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.

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

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

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/

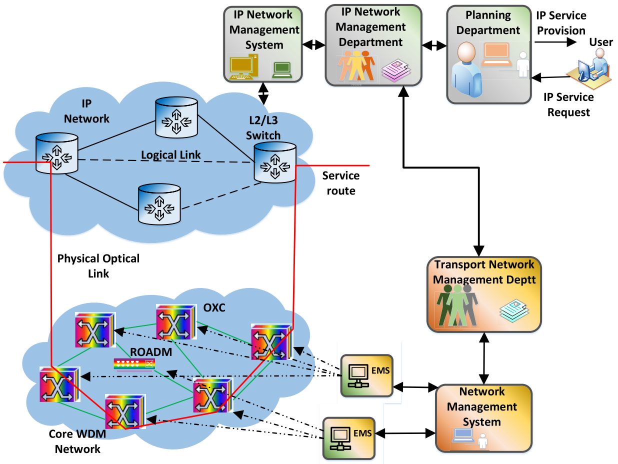


FIGURE 1. Overview of IP and optical layer management system.

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 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.

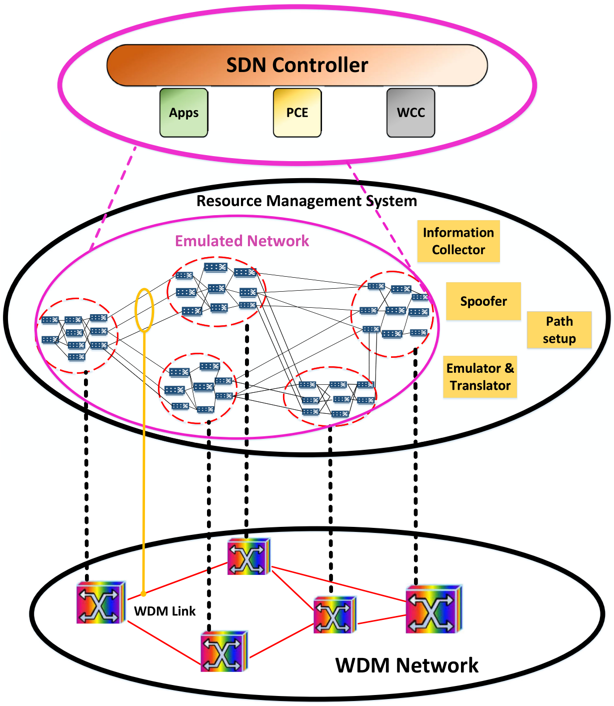


FIGURE 2. Proposed model overview.

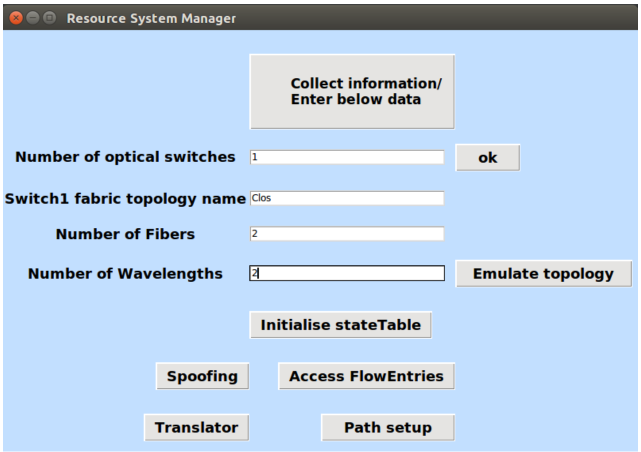
A. RESOURCE MANAGEMENT SYSTEM (RMS)

