDN controller can have a global view of all the possible paths within the switch together with the end-to-end links between the devices. We use a popular emulator namely ''Mininet'', further discussed in Section V. The SDN controller can have knowledge of all the available paths and can dynamically provision a connection when a request arrives. Our work is based on emulating the network connectivity to the SDN controller as a network of layer two OpenFlow switches, using Mininet. The controller thus does not have any knowledge of the underlying optical switches nor the wavelengths. Therefore no extensions of the SDN controller as well as OpenFlow protocol are required. A middle-ware application is developed to do all the translations between the controller and the optical network. Details of the model's implementation and tests are provided in Sections VI and VII respectively. Section VIII describes the creation and mapping of layer one connectivity to the controller. Section IX reports the performance results while Section X provides discussion about the graphs. The paper summarizes with conclusions and future work in Section XI.

II. BACKGROUND

SDN's potential in the packet domain has already been proven, initially by the researchers and now by the fact that some major companies are providing services and products based on SDN architecture and technology.[1] For optical networks, SDN has been suggested to be advantageous over existing control plane solutions in many aspects, see e.g., [13]–[15]. There have been efforts to extend SDN to optical networks; however, for multilayer convergence and control, layer one (electronic switching) as well as layer zero, i.e., wavelength switching layer should also be included in the SDN umbrella. Extending SDN to WDM layer has several challenges; though, from the physical characteristics of the communication signals to how the resources are allocated in vendors' proprietary equipment.

A. PACKET AND CIRCUIT SWITCHED NETWORKS

There is an essential and fundamental difference between circuit-switched and packet-switched networks, i.e., a packet-switched network do not pre-allocate resources before the first data flows. Instead, it allocates resources on-demand as datagrams arrive. On the other hand, circuit-switched systems pre-allocate resources before the first data flows. Those resources can then only be used for that instance of data flow. Optical transport networks are circuit-switched networks, which use light signals for faster delivery of packets from the source to the destination ports of layer two or layer 3 devices as shown in Figure 1. LightPaths are pre-computed and established by the management system and provisioning a new optical connection involves operator intervention. These mechanisms allocate the entire wavelength to a single connection, and the connections remain 'on' even when they are not utilized fully.

B. PATH ESTABLISHMENT IN OPTICAL WDM NETWORKS

Todays' commercial long-haul or core optical transport networks are implemented in a mesh topology with path planning and management done through a centralized network management system (NMS). NMS employs multiple element management systems (EMSs) to manage the whole network [1]. Usually, a single vendor's network elements are managed by a separate EMS. The EMS can have a view of only its attached network devices and does not have the overall network view. Therefore, for the management

[1] http://www.sdxcentral.com/directory/

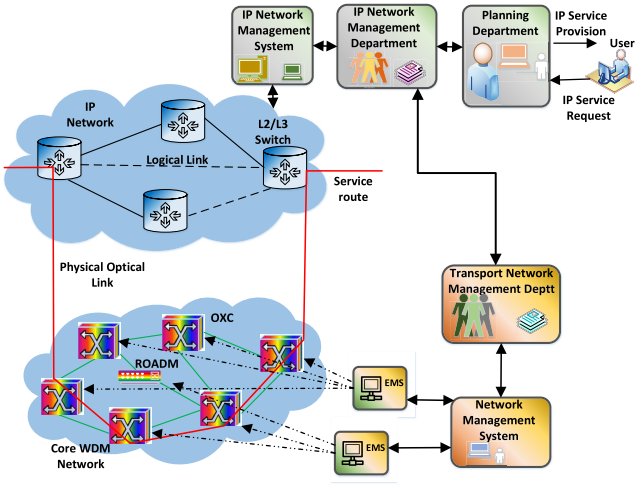


FIGURE 1. Overview of IP and optical layer management system.

of whole network, all EMSs, are supposed to connect and report to the NMS. The NMS then manages all types of network elements, see Figure 1. Additionally, a simple device management system is needed to enable the craftspeople and other technical staff to configure and manage individual network equipment.

Two main components of a WDM network are reconfigurable optical add/drop multiplexers (ROADMs), and optical cross-connects (OXCs). ROADM is a wavelength switching device which is required when some of the wavelengths need to be dropped or added locally in the network while others need to pass through to their destinations. OXCs are designed to perform almost same tasks as ROADMs but they have larger number of ports and wavelengths involved and are deployed in core optical networks. The OXC's core is its switch fabric. The switch fabric is a combination of switching cells which are arranged in the form of a either a crossbar matrix or multistage interconnections (MIN). The fabric usually exists in one of the popular form of MIN topologies, e.g., Clos, Benes, Banyan, or Tree. When a new connection is to be made, the local switch controller has to find an available connecting path in the switch fabric and issue respective electronic control signals to set up switching cells. Algorithms performing such tasks are referred as control algorithms [1], [16]. The circuit-switched connection between WDM networks' client terminals is called LightPath and the process of finding the path and assigning a wavelength is designated as Routing and Wavelength Assignment (RWA) problem [17], [18]. Connection management tasks, e.g., setting/releasing and monitoring LightPath connections in a network is done by NMS and EMS.

For a new service provision, the user needs to request to the "Sales and Planning" department of the IP network. The Sales department puts up a request to the management department, which collaborates with the management of transport network and works out the feasibility of service provision, see Figure 1. As a result, it may take several months for a service provider to supply a new service in response to a user's new connection request. The span of these connections is generally for years. With the evolution of optical networks, connection demands are becoming more dynamic and networks are turning more extensive and complex. Carriers prefer to provide connections to their customers quickly and not hold them for a long time. The current mechanism of LightPath management has proven to be very complex and slow. It involves human labor, time, and is error-prone [8]. To a certain extent, dynamic connection provisioning is achieved by the wavelength switching and routing at optical switches; however, more research, technology and new methods, are required in this domain.

C. SDN AND MININET

SDN paradigm separates the control plane from the forwarding plane of a traditional network and brings the intelligence in a logically centralized controller that keeps the global network information and does strategic changes in routing and switching devices dynamically, based on different requirements, e.g., quality of service (QoS) or traffic engineering. The OpenFlow protocol, first introduced by N. McKeown *et al.* implements SDN and is a communication standard between the controller and the forwarding devices. For testing SDN solutions, an emulator called "Mininet" is mostly used which is a rapid prototyping environment that offers lightweight virtualization and flexibility. For further understanding, the working, benefits, limitations, and features of SDN, OpenFlow and Mininet, the reader is referred to [19]–[21].

III. ADVANTAGES

Employing SDN controller for dynamic control of optical networks brings many advantages with it, some of which are discussed below.

A. MULTILAYER CONVERGENCE

SDN has already shown its benefits in layers 2-4. A considerable benefit of encompassing optical networks with SDN will be the combined dynamic provision of services as well as optimization of multiple network layers. A central SDN controller based on its global view of all the layers can play a role in optimal resource allocation. If a service needs on-demand connectivity, the SDN controller can calculate an optimum path through multiple layers and can dynamically establish the connection, see e.g., [22].

B. DYNAMIC CUSTOMIZED PROVISION OF SERVICES

Provisioning of dynamic on-demand bandwidth pipes between end-to-end systems can benefit economically and improve network performance by dynamically giving preference to the customer's preferred service(s), based on instantaneous network status, e.g., see a use-case by [6].

C. TRAFFIC ENGINEERING

In SDN, the controller has a bird's-eye view of the network's resources and can improve its performance by providing effective traffic engineering solutions. The SDN controller can choose paths in order to balance the load on various links and devices in the network where multiple alternate paths are available, see examples in [23] and [24].

