Mathematics and Computer Science
2020; 5(3): 64-71
http://www.sciencepublishinggroup.com/j/mcs
doi: 10.11648/j.mcs.20200503.11
ISSN: 2575-6036 (Print); ISSN: 2575-6028 (Online)

MPF-MLBS: A Multi-path Load Balancing Strategy for SDN Networks Based on Multiple Performance Factors

Daoquan Li, Haoxin Liu*, Yingnan Jin

School of Information and Control Engineering, Qingdao University of Technology, Qingdao, China

Email address:

lidaoquan@qtech.edu.cn (Daoquan Li), 2536855669@qq.com (Haoxin Liu), nan723@163.com (Yingnan Jin)

*Corresponding author

To cite this article:

Daoquan Li, Haoxin Liu, Yingnan Jin. MPF-MLBS: A Multi-path Load Balancing Strategy for SDN Networks Based on Multiple Performance Factors. *Mathematics and Computer Science*. Vol. 5, No. 3, 2020, pp. 64-71. doi: 10.11648/j.mcs.20200503.11

Received: August 13, 2020; **Accepted**: August 22, 2020; **Published**: August 25, 2020

Abstract: Aiming at the common load balancing problems in the network, a multi-path load balancing strategy (MPF-MLBS) based on multi-performance factor (MPF) for Software Defined Network (SDN) networks is proposed, which can be divided into two stages: algorithm design and strategy implementation. In the algorithm design stage, the advantages and disadvantages of the existing load balancing algorithms are analyzed, combined with the characteristics of the SDN network architecture, and the bandwidth, delay and link rate of the network link are comprehensively considered. Based on this, multiple performance factors are defined, and a load balancing algorithm based on multiple performance factors (MPF-CMP) is designed and implemented. In the strategy implementation stage, build a multi-path network topology based on the SDN architecture, and use the depth-first traversal algorithm to traverse the global network to obtain the required link information; Subsequently, the MPF-CMP algorithm and OpenFlow group table technology are combined to complete the proportional distribution of network traffic to each available path, thereby achieving multi-path load balancing of the SDN network. The simulation experiment results show that the strategy effectively exerts the SDN controller's overall network monitoring and scheduling functions, can obtain link information in real time and distribute and transmit network traffic according to the situation. The overload of a single path is avoided, the data packet transmission volume of all available paths is effectively increased, the flow transmission efficiency of the entire network is improved, and the multi-path load balancing of the SDN network is realized.

Keywords: Load Balancing, Multiple Performance Factors, Multipath Routing, OpenFlow, SDN

1. Introduction

With the diversified development of network applications and the increasing demand of network users for service quality, how to efficiently improve network performance has become a popular direction for network research in recent years. Among them, load balancing, as one of the common strategies to improve network performance, has always been a hot issue but also a difficult issue [1]. Load balancing means to balance and distribute the load (work tasks) to multiple operation units for operation, so as to work together to complete tasks. Load balancing can improve network performance because it provides a transparent, inexpensive and effective method to expand the bandwidth of servers and network devices, strengthens the processing capacity of network data, and improves network availability and flexibility [2].

There are many researches on load balancing, among which the research on new types of networks represented by Software Defined Network (SDN) is the most significant. Due to the static state of traditional networks, network administrators cannot effectively obtain real-time network link conditions, and thus cannot allocate and schedule network traffic transmission in real time according to actual conditions. Many problems such as single path overload, uneven link resource allocation, and overall low flow transmission efficiency are prone to occur, and it is difficult to adapt to the development and needs of today's network environment [3]. In this context, the research on load balancing based on software-defined networking came into being, and it was gradually accepted and valued by everyone with its new network concept. As a new type of network architecture, logically centralized control planes and flexible open interfaces will help SDN change the static state of traditional networks. This provides theoretical and technical support for

realizing the transformation from single-path routing strategy to multi-path routing strategy in traditional networks, thereby increasing the real-time and efficiency of network traffic transmission. It solves the problem of excessive load on a single path and improves the utilization of link resources in the entire network [4].

Nevertheless, the existing load balancing strategies based on software-defined networks still have many shortcomings. Some multi-path routing strategies are improved on the basis of traditional single-path routing strategies. The optimal path among multiple paths is selected for network traffic transmission, which does not fundamentally solve the problem of uneven distribution of link resources [5]. Some multipath routing strategies use multiple paths to evenly distribute traffic for network traffic transmission. The disadvantage of this strategy is that there is no on-demand distribution of link resources based on the performance and actual conditions of multiple paths. It is easy to cause problems such as partial link congestion and overall low flow transmission efficiency [6].