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



Number of optical switches: 1

Switch1 fabric topology name: Clos

Number of Fibers: 2

Number of Wavelengths: 2

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

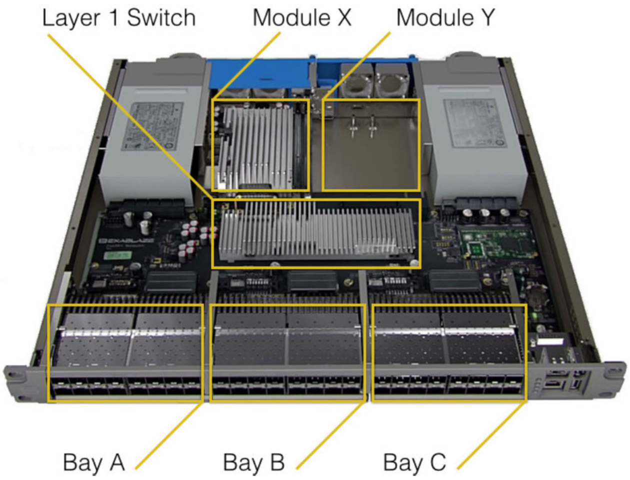


FIGURE 4. Cisco Nexus 3550-F Fusion internal structure (Source: ExaLink Fusion documentation).

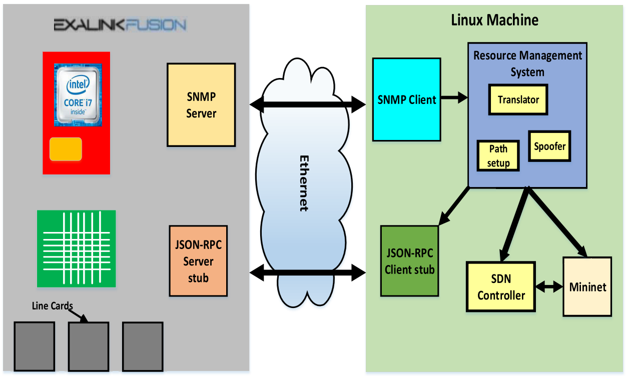


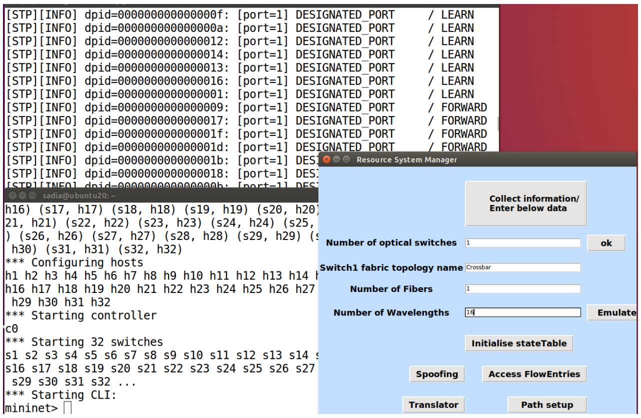
FIGURE 5. Testing setup.

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,

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



```
[STP][INFO] dpid=000000000000000f: [port=1] DESIGNATED_PORT     / LEARN
[STP][INFO] dpid=000000000000000a: [port=1] DESIGNATED_PORT     / LEARN
[STP][INFO] dpid=0000000000000012: [port=1] DESIGNATED_PORT     / LEARN
[STP][INFO] dpid=0000000000000014: [port=1] DESIGNATED_PORT     / LEARN
[STP][INFO] dpid=0000000000000013: [port=1] DESIGNATED_PORT     / LEARN
[STP][INFO] dpid=0000000000000016: [port=1] DESIGNATED_PORT     / LEARN
[STP][INFO] dpid=0000000000000001: [port=1] DESIGNATED_PORT     / LEARN
[STP][INFO] dpid=0000000000000009: [port=1] DESIGNATED_PORT     / FORWARD
[STP][INFO] dpid=0000000000000017: [port=1] DESIGNATED_PORT     / FORWARD
[STP][INFO] dpid=000000000000001f: [port=1] DESIGNATED_PORT     / FORWARD
[STP][INFO] dpid=000000000000001d: [port=1] DESIGNATED_PORT     / FORWARD
[STP][INFO] dpid=000000000000001b: [port=1] DESI
[STP][INFO] dpid=0000000000000018: [port=1] DESI
```

```
sadia@ubuntu20: ~
h16) (s17, h17) (s18, h18) (s19, h19) (s20, h20)
21, h21) (s22, h22) (s23, h23) (s24, h24) (s25,
) (s26, h26) (s27, h27) (s28, h28) (s29, h29) (s
 h30) (s31, h31) (s32, h32)
*** Configuring hosts
h1 h2 h3 h4 h5 h6 h7 h8 h9 h10 h11 h12 h13 h14 h
h16 h17 h18 h19 h20 h21 h22 h23 h24 h25 h26 h27
 h29 h30 h31 h32
*** Starting controller
c0
*** Starting 32 switches
s1 s2 s3 s4 s5 s6 s7 s8 s9 s10 s11 s12 s13 s14 s
s16 s17 s18 s19 s20 s21 s22 s23 s24 s25 s26 s27
 s29 s30 s31 s32 ...
*** Starting CLI:
mininet> 
```

