A Dynamic Method of Constructing MST Based on OpenFlow Protocol

Haike Liu*, Jilin Li, Kai Yang
Beijing Institute of Satellite Information Engineering, No. 82, Zhichun Road, Haidian District, Beijing, China.

* Corresponding author. Tel.: +8613426236020; email: ko5330@163.com
Manuscript submitted August 28, 2015; accepted April 25, 2016.
doi: 10.17706/ijcce.2016.5.6.398-408

Abstract: The MST algorithm has the disadvantage of slow convergence and low resource utilization in the traditional two layer network. The forwarding topology cannot be adjusted in real time to realize the load balance of the whole network. In the SDN architecture, a dynamic construction method for MST based on OpenFlow is proposed in this paper. The controller can adjust no-loop forwarding topology of the underlying network dynamicly based on the current network traffic distribution to achieve the whole network load balancing. Through simulation, the whole network traffic distribution is more balanced after the topology adjustment, and the index of delay jitter and packet loss ratio has greatly improved.

Key words: OpenFlow protocol, load balancing, dynamic topology, STP.

1. Introduction

In the traditional two layer network, the spanning tree protocol is used to generate a no-loop topology of network. In a large-scale network, the STP algorithm generally has a long convergence time. When a switch is added in the network, a new minimum spanning tree structure should be constructed through recalculation of the network. And the traffic transmitted in the network should be blocked. Therefore, in order to avoid long convergence time as well as traffic interruption, the network generally does not change the STP structure which is calculated in the network initial stage [1], [2].

Under the SDN network architecture, centralized management and control system can achieve flexible network load balancing function [3]. In order to realize the target of network traffic load balancing and shortest path forwarding, the different minimum spanning tree structures are constructed for underlying network according to the distribution of the current network traffic load.

In the spanning tree structure, the traffic can only be forwarded along the branch of the tree. So it may lead the root node of the tree and the middle nodes which near the root node to carry larger load than the other nodes, as shown in Fig. 1.

From the picture above, for the same traffic flow, the network is more balanced in the STP topology 2. In the topology 1, it may cause too much link load between nodes 2 and node 1, which affects the user experience. The different topologies which even have the same traffic distribution can lead to different effects on load balance.

Therefore, when the network is load imbalance, a certain mechanism can be used to change the topology of the network in order to use some of the backup link caused by STP, so as to achieve the goal of improving the utilization of network resources [4].

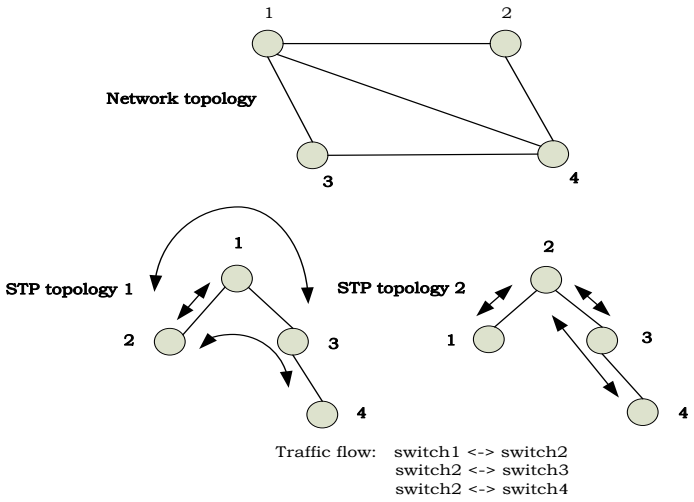


Fig. 1. Effect of different STP topologies on network load.

2. Dynamic Construction Method of Balanced MST Based on OpenFlow

2.1. Function Description

Minimum spanning tree function module with characteristic of dynamic topology based on OpenFlow is shown in Fig. 2 below. In order to dynamically adjust the network's minimum spanning tree structure based on the network load distribution, the OpenFlow controller is required to collect the whole network traffic statistical information periodically [5], [6]. The collected statistical information is mapped into the weight parameters of the network, and a load balanced forwarding topology without loop can be constructed by minimum spanning tree algorithm.