This paper comprehensively considers the characteristics of SDN network architecture and the advantages and disadvantages of existing load balancing strategies, and proposes a multi-path load balancing strategy for SDN networks based on multiple performance factors. Combining the centralized control characteristics of the SDN architecture, the SDN controller is used to obtain the topology of the entire network and collect multiple network performance parameters such as bandwidth, delay and link rate of each link in real time. Based on this, multiple performance factors are defined, and a load balancing algorithm based on multiple performance factors (MPF-CMP) is designed and implemented. Then combined with MPF-CMP algorithm and OpenFlow group table technology [7] to complete the proportional distribution of network traffic to each available path. The simulation experiment results show that the strategy effectively increases the total amount of data packet transmission of all available paths, reduces the load of a single path in the SDN network, improves the traffic transmission efficiency of the entire network, and realizes the multi-path load balancing of the SDN network.

2. Theoretical Background

2.1. SDN and Load Balancing

As a new type of network architecture, SDN has multiple features such as separation of transfer and control, centralized control, open interfaces, strong scalability and security, and has become one of the network architectures commonly used by network researchers. Its architecture is shown in Figure 1. The biggest difference between the SDN network and the traditional network is the separation of control and forwarding. The centralized monitoring and scheduling of network equipment at the infrastructure layer is carried out through the controller, which breaks the problem of the binding of equipment by suppliers in traditional networks. Network administrators can optimize the overall performance of the network architecture and schedule functions from the control level, which improves the deployment speed of new services [8].

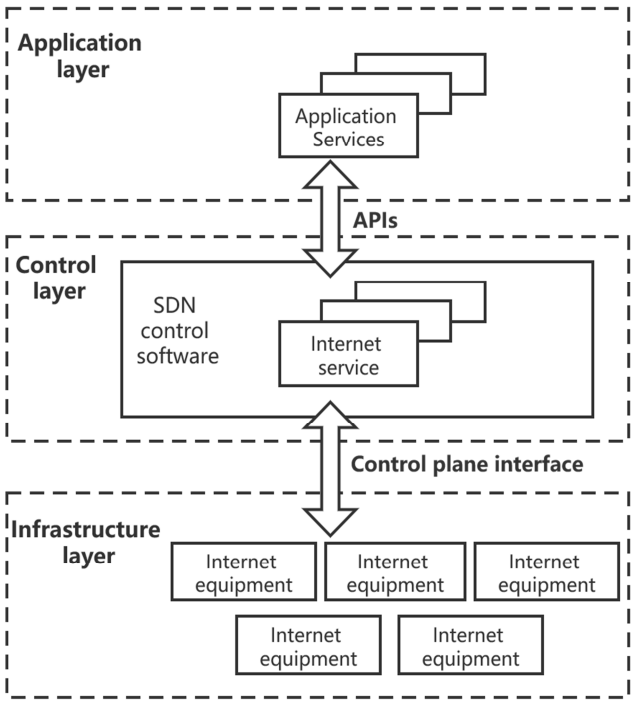

Figure 1. *SDN architecture diagram.*

Using the SDN architecture to study load balancing strategies can give full play to the monitoring and scheduling functions of the SDN controller for the global network, obtain link information in real time, and distribute and transmit network traffic according to the load balancing algorithm; At the same time, the load balancing strategy can be adjusted according to user needs, and the load balancing can be realized by the way of sending the flow table by the SDN controller [9].

2.2. Research on Load Balancing Based on Multipath Routing

With the development of network applications, network topologies are becoming more and more complex, and there are usually multiple links between different network nodes, which extends many load balancing studies based on multipath routing. In recent years, the research on load balancing for multipath routing has made the following progress: Chiesa et al. conducted related research on the application of Equal-Cost-Multipath (ECMP) in large-scale networks [10]. The ECMP algorithm means that in the transmission process to the same destination IP or destination network segment, there are multiple different routing paths with the same cost value simultaneously participating in the transmission of traffic. When some of the paths fail, other available paths can be substituted to complete the data forwarding process, which can improve the link utilization rate and realize the route redundancy backup function. As one of the commonly used algorithms for load balancing, ECMP has a wide range of applications. Since the core of the ECMP algorithm is to use all equal-cost paths in a balanced manner,

this will recognize the Cost value of the transmission path as the same. However, performance parameters such as bandwidth and delay of multiple paths in a real network are different, and the use of traditional ECMP algorithms will result in lower overall bandwidth utilization, especially when the path difference is large, the effect is very unsatisfactory [11]. Therefore, how to assign link weights according to the current network load status has become an important issue for improving the ECMP routing algorithm.

Aiming at the shortcomings of ECMP, Wang et al. used Weighted-Cost-Multipath (WCMP) to solve the load balancing problem under the resource constraints of the data plane [12]. The WCMP algorithm is proposed to solve the load balancing problem of data center traffic, and is mainly aimed at improving the problems of the ECMP algorithm. WCMP can deliver traffic proportionally according to the bandwidth of each path, which solves the weakness of ECMP in handling topology asymmetry, thereby improving the transmission efficiency of traffic bandwidth in the network. However, WCMP generally sets weights based on the bandwidth attributes of the paths, and does not take into account the delay changes of each link during the traffic transmission process, and there is still much room for improvement [13].