Resource System Manager

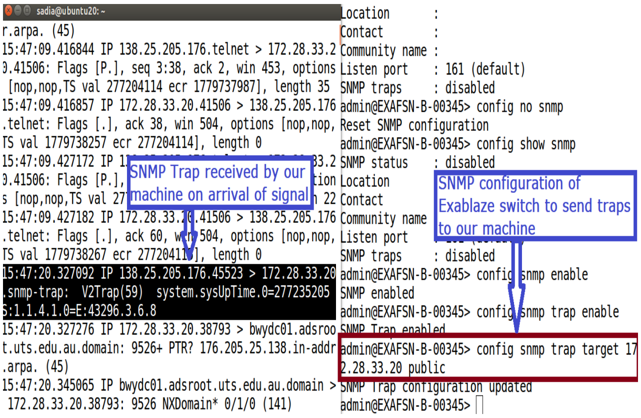
Number of optical switches: 1

Switch1 fabric topology name: Crossbar

Number of Fibers: 1

Number of Wavelengths: 16

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



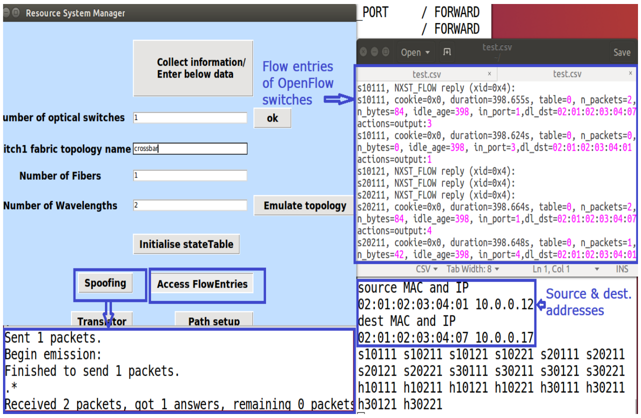
```
sadia@ubuntu20: ~
r.arpa. (45)
15:47:09.416844 IP 138.25.205.176.telnet > 172.28.33.2
0.41506: Flags [P.], seq 3:38, ack 2, win 453, options
 [nop,nop,TS val 277204114 ecr 1779737987], length 35
15:47:09.416857 IP 172.28.33.20.41506 > 138.25.205.176
.telnet: Flags [.], ack 38, win 504, options [nop,nop,
TS val 1779738257 ecr 277204114], length 0
15:47:09.427172 IP 1
0.41506: Flags [P.],
s [nop,nop,TS val 27                                  22
15:47:09.427182 IP 172.28.33.20.41506 > 138.25.205.176
.telnet: Flags [.], ack 60, win 504, options [nop,nop,
TS val 1779738267 ecr 277204115], length 0
15:47:20.327092 IP 138.25.205.176.45523 > 172.28.33.20
.snmp-trap:  V2Trap(59)  system.sysUpTime.0=277235205
S:1.1.4.1.0=E:43296.3.6.8
15:47:20.327276 IP 172.28.33.20.38793 > bwydc01.adsroo
t.uts.edu.au.domain: 9526+ PTR? 176.205.25.138.in-addr
.arpa. (45)
15:47:20.345065 IP bwydc01.adsroot.uts.edu.au.domain >
 172.28.33.20.38793: 9526 NXDomain* 0/1/0 (141)
```

```
Location        :
Contact         :
Community name  :
Listen port     : 161 (default)
SNMP traps      : disabled
admin@EXAFSN-B-00345> config no snmp
Reset SNMP configuration
admin@EXAFSN-B-00345> config show snmp
SNMP status     : disabled
Location
Contact
Community name
Listen port
SNMP traps      : disabled
admin@EXAFSN-B-00345> config snmp enable
SNMP enabled
admin@EXAFSN-B-00345> config snmp trap enable
SNMP Trap enabled
admin@EXAFSN-B-00345> config snmp trap target 17
2.28.33.20 public
SNMP Trap configuration updated
admin@EXAFSN-B-00345> 
```