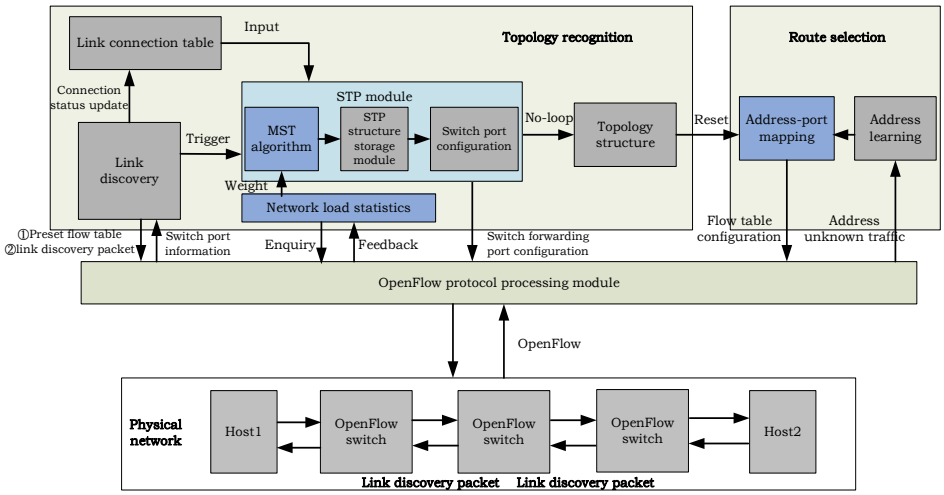


Fig. 2. Link discovery and dynamic topology computing module based on OpenFlow.

In Fig. 2, the OpenFlow controller mainly completes the function of link discovery, no-loop topology construction and route forwarding. The connection of the underlying network is found by LLDP protocol, and a no-loop forwarding topology is constructed by minimum spanning tree module based on the result of link discovery. Business forwarding is achieved mainly through self-learning method [7], [8].

The traffic load information of OF switches in the underlying network is collected periodically by the controller [9]. The minimum spanning tree algorithm considers the link weight and switch node weight to construct the MST structure, and then the configuration of switch ports is deployed by the controller. The

minimum spanning tree algorithm mainly based on link weight and node weight. The link weight represents the initial state of the network link, which is calculated by considering the bandwidth and delay of the link [10]. The node weight mainly represents the network traffic distribution, which is calculated through the changing network load.

2.2. Theory Model

2.2.1. The definition

The MST structure of the underlying network is constructed by considering the link weights and node weights value. So the problem can be summarized as the minimum spanning tree with weighted nodes.

Definition 2.1: Given undirected connected graph $G(V,E)$, the node weight function is $f:V\to R+$, the edge weight (link weight) function is $g:E\to R+$, suppose T is the minimum spanning tree of the graph G, define the degree of node v in G is $d(v|G)$, Then the total weight function of the spanning tree T is defined as follows:

$$W(T)=\sum_{e\in E(T)} w(e)+\sum_{v_k\in V} d(v_k|G)f(v_k) \tag{1}$$

The degree of node v in G is the number of adjacent nodes of v in the graph G.

The spanning tree with the minimum weight $W(T)$ in all of the spanning trees of graph G is called the minimum spanning tree with weighted nodes. The definition of the function $f(v)$ represents the traffic load forwarded by node v. The degree $d(v|G)$ of the leaf nodes is 1. It means that the degree of the leaf node is the smallest.

The problem of the minimum spanning tree with weighted nodes can be attributed to the following question:

Question 2.1: Given a graph $G(V,E)$ and positive integer K, is there a spanning tree with weighted nodes of which the total weight $W(T)$ is less than or equal to K in graph $G(V,E)$?

2.2.2. Complexity analysis

In order to solve the question 2.1, first to prove the question is NP-hard problem, the proof is described below.

Firstly, the nodes in graph G should be classified, we suppose the set of leaf nodes is V_1, and the set of non leaf nodes is V_2. Suppose the degree of the node $\forall v\in V_1$ is 1, the degree of the node $\forall v\in V_2$ is $d(v_2|G)$.

1) Without considering the weight of the leaf nodes

The total weight function is:

$$W_2(T)=\sum_{e\in E(T)} w(e)+\sum_{v_k\in V_2} d(v_k|G)f(v_k) \tag{2}$$

In this case, question 2.1 can be converted to question 2.2.