In addition to the algorithm research of multi-path routing, Manzanares-Lopez et al. once proposed the research of multi-path routing protocol MultiPathTCP (MPTCP) and OpenFlow [14]. MPTCP adds MPTCP components and Transmission Control Protocol (TCP) sub-flow components for each OpenFlow switch port to the traditional TCP kernel components. This can transfer the traffic of a congested path to a link with less congestion, which can be applied to network environments with different path bandwidths. MPTCP's solution uses multiple randomly selected paths. It works as a TCP process and does not support other protocols such as User Datagram Protocol (UDP). In addition, MPTCP itself cannot perform global link awareness. How to perfectly integrate with SDN is still a difficult problem. In the communication process, it is necessary to ensure that both parties have installed the MPTCP protocol stack, which brings difficulties to the widespread deployment of MPTCP [15].

In general, the research of load balancing based on multipath routing has become an important research direction to improve network performance. This not only requires the realization of multi-path load balancing and link backup, but also needs to consider multiple network performance such as bandwidth and delay in the network link to deal with a variety of complex network conditions, thereby improving the availability and flexibility of the network [16].

3. Theory and Model

This article proposes a multi-path load balancing strategy (MPF-CMP) for SDN networks based on multiple performance factors. The implementation process is divided into the following three steps:

i. Firstly the controller uses the depth-first traversal algorithm [17] to obtain the multipath topology and the status of each link, and simultaneously collects and saves the bandwidth, delay, and link rate information of each path.

ii. Secondly the multiple performance factors (MPF) of each path are calculated based on the bandwidth, delay, and link rate parameters of each path.

iii. Finally, the OpenFlow group table is installed in each path, and the calculated multiple performance factors is set to the bucket value in the OpenFlow Select group table, so that network traffic is distributed to all available paths according to the MPF ratio of each path.

Simulation results show that this strategy can effectively improve the utilization of multipath resources in SDN networks and significantly improve the transmission efficiency of traffic, and achieve load balancing of multipath routing.

3.1. Multipath link Discovery

SDN technology enables the SDN controller to obtain and manage global network topology information due to its numerical control separation and programmability characteristics. The topology discovery and management mechanism is implemented by the *Switches* module in the SDN controller sending a *packet_out* packet containing the link layer discovery protocol (LLDP) to the underlying network (OpenFlow switch). After receiving the LLDP packet, the OpenFlow switch will send a *packet_in* packet containing link information between switches to the controller, and the SDN controller then identifies and manages the network topology based on the feedback information collected by the link discovery protocol [18]. Such a link discovery mechanism consumes a large amount of communication traffic and generates a certain delay cost.

Table 1. *Depth-first traversal algorithm.*

Algorithm 1 Depth first traversal

```
Input:self,src,dst
Output:paths
    if src == dst then
      return [[src]]
    paths = []
    stack = [(src,[src])]
    while stack
      (node,path) = stack.pop()
      for next in set(self.adjacency[node].keys()) - set(path)
        if next is dst
          paths.append(path + [next])
      else stack.append((next,path + [next]))
```

In order to simplify the multipath topology discovery process, reduce communication consumption, and reduce the cost of delay, this paper uses the depth-first traversal algorithm to implement multipath topology discovery. The process of depth-first traversal path acquisition is described as follows:

Assume a given network topology $G(H,S)$, where H represents the set of hosts, and S is the set of OpenFlow switches managed by the SDN controller. To find all available multipath between any two hosts, the depth-first traversal path search algorithm is shown in Table 1.

By implementing this algorithm, you can use the SDN controller to accurately obtain the entire network topology and

link connection conditions, and provide usable information for subsequent load balancing.

3.2. Path Information Collection

During the discovery of multipath topology, the SDN controller can use the interactive information provided by the OpenFlow protocol to collect information on the parameters of each OpenFlow switch port. The OpenFlow protocol mainly supports three message types, which are $controller-to-switch$, $symmetric$ (symmetric messages) and $asynchronous$ (asynchronous messages). Each message corresponds to multiple events. The Read-Stats event in the message type $controller-to-switch$ can help the SDN controller to obtain the status information of each OpenFlow switch, including the bandwidth and delay of each port [19].

Considering that in a real network environment, the parameters such as bandwidth, delay, and link rate of each link may not be exactly the same. To ensure that the traffic allocated to each path is more reasonable, the bandwidth, delay, and link rate information of each OpenFlow switch port needs to be collected, and combined with the multiple performance factors proposed later to achieve load balancing of network traffic.