SNMP Trap received by our machine on arrival of signal

SNMP configuration of Exablaze switch to send traps to our machine

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



Resource System Manager

Number of optical switches: 1

Switch1 fabric topology name: crossbar

Number of Fibers: 1

Number of Wavelengths: 2

Flow entries of OpenFlow switches

```
PORT     / FORWARD
         / FORWARD
```

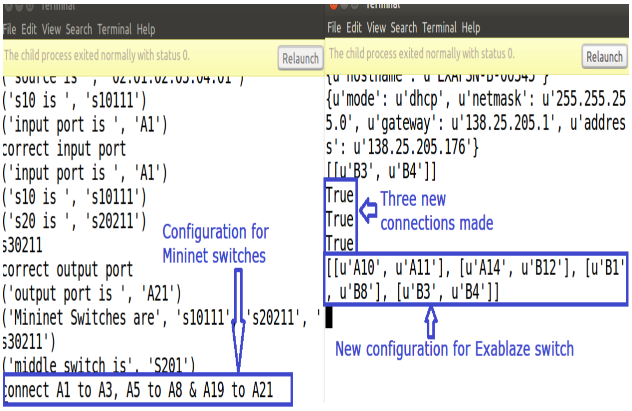
```
test.csv
s10111, NXST_FLOW reply (xid=0x4):
s10111, cookie=0x0, duration=398.655s, table=0, n_packets=2,
n_bytes=84, idle_age=398, in_port=1,dl_dst=02:01:02:03:04:07
actions=output:3
s10111, cookie=0x0, duration=398.624s, table=0, n_packets=0,
n_bytes=0, idle_age=398, in_port=3,dl_dst=02:01:02:03:04:01
actions=output:1
s10121, NXST_FLOW reply (xid=0x4):
s20111, NXST_FLOW reply (xid=0x4):
s20211, NXST_FLOW reply (xid=0x4):
s20211, cookie=0x0, duration=398.664s, table=0, n_packets=2,
n_bytes=84, idle_age=398, in_port=1,dl_dst=02:01:02:03:04:07
actions=output:4
s20211, cookie=0x0, duration=398.648s, table=0, n_packets=1,
n_bytes=42, idle_age=398, in_port=4,dl_dst=02:01:02:03:04:01
```

```
Sent 1 packets.
Begin emission:
Finished to send 1 packets.
.*
Received 2 packets, got 1 answers, remaining 0 packets
```

```
source MAC and IP
02:01:02:03:04:01 10.0.0.12
dest MAC and IP
02:01:02:03:04:07 10.0.0.17
s10111 s10211 s10121 s10221 s20111 s20211
s20121 s20221 s30111 s30211 s30121 s30221
h10111 h10211 h10121 h10221 h30111 h30211
h30121 h30221
```

Source & dest. addresses

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



File Edit View Search Terminal Help
The child process exited normally with status 0.
Relaunch

```
('source is ', '02:01:02:03:04:01')
('s10 is ', 's10111')
('input port is ', 'A1')
correct input port
('input port is ', 'A1')
('s10 is ', 's10111')
('s20 is ', 's20211')
s30211
correct output port
('output port is ', 'A21')
('Mininet Switches are', 's10111', 's20211', '
s30211')
('middle switch is', 'S201')
connect A1 to A3, A5 to A8 & A19 to A21
```

```
{u'hostname': u'EXAFSN-B-00345'}
{u'mode': u'dhcp', u'netmask': u'255.255.25
5.0', u'gateway': u'138.25.205.1', u'addres
s': u'138.25.205.176'}
[[u'B3', u'B4']]
True
True
True
[[u'A10', u'A11'], [u'A14', u'B12'], [u'B1'
, u'B8'], [u'B3', u'B4']]
```