Question 2.2: Given a graph $G(V,E)$ and positive integer K, is there a spanning tree with weighted non-leaf nodes of which the total weight $W_2(T)$ is less than or equal to K in G?

Firstly CDS (Connected Dominating Set) issue [11] is: Given a graph $G(V,E)$ and positive integer K, is there a CDS of which the number of nodes is less than or equal to K in G?

The method of judging the complexity of question 2.2 is firstly proving that the CDS problem can be reduced to the problem 2.2. Because of the CDS issue is NPC problem, then the question 2.2 is NP hard.

Theorem 2.1: question 2.2 is NP hard.

Proof: The spanning tree T, all the solutions to question 2.2, can be verified whether it is the spanning tree of the original graph in polynomial time. Calculate whether $W_2(T)$ is less than or equal to constant K or not. So the question 2.2 is NP hard.

Suppose a CDS example is: $G(V,E)$ and constant K. Given an instance of the question 2.2: $G'(V',E')$ and constant K'. Suppose $V'=V$, $E'=E$, $K'=K$, and $\forall v \in V'$, $f(v_k)=1/d(v_k \mid G)$, $\forall e \in E'$, $w(e)=0$. By transforming, if let the CDS in original G correspond to the inner nodes set of the spanning tree in graph G', then the leaf nodes of spanning tree in G' correspond to the other nodes dominated by the dominating set. So, there is a CDS of which the number of nodes is less than or equal to K in original graph G when and only when there is a spanning tree with weighted nodes but without leaf nodes of which the total weight is less than or equal to K' in graph G'.

At the same time, the transformation process is in solvable polynomial time, so the question 2.2 is NP-hard.

2) Without considering the weight of the non-leaf nodes

In this case, the total weight function is:

$$W_1(T)=\sum_{e\in E(T)} w(e)+\sum_{v_k\in V_1} f(v_k) \tag{3}$$

The question 2.1 can be converted into question 2.3:

Question 2.3: Given a graph $G(V,E)$ and positive integer K, is there a spanning tree with weighted leaf nodes of which the total weight $W_1(T)$ is less than or equal to K in graph G?

Theorem 2.2: question 2.3 is NP-hard.

Proof: Because of the most leaf spanning tree problem is NP-hard in undirected connected graph, theorem 2.2 can be proved if it can be reduced to the question 2.3.

Given an instance of the most leaf spanning tree: $G=(V,E)$ and constant K. That is, the number of leaf nodes in this spanning tree is less than or equal to K.

Given an undirected connected graph $G'=(V',E')$ and K'. Suppose $|V'|=n$, $V'=V$, $E'=E$, $K'=K$. And $\forall e \in E'$, suppose $w(e)=0$, $\forall v \in V'$, $f(v)=1$.

Suppose T' is a spanning tree with m leaf nodes in G', and $|V_1|=m$, then the equation (3) can be transformed into:

$$W_1(T)=\sum_{(v_i,v_j)\in T} w(v_i,v_j)+\sum_{v_k\in V_1} f(v_k)=m \tag{4}$$

Through the transformation, the number of leaf nodes in the spanning tree of G corresponds to the number of leaf nodes in the spanning tree of G'. So, there is a spanning tree of which the number of leaf nodes is less than or equal to K in G when and only when there is a spanning tree of which the total weight is less than or equal to K'. Because of the most leaf spanning tree problem is NP-hard, the question 2.3 is NP-hard.

3) Consider all the weights of the nodes

For considering all the weights of the nodes, the equation (1) can be transformed into:

$$W(T) = \sum_{(v_i, v_j) \in T} w(v_i, v_j) + \sum_{v_k \in V_1} f(v_k) + \sum_{v_l \in V_2} d(v_l \mid G) f(v_l) \tag{5}$$

The function $f(v_i)\ (1 \le i \le n)$ is a variable changed with time and statistical information, meanwhile $d(v \mid G)$ represents the degree of the node which is corresponding to the number of edges. So the equation (5) can be transformed into:

$$W(T) = \sum_{(v_i, v_j) \in T} [w(v_i, v_j) + f(v_i) + f(v_j)] \tag{6}$$

2.3. Dynamic Construction Algorithm of Balanced MST Topology

From the complexity analysis, the problem of the minimum spanning tree with weighted nodes is NP-hard. The solution of this kind of problem is the optimal search algorithm. Prim is an algorithm in graph theory by which the minimum spanning tree can be searched in the weighted connected graph. The main idea is to select the nearest vertex v_1 from any vertex v_0 to construct tree T_1, then connect to the nearest vertex v_2 from T_1 to construct tree T_2. Repeat this process until all the vertices are in the tree.

