Resource Allocation Scheme for SDN-Based Cloud Data Center Network

Nwe Thazin†, Khine Moe Nwe†, Yutaka Ishibashi†‡
†*University of Computer Studies, Yangon* ‡*Nagoya Institute of Technology, Japan*
nwethazin@ucsy.edu.mm, *khinemoenwe@ucsy.edu.mm*, *ishibasi@nitech.ac.jp*

Abstract

Cloud Data Center Network (CDCN) is receiving considerable attention and becomes one of the hottest topics among academia and industry. It is difficult to deliver the new emerging network services in a flexible way and to fulfill the huge amount of demand with better performance in a current data center network. Traditional routing cannot provide accessible performance for all traffic types, and it can cause excessive link utilization in the network. In this paper, we propose the resource allocation scheme to support Quality of Service (QoS) for various types of traffic while maintaining the network link utilization as much as high. We present a resource allocation scheme based on Software Defined Networking (SDN) that integrates the proposed scheme to provide better performance in CDCN. We demonstrate the effectiveness of the proposed scheme on the emulated SDN network. The proposed scheme is compared with the conventional shortest path scheme. Emulation results showed that the proposed scheme provides better performance in term of packet loss rate for the QoS traffics and great improvement in link utilization.

Key Words- *Cloud Data Center Network, Quality of Service, Software Defined Networking, Link Utilization.*

1. Introduction

In recent years, cloud computing has drawn much attention to the researchers in the Data Center Network (DCN) architecture research work. Cloud Data Center Network (CDCN) is a modern DCN which can offer a shared computing resource model with flexibility, agility, and control. The main difference between CDCN and traditional DCN is virtualization that allows for great scalability, flexibility, cost-saving, energy-saving, and on-demand utility computing. Even though the CDCN architecture has great capacity in provisioning process, Quality of Service (QoS) is still playing an important role in the CDCN due to the various specific network requirements of cloud applications and services. To address this issue and to modernize the capabilities of the data center network, Software Defined Network (SDN) has been approached by many researchers in these days.

SDN is an emerging architecture that may play a critical role in future network architectures. SDN can provide a global network view of the network resources and their performance indicators such as link utilization and the network congestion level. The main idea of SDN is to separate the network intelligence from the forwarding device and logically place it in the external entity which is called controller. OpenFlow protocol (OF) [1] is used to exchange data between controller and forwarding devices in SDN architecture. Due to its flexibility and responsive to rapid changes, SDN is proper for an emerging technology like 5G and cloud data center network (CDCN).

Leveraging the advantage of centralized control in SDN, network-wide monitoring and flow-level scheduling can be used to achieve high QoS for cloud applications and services such as voice over IP, video conferencing and online gaming. The utilization of the link resource allocation that meets the QoS requirements for all the services is one of the critical issues in the current CDCN. When traditional routing methods are used in DCN, flows are forced to preempt the shortest path to be routed and forwarded. Only sharing the shortest path may lead the link load overhead while other links are under low load [2]. Another major challenge of CDCN is a suitable path selection among the multiple paths to optimize the overall network link utilization. To solve these problems, SDN has a possible solution by using Traffic Engineering (TE) with multipath routing. TE can be used to optimize the overall network resource utilization and to improve network performance by measuring and controlling network traffic.

This paper focuses on the problem of CDCN traffic management and attempts to maintain the link utilization as much as high. As network resources, we emphasize only on the links utilization in this work. Link utilization is one of the main factors that can impact to the performance of network applications and services. When the network link utilization is too low for all the types of traffic to move through at once, there becomes congestion. If congestion happens in the network, the network will face the packet loss. When the packet loss is present on the network, the users will experience large delays and service degradation. A solution is needed to support the network communication quality for cloud users to use the services without service degradation too much. We propose a resource allocation scheme based on SDN for TE in SDN based CDCN to support the QoS requirements for individual flow. Our goal is to improve the link utilization while providing the requires bandwidth resources and less packet loss rate as the QoS factor in the overall network. To realize our proposed scheme, we design a resource allocation system based on SDN with four main modules: topology discovery module, network monitoring module, multipath route calculation module and queue scheduling module. We implement the proposed scheme on the SDN test bed to support our idea and evaluate its performance.