Configuration for Mininet switches

Three new connections made

New configuration for Exablaze switch

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

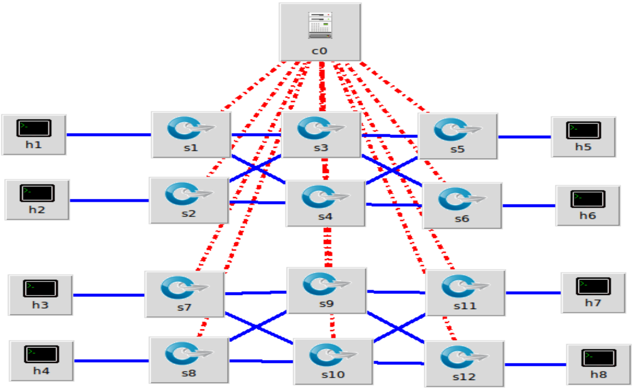


FIGURE 10. Mapped Mininet topology for Clos switch fabric for two fibers and two wavelengths.

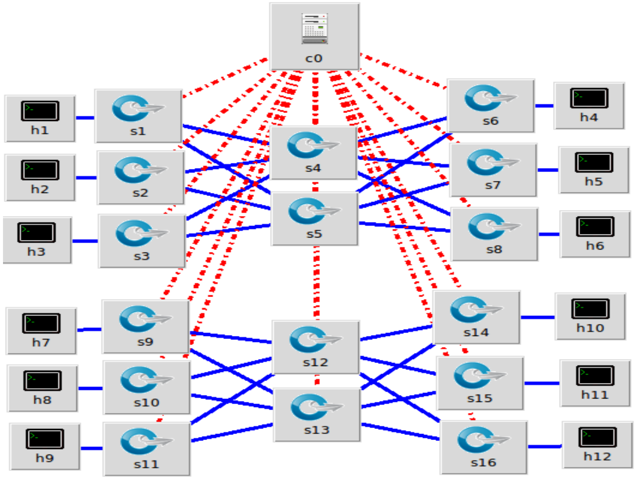


FIGURE 11. Mapped Mininet topology for Clos switch fabric for three fibers and two wavelengths.

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

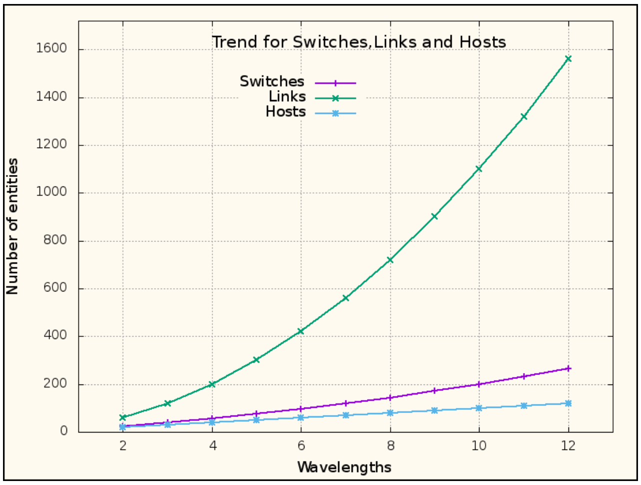


FIGURE 12. Trend for number of switches, hosts and links for Clos fabric topology in Mininet with wavelengths.