An improved prim algorithm is proposed to construct the dynamic balanced MST topology.

From the (6):

$$w'(v_i, v_j) = w(v_i, v_j) + f(v_i) + f(v_j) \tag{7}$$

Then:

$$W(T) = \sum_{(v_i, v_j) \in T} w'(v_i, v_j) \tag{8}$$

$w'(v_i, v_j)$ can be regarded as the weight of equivalent edge in $G(V, E)$. Because of $e = (v_i, v_j) \in E$, Prim algorithm can be used to solve the problem.

In order to realize the construction of the dynamic MST topology which can guarantee the load balancing of the whole network, OpenFlow controller is required to periodically obtain the statistical information of each switch. First the definition of transit traffic is given, which represents all of the traffic flew into the switch minus the traffic of which the source address or destination address is in the same LAN with this switch. So the transit traffic is the sum of the traffic which needs to be forwarded to the next switch.

Suppose the transit traffic of the switch is B, and the traffic from all of the ports except the local LAN port is b_{in}, the traffic of which the destination address is the host in this local LAN is b_0, so:

$$B = b_{in} - b_0 \tag{9}$$

According to the traffic statistics of each switch, the controller calculates the transit traffic of all the non-leaf nodes in the MST topology of the current network as the basis of generating the node weight.

Controller calculates the transit traffic of each node, and mapping it proportional into the value of which the range is specified by the weight regulation, and the value is used as the weight of this non-leaf node. Therefore, the weight value can reflect the throughput of the transit traffic of this node. Assuming the mapping function is h, the weight of this non-leaf node is P_{middle}, then:

$$P_{middle} = h(B) \tag{10}$$

At the network initialization stage, set an initial weight value for each switch, this value is mainly used as the node weights of the initial network and calculating the MST topology with the edge weight. The method of setting initial weight for switch node is similar to the link weight. It depends on the performance and the importance of the switch.

When the no-loop forwarding topology needs to be recalculated, the weight of the leaf nodes in the current forwarding topology is set to the initial weight of the switch. Or according to the situation, the weight of leaf nodes can be set to zero before the recalculation of the no-loop forwarding topology. Set the weight of the current leaf node to P_{leaf}.

So, the edge weight $w'(v_i, v_j)$ is:

$$w'(v_i, v_j) = w(v_i, v_j) + P(v_i) + P(v_j) \tag{11}$$

If v is a leaf node, then $P(v) = P_{leaf}$. If v is a non-leaf node, then $P(v) = P_{middle}$.

Then, a new no-loop forwarding topology can be calculated by Prim algorithm.

3. Simulation Verification

We build simulation environment in the mininet simulation software, and choose OVS switch type as the switch nodes. The POX is used as the OpenFlow controller. The dynamic construction algorithm of balanced MST topology proposed in this paper is implemented in the POX controller. In order to test the performance of the algorithm, we deploy multiple traffic flow in the network [12]. In the simulation, the sFlow-rt is used to collect the traffic statistics of the OF switches, which can be used as the evaluation criteria of the simulation results.

3.1. Simulation Topology

Firstly, the topology of the OF network is constructed in the mininet. And set the initial edge weight to this topology. The topology is composed of seven OF switch nodes. The bandwidth of each link is 5Mbps, the link delay is 5ms. The edge weight is shown in Fig. 3 below.

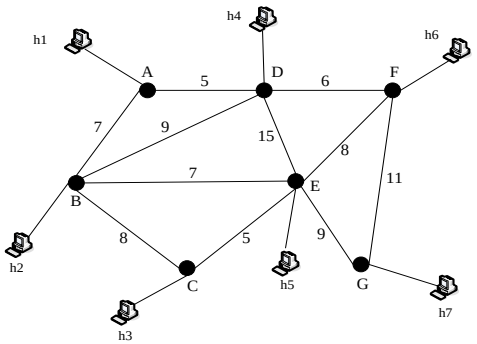


Fig. 3. Simulation topology.