The rest of the paper is organized as follows. Related work is presented in Section 2. Section 3 describes the proposed scheme and explains the proposed system modules. The test-bed environment and evaluation method are shown in Section 4. Evaluation results are presented in Section 5. Finally, Section 6 concludes the paper.

2. Related work

SDN has become a promising network technology, and it has been deployed in CDCN networks. SDN provides the opportunity to achieve better management of network resources by leveraging the advantage of centralized control, network-wide monitoring and flow level scheduling. This section covers an existing literature review of traffic engineering and resource allocation for SDN based networks. Generally, network paths are often highly congested in CDCN due to a large number of users. Service provider allocates the minimum bandwidth by applying policy based on the type of flows.

Queue scheduling is the most common tool to enforce QoS in the data plane and it allows to shape and prioritize traffic to share the bandwidth. In [3], Baek et al. verified and implement a QoS guarantee function and demonstrated how to provide bandwidth guarantee with OpenFlow by using queuing techniques. In [4], Li et al. implemented a queue scheduling technique on SDN switches to achieve quality of service (QoS) for cloud applications and services. In [5], Krishna et al. proposed the system design to provide bandwidth guarantee used queue and meter for traffic shaping.

Some research work studies the SDN control framework approach. Chenhui et al. [6] proposed a QoS-enable management framework to guarantee the QoS of specific flow and employ the queue technique and policy to satisfy the requirement of service. Tomovic et al. [7] presented a new SDN control framework for QoS provisioning. The framework could provide required QoS level for multimedia applications automatically and flexibly. Their work focuses on the framework not queuing technique. In [8], Boley et al. developed a QoS framework to achieve optimal throughput for all QoS flows with the help of meters function. They presented a static meter to improve the throughput over a network by using meter-based rate limiter. In [9], Jaward et al. proposed the policy-based QoS management framework to achieve end-to-end QoS with rerouting and rate limiting. User-reservation based end-to-end dynamic bandwidth allocation procedure in a pure SDN environment was proposed in [10, 11]. Akella et al. [12] studied bandwidth allocation for ensuring the end-to-end QoS guarantee of each cloud user based on SDN. Their work emphasizes on bandwidth allocation with queueing techniques considered the performance metrics of response time and the number of hops. Durairaj et al. [13] proposed a new effective cloud resource allocation with the improvised genetic approach. Their work focused on the resource utilization problem among the virtual machine.

Most of the existing works focus on traffic scheduling and queuing mechanism to guarantee network bandwidth for different types of services. In the traditional TE, the average link utilization and the maximum link utilization are used to measure the distribution of the overall network. Base on the existing work, we propose a resource allocation scheme by combining the queuing technique to support QoS for different types of traffic in SDN-based CDCN by using multipath to provide a way to use network resources better especially in link utilization. The goal of our proposed scheme is to support QoS for various types of traffic while maintaining the link utilization.

3. Proposed resource allocation scheme

In this section, the proposed resource allocation scheme based on SDN is described in Figure 1.

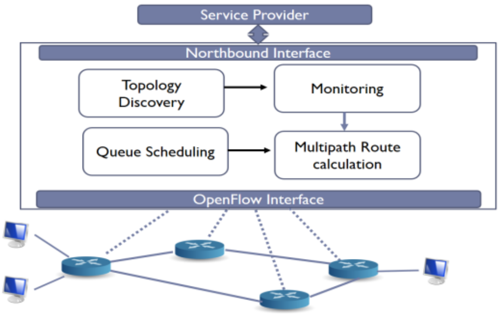

Figure 1. Proposed resource allocation system design

The system includes four main modules: topology discovery module, network monitoring module, multipath route calculation module and queue management module. The overall system modules are explained in section 3.1.