3.2.1. Maximum Remaining Bandwidth

In the process of data transmission, in order to prevent path congestion, it needs to use the SDN controller to obtain the bandwidth information of the OpenFlow switch ports and calculate the maximum remaining bandwidth of each path.

First calculate the used bandwidth of the path. The calculation formula is as follows:

$$UsedBand_s = \frac{bytes_{rx} + bytes_{tx}}{T} \tag{1}$$

Among them, $UsedBand_s$ represents the used bandwidth of port s in the acquisition interval T, $bytes_{rx}$ represents the total number of data bytes received by the port during interval T, and $bytes_{tx}$ represents the total number of data bytes transmitted by the port during interval T.

For each path $p = \{s_1, s_2, ..., s_n\}$, the maximum remaining bandwidth is the minimum remaining bandwidth value of all OpenFlow switch ports included in p, so the path maximum remaining bandwidth B_p calculation formula is as follows:

$$B_p = \min\{B_{s_i} - UsedBand_{s_i} \mid 1 \le i \le n\} \tag{2}$$

Among them, B_{s_i} represents the total bandwidth of OpenFlow switch s_i on path p.

3.2.2. Delay

In an SDN network, the link set of any path $p = \{s_1, s_2, ..., s_n\}$ should be $l = \{l_1, l_2, ..., l_{n-1}\}$, and the delay of path p should be the sum of the delays of all links l included in the path.

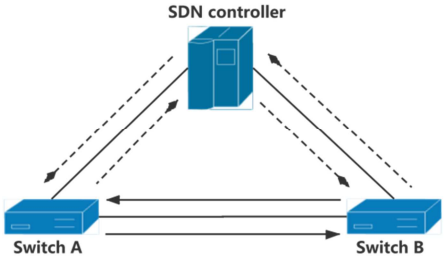

Figure 2. *Delay principle.*

The principle of obtaining a link delay in an SDN network is shown in Figure 2. Assuming that a single link contains two OpenFlow switches S_A and S_B, the SDN controller can send a data packet containing LLDP to S_A, and instruct it to forward the data packet to S_B, while recording the time stamp of data transmission. Then subtract this timestamp from the timestamp of the packet_in packet from S_B, to get the delay D_1 of switch S_A. Similarly, the delay D_2 of the switch S_B can be obtained. In addition, the round-trip delay between the controller and the switch is shown by the dashed line in Figure 2 [20]. This part of the delay is obtained by echo packet test, which is D_a and D_b respectively. Therefore, the delay calculation formula for link l is shown in equation (3):

$$D_l = (D_1 + D_2 - D_a - D_b) \tag{3}$$

Therefore, the formula for calculating delay D_p of path p is shown in equation (4):

$$D_p = \sum_{j=1}^{n-1} D_{l_j} \tag{4}$$

3.2.3. Link Rate

Different network link rates are usually different, which is determined by the nature of the hardware itself [21]. Considering the implementation difficulty and real situation comprehensively, since a path may include multiple segments of links, it is assumed that the link rate of the i segment link is V_i, the link bandwidth is B_i, and the link width is W_i. Therefore, the calculation formula of link rate V for each path is shown in equation (5):

$$V = \frac{\sum_{i=1}^{n} V_i}{n} \tag{5}$$

Among them, the calculation formula of the link rate V_i of the i segment link is shown in equation (6):

$$V_i = \frac{B_i}{W_i} \tag{6}$$

3.3. Multiple Performance Factor (MPF)

Existing multi-path routing widely uses equal-cost

multi-path (ECMP) or link-weighted multi-path (WCMP) when distributing traffic. However, in a network environment, the differences between various paths are not only reflected in the size of the bandwidth, but also characteristics such as delay and link rate will also affect the transmission performance of the path [22].

This paper comprehensively considers the parameters of multi-path bandwidth, delay, and link rate, and sets the multi-performance factor (MPF). Among them, let the multi-performance factor of path p be I_p, which is defined as formula (7):

$$I_p = \frac{B_p}{D_p} * V \tag{7}$$

In the network topology, the bandwidth of the link is usually more than 10Mbps, and the delay of each link in the traffic transmission process is usually a decimal number. When setting multiple performance factors, this article expects that the higher the bandwidth of the path, the lower the delay, and the faster the speed, the higher the multiple performance factor will be. Therefore, in formula (7), the path delay D_p is used as the denominator, and the product of the maximum remaining bandwidth B_p of the path and the link rate V is used as the numerator.