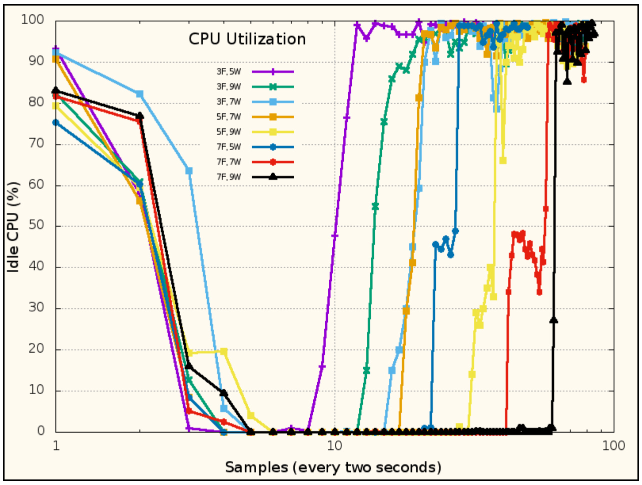


FIGURE 13. CPU utilization for creating Mininet topologies with various number of fibers and wavelengths.

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.

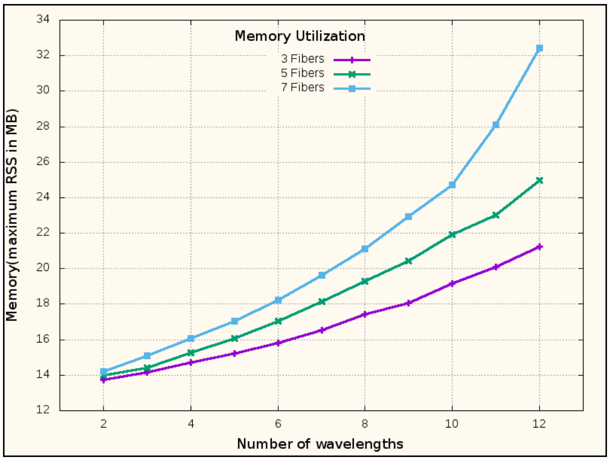


FIGURE 14. Memory utilization for creating Mininet topologies with various number of fibers and wavelengths.

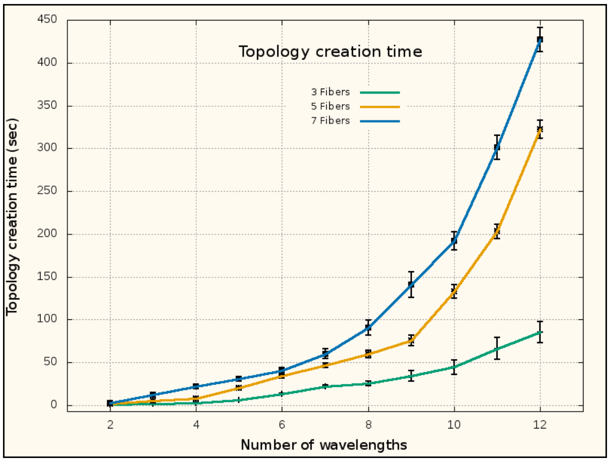


FIGURE 15. Mininet topology creation time (mean and standard error) for various number of fibers and wavelengths.

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.

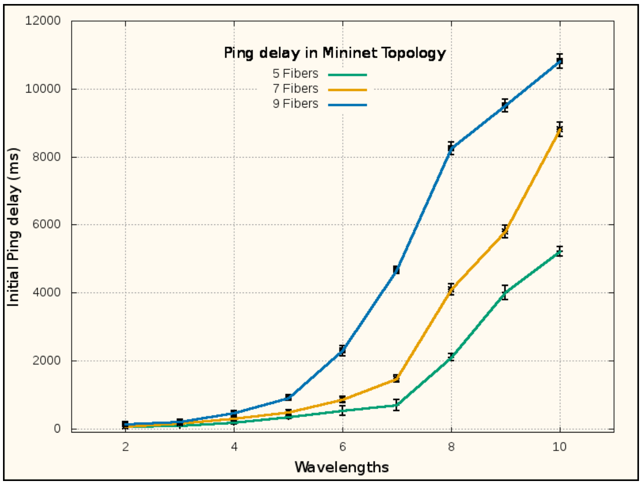


FIGURE 16. Initial Ping delay for creating Mininet topologies for various number of fibers and wavelengths.

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).