3.1 System modules

As shown in Figure 1, each module has its own functions and its functions are linked with each other in the proposed scheme. Below we will explain the proposed system modules individually.

A. Topology discovery module

This module is used to discover the SDN switches connected to the controller and have knowledge of the links between them to calculate a route for the network connection. The route cannot contract without discovering the information about hosts, switches and links in the network. To acquire topology and connection information, the module firstly sent out the Link Layer Discovery Protocol (LLDP) packets to all the connected switches through packet_out messages. After that, the messages instruct the connected switches to send LLDP packet_out messages over all its ports to other connected devices. Then this message would be delivered to the controller as packet_in messages since the switch does not have a flow entry for this LLDP message. These packet_in messages contain information about switch's port that the specific host connects to. SDN controller creates a connection based on these packet_in message. In this way, global topology information can be gained. LLDP messages are periodically exchanged to check whether the connection links go up or down. The collected information of switches and links, including MAC and IP address of all the connected hosts in a database called topology database.

B. Network monitoring module

This module is used to do real-time measurements of the network. The controller keeps tracks of how much bandwidth is allocated in the network. In order to achieve some information like link utilization and the network topology updates, the controller listens to asynchronous messages such as OFPT_PACKET_IN message, OFPT_FLOW_ REMOVED message and OFPT_PORT_STATUS message from each switch. Monitoring module tracks the amount of traffics by periodically polling flows statistics such as received and transmitted bytes or packets from all connected switches and takes snapshot the current network status. The module calculates link utilization and available bandwidth for bandwidth allocation. To calculate the link utilization of each link i for every time unit can be computed by using the number of transmitted bytes from the port statistic as follows:

$$LU_i=[B(i,t_{j+1}) - B(i,t_j)] \,/\, [(t_{j+1}) - (t_j)] \quad (1)$$

Where t_j , t_{j+1} indicate the two consecutive responses time and the number of transmitted bytes reported at time t_j for link i is denoted as $B(i,t_j)$. $B(i,t_{j+1})$ indicate the number of transmitted bytes reported at time t_{j+1}. Then, the available bandwidth (ABW_i) of each link can be computed simply by subtracting the link utilization (LU_i) from the network bandwidth capacity (BW_i) as follows:

$$ABW_i=BW_i - LU_i \quad (2)$$

After calculating available bandwidth, the monitoring module sends this information to the multipath route calculation module to compute routes to deliver the traffic form the source node to the specific destination node.

C. Multipath route calculation module

This module uses the network topology information from the topology discovery module and the traffic statistics from the monitoring module to compute multiple paths and push the resulting computation as flow rules to the SDN switches. Route calculation module calculates the shortest path tree from each source node to all the destinations by applying a shortest path finding algorithm. The modified Dijkstra's shortest path algorithm [14] is used to find a set of candidate paths between a pair of source and destination. All the paths are stored in HashMap <key, value> form [15] is used to store all the paths, and later the controller will use to determine the routes for different types of traffic with their QoS constraint. When a new flow arrives to the OF switch, it will send to the controller if the OF switch does not have flow entry for it. According to the flow information including in the packet header fields, the controller will select a suitable path with sufficient amount of bandwidth available for it and send back to the OF switch as flow entries for packet forwarding.

D. Queue scheduling module

This module is responsible for configuring queues on the output interfaces and maintains the queue configuration information. Each output interface can configure eight queues as a maximum number of queues per interface. Only three queues are created for each output interface of the switch for our study. The flows are classified into different levels and allocate network resources dynamically to provide high QoS for each traffic. Different type of traffic will be transmitted through different queues. For example, the QoS-flows will queue into the high priority queue to acquire sufficient bandwidth resources since the cross-traffic queue has the lowest priority.