3.4. Multipath Traffic Distribution and Load Balancing

In a multipath network environment, this paper installs each OpenFlow switch that includes multipath and uses the select group table in the OpenFlow group table technology. And applying the set multiple performance factor (MPF) to the bucket weight of the action bucket in the select group table. Set the action bucket weight to w_p, and its definition is shown in formula (8):

$$w_p = I_p \tag{8}$$

In order to explain the multi-path traffic distribution process, a mathematical model should be established and simulation experiments performed. Suppose that there are n available paths $p_i (1 \le i \le n)$ between two communication terminals h_1 and h_2, and the action bucket weight is w_{p_i} and is equal to its MPF value of I_{p_i}. Whenever new traffic $T_j (j \ge 1)$ arrives, the select group table will process all the data packets contained in traffic w_{p_i} according to the different paths T_j, and hash the packets to be forwarded to all available paths p_i. Assuming that the number of data packets contained in the traffic T_j is Q, the definition of the data packet traffic allocated by each available path is shown in formula (9):

$$q_{p_i} = \frac{w_{p_i}}{\sum_{i=1}^{n} w_{p_i}} * Q \tag{9}$$

$$\sum q_{p_i} = Q \tag{10}$$

Among them, formula (10) is a constraint condition, which means that the sum of the data packets forwarded by all available paths should be the total number of data packets included in traffic T_j.

By using the select group table in the OpenFlow group table and the multi-performance factor MPF, the OpenFlow switch including the multi-forwarding path can achieve a proportional hash of the data packet during each flow forwarding process. This achieves a balanced distribution of traffic and achieves the goal of multipath load balancing.

4. Experiment and Analysis

In order to verify the previously proposed MPF-CMP, this section conducts simulation experiments on this strategy. This experiment uses Ubuntu 16.04 as the operating system, uses Mininet as the network simulation software to build the network topology, and uses Ryu software as the SDN controller. Mininet and Ryu are installed on a computer with an Intel i5 6500 processor, a main frequency of 3.2GHz, 4G memory and a 64-bit Linux operating system. The operating environment of the Ryu controller is 64-bit Python 2.7.

The MPF-CMP proposed in this paper is implemented by writing Ryu controller application files, using the Python programming language.

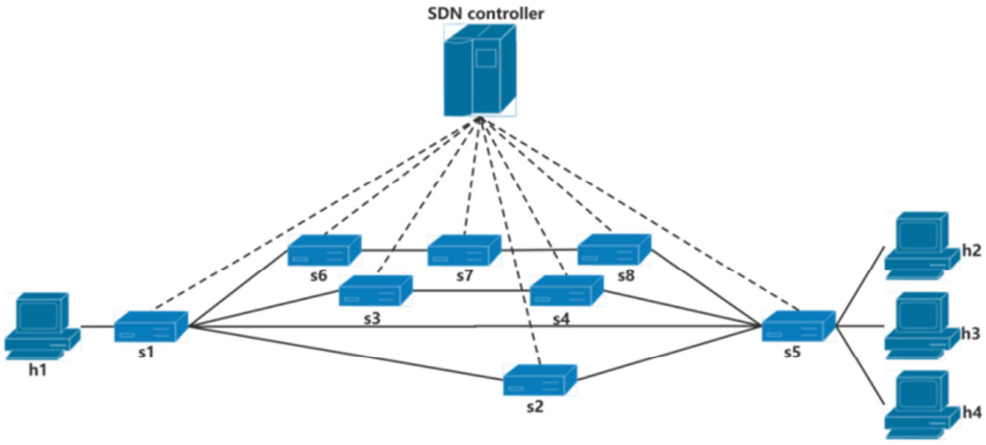

Figure 3. *Experimental network topology.*

In order to show the obvious experimental results, the network topology shown in Figure 3 is applied in this experimental topology. It contains an SDN controller, $H=\{h_1,h_2,h_3,h_4\}$ represents the host set, and $S=\{s_1,s_2,s_3,s_4,s_5,s_6\}$ is the OpenFlow switch set managed by the SDN controller. Each link applies default bandwidth, delay, and link rate parameters to simply simulate a real network environment.

To simulate the communication between hosts, this experiment uses *iperf* command to set h_1 as the service host and h_2-h_4 as the client host.

The experiment in this paper is to simulate the three client hosts of h_2-h_4 sending traffic to server host h_1 at the same time. By comparing the changes of the four forwarding ports $Pt=\{port_1,port_2,port_3,port_4\}$ of switch s_5 before and after using MPF-CMP, the effect of load balancing is verified.

The experiment first finds a total of four available paths between h_2-h_4 and h_1 through depth-first traversal: (1) $s_5\rightarrow s_8\rightarrow s_7\rightarrow s_6\rightarrow s_1$; (2) $s_5\rightarrow s_4\rightarrow s_3\rightarrow s_1$; (3) $s_5\rightarrow s_2\rightarrow s_1$; (4) $s_5\rightarrow s_1$. By using the instructions $dump-flows$, $dump-groups$ and $dump-ports$ that come with Open vSwitch to view the status information of the data flow and group table and the packet forwarding status of the port, it can be determined that $Pt=\{port_1,port_2,port_3,port_4\}$ corresponds to the path (1, 2, 3, 4).