3.2. Simulation Result

After the network initialization, Three traffic flows are introduced in the initial minimum spanning tree topology: h1<->h2, h3<->h4, h4<->h5. The bandwidth of h1<->h2 is 1Mbit/s, 2Mbit/s for h3<->h4, and 2Mbit/s for h4<->h5. The traffic distribution is shown as Fig. 4.

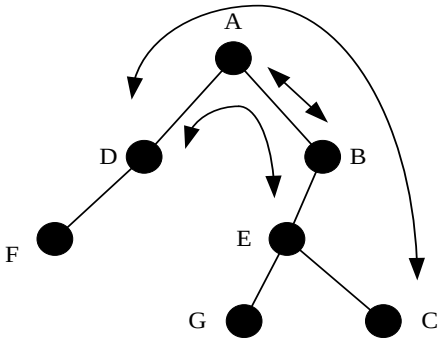


Fig. 4. The traffic distribution of initial MST topological.

In the picture above, the link load between A and B is too heavy in the current forwarding topology. Because the bandwidth between A and B is 5Mbit/s, the topology optimization algorithm is triggered. And the new MST topology is shown as Fig. 5.

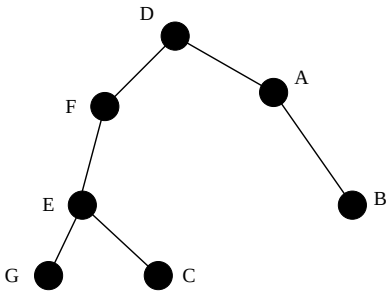


Fig. 5. The new MST topology.

The sflow-rt software can monitor the traffic flow. Three monitoring points are deployed in the network as Fig. 6 which are used to monitoring the output traffic from B to A, the output traffic from A to D, and the output traffic from F to D.

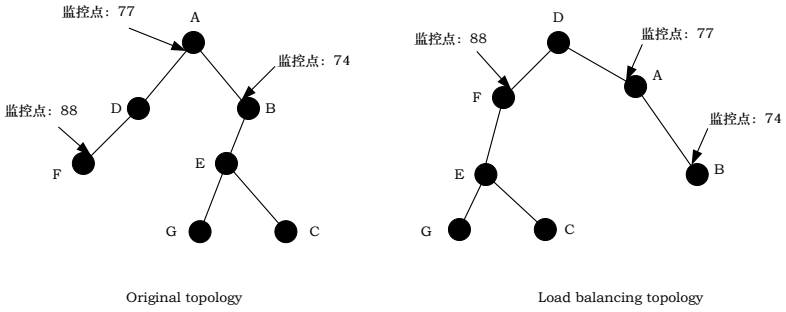


Fig. 6. Sflow-rt monitoring point deployment diagram.

In the network, the traffic of monitoring points is shown in Fig. 7, Fig. 8 and Fig. 9 below.

In Fig. 7, vertical coordinate is output traffic from the monitoring point for real-time monitoring. The unit is byte/s, and each cell is 4Mbit/s. Cross coordinates is time, and each cell is 5s. The topology has changed after the time of 07:04:37. Before the change of the topology, the average output throughput is 5Mbit/s from nod B to node A, and the jitter of the traffic is big. It shows that the link load of A<->B is too much. After the change of the topology, the output traffic from node A to node B is about 1Mbit/s, the throughput of the flow is relatively stable. It shows that some of the traffic is directed to other links.