The type of flow can be differentiated in various ways based on the parameter of the internet protocol such as source IP address and port number, MAC address, type of service (ToS) bits and differentiated services code point (DSCP). In our study, we simply use <IP source address, IP destination address, port number, flow priority> as a matching field of the flow tuple. We differentiate the traffic flows as QoS-flow and cross traffic flow. Cross traffic flow is served with simple Best-effort fashion without any QoS requirement while QoS-flow has some constraint QoS action. QoS flow can be also separated into different types with different QoS.

3.2 Procedure of proposed resource allocation scheme

The overall procedures and the outline of the proposed resource allocation system work is presented in Algorithm 1 (see Figure 2). Let us consider the network traffic as in two types: QoS and non-QoS. Whenever a new flow is arrived, the controller extracts flow information such as source node, destination node and request bandwidth. The controller checks the flow priority information which can show the incoming flow is QoS or not. If it is QoS flow, the multipath route calculation module calculates all the available possible paths between the source and the destination nodes.

After calculation possible path lists, check the available bandwidth of the path which can be implemented by using the statistic of the network monitoring module. If it is enough for the bandwidth guarantee rate, the controller selects the path as the optimal candidate path for routing. If it is not enough, we simply remove the link to avoid the link performance degradation. After selecting the routing path, the controller updates the flow table of the switches along the path. Then, QoS mapping is implemented for QoS flow with priority queue to provide bandwidth guarantee. If the flow is non-QoS type, we simply use the traditional routing to route the flow from the source node to the destination node. There is no QoS implementation of these type of flow and they will serve as traditional best effort fashion.

Algorithm 1: Outline

```
while true do
  If new flow arrives then
    extract the flow information
    check the flow priority
      If flow priority = QoS then
        calculate all possible paths between source and
        destination by using multipath routing
        If the possible paths > 1 then
          check the available bandwidth
            If the available bandwidth < guarantee
            bandwidth then
              remove the path
            else select as optimal candidate path
            endif
        endif
        update the flow table along the path
        queue in priority queue interface of each switch
        reserve request bandwidth along the path
      else calculate the route with traditional routing
      endif
  endif
end while
```

Figure 2. Outline of proposed resource allocation scheme

4. Evaluation method

In this section, we introduce the prototype implementation and demonstration of our proposed resource allocation scheme with simple network topology. An emulated OpenFlow environment is configured and used to validate the proposed solution by using Mininet [16] emulator. The Ryu [17] controller is used as an SDN remote controller in the control plane of the OVS for our proposed system.

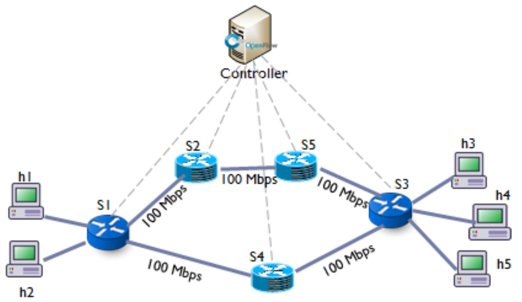

Figure 3. Network topology

The network topology is implemented by using python script in the Mininet emulator. Figure 3 shows the simulated network topology for our experiment. The proposed resource allocation scheme is demonstrated with simple network topology which consists of five switches and five hosts. The Open Virtual Switches (OVS) are used in the network topology. All the network link between OVS switches have been set to 100 Mbps, and the capacity of the output ports in all the OVS switches are also100 Mbps. Hosts h1 and h2 are traffic senders via OVS switch S1 and h3, h4 and h5 are the traffic receivers via OVS switch S3. OVS switches S2, S4 and S5 are the intermediate switches in the prototype.