D. OPTIMUM PATH CALCULATION

Centralized path computation has always been known to be more effective when compared to the traditional distributed path computation. Combining this advantage of SDN with others, e.g. taking into account the physical layer impairments, can improve end-to-end optimum path calculation, see e.g., [25]. Layer one transport networks also face another major issue of RWA. RWA can be handled better by incorporating contemporary RWA algorithms within path computation engine of SDN controller, see e.g., [26].

E. CONTROL OF ELASTIC OPTICAL NETWORKS (EONs)

To meet the highly dynamic traffic demands of the future, an elastic optical network has been regarded as a promising paradigm, which can offer a flexible data rate/channel allocation and high resource efficiency. The main purpose of EONs is to replace the fixed spectrum spacing with a flexible grid where an optical channel can span a smaller grid size for low data rates, whereas multiple slots to accommodate higher bit rates, e.g., 400 Gbps, thus supporting various data rates dynamically in a spectrum-efficient manner [27]. Technology advances associated with EON nodes have been identified as flexible grid add/drop and switching (essentially Bandwidth Variable Transponders and flexible OXCs), as well as spectrum conversion. To harness such a flexible network, a programmable control plane technique is required to program network functions dynamically according to the applications. However, this isn't easy to implement, as the control and data planes are currently integrated within the network nodes. Newly emerging software-defined optical networking (SDON) paradigm has a separated programmable control and data plane (DP) in which network functions and protocols are manageable according to the user's demand and applications [28]. The first feasibility and validation check of an OpenFlow-based control plane for an EONs has been presented in [29]. Further studies regarding software-defined elastic optical networks are reported in [30] and [28].

F. PROTECTION AND RESTORATION

A well-established key feature of fixed-grid optical core networks is their protection and agile restoration when link failures occur. The critical factor that needs to be considered while planning protection schemes is a trade-off between redundant capacity and restoration time [31]. Most popular protection schemes are based on dedicated path protection; however, research studies are exploring the ways to improve the performance of these techniques for fixed-grid and EONs. Network-coding-based protection [32] is one of such proposed techniques; see for example [31]. With the paradigm shift to SDON, the protection and restoration tasks are performed by a centralized controller, which can perform optimum computations based on the global map of the network. The reader can find references to various studies with a focus on SDN-based protection and recovery in [33].

G. OPPORTUNITY FOR NEW BUSINESS MODELS

SDN's programmability and flexibility have opened up opportunities for the network carriers' to try new business models. Network operators can investigate, test, and implement new models using the controller to optimize optical network's cost and performance, e.g., better bandwidth provision or energy effectiveness, see examples in [34] and [35].

IV. RELEVANT WORK

SDONs are being investigated and studied at the industrial as well as academic level. However, these investigations have varied scales and aim for different objectives. A good survey of previous research efforts is available in [7], [8], [36]–[38]. In this section, we will discuss the works that are explicitly done at the WDM layer.

In [12], the authors propose to use IP router to invoke the request to the SDN controller for path establishment. The work proposes OpenFlow-enabled optical switches which can understand commands from the extended-NOX controller. The optical switch interfaces are mapped using virtual Ethernet interfaces 'veths' of an OF switch; however, internal switching constraints are invisible to the controller. Also, the optical switch is incapable of generating the request itself, and thus full potential of the dynamicity of the OpenFlow protocol is not exploited.

References [11] and [15] have used an agent-based approach and developed an extended controller with an application to compute LightPaths. It is not clear how the optical devices' constraints are considered by the resource model and their way of abstracting the devices' hardware requires them to use an extended OpenFlow protocol and an extended controller. Reference [10] uses the same agent-based concept and utilises a Path Computation Engine (PCE) and a Wavelength Assignment (WA) algorithm in the controller for computing LightPaths. The authors use a port emulation entity which maintains and receives updates about port status and is used to calculate LightPath but no data is given for the switch fabric's limitations. They have also used an extended OF protocol.

Reference [39] uses a proxy application to control a WDM network device with the SDN controller. The authors have considered an end-to-end connection between devices as an optical link and used Simple Network Management Protocol (SNMP) to get the features of the switch; hence, the internal fabric connectivity is hidden from the controller. The work is conducted for integrating ROADMS, much simpler

than OXCs, which are the main operating devices in the optical core networks.

The Open Networking Foundation Transport Working Group (ONFTWG) is a team, working towards extending SDN and OpenFlow protocol for transport networks. Some initial efforts are being made by ONFTWG as well as other emerging organizations for standardization. In [40] and [41], authors have presented an ONOS control method using a dis-aggregated transport network.

V. PROPOSED DESIGN

The idea of parameterizing the WDM layer as a layer two system which allocates wavelength paths in an on-demand fashion, is presented in this paper. The proposed framework demonstrates a way to abstract layer one network connectivity so that it can be used in a layer two paradigm. The abstraction also includes the internal connectivity of the switch fabrics of OXCs and ROADMs and hence incorporating switching constraints as discussed in [8]. Switching constraints are imposed by the topology of the switch fabric, which is often chosen to reduce the cost and may result in congestion and hence non availability of the path through the fabric. This leads to declining a service provision.

Our model uses a central SDN controller and the emulated connectivity of layer one to control the end-to-end network connections. Incorporating the connectivity of switch fabrics helps the SDN controller to make ''Network Graph''. OXCs use different control algorithms to find paths through their switch fabrics. Algorithm for one fabric, may not work for others. Abstracting the connectivity eliminates the need for different equipment-specific algorithms, and one algorithm can be used to find end-to-end path in different classes of equipment.

In WDM optical networks, another limitation is ''Wavelength Continuity Constraint'' (WCC), which is catered in a component of our model called Resource Management System (RMS). The RMS component serves as a portal or application that maintains and manages virtual abstraction of paths of an optical network, so that customers are served dynamically as per their needs. With dynamic path establishment using SDN controller, the requirement for the always-on connections can be eliminated thus avoiding wastage of resources, when not being used. Our way of abstraction does not need to use any extended controller nor any extensions to the OF protocol.

Figure 2 represents the overview of the proposed design. RMS is used to parameterize the layer one optical network to the SDN controller, as a network of layer two OF switches. The WDM link is emulated as a combination of Ethernet-linked interfaces, each representing an individual wavelength. The controller is used to find a path from source to destination through this network. The RMS uses the path information in the emulated network and sends commands to Layer one devices to set up the physical path through the optical network. The proposed model assumes the invocation of

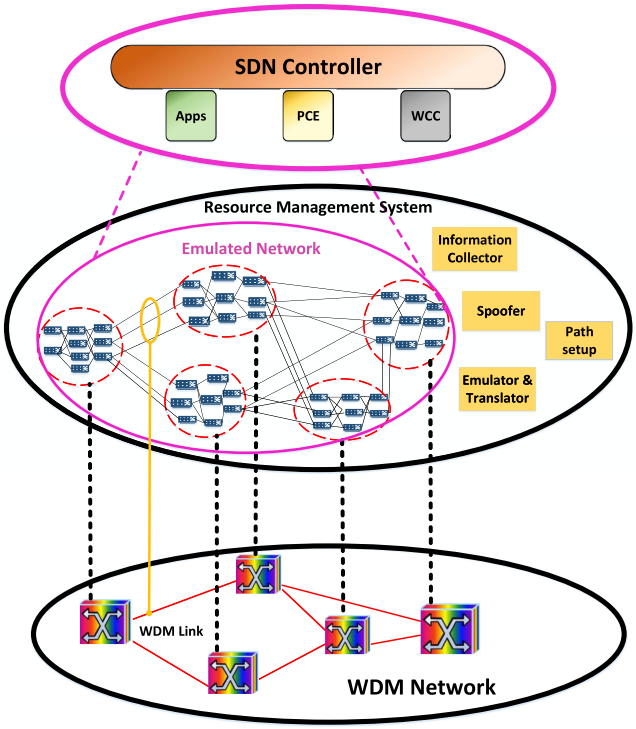


FIGURE 2. Proposed model overview.

path-requests from the optical switch itself. The model also assumes that the optical device can extract destination address from the incoming signals. The RMS can also include a ''scheduler'' component to save future requests. Details of the components of the proposed model are as below.

