Architecture Design and Experimental Platform
Demonstration of Optical Network based on OpenFlow Protocol

Fangyuan Xing¹, Honghuan Wang¹, Hongxi Yin¹*, Ming Li¹, Shenzi Luo¹, Chenguang Wu²
¹Lab of Optical Communications and Photonic Technology, School of Information and Communication Engineering, Dalian University of Technology, Dalian 116024, Liaoning, China
²HAEPC Information and Telecommunication Company, Zhengzhou 450052, Henan, China

E-mail: hxyin@dlut.edu.cn

Abstract. With the extensive application of cloud computing and data centres, as well as the constantly emerging services, the big data with the burst characteristic has brought huge challenges to optical networks. Consequently, the software defined optical network (SDON) that combines optical networks with software defined network (SDN), has attracted much attention. In this paper, an OpenFlow-enabled optical node employed in optical cross-connect (OXC) and reconfigurable optical add/drop multiplexer (ROADM), is proposed. An open source OpenFlow controller is extended on routing strategies. In addition, the experiment platform based on OpenFlow protocol for software defined optical network, is designed. The feasibility and availability of the OpenFlow-enabled optical nodes and the extended OpenFlow controller are validated by the connectivity test, protection switching and load balancing experiments in this test platform.

1. Introduction
Optical networks have increasingly become important information infrastructure because of having many advantages such as the ultra-high speed rate, the ultra-wide bandwidth, the ultra-great capacity and the long transmission distance. The software defined optical network (SDON) is a novel optical network architecture, in which the users and operators can dynamically design the structures and functions through software programming in order to respond to the requests rapidly, utilize the resources effectively and provide the services flexibly [1].

In 2008, the concept of software defined network (SDN) is firstly proposed by Professor Nick McKeown in Stanford University [2]. SDN, a novel network architecture, enables the network control to become programmable and the underlying infrastructure to be abstracted from applications and network services through decoupling network control and forwarding [3]. By the year 2011, KDDI R&D Laboratory in Japan and Beijing University of Posts and Telecommunications firstly introduced SDN into optical networks [4]. In 2013, High Performance Networks Group (HPNG) in UK applied SDON to the cloud service architecture [5]. Though the researches on the SDON have made some progresses recently, the layered architecture of the SDON optical nodes and the experimental verification on the protocols, the strategies and the functions of the more complex network topology
still need further exploring. Consequently, a novel OpenFlow-enabled optical node is proposed and an open source OpenFlow controller is extended on routing strategies in this paper. Moreover, the experiment platform based on OpenFlow protocol for software defined optical network is designed and the performance of the OpenFlow-enabled optical nodes and the extended OpenFlow controller are experimental evaluated.

2. Architecture design of OpenFlow-enabled optical node

2.1. Architecture of optical node

The architecture of OpenFlow-enabled optical node is shown in Figure 1.

The OpenFlow-enabled optical node consists of an Open vSwitch software, a translation layer software, an FPGA and an OXC/ROADM device. The Open vSwitch is a multilayer virtual switch in accord with the open source Apache 2.0 [6]. Open vSwitch software comprises a secure channel, a flow table and an OpenFlow protocol [7]. The OpenFlow controller communicates with the OpenFlow-enabled optical nodes via the secure channel of the Open vSwitch. The flow table is used to store commands from the controller while the OpenFlow protocol acts as the signaling between the OpenFlow controller and optical nodes [8]. The translation layer software composed of optical-switch-device control-table and serial communication module, is responsible for reading out the input, output and wavelength information from the flow table. The FPGA consisting of optical switch-matrixes level configuration table and serial communication module, communicates with the translation layer software via serial ports and controls the OXC/ROADM devices according to the level configuration table.

The signaling proceeding of the OpenFlow-enabled optical node is shown as below. When the OpenFlow-enabled optical node receives a packet that does not match with any flow entry in the flow table, it sends the packet to the OpenFlow controller. The OpenFlow controller performs routing strategies based on its knowledge of the whole network and then, inserts a new flow entry into the flow table of the Open vSwitch via the secure channel. The translation layer software reads out the input, output and wavelength information from the flow entry and acquires commands through looking up the optical-switch-device control-table. Afterwards, the command is passed to the FPGA via serial ports. According to the command, FPGA obtains the states of optical switch matrixes in the OXC by searching the level configuration table. Finally, the FPGA sends the states of optical switch matrixes to OXC/ROADM device via general purpose input and output ports (GPIO).
2.2. Extension of controller
An open source floodlight [9] extended on the routing strategies, is utilized as an OpenFlow controller. The routing and wavelength assignment (RWA) algorithm is added into the routing and forwarding modules, in which the Dijkstra Algorithm to find the shortest path between each node pair and the First-Fit Algorithm to solve wavelength assignment issues are deployed.

The extended OpenFlow controller workflow is shown in Figure 2. The controller discovers the link resources through sending Link Layer Discovery Protocol (LLDP) periodically. When the network adds or deletes links, the link discovery manager calls the topology manager. The topology manager computes the network routing resources to create routing list based on the Dijkstra Algorithm. When a service request is produced from users, the forwarding module receives the request, chooses the shortest path according to the routing list, assigns the wavelength utilizing the First-Fit Algorithm and passes a flow entry to the OpenFlow-enabled optical nodes.

![Figure 2. Extended OpenFlow controller workflow.](image)

In the flow entry, the cookie field comprising 56 bits binary numbers, stores the statistic information of the flow table. After numerous tests, the 16-19 bits of the cookie field are found to be empty. Consequently, the wavelength information is stored in the 16-19 bit. For instance, in Figure 3, the flow entry shows that the input port is No. 4, the output port is No. 2 and the wavelength is 1.