Then, based on the path information collected by the controller, the multiple performance factor (MPF) ratio of the four paths was calculated to be 6: 7: 8: 9.

To compare the effects of load balancing, use Open vSwitch instructions to view and record the changes in the amount of forwarded data packets on port $\{port_1,port_2,port_3,port_4\}$ before and after implementing the load balancing policy.

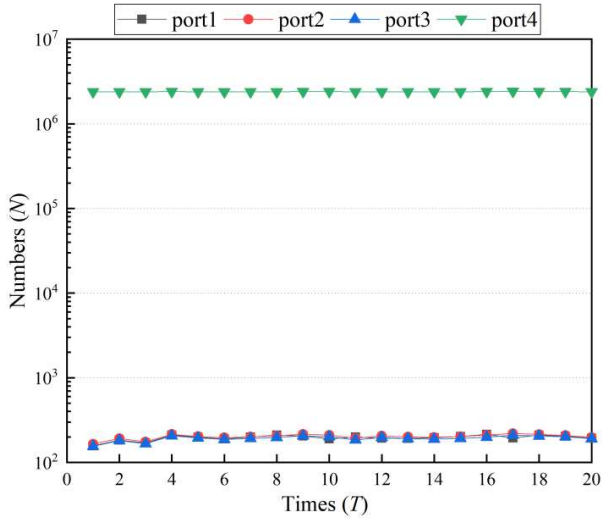

Figure 4. *Packet forwarding on each port of the switch before using MPF-CMP.*

As shown in Figure 4, before using MPF-CMP, the path corresponding to port $port_4$ is the default shortest path, and the number of forwarded data packets has reached the order of magnitude 10^6. Although the paths corresponding to $port_1$, $port_2$, and $port_3$ have higher bandwidth and lower delay parameters, they only forward packets of order 10^2. This makes the load on the path corresponding to $port_4$ high, while the load on the other three paths is low. The result of this kind of traffic distribution will cause a severe load imbalance during the peak period of traffic requests, which will greatly affect the quality of network services.

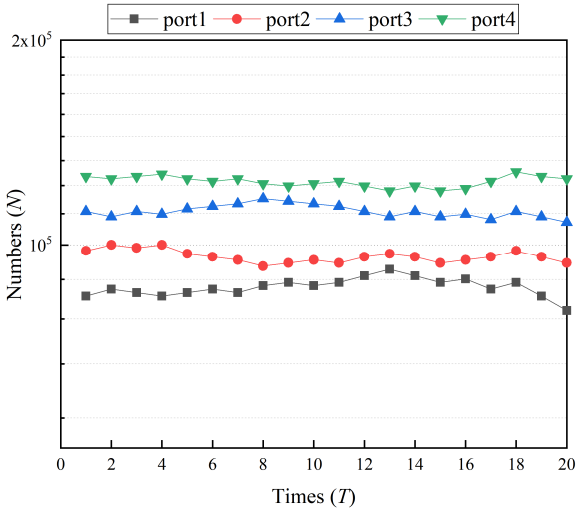

Figure 5. *Packet forwarding of each port of Switch5 after using MPF-CMP.*

Figure 5 shows the comparison of the number of forwarded packets on the four ports after using MPF-CMP in this article. Although $port_4$ is the default shortest path, because the link performance (depending on bandwidth, delay, and link rate settings) is not high, and the MPF value is not much different from other ports, the number of forwarded packets is significantly reduced; Because $port_1$, $port_2$ and $port_3$ have better link performance and higher MPF value, they can also forward a large number of data packets, thus achieving multi-path based load balancing.

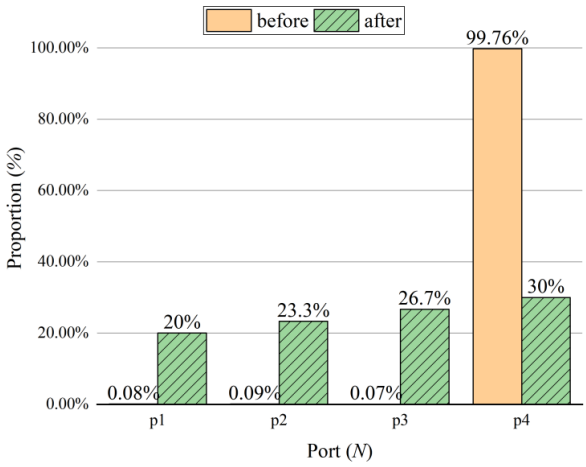

Figure 6. *Percentage of forwarded packets on each port of Switch5 before and after load balancing.*

Figure 6 shows the percentage of data packets forwarded by port $port_1 - port_4$ before and after MPF-CMP is used. From this figure, it can be clearly seen that before the implementation of load balancing, the three ports $port_1$, $port_2$, and $port_3$ only forwarded 0.08%, 0.09%, and 0.07% of packets, while port $port_4$ forwarded up to 99.76% of packets. After the implementation of load balancing, it can be found from the experimental graph that the proportion of data packets forwarded by the four ports $port_1 - port_4$ is 5.9: 6.89: 8.11: 9.1, which basically meets their corresponding MPF ratio.