Table 1: Network experiment flows

Src	Dst	Flow Type	Rate
h1	h3	QoS Flow 1	80 Mbps
h1	h4	QoS Flow 2	80 Mbps
h2	h5	Cross-Traffic	80 Mbps

In order to simulate the UDP traffic, we customize the iperf [18] network test tool and generate three flows. To observe the flow performance for our proposed scheme, we use two QoS flows and a cross-traffic. The host pair (h1-h3) means the traffic from h1 to h3 and it is denoted as QoS Flow 1. The host pair (h1-h4) means the traffic from h1 to h4 and it is denoted as QoS Flow 2. Then the host pair (h2-h5) means the traffic from h2 to h5 and it denoted as cross-traffic. The minimum bandwidth request ratios for both QoS flows and cross-traffic are shown in Table 1.

If the flow request exceeds the available bandwidth which is defined by the queue scheduling module to serve for this specific type of flow, the service will not be guaranteed. We set the flow priority for QoS flow 1 as a higher priority than QoS flow 2. Cross-traffic flow is a simple best effort flow and it has the lowest priority. The cross-traffic is generated to change the congestion level for the demonstration purpose. The multipath routing module will calculate the new path to reroute the high priority flow when the network is highly congested.

To limit the maximum traffic rates for different flows, we set three different queues with different rates for all the interfaces of the intermediate switches (S2, S4 and S5). Queue configuration setting and queue mapping with specific flow type is shown in Table 2.

Table 2: Queue configuration and queue mapping with flows

Flow Type	Queues	max-rate	min-rate
cross-traffic	q0	10 Mbps	–
QoS Flow 2	q1	30 Mbps	–
QoS Flow 1	q2	60 Mbps	–

The example command to create the queues is shown below.

```
ovs-vsctl -- set port s2-eth2 qos=@newqos -- --id=
@newqos create qos type=linux-htb other-config:max-
rate=100000000 queues=0=@q0,1=@q1,2=@q2 -- \
--id=@q0 create queue other-config:max-rate=10000000
–
--id=@q1 create queue other-config:max-rate=30000000
–
--id=@q2 create queue other-config:max-rate=60000000
```

The priority flow rules are set in Mininet topology script file for both QoS flows. QoS flow 1 is mapped with queue 2 and QoS flow 2 is mapped with queue 1 of output interface in all intermediate switches. Example flow rule command is shown below.

```
ovs-ofctl -O OpenFlow13 add-flow s2
priority=10,table=0,ip,ip_src={0},ip_dst={1},\
actions=set_queue:2,output:2'.format(h1.IP(),h3.IP())
```

After setting flow rule in all the intermediate switches, QoS flow 1 (h1-h3) is sent at time zero for 30 seconds. After five seconds later, QoS flow 2 (h1-h4) is sent for 30 seconds while cross-traffic start at another 5 seconds later for 10 seconds. According to our prototype network topology, there are two possible traveling paths between traffic senders and traffic receivers.

First Path: S1 → S4 → S3

Second Path: S1 → S2 → S5 → S3

Initially, QoS flow 1 selects the first path (S1-S4-S3) among these two possible paths since the first path has the shortest length. After 5 seconds, the QoS flow 2 request starts and then it selects the second path (S1-S2-S5-S3) since the first path does not have enough bandwidth for it. After 10 seconds later from QoS flow 1, we generate the cross-traffic that consumes all the available bandwidth over the path. When the cross-traffic gets into the network, the QoS flows increase jitter due to the total bandwidth over a link is exceeded than the link bandwidth. The network will highly congest since the current bandwidth cannot provide the request of three test flows simultaneously.

5. Evaluation results

Here, we make a comparison of our experimental results by using the proposed scheme with the traditional shortest-path routing scheme. We calculated the available bandwidth to know the overall link utilization of the intermediate switches, S2, S4 and S5. We calculated the link utilization from the polling statistics of switch ports and we set the polling interval at 5 sec. The overall link utilization in two approaches: single path and multipath approaches, are shown in Figures 4 and 5, respectively.