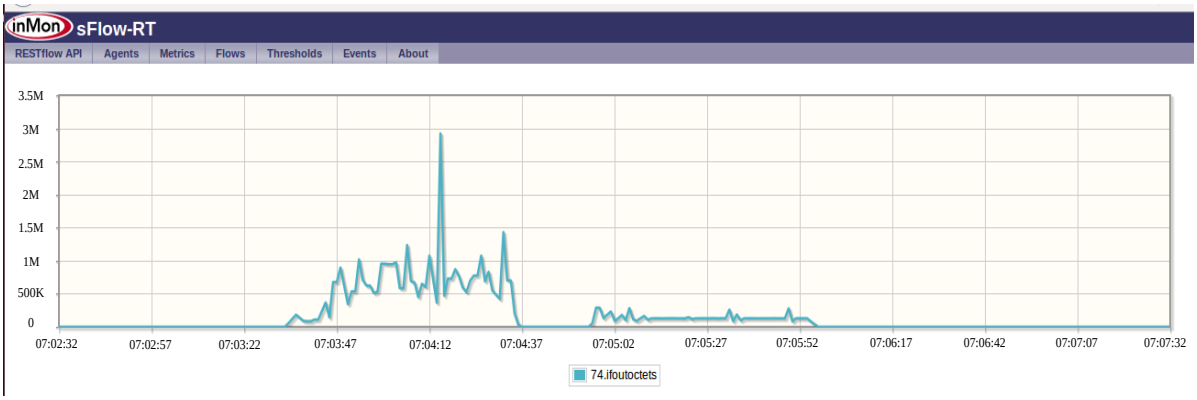


Fig. 7. The traffic of monitoring point 74.

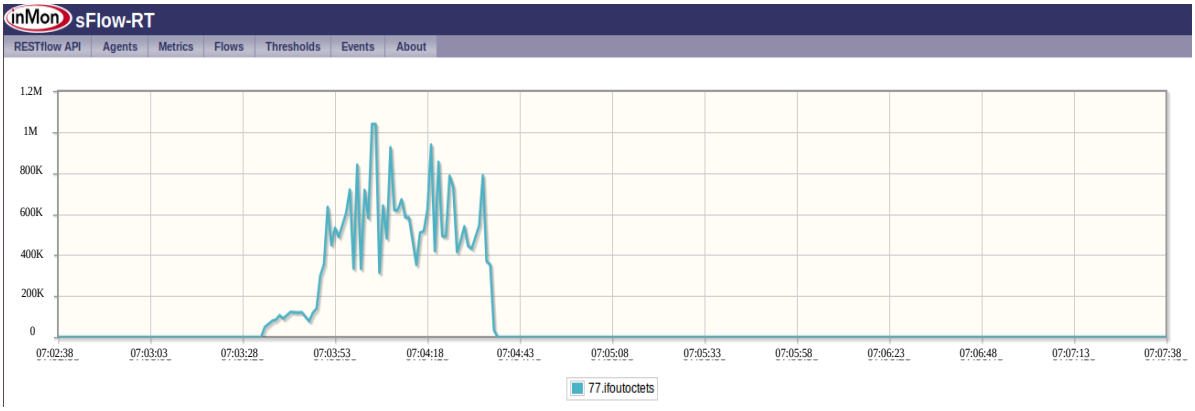


Fig. 8. The traffic of monitoring point 77.

In Fig. 8, the monitoring point 77 is mainly used to monitoring the output traffic from node A to node D, the average throughput is 4Mbit/s. But after the time of 07:04:37, the throughput of output traffic is 0. So it shows that A node has no longer forwarding the traffic of h3<->h4, h4<->h5.

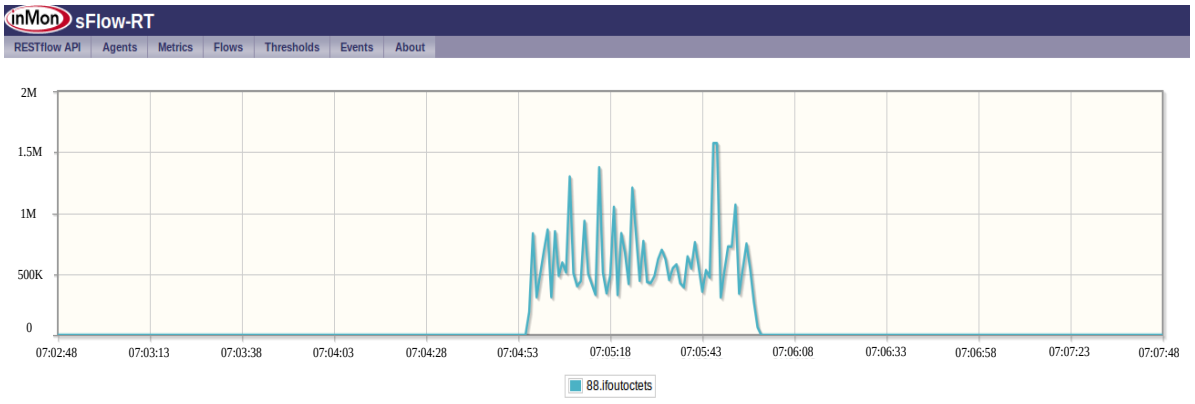


Fig. 9. The traffic of monitoring point 88.