A. RESOURCE MANAGEMENT SYSTEM (RMS)

The RMS is responsible for performing multiple tasks and acts as a middle-ware between the SDN controller and optical network. It has many components and performs actions like collecting information from the optical switches, abstracting layer one connectivity, translating path found by the SDN controller, and wavelength availability check. It also maintains a table to store the status (idle/busy) of the ports and check their availability every time the path is requested.

B. SDN CONTROLLER

This is any conventional SDN controller with path computation ability. It can be loaded with customized applications for advanced control of this system, such as a few mentioned in the Section III.

The proposed design achieves the goal of dynamic path establishment in few steps. The first step involves ''Network Graph'' creation. This step can be challenging as there is no network connectivity initially (within switches), and resources will be allocated once the traffic flow starts. However, if all the possible paths are abstracted, see, e.g., [42], the SDN controller is able to make a network graph. Thus the SDN controller can have a global view of the layer one network, emulated as a layer two network. The second

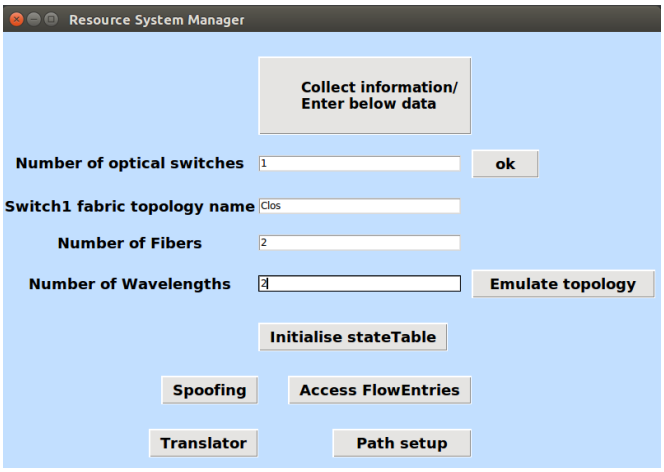


FIGURE 3. GUI for resource management system.

step follows when a request signal arrives at an optical switch. The relevant information of this request is sent to the RMS by the switch, which triggers the path calculation process in the RMS using the SDN controller. The controller computes the path for the emulated network, and flows are installed in the OF switches. In the next step, this path is translated back by the RMS to a form that is understandable by the optical devices, and finally, it sends commands to physically set up the path through them. This model appears to be an agent-based approach, however the way we implemented it (see sectionVI), is instead a middle-ware approach. Flow chart and timing diagram for this process are discussed and represented in [43].

VI. IMPLEMENTATION

Our RMS is in the form of a GUI application to implement the proposed design. We used the RYU controller and ran the application namely, ''simpleswitch1.3'' along with the ''Spanning Tree Protocol'' application to avoid any loops in the topology. Python and Bash scripting have been used to implement the components of the RMS. The components of this application are shown in Figure 3 and are explained as below.

A. INFORMATION COLLECTION COMPONENT

The information collection component serves the purpose of the topology discovery. Layer one switches are required to have knowledge about their neighbors either through a protocol like Link Layer Discovery Protocol (LLDP) or installing static files. Every switch has an interconnection of switching cells to make input-output port connections through it, called switch fabric, common examples of which are Clos, Benes, and Banyan [1]. Information collection components can be made to use any protocol to extract switch fabric information from the switch, for example, SNMP. However currently, switch fabrics are not managed in this way, so we have implemented it as a manual process to provide information such as, type of the switch fabric, number of fibers, number of allowed wavelengths per fiber, and a neighbors table.

B. EMULATE NETWORK TOPOLOGY

In order to abstract the layer one network topology, we have used a popular emulator, called ''Mininet''. The emulated network represents all layer one possible connections in the form of a layer two devices' (OF switches) network to the SDN controller. For every possible wavelength route, a packet based Ethernet link between two OF switches is formed in an emulated topology and each port of the optical switch is represented by an individual OF switch. The component emulates Layer one connectivity using an ''Emulation table'' which has information for mapping of the optical paths of WDM layer in Mininet. For this work, we have chosen the ''Clos'' and ''Crossbar'' fabric topologies for demonstration, however any other topology can be emulated. The Mininet topology is emulated as a wavelength selective architecture, further details of which are explained in [42]. Once the ''Emulate topology'' button is pressed, the topology is constructed in Mininet and OF switches will start registering themselves with the SDN controller.

C. STATE TABLE INITIALIZATION

The state table maintains the status of all the current, past and future connections and wavelengths assigned to them. The path found by the 'Translator' component is checked for its availability against the state table. The state table can be initialized at any stage of the simulation.

D. SPOOFING

Once the Mininet topology is created, the SDN controller will have a global view of the network. The controller can now find a path between any source and destination port (OF switches). When the light signal arrives at any of the optical switches, the switch will send the request for path and also the destination address to the RMS. The RMS will initiate the process of spoofing a frame arrival at the OF switch, corresponding to the same optical port that received the signal. After this, a standard OF protocol procedure involving ''PacketIn'' and ''PacketOut'' messages will start.

To implement spoofing, we have used the ''Ping'' command. When the ''Ping'' process is invoked, the OF switch looks for a flow entry in its table. If no path is found, a 'Packet In' message is sent to the controller. The SDN controller finds a path between source and destination ports and installs flow entries in the OF switches. A successful ping means that there is a possible route between the source and destination ports of the optical switch (es).

E. ACCESS FLOW ENTRIES

This component accesses flow entries of the OF switches after the 'Ping' process completes. It then verifies if the path is a valid path between the required source and destination ports. For this purpose it uses the topology information of the network.

F. TRANSLATOR

Translator uses the "Emulation table" and translates the layer two network path into a layer one path. It then checks if this path is idle (using state table) and also availability of a single wavelength along this path to satisfy the WCC. If both conditions are satisfied, it can then trigger the "Path setup" component.

G. PATH SETUP

This component is responsible for sending commands to the optical switches that are understandable by them and establishing a LightPath. These commands tell the switch to make connections between specific switch cells and the ports' interfaces, thus establishing circuit switched LightPath.

Since the whole process initiates when the requests from the optical switches arrive, this is used to dynamically find the path with no need for pre-allocation of resources as in conventional WDM networks. Another worth considering aspect of the WDM networks, while moving from static to dynamic path allocation, is to take care of the protection strategy. With our framework, it is easy to add a "link failure" component in the RMS, which will store alternate paths computed by the SDN controller. Upon receiving a link failure alarm from the switch, this component can communicate the alternate route to the optical switch and restore the traffic. We left the implementation of such a feature for our future work. Here, the RMS GUI application has been designed to have buttons for performing the respective tasks. However this is only done for demonstration purposes. The RMS only needs to receive a request from the switch and the rest of the path establishing process is automatic. If the light signal stops striking a port for a certain threshold time, the optical switch needs to inform the RMS and the path setup component will send commands to reset the switching cells of the optical switch to terminate the connection and will also update the state table.

VII. TESTING

In order to prove the concept of the dynamic path establishment process using our model, we have tested our developed RMS with a layer one switch by Cisco namely Nexus 3550-F Fusion (formerly called ExaLINK Fusion),[2] see Figure 4.