In Figure 4, we see that the link utilization drops as soon as QoS flow 1 enters the network at polling interval 2, and it consumes the available bandwidth along the path. It passes through the network via the first path. After 5 seconds later, polling interval 3, QoS flow 2 is started and it also consumes the link utilization and it selects the second path. When the cross traffic is started in polling interval 4, there is not enough bandwidth for these three types of flows simultaneously. Before cross traffic release the path, the network is highly congested and there is no available bandwidth in this stage till to the polling interval 11 because they are sharing a single path to route flows from sources to destinations. It cannot achieve the proper link utilization with single path routing. On the other hand, Figure 5 reveals that the multipath routing provides great improvement of link utilization and balance network traffic as the traffic bound to a destination is split across multiple paths to that destination. According to the results of Figure 4 and Figure 5, the proposed resource allocation scheme provides better link resource utilization over the network.

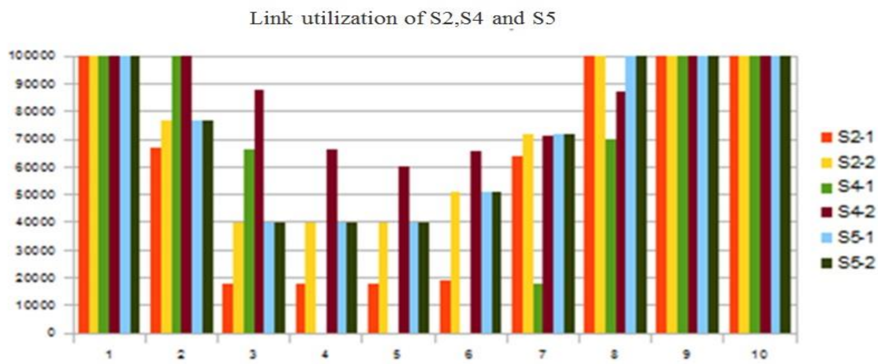

Figure 4. Overall link utilization of S2, S4 and S5 with traditional single path routing

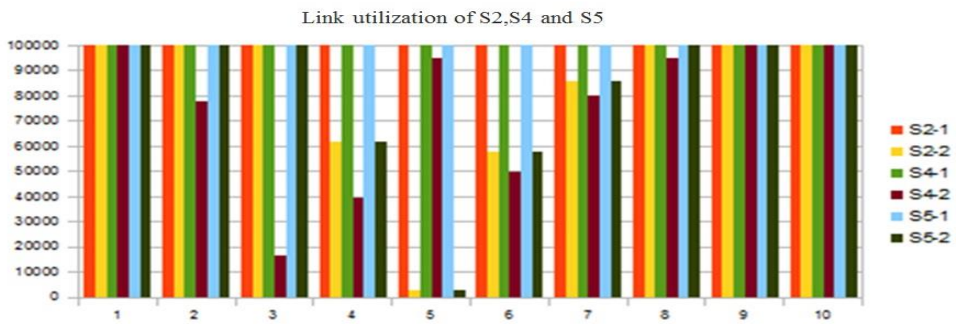

Figure 5. Overall link utilization of S2, S4 and S5 with multipath routing.

We also measured the number of packets (i.e., throughput) over queues in all the intermediate switches. The number of packets over different queues for both single path and multipath approach are shown in Figures 6 and 7, respectively. Figure 6 shows that QoS flow 1 selects the first shortest path (S1-S4-S3) and it mapped with the high priority queue q2 of the intermediate switch S4's interface. So, only queue q2 of the S4 has the packets of the QoS flow 1. When QoS flow 2 starts, there is no available bandwidth in the first shortest path, the second possible path (S1-S2-S5-S3) is selected by QoS flow 2. Then, the QoS flow 2 is mapped with the queue q1 of all the intermediate switches S2 and S5's interfaces.