The above experiment only considered the load of multipath routing under the number of four paths. To prove the universality of this strategy, Figure 7 shows the percentage of packets forwarded by port $port_1 - port_4$ under different numbers of paths. From this figure, it can be seen that when there are two paths, the proportion of forwarded packets on two ports is about 4: 6; when there are three paths, the proportion of forwarded packets on three ports is about 3: 4: 5; when there are four paths, the proportion of forwarded packets on the four ports is about 6: 7: 8: 9, which basically accords with the set multi-factor ratio. By comparing the proportion of packets forwarded by each port under different numbers of paths, it can be determined that the multi-path load balancing based on the multi-performance factor of the SDN network proposed in this paper has good applicability.

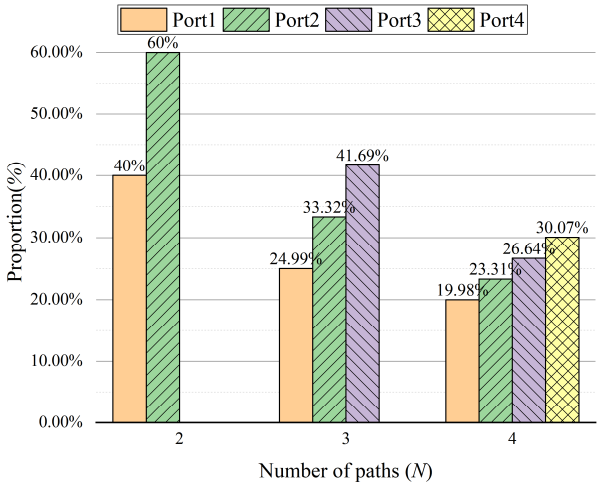

Figure 7. *Proportion of packets forwarded by each port of Switch5 under different numbers of paths.*

All the above experiments are the experiments and results of MPF-CMP. In order to prove the superiority of this strategy, Figure 8 shows a comparison of the traffic transmission time of different load balancing algorithms. It can be seen from the figure that when the traffic transmission volume is small, the traffic transmission time of the ECMP, WCMP and MPF-CMP algorithms is not much different; but as the traffic transmission volume increases, the difference in transmission time becomes more obvious. Because the MPF-CMP algorithm proposed in this paper is developed on the basis of the WCMP algorithm, which combines the advantages of the WCMP algorithm and improves the disadvantages, the MPF-CMP algorithm's traffic transmission efficiency is significantly higher than the traditional ECMP algorithm. At the same time, this has advantages over the WCMP algorithm. By comparing the traffic transmission time under different load balancing algorithms, we can confirm that the MPF-CMP proposed in this paper has good performance in improving traffic transmission efficiency and improving load balancing.

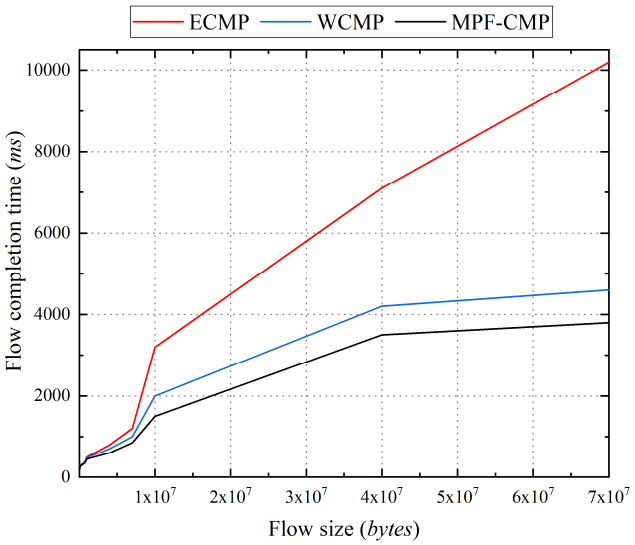

Figure 8. *Comparison of traffic transmission time of different load balancing algorithms.*

5. Conclusion

This paper proposes a multi-path load balancing strategy MPF-MLBS based on multi-performance factor (MPF) for SDN networks. This strategy first realizes multi-path search and information collection through the SDN controller, and then combines OpenFlow group table technology to achieve multi-path-based load balancing in SDN networks. When using the OpenFlow group table, this paper designs multiple performance factors. This factor comprehensively considers the bandwidth, delay, and link rate information of each path, so that network traffic can be distributed to all available paths in proportion to the multiple performance factors. The load of a single path is reduced, and the load balancing of multiple paths is realized. The final simulation experiment results also verify the feasibility and superiority of the strategy. Future research will consider how to apply this strategy dynamically when multipath performance changes.