The ExaLINK Fusion has layer one switch characteristics and consists of a crossbar switch fabric that can be dynamically set up to provide a connection between any of its ports. Each of the three front panel line card bays designated as A, B, and C, have 16 inputs and outputs to the layer one switch fabric. Ports are named as A1-A16, B1-B16, and C1-C16. The switch offers SNMP and JSON remote procedure call services that we have used in our testing. Figure 5 represents the testing scenario. ExaLink Fusion is assumed to have one fiber having 16 wavelengths comprising of A1-A16 (input), and B1-B16 (output) crossbar

[2]https://fusion.exablaze.com/detailed/

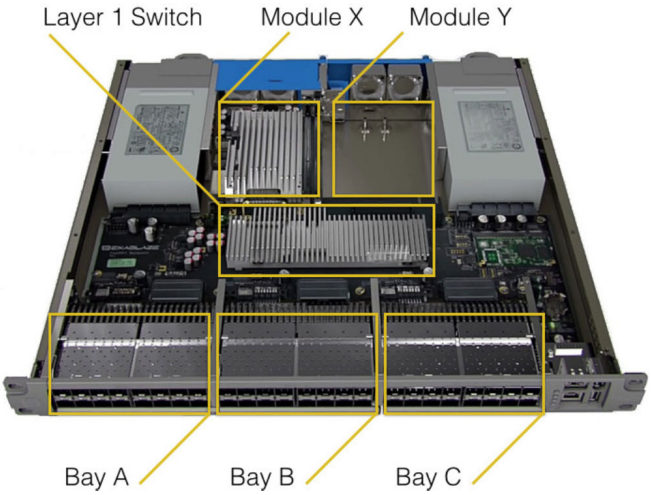


FIGURE 4. Cisco Nexus 3550-F Fusion internal structure (Source: ExaLink Fusion documentation).

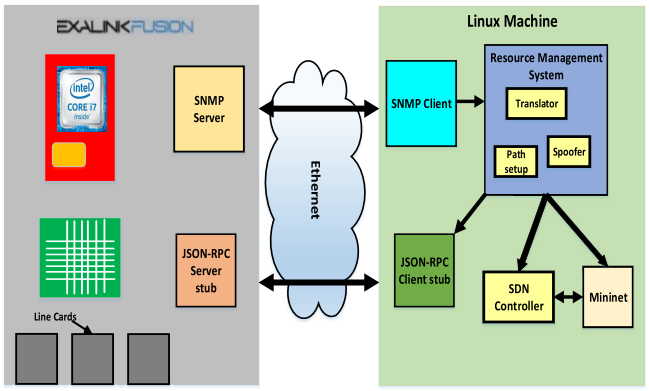


FIGURE 5. Testing setup.

fabric for this experiment. Steps followed for the testing are described below.

1) The RMS is provided with the information required for emulating a crossbar fabric topology, manually. Once the button "Emulate topology" is pressed, a cross bar topology is created in Mininet, see Figure 6.
2) When the signal arrives at one of the input ports of the ExaLink Fusion, it sends SNMP "trap" notification to the SNMP engine running in our Linux Machine which is integrated with the RMS, see Figure 7.
3) After analyzing this trap notification, the RMS uses spoofing mechanism (destination address is chosen random) to find path through the switch, see Figure 8.
4) When an available path is found and translated for the switch, the system uses remote procedure call (RPC) JSON interface to configure ExaLink Fusion to establish path between ports, e.g., path established between ports A1 and A3, as shown in Figure 9.

VIII. TOPOLOGY CREATION

Figure 3 represents the case where RMS GUI is given parameters of a "Clos" type of topology, and the number of switches/OXC in the network is one. At the same time,

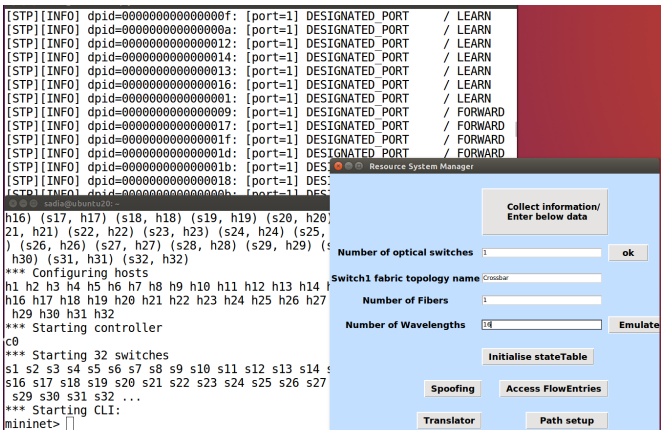


FIGURE 6. Mininet topology generated and switches being registered with controller.

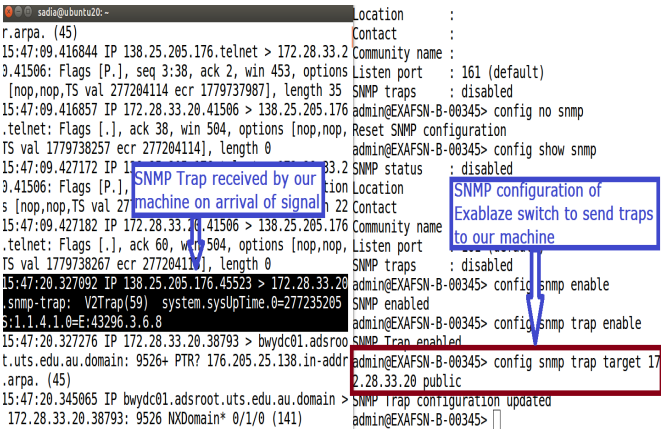


FIGURE 7. SNMP trap received from ExaLINK Fusion on arrival of signal.

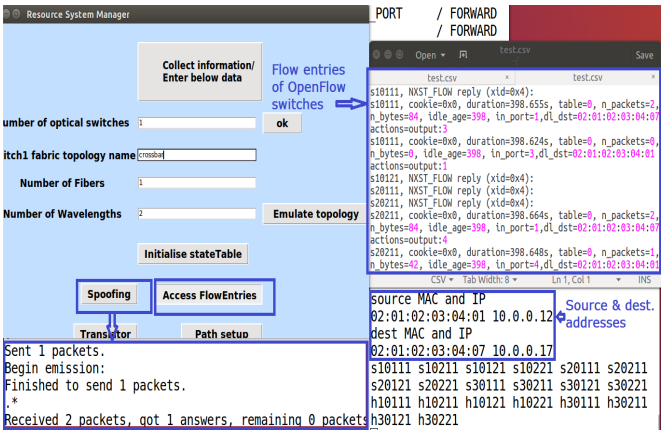


FIGURE 8. Ping test invoked by spoofing step and flow entries in OpenFlow switches being accessed.

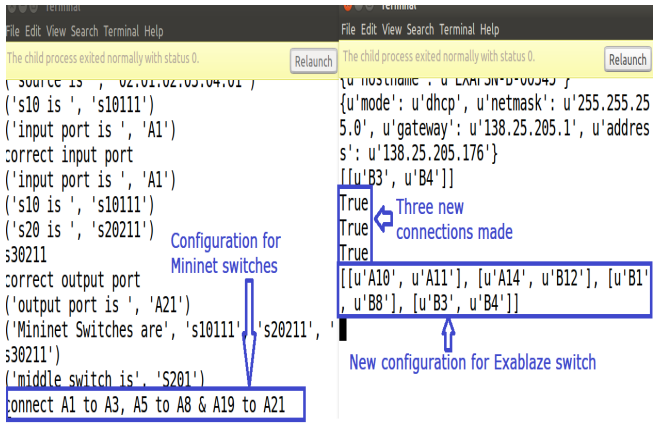


FIGURE 9. Path found between source and host of Mininet topology and the switch is configured after the path translation.