When the cross-traffic starts, there is no available bandwidth in both paths, it shared the second possible path (S1-S2-S5-S3) with QoS flow 2 since the priority of the QoS flow 2 is lower than the QoS flow 1. Then, the number of packets of the cross-traffic is mapped with the interface q0 of S2 and S5. The link load distribution nature of the multipath with priority flow is shown in the interface q0 of S2 and S5. Although QoS flow 1 selects the first shortest path (S1-S4-S3), the flow is distributed to second shortest path and map with the q2 in all the intermediate switches S2, S4 and S5's interfaces. Both QoS flow 2 and the cross-traffic are also shared and distributed among the path. Compare with Figure 6, the number of packets counts in the multipath routing is obviously larger than the single path routing. Therefore, the results of the average throughput of the overall queues is better in our proposed resource allocation scheme.

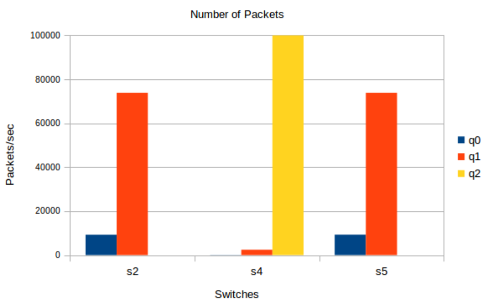

Figure 6. Number of packets over different queue with traditional single path routing

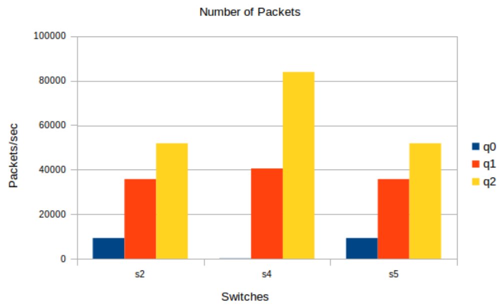

Figure 7. Number of packets over different queue with multipath routing

Since the packet loss rate is also a QoS parameter which is widely used to meet the satisfaction of QoS requirements for the cloud users, we demonstrated the comparison of packet loss rate in the two aforementioned schemes. The packet loss is unavoidable since the total requested bandwidth of the three types of flows is higher than the maximum available bandwidth of the links (100 Mbps). According to the results of Figure 8, the total packet loss rates of QoS flow 1 and QoS flow 2 are decreased in our proposed scheme than the traditional routing scheme. Therefore, the proposed allocation scheme can provide better performance in term of packet loss rate for the QoS traffics.

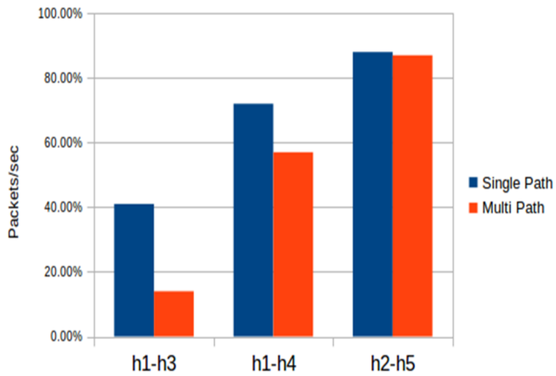

Figure 8. Comparison of packet loss rate

6. Conclusion

In this paper, we focused on the resource allocation of SDN-based CDCN by using multipath and queuing technique. Our goal is to improve the link utilization while reducing the packets loss rate as the QoS factor in the overall network. To realize this goal, we proposed the resource allocation scheme to allocate the network traffic dynamically by using the available bandwidth which is provided from the network monitoring module. We also demonstrated our proposed scheme with the simple network topology in the Mininet emulation environment. The results of the experiments showed that the proposed resource allocation scheme achieves better performance in term of link utilization and packet loss rate than traditional shortest-path routing scheme. We are still implementing the prototype system of providing a bandwidth guarantee network. The utilization of link resources is insufficient; this is still a challenging problem in CDCN. We still need to implement more realistic techniques such as effective queue scheduling in the data plane and apply meter feature of the OpenFlow protocol in the control plane. In our future research, the congestion management scheme will be implemented to improve the throughput for all the QoS flows in SDN-based environments.