References

[1] Liu, Yuxin, et al. "A novel load balancing and low response delay framework for edge-cloud network based on SDN." IEEE Internet of Things Journal (2019).

[2] AlKhatib, Ahmad AA, Thaer Sawalha, and Shadi AlZu'bi. "Load Balancing Techniques in Software-Defined Cloud Computing: an overview." 2020 Seventh International Conference on Software Defined Systems (SDS). IEEE, 2020.

[3] Li, Tong, Jinqiang Chen, and Hongyong Fu. "Application Scenarios based on SDN: An Overview." Journal of Physics: Conference Series. Vol. 1187. No. 5. IOP Publishing, 2019.

[4] Belgaum, Mohammad Riyaz, et al. "A Systematic Review of Load Balancing Techniques in Software-Defined Networking." IEEE Access (2020).

[5] Hossen, Md Sajid, et al. "Enhancing Quality of Service in SDN based on Multi-path Routing Optimization with DFS." 2019 International Conference on Sustainable Technologies for Industry 4.0 (STI). IEEE, 2019.

[6] Lu, Lihua. "Multi-path Allocation Scheduling Optimization Algorithm for Network Data Traffic Based on SDN Architecture." IMA Journal of Mathematical Control and Information (2020).

[7] Priya, A. Vishnu, and N. Radhika. "Performance comparison of SDN OpenFlow controllers." International Journal of Computer Aided Engineering and Technology 11.4-5 (2019): 467-479.

[8] Rehman, A. U., Rui L. Aguiar, and João Paulo Barraca. "Fault-Tolerance in the Scope of Software-Defined Networking (SDN)." IEEE Access 7 (2019): 124474-124490.

[9] Chahlaoui, Farah, Mohammed Raiss El-Fenni, and Hamza Dahmouni. "Performance analysis of load balancing mechanisms in SDN networks." Proceedings of the 2nd International Conference on Networking, Information Systems & Security. 2019.

[10] Chiesa, Marco, Guy Kindler, and Michael Schapira. "Traffic engineering with equal-cost-multipath: An algorithmic perspective." IEEE/ACM Transactions on Networking 25.2 (2016): 779-792.

[11] Zaw, Hnin Thiri. Delay-Aware Elephant Flow Rerouting in Software-Defined Networking (SDN). Diss. University of Computer Studies, Yangon, 2019.

[12] Wang, Haibo, et al. "PrePass: Load Balancing with Data Plane Resource Constraints using Commodity SDN Switches." Computer Networks (2020): 107339.

[13] Cheng, Yingying, and Xiaohua Jia. "NAMP: Network-Aware Multipathing in Software-Defined Data Center Networks." IEEE/ACM Transactions on Networking 28.2 (2020): 846-859.

[14] Manzanares-Lopez, Pilar, Juan Pedro Muñoz-Gea, and Josemaria Malgosa-Sanahuja. "An MPTCP-compatible load balancing solution for pools of servers in OpenFlow SDN networks." 2019 Sixth International Conference on Software Defined Systems (SDS). IEEE, 2019.

[15] Liu, Yanbing, et al. "Improve MPTCP with SDN: From the perspective of resource pooling." Journal of Network and Computer Applications 141 (2019): 73-85.

[16] Chen, Zhongping. "Load Balancing Dynamic Source Routing Protocol Based on Multi-Path Routing." Journal of computing and information technology 27.2 (2019): 17-27.

[17] Zaki, Fatimah Audah Md, and Nurul Fariza Zulkurnain. "Frequent Itemset Mining in High Dimensional Data: A Review." Computational Science and Technology. Springer, Singapore, 2019. 325-334.

[18] Balmakhtar, Marouane, Arun Rajagopal, and Carl Joseph Persson. "Software defined network (SDN) information distribution across an SDN data-plane." U.S. Patent No. 10,623,260. 14 Apr. 2020.

[19] Alghamdi, Khaled, and Robin Braun. "Software defined network (SDN) and OpenFlow protocol in 5G network." Communications and Network 12.01 (2020): 28.

[20] Wang, Xiong, et al. "Efficient measurement of round-trip link delays in software-defined networks." Journal of Network and Computer Applications 150 (2020): 102468.

[21] Akin, Erdal, and Turgay Korkmaz. "Rate-Based Dynamic Shortest Path Algorithm for Efficiently Routing Multiple Flows in SDN." ICC 2019-2019 IEEE International Conference on Communications (ICC). IEEE, 2019.

[22] Al-Saadi, Muna, et al. "A novel approach for performance-based clustering and anagement of network traffic flows." 2019 15th International Wireless Communications & Mobile Computing Conference (IWCMC). IEEE, 2019.