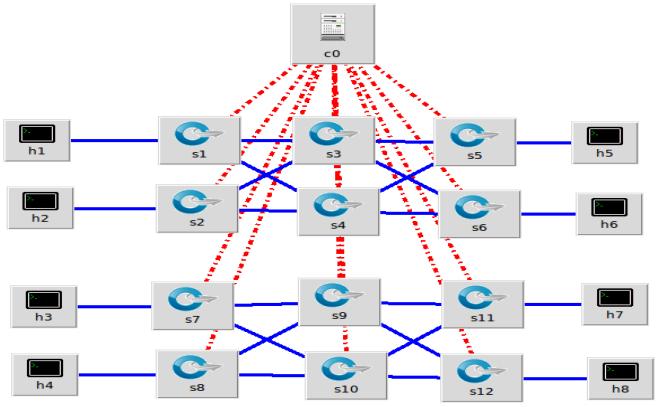


FIGURE 10. Mapped Mininet topology for Clos switch fabric for two fibers and two wavelengths.

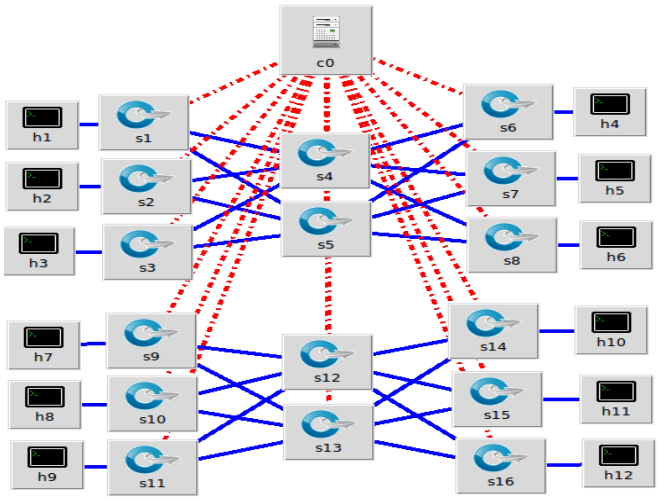


FIGURE 11. Mapped Mininet topology for Clos switch fabric for three fibers and two wavelengths.

the number of fibers and wavelengths/fiber in that switch are also provided. Pressing the "Emulate topology" button of the RMS GUI runs a python script that takes these input parameters and uses an embedded topology creation algorithm to emulate the WDM network connectivity in Mininet. Figures 10 and 11 show the cases for two fibers, two wavelengths (2F, 2W) and three fibers, two wavelengths (3F, 2W) for Clos type of OXC's switch fabric, respectively.

To see the case for (3F, 3W) and understand how this topology maps the actual optical switch fabric, the reader is referred to [42].

To find out the effect of the number of wavelengths and fibers on Mininet topology creation, we analyzed the "Generate Mininet topology" algorithm [42]. For example, we noticed that to map the wavelength-selective architecture

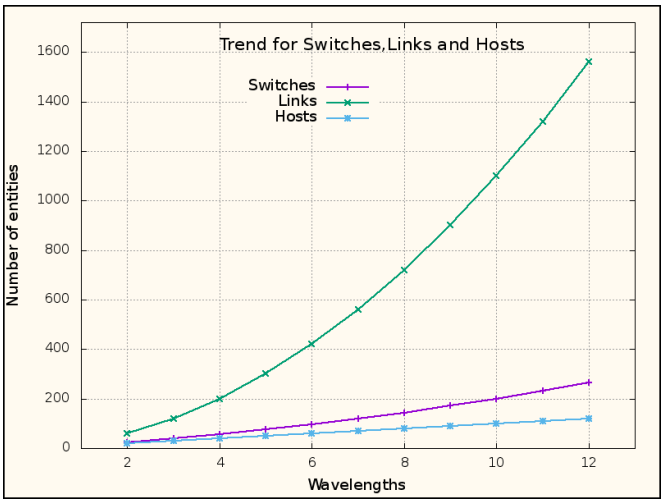


FIGURE 12. Trend for number of switches, hosts and links for Clos fabric topology in Mininet with wavelengths.

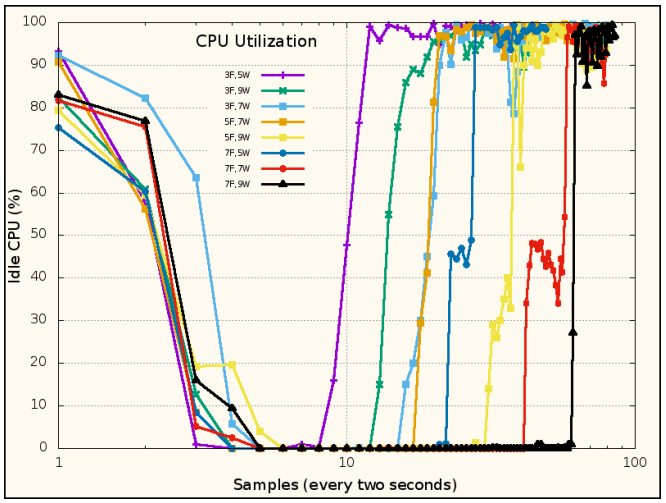


FIGURE 13. CPU utilization for creating Mininet topologies with various number of fibers and wavelengths.

of Clos fabric in Mininet; we need twice the number of hosts for a single increase in the number of (F) or (W), hence the equation (3). We noticed that three entities of a Mininet topology, i.e., OpenFlow switches (S), hosts (H), and links (L) depend on two factors namely fibers (F) and wavelengths/fiber (W), see equations (1), (2), and (3). Figure 12 represents the trend for these entities with the increase in the number of wavelengths (F is kept constant) for a Clos type of optical switch fabric. It is observed that the effect of varying F is not as prominent as with W, as apparent from the equations as well, thus here, we are only showing the trend with the number of wavelengths while keeping the number of F constant (e.g., five).

$$S = W^2 + 2F.W \tag{1}$$

$$L = 2F.W + F.W^2 \tag{2}$$

$$H = 2F.W \tag{3}$$

It can be seen from the graph that the increase in the number of links is huge with the increasing number of wavelengths. For example, increasing wavelengths from 4 to 6, increases the number of links from 200 to 410, this is because every possible path for a wavelength in an optical network is represented as an individual Ethernet link in the Mininet emulation. Similar equations can be found for emulating other types of fabrics, and their effect can be observed. Here, it is important to consider the effects of W and F because they help understand the performance parameters' trends. A higher number of F and W means bigger and complex Mininet topologies.

IX. PERFORMANCE RESULTS AND ANALYSIS

The emulated Mininet topology gets more complex with the complexity of the OXC's fabric topology and with the increase in the number of the wavelengths and fibers. We observed that more the fabric structure is complex, more the computational resources are required by our designed system. We therefore did an experimental analysis of some main computational resource factors, e.g., memory, CPU usage, and time required for creating the Mininet topology. For this analysis we chose "Clos" switch fabric topology as an example.

Our system specifications are Intel Core i7-7600 CPU 2.80GHz & 2.90GHz with 16GB installed memory.

A. CPU UTILIZATION

Figure 13 shows that the percentage of the idle CPU (mean values) decreases very quickly as soon as the topology starts getting created. As long as the topology is being created the CPU is utilized almost 100% and then comes back to become almost free. Smaller the number of F and W, quicker is the restored CPU usage, this is clearly because of less number of OpenFlow switches, links and hosts and thus less required processing. Same trend is followed when the topology is destroyed (not shown in the graph).

B. MEMORY UTILIZATION

Figure 14 shows the graph for maximum Resident Set Size (RSS) memory allocated for creating Mininet Clos topologies with various number of fibers and wavelengths. Increasing the number of fibers and wavelengths increase the number of switches, links, and hosts and thus more memory is utilized. The graphs become steeper with higher values of W, since the number of links and switches increase as the square power of W, (see (1), (2), and (3)). Increasing the number of wavelengths in a sequence of one, increases the memory utilization by nearly more than 1.5MB for topologies with 3, 5, and 7 fibres, as seen in the graph. The memory utilization increases by 10.8% with increase in the number of fibers from 3 to 5, while the increase is 9.7% with increase in the number of fibers from 5 to 7, with the number of wavelengths equal to 8. However, this gain slightly increases with the number of wavelengths because of more memory required.