References

[1] "OpenFlow Switch Specification". Available: https://www.opennetworking.org/wp-content/uploads/.../ OpenFlow-spec-v1.3.1.pdf. Accessed: August1,2015

[2] T. Wan, L. Guo, Y. Ximin, and C. Fan, "OpenFlow-based path-planning with bandwidth guarantee in the inter-datacenter network," Journal of South -Central University for Nationalities (Nat. Sci. Edition), vol.35, no. 2 (2016).

[3] S. U. Baek, C. H. Park, E. Kim and D. Shin, "Implementation and verification of QoS priority over software-defined networking," In Proc. of the International Conference on Internet Computing (ICOMP). The Steering Committee of the World Congress in Computer Science, Computer Engineering and Applied Computing (WorldComp), 2016.

[4] F. Li, J. Cao, X. Wang, Y. Sun and Y. Sahni, "Enabling Software Defined Networking with QoS Guarantee for Cloud Applications," *2017 IEEE 10th International Conference on Cloud Computing (CLOUD)*, Honolulu, CA, 2017, pp. 130-137.

[5] H. Krishna, N. L. M van Adrichem, and F. A. Kuipers, "Providing bandwidth guarantees with OpenFlow," in Proc. of IEEE 2016 Symposium on Communications and Vehicular Technologies (SCVT), pp. 1-6, 2016

[6] X. Chenhui, B. Chen, and H. Qian. "Quality of service guaranteed resource management dynamically in software defined network," Journal of Communications pp. 10.11 (2015), 843-850.

[7] S. Tomovic, N. Prasad, and I. Radusinovic, "SDN control framework for QoS provisioning," in Proc. Telecommun. Forum Telfor (TELFOR), Nov. 2014, pp. 111-114.

[8] J. M. Boley, E. S. Jung, and R. Kettimuthu, "Adaptive QoS for data transfers using software-defined networking," in Proc. IEEE International Conference on Advanced Networks and Telecommunications Systems (ANTS), pp. 1-6, IEEE, 2016.

[9] A..J. Ahmed, P. Shah, O. Gemikonakli, and R. Trestian. "Policy-based QoS management framework for software-defined networks," (2018).

[10] G. Zhang, D. Zhou, L. Zhou and X. Liu, "End-to-end dynamic bandwidth allocation based on user in software-defined networks," International Journal of Future Generation Communication and Networking. Sep. 2016, pp. 67-76.

[11] C. Zhang, X. Huang, G. Ma and X. Han, "A dynamic scheduling algorithm for bandwidth reservation requests in software-defined networks," 2015 10th International Conference on Information, Communications and Signal Processing (ICICS), Singapore, 2015, pp. 1-5.

[12] A. V. Akella and K. Xiong, "Quality of service (QoS)-guaranteed network resource allocation via software defined networking (SDN)," in Proc. IEEE 12th Int. Conf. Dependable, Autonomic Secure Comput. (DASC), Aug. 2014, pp. 7-13.

[13] M. Durairaj, and P. Kannan. "Improvised Genetic Approach for an Effective Resource Allocation in Cloud Infrastructure." International Journal of Computer Science and Information Technologies (IJCSIT) vol.6, no. 4 (2015).

[14] M. Bahr, F-J. Goetz. "Multi-path link-state routing", available at http://www.ieee802.org/1/files/public/docs 2012/new-avb-bahr linkstate-multipath-routing-0112.pdf - accessed on 2012-04-1

[15] M. Singh, N. Varyani, J. Singh, K. Haribabu "Estimation of End-to-End Available Bandwidth and Link Capacity in SDN," 2017 International Conference on Ubiquitous Communications and Network Computing. Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering, vol 218. Springer, Cham

[16] Mininet: An Instant Virtual Network on your Laptop. Available: http://mininet.org/.

[17] Ryu SDN Framework Community. (2014) Ryu SDN Framework , Available: https://osrg.github.io/ryu/

[18] "Iperf" , Abailable: http://iperf.sourceforge.net.