In Fig. 9, the monitoring point 88 is mainly used to monitoring the output traffic from node F to node D, the average throughput is 4Mbit/s. Before the time of 07:04:37, there is no traffic from this monitoring point. But after the time of 07:04:37, the traffic of h3<->h4, h4<->h5 is forwarded by node F.

The traffic distribution with the same flows in the network before and after the changing of MST topology is shown in Fig. 10 below.

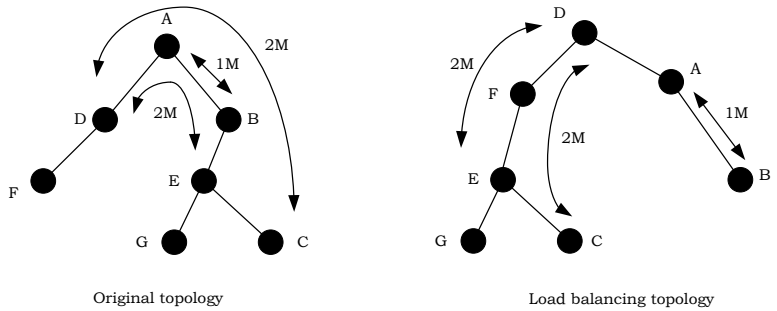


Fig. 10. The load distribution before and after the changing of MST topology.

The Fig. 10 shows that the distribution of network traffic in the second topology is more balanced. There is too much load in the link between node A and node B in the original topology, which reaches the bandwidth limit of 5Mbit/s. And in the adjusted topology, each link is more balanced and reasonable.

The Fig. 11 shows the comparison of packet loss rate and delay jitter of UDP traffic received by h1 before and after the changing of MST topology. The traffic is from h2 to h1. Before changing, the delay jitter is 3.677ms, the packet loss rate is 6.7%. After changing, the delay jitter is 1.075ms, the packet loss rate is 0.0039%.

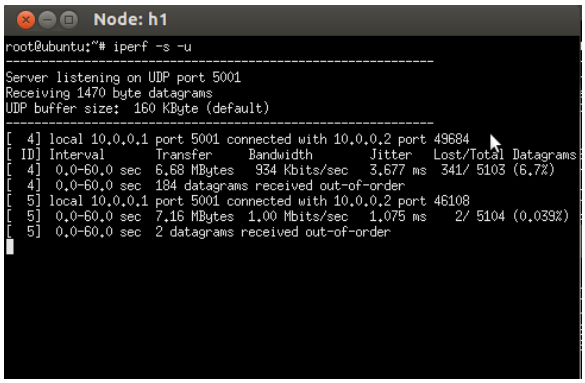


Fig. 11. The comparison of packet loss rate and delay jitter in h1.

The Fig. 12 shows the comparison of packet loss rate and delay jitter of UDP traffic received by h4 before and after the changing of MST topology. One flow is from h3 to h4, and the other is from h5 to h4. It shows that the change of delay jitter is not big. The packet loss rate is 24% and 20% before changing, 0.23% and 0.11% after changing.

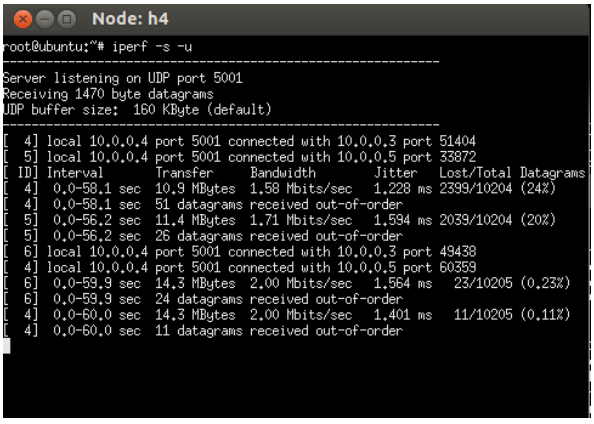


Fig. 12. The comparison of packet loss rate and delay jitter in h4.