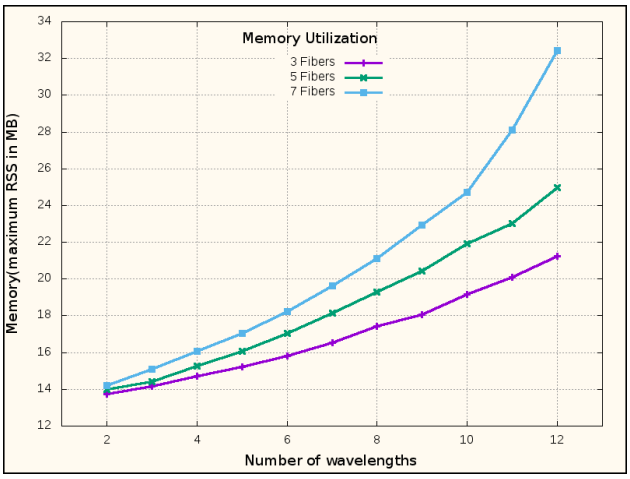


FIGURE 14. Memory utilization for creating Mininet topologies with various number of fibers and wavelengths.

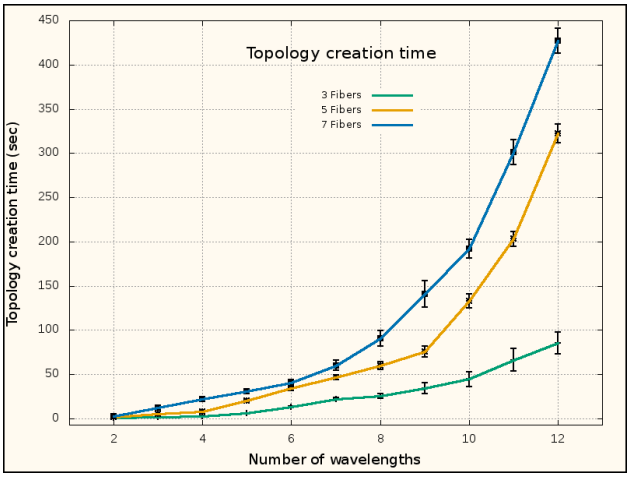


FIGURE 15. Mininet topology creation time (mean and standard error) for various number of fibers and wavelengths.

C. TOPOLOGY EMULATION TIME

Figure 15 shows the increase in the time required for creating a topology in Mininet, with the increase in the number of fibers and wavelengths. It can be seen that the graph becomes steeper with higher values of F and W, this is compliant with our equations, (1-3) and Figure 12, where the number of switches and links increase as a square power of W. We observed that there is no specific approximate time for the creation of topology and the creation time is highly dependent on the state of the machine on which experiments are being conducted. For every fresh startup of the machine, less time is taken for the topology creation than if the machine is already running other processes or when cache memory is full. For our experiments, the topology creation time increases by 96% when increasing fibers from 3 to 5, and by 43% when increasing fibers from 5 to 7, while the number of the wavelengths is 8.

D. LATENCY

The overall path setup delay depends on the delay in the links between OXCs, performance of the controller, time for translation of flows and configuration of optical switches. To measure the time required for the whole path setup process is difficult as it involves different systems; however, the most contributing and varying factor in the path setup delay is the path finding through the Mininet topology. Thus if we have an estimate of how much time is taken by the path finding process in the Mininet topology, then we can have a estimation of the overall path setup delay. In our model, RMS uses a spoofing mechanism to invoke path finding process which uses the 'Ping' command to find the path between the source and the destination hosts. Thus we measure the time taken by a successful 'Initial Ping' (Iping) [44], [45], by performing the Ping test between end hosts of the given network topologies for calculating round trip time (RTT) between hosts.

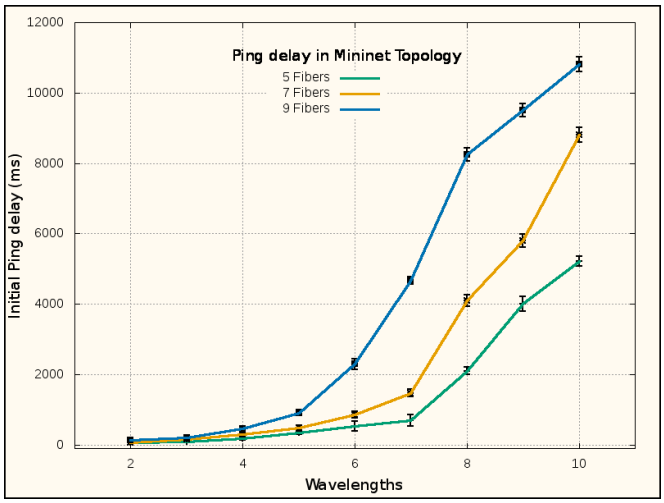


FIGURE 16. Initial Ping delay for creating Mininet topologies for various number of fibers and wavelengths.

Figure 16 shows the Iping delays (mean values) between the end hosts for topologies with 5, 7, and 9 fibers. It is obvious from the graph that the latency has an increasing trend for bigger topologies; however, the increase is not smooth because of the high dependence on the state of the machine. For exact number of switches, links, and hosts in a topology, see equations (1-3). Standard error for bigger topologies is between 100-300 milliseconds approximately. Increase in the ping delay doubles when increasing the number of fibers from 5 to 7, and 7 to 9, for the number of wavelengths being equal to 8.

X. DISCUSSION

The graphs shown here are only for a single switch having Clos topology switch fabric. For any other topology, re-analysis will be required. Moreover, there may exist different types of switches within the optical network, so overall network connectivity will be the combination of all the switch fabrics and hence will be the combined required resources. Our results show compliance with the outcomes and discussions in some other performance analysis works, e.g., [44], [46], [47].

Figure 15 shows some large error bars especially for higher values of wavelengths, this is mainly because the speed of experiments depend on the background processes running in the machine (laptop) at the same time. This is an unavoidable situation in our case as the processor has to continue its back-end processes for its normal operation. Thus in cases, where automation of the process depends on the time of topology creation (as in our case), no certain time can be assumed and there is need of an alert system to tell that the topology creation is done and the system is ready for the next ''path finding'' step.

Dynamically establishing paths using SDN, seems to take time; however, considering core transport networks, the incoming requests do not arrive very often, and also once the path is established, it remains for longer times, e.g., months or years [48]. Although, there will be an initial loss of data until the path is found and established, but this is worth of the flexibility of the dynamic resources allocation, and offers a range of other benefits of SDN, as discussed previously in Section III.

XI. CONCLUSION AND FUTURE WORK

A model for path establishment in WDM network is presented in this paper which provides a flexible and configurable SDN solution for dynamic circuit-switching in WDM devices in core optical networks. It gives an advantage over other path establishment approaches presented in the literature in terms of considering switching constraints. None of the designs considered the congestion inside a switch and hence the non-availability of the path. This design is an effort towards a dynamic on-demand SDN based path establishment in WDM network without enhancing controller or OF protocol. However, the model considers the optical switch to be able to extract the destination addresses from the incoming requests. The model has been tested with Cisco layer one switch, and its performance is evaluated in terms of CPU and memory utilization, topology creation time, and latency. It is found that with more complicated switch fabric's topology, more computational resources are required and also path establishment time increases. Gain in the memory utilization, topology creation time, and latency is 9.7%, 43%, and 100% respectively, when increasing the number of fibers from 5 to 7 and the number of wavelengths is set to 8. Future work involves extension of the model for the optical switches with other fabric topologies and any-to-any port connections. Furthermore incorporating new modules for implementation of other optical network functionalities, e.g., protection and restoration.