![Figure 3.](image)

3. Experimental platform demonstration of optical network based on OpenFlow protocol
3.1. Style and spacing
The experimental platform of optical network based on OpenFlow protocol is shown in Figure 4.

The experiment platform consists of a control plane and a data plane. The control plane comprising an extended OpenFlow controller, an Open vSwitch software and a translation layer software, is responsible for the routing decision, the wavelength assignment and the configuration of optical switch
states. In this control plane, the controller runs on the Linux operation system embedded in the IBM server while the Open vSwitch software and the translation layer software run on the Linux operation system embedded in the computer. The Open vSwitch software creates several virtual switches corresponding to OXC/ROADMs in the data plane. The topology of the virtual switches is consistent with the data plane, as shown in the virtual plane of the inset in the Figure 4. The data plane composed of FPGA and several optical nodes such as OXC and ROADM, is employed to implement the packet forwarding according to the optical switch-matrix level configuration table. The topology of the data plane includes ring, mesh and star networks. The performance of several network protocols and algorithms can be evaluated in this experiment platform, and many functions such as the multicast, the multi-granularity switching, the load balancing, the protection switching and etc., are verified.

![Figure 4. Experimental platform of optical network based on OpenFlow protocol.](image1)

3.2. Experimental demonstration

3.2.1. Connectivity test. The structure of the data plane under the connectivity test is shown in Figure 5. Two hosts (h1 and h4) are connected to the virtual switches (S1 and S3) corresponding to OXC1 and OXC3. The IP address of h1 and h4 is 10.0.0.1 and 10.0.0.4 respectively. On the host h1, ping 10.0.0.4 command is utilized to test the network connectivity. The virtual switch S1 receives the request information and looks up the flow table. If the matched flow entry is not found, S1 passes the request to the OpenFlow controller. The controller performs RWA Algorithm, chooses the shortest path OXC1-OXC3, assigns wavelength 1 and sends flow entries to S1 and S3. Afterwards, the translation layer software reads out the input, output and wavelength information, then controls the OXCs via serial ports. The OXCs toggle the switches and the light path from OXC1 to OXC3 is connected. The output waveform observed at OXC3, is shown in Figure 6.

![Figure 5. The structure of the data plane under the connectivity test.](image2)
3.2.2. Protection switching experiment. The structure of the data plane under the protection switching experiment is shown in Figure 7. On the basis of the connectivity test above, the link between OXC1 and OXC3 is disconnected. When the topology of the network is changed, the controller performs the routing strategy again and sends the flow entries. The link is switched as OXC1-OXC2-OXC3, and the service continues to be transmitted.

The experimental result of the protection switching experiment is shown in Figure 8. When the link between OXC1-OXC3 is broken, the network loses packets. After 6.98 ms, the network reestablishes the links and switches the service.
3.2.3. **Load balancing experiment.** The structure of the data plane under the load balancing experiment is shown in Figure 9. The hosts h8 and h9 are both connected to the virtual network of which IP address is 10.0.0.100.

The experimental result of the load balancing experiment is shown in Figure 10. On the host h1, the command ping 10.0.0.100 is executed. The h1 receives the replies from the host h9 and the output waveform is observed at OADM1. This result illustrates that the host h1 is connected to 10.0.0.100 via the link OXC1-OXC2-OADM1. However, when the command ping 10.0.0.100 is executed on the host h4, the h4 receives the replies from the host h8 and the output waveform is observed at OADM2. This result illustrates that the host h4 is connected to 10.0.0.100 via the link OXC3-OXC2-OADM2. The experiment above shows that the network can choose different hosts and different switches to process services when the same services are received.

![Figure 9. The structure of the data plane under the load balancing experiment.](image1)

![Figure 10. The experimental result of the load balancing experiment.](image2)

4. **Conclusion**

In this paper, a novel OpenFlow-enabled optical node is proposed. An open source OpenFlow controller is extended on the routing strategies and the experiment platform based on OpenFlow protocol for software defined optical network is designed. The feasibility and availability of the OpenFlow-enabled optical nodes and the extended OpenFlow controller are validated by the connectivity test, protection switching and load balancing experiments in this test platform. The structure of the OpenFlow-enabled optical node is clear and the function extension is flexible. The OpenFlow-enabled optical node can connect to several different optical devices via serial ports and the versatility is enhanced. The experiment platform includes different topologies and can be utilized to verify numerous protocols, algorithms and functions.

5. **References**

[1] Simeonidou D, Nejabati R and Azodolmolky S 2011 Enabling the Future Optical Internet with OpenFlow: A Paradigm Shift in Providing Intelligent Optical Network Services In Proceedings
of 13th International Conference on Transparent Optical Networks (ICTON) (Stockholm, Sweden) pp 1-4

[2] McKeown N et al. 2008 OpenFlow: enabling innovation in campus networks ACM SIGCOMM Computer Communication Review 38 pp 69-74

[3] https://www.opennetworking.org/sdn-resources/sdn-definition.

[4] Liu L Tsuritani T Morita I Guo H and Wu J 2011 Experimental validation and performance evaluation of OpenFlow-based wavelength path control in transparent optical networks Optics Express 19 pp 26578-26593

[5] Channegowda M Nejabati R and Simeonidou D 2013 Software-Defined Optical Networks Technology and Infrastructure: Enabling Software-Defined Optical Network Operations [Invited] Journal of Optical Communications and Networking 5 pp A274-A282

[6] http://openvswitch.org/.

[7] http://archive.openflow.org/.

[8] https://www.opennetworking.org/images/stories/downloads/specification/openflow-spec-v1.3.1.pdf

[9] http://www.projectfloodlight.org/.

Acknowledgments

This work is supported in part by the National Natural Science Foundation of China (NSFC) under Grant 61071123, the First HAEP Science and Technology Project in 2015 (5217Q014006V).