4. Conclusion

In this paper, a dynamic construction method of MST which can realize the network load balancing based on the OpenFlow protocol is proposed. The function module and theoretical analysis is described in detail. From the results of simulation, it shows that the proposed method can dynamically optimize the underlying network forwarding topology in real-time according to the traffic distribution in the network. As the result, the users can have a better experience. According to the different traffic distribution, some of the traffic can be directed to the link with less load by changing the no-loop forwarding topology.

Acknowledgment

First and foremost, I would like to show my deepest gratitude to my supervisor, Dr. Li Jilin, a respectable, responsible and resourceful scholar, who has provided me with valuable guidance in every stage of the writing of this thesis. Without his enlightening instruction, impressive kindness and patience, I could not have completed my thesis. His keen and vigorous academic observation enlightens me not only in this thesis but also in my future study. I shall extend my thanks to Mr. Yang for all his kindness and help.

References

[1] Huang, W., Chen, Y., & Hee, J. (2006). STP technology: An overview and a conceptual framework. *Information & Management*, *43(3)*, 263-270.

[2] Prete, L., Farina, F., Campanella, M., *et al.* (2012). Energy efficient minimum spanning tree in OpenFlow networks. *Proceedings of European Workshop on Software Defined Networking* (pp. 36-41).

[3] OpenFlow switch specification (version 1.3.2). (2013). From https://www.opennetworking.org/images/stories/downloads/sdn-resources/onfspecifications/OpenFlow/OpenFlow-spec-v1.3.2.pdf

[4] Open networking foundation. (2013). From http://www.opennetworking.org

[5] Wang, Q., Zhao, H., & Xie, Y. (2013). Standardization and deployment of SDN. *Zte Technology Journal*.

[6] Long, H., Shen, Y., Guo, M., & Tang, F. (2013). LABERIO: Dynamic load-balanced routing in OpenFlow-enabled networks. *Proceedings of IEEE 27th International Conference Advanced Information Networking and Application*.

[7] Tewarie, P., Van, D. E., Hillebrand, A., *et al.* (2015). The minimum spanning tree: An unbiased method for brain network analysis. *Neuroimage, 104,* 177-188.

[8] Jian, Z., Lu, C., & Ke, W. (2015). Path optimality conditions for minimum spanning tree problem with uncertain edge weights. *Int. J. of Uncertainty, Fuzziness and Knowledge-Based Systems*, *23(1),* 49-71.

[9] Androulidakis, G., Papagianni, C., & Papavassiliou, S. (2014). Virtual topology mapping in SDN-enabled clouds. *Proceedings of IEEE 3rd Symposium on Network Cloud Computing and Applications* (pp. 62-67).

[10] Wang, Z., Wu, J. X., Wang, Y., *et al.* (2014). Survivable virtual network mapping using optimal backup topology in virtualized SDN. *Communications China, 11(2)*, 26-37.

[11] Thai, M. T., Wang, F., Liu, D., *et al*. (2007). Connected dominating sets in wireless networks with different transmission ranges. *IEEE Transactions on Mobile Computing*, 721-730.

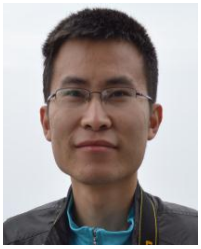
[12] Pox homepage. (2013). From http://www.noxrepo.org/pox

Haike Liu was born in China. He received the B.S. degree in electronic and information engineering from Chongqing University of Posts and Telecommunications, Chongqing, China, in 2009. He received the M.S. degree in electronic and information engineering from Beijing Institute Of Satellite Information Engineering, China, in 2012. He is currently pursuing his Ph.D. degree in computer network technology at Beijing Institute of Satellite Information Engineering, China.

His research interests include software defined network technology and satellite communication network.

Jilin Li received his M.S. and B.S. degree in communication engineering from Harbin Institute of Technology, Harbin, China in 1985 and 1982. He is currently working as a professor in Beijing Institute of Satellite Information Engineering.

His research interests include satellite communication technology, computer network and wireless communication technology.



Kai Yang earned a B.S. degree in 2007, and an M.S. degree in 2009, all in computer science from University of Electronic Science and Technology of China.

He was a member of the technical staff at Microsoft in 2010. Since 2011, he has been with the Department of Satellite Communication at Beijing Institute of Satellite Information Engineering, where he is currently an associate professor.