REFERENCES

[1] T. S. El-Bewab, *Optical Switching*. New York, NY, USA: Springer, 2006, doi: 10.1007/0-387-29159-8.

[2] N. McKeown, T. Anderson, H. Balakrishnan, G. Parulkar, L. Peterson, J. Rexford, S. Shenker, and J. Turner, ''OpenFlow: Enabling innovation in campus networks,'' *ACM SIGCOMM Comput. Commun. Rev.*, vol. 38, no. 2, p. 69, 2008. [Online]. Available: http://portal.acm.org/citation.cfm?doid=1355734.1355746

[3] ''Software-defined networking: The new norm for networks [white paper],'' Open Netw. Found., ONF, Menlo Park, CA, USA, White Paper, 2012, pp. 1–12. [Online]. Available: https://opennetworking.org

[4] S. Das, G. Parulkar, and N. McKeown, ''Unifying packet and circuit switched networks,'' in *Proc. IEEE Globecom Workshops*, Nov. 2009, pp. 1–6. [Online]. Available: http://ieeexplore.ieee.org/document/5360777/

[5] S. Gringeri, N. Bitar, and T. J. Xia, ''Extending software defined network principles to include optical transport,'' *IEEE Commun. Mag.*, vol. 51, no. 3, pp. 32–40, Mar. 2013. [Online]. Available: http://ieeexplore.ieee.org/document/6476863/

[6] *OpenFlow-Enabled Transport SDN*, ONF Solution Brief, Menlo Park, CA, USA, 2014.

[7] P. Bhaumik, S. Zhang, P. Chowdhury, S.-S. Lee, J. H. Lee, and B. Mukherjee, ''Software-defined optical networks (SDONs): A survey,'' *Photonic Netw. Commun.*, vol. 28, no. 1, pp. 4–18, Aug. 2014.

[8] M. Chamania and A. Jukan, ''Springer handbook of optical networks,'' in *Springer Handbooks*, B. Mukherjee, I. Tomkos, M. Tornatore, P. Winzer, and Y. Zhao, Eds. Cham, Switzerland: Springer, 2020, doi: 10.1007/978-3-030-16250-4.

[9] B. Lantz, A. A. Díaz-Montiel, J. Yu, C. Rios, M. Ruffini, and D. Kilper, ''Demonstration of software-defined packet-optical network emulation with mininet-optical and ONOS,'' in *Proc. Opt. InfoBase Conf. Papers*, Mar. 2020, pp. 1–3.

[10] M. Bahnasy, K. Idoudi, and H. Elbiaze, ''OpenFlow and GMPLS unified control planes: Testbed implementation and comparative study,'' *J. Opt. Commun. Netw.*, vol. 7, no. 4, p. 301, 2015. [Online]. Available: http://www.opticsinfobase.org/abstract.cfm?URI=jocn-7-4-301

[11] M. Channegowda, P. Kostecki, N. Efstathiou, S. Azodolmolky, R. Nejabati, P. Kaczmarek, A. Autenrieth, J.-P. Elbers, and D. Simeonidou, ''Experimental evaluation of extended OpenFlow deployment for high-performance optical networks,'' in *Proc. Eur. Conf. Exhib. Opt. Commun.*, Sep. 2012, pp. 1–6. [Online]. Available: https://www.osapublishing.org/abstract.cfm?uri=ECEOC-2012-Tu.1.D.2

[12] L. Liu, T. Tsuritani, I. Morita, H. Guo, and J. Wu, ''Experimental validation and performance evaluation of OpenFlow-based wavelength path control in transparent optical networks,'' in *Proc. Opt. InfoBase Conf. Papers*, vol. 19, no. 27, 2011, pp. 26578–26593.

[13] S. Das, G. Parulkar, and N. McKeown, ''Why OpenFlow/SDN can succeed where GMPLS failed,'' in *Proc. Eur. Conf. Exhib. Opt. Commun.*, Sep. 2012, pp. 31–33.

[14] S. Das, G. Parulkar, N. McKeown, P. Singh, D. Getachew, and L. Ong, ''Packet and circuit network convergence with OpenFlow,'' in *Proc. Opt. Fiber Commun. Conf.*, Mar. 2010, pp. 7–9.

[15] D. Simeonidou, R. Nejabati, and M. P. Channegowda, ''Software defined optical networks technology and infrastructure: Enabling software-defined optical network operations,'' in *Proc. Opt. Fiber Commun. Conf./Nat. Fiber Optic Eng. Conf.*, Mar. 2013, pp. 274–282.

[16] R. Rejeb, I. Pavlosoglou, M. S. Leeson, and R. J. Green, ''Management issues in transparent optical networks,'' in *Proc. 6th Int. Conf. Transparent Opt. Netw.*, Jul. 2004, pp. 248–254.

[17] B. Barán and D. P. Pinto-Roa, ''Multiobjective optimization in optical networks,'' in *Bio-Inspired Computation in Telecommunications*, X.-S. Yang, S. F. Chien, and T. O. Ting, Eds. Amsterdam, The Netherlands: Elsevier, 2015, ch. 10, pp. 205–244. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/C20140005011

[18] H. Zang, J. P. Jue, and B. Mukherjee, ''A review of routing and wavelength assignment approaches for wavelength- routed optical WDM networks,'' *Opt. Netw. Mag.*, vol. 1, no. January, pp. 47–60, 2000. [Online]. Available: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.127.5171&re%p=rep1&type=pdf

[19] P. Goransson and C. Black, *Software Defined Networks: A Comprehensive Approach*, 2nd ed. Amsterdam, The Netherlands: Elsevier, 2016.

[20] D. Kreutz, F. Ramos, P. E. Veríssimo, C. E. Rothenberg, S. Azodolmolky, and S. Uhlig, ''Software-defined networking: A comprehensive survey,'' *Proc. IEEE*, vol. 103, no. 1, pp. 14–76, Jan. 2015. [Online]. Available: http://ieeexplore.ieee.org/document/6994333/

[21] D. F. Macedo, D. Guedes, L. F. M. Vieira, M. A. M. Vieira, and M. Nogueira, ''Programmable networks-from software-defined radio to software-defined networking,'' *IEEE Commun. Surveys Tuts.*, vol. 17, no. 2, pp. 1102–1125, 2nd Quart., 2015. [Online]. Available: http://ieeexplore.ieee.org/document/7039225/

[22] L. Liu, D. Zhang, T. Tsuritani, R. Vilalta, R. Casellas, L. Hong, I. Morita, H. Guo, J. Wu, R. Martinez, and R. Munoz, "Field trial of an OpenFlow-based unified control plane for multilayer multigranularity optical switching networks," *J. Lightw. Technol.*, vol. 31, no. 4, pp. 506–514, Feb. 15, 2013.

[23] Z. Shu, J. Wan, J. Lin, S. Wang, D. Li, S. Rho, and C. Yang, "Traffic engineering in software-defined networking: Measurement and management," *IEEE Access*, vol. 4, pp. 3246–3256, 2016.

[24] Q. Dong, J. Li, Y. Ma, and S. Han, "A path allocation method based on source routing in SDN traffic engineering," in *Proc. IEEE Int. Conf. Smart Cloud (SmartCloud)*, Dec. 2019, pp. 163–168.

[25] T. Zhang, A. Samadian, A. Shakeri, B. Mirkhanzadeh, C. Shao, M. Razo, M. Tacca, A. Ferrari, M. Cantono, V. Curri, G. Martinelli, G. M. Galimberti , T. Xia, G. Wellbrock, and A. Fumagalli, "A WDM network controller with real-time updates of the physical layer abstraction," *J. Lightw. Technol.*, vol. 37, no. 16, pp. 4073–4080, Aug. 15, 2019.

[26] I. Martin, S. Troia, J. A. Hernandez, A. Rodriguez, F. Musumeci, G. Maier, R. Alvizu, and O. G. de Dios, "Machine learning-based routing and wavelength assignment in software-defined optical networks," *IEEE Trans. Netw. Service Manage.*, vol. 16, no. 3, pp. 871–883, Sep. 2019. [Online]. Available: https://ieeexplore.ieee.org/document/8758853/

[27] D. T. Hai, "On the spectrum-efficiency of QoS-aware protection in elastic optical networks," *Optik*, vol. 202, Feb. 2020, Art. no. 163563, doi: 10.1016/j.ijleo.2019.163563.

[28] D. Chada, *Optical WDM Networks: From Static to Elastic Networks*. Hoboken, NJ, USA: Wiley, 2019, doi: 10.1002/9781119393399.

[29] L. Liu, R. Muñoz, R. Casellas, T. Tsuritani, R. Martínez, and I. Morita, "OpenSlice: An OpenFlow-based control plane for spectrum sliced elastic optical path networks," *Opt. Exp.*, vol. 21, no. 4, pp. 4194–4204, 2013.

[30] L. Liu, W.-R. Peng, R. Casellas, T. Tsuritani, I. Morita, R. Martínez, R. Muñoz, and S. J. B. Yoo, "Design and performance evaluation of an OpenFlow-based control plane for software-defined elastic optical networks with direct-detection optical OFDM (DDO-OFDM) transmission," *Opt. Exp.*, vol. 22, no. 1, p. 30, 2014.

[31] D. T. Hai, L. H. Chau, and N. T. Hung, "A priority-based multiobjective design for routing, spectrum, and network coding assignment problem in network-coding-enabled elastic optical networks," *IEEE Syst. J.*, vol. 14, no. 2, pp. 2358–2369, Jun. 2020.

[32] A. E. Kamal and M. Mohandespour, "Network coding-based protection," *Opt. Switching Netw.*, vol. 11, pp. 189–201, Jan. 2014.

[33] D. B. Swarna and V. Muthumanikandan, "Nested failure detection and recovery in software defined networks," in *Proc. IEEE Int. Conf. Electr., Comput. Commun. Technol. (ICECCT)*, Feb. 2019, pp. 1–6.

[34] I. Seremet, S. Hadzovic, S. Mrdovic, and S. Causevic, "SDN as a tool for energy saving," in *Proc. 27th Telecommun. Forum (TELFOR)*, Nov. 2019, pp. 2019–2022.

[35] A. Fernandez-Fernandez, C. Cervello-Pastor, and L. Ochoa-Aday, "Achieving energy efficiency: An energy-aware approach in SDN," in *Proc. IEEE Global Commun. Conf. (GLOBECOM)*, Dec. 2016, pp. 1–7.

[36] M. C. Nkosi, L. Mamushiane, A. A. Lysko, D. L. Johnson, and A. P. Engelbrecht, "Towards programmable on-demand lightpath services: Current state-of-the-art and open research areas," *IET Netw.*, vol. 8, no. 6, pp. 347–355, Nov. 2019.

[37] R. Alvizu, G. Maier, N. Kukreja, A. Pattavina, R. Morro, A. Capello, and C. Cavazzoni, "Comprehensive survey on T-SDN: Software-defined networking for transport networks," *IEEE Commun. Surveys Tuts.*, vol. 19, no. 4, pp. 2232–2283, 4th Quart. 2017.

[38] Y. Li and D. C. Kilper, "Optical physical layer SDN [invited]," *J. Opt. Commun. Netw.*, vol. 10, no. 1, p. A110, Jan. 2018. [Online]. Available: https://www.osapublishing.org/abstract.cfm?URI=jocn-10-1-A110

[39] I. Alawe, B. Cousin, O. Thorey, and R. Legouable, "Integration of legacy non-SDN optical ROADMs in a software defined network," in *Proc. IEEE Int. Conf. Cloud Eng. Workshop (IC2EW)*, Apr. 2016, pp. 60–64.

[40] M. De Leenheer, Y. Higuchi, T. Furusawa, and G. Parulkar, "SDN control of optical networks," in *Proc. Eur. Conf. Opt. Commun. (ECOC)*, Sep. 2017, pp. 1–3.

[41] M. De Leenheer, G. Parulkar, and T. Tofigh, "SDN control of packet-over-optical networks," in *Proc. Opt. Fiber Commun. Conf.*, Mar. 2015, pp. 1–27.

[42] S. Qureshi and R. Braun, "Mininet topology: Mirror of the optical switch fabric," in *Proc. 29th Int. Telecommun. Netw. Appl. Conf. (ITNAC)*, Nov. 2019, pp. 1–6.

[43] S. Qureshi and R. Braun, "Dynamic light path establishment in switch fabric using OpenFlow," in *Proc. 26th Int. Conf. Syst. Eng. (ICSEng)*, Dec. 2018, pp. 1–4. [Online]. Available: https://ieeexplore.ieee.org/document/8638220/

[44] P. Isaia and L. Guan, "Performance benchmarking of SDN experimental platforms," in *Proc. IEEE NetSoft Conf. Workshops (NetSoft)*, Jun. 2016, pp. 116–120. [Online]. Available: http://ieeexplore.ieee.org/document/7502456/

[45] A. K. Arahunashi, S. Neethu, and H. V. R. Aradhya, "Performance analysis of various SDN controllers in mininet emulator," in *Proc. 4th Int. Conf. Recent Trends Electron., Inf., Commun. Technol. (RTEICT)*, May 2019, pp. 752–756.

[46] A. Nguyen-Ngoc, S. Lange, T. Zinner, M. Seufert, P. Tran-Gia, N. Aerts, and D. Hock, "Performance evaluation of selective flow monitoring in the ONOS controller," in *Proc. 13th Int. Conf. Netw. Service Manage. (CNSM)*, Nov. 2017, pp. 1–6.

[47] D. Muelas, J. Ramos, and J. E. L. D. Vergara, "Assessing the limits of mininet-based environments for network experimentation," *IEEE Netw.*, vol. 32, no. 6, pp. 168–176, Nov. 2018.

[48] F. Cugini and P. Castoldi, "Software defined networking (SDN) in optical network," in *Elastic Optical Networks* (Optical Networks), vol. 56, V. López and L. Velasco, Eds. Cham, Switzerland: Springer, 2016, ch. 9, pp. 101–193, doi: 10.1007/978-3-319-30174-7_9.

SADIA QURESHI received the M.S. degree in electrical engineering from the University of Engineering and Technology, Peshawar, Pakistan. She is currently pursuing the Ph.D. degree with the School of Electrical and Data Engineering, University of Technology Sydney, Australia. She is also a full-time Research Scholar with the School of Electrical and Data Engineering, University of Technology Sydney. Her research interests include optical networks, software-defined networking, heterogeneous networks, and wireless sensor networks.

ROBIN M. BRAUN (Life Senior Member, IEEE) received the B.Sc. degree (Hons.) from Brighton University, Brighton, U.K., in 1980, and the M.Sc.(Eng.) and Ph.D. degrees from the University of Cape Town, Cape Town, South Africa, in 1982 and 1986, respectively. He started his academic career in 1986 at the University of Cape Town. In 1998, he moved to the University of Technology Sydney, Ultimo, NSW, Australia, where he is currently an Honorary Professor with the School of Electrical and Data Engineering. Prior to moving to academia, he spent ten years in the industry, mostly with Philips, Johannesburg and Plessey, Cape Town, where he worked on the design of precision electronic distance measuring equipment. His recent work has been in network protocols and the management of complex next-generation networks. He is very active in software-defined networks. He is also a Founder Member of Australia and New Zealand Software Defined Networking (ANZSDN).