Acta Technica *62* (2017), No. 5B, 1049–1056

An improved algorithm for link price calculating based on information entropy and topsis method

SHAN CHUN[2]

Abstract. The current calculating method of the link congestion price is unable to meet the dynamically figure of data center network. Therefore, we will make full use of the character that the global resource information can be grasped by the Opendaylight controller of SDN architecture and put forward a kind of two layer ELPC Entropy Link price calculating algorithm based on the entropy and TOPSIS(Technique for Order Preference by Similarity to an Ideal Solution) method. The upper algorithm is deployed in the central controller such as Opendaylight and the proportion parameter vector B is given by the upper algorithm according to the information entropy of the global network link load with the TOPSIS method.The lower algorithm is deployed in Openflow switch and the link congestion price is calculated according proportion parameter vector B given by the upper algorithm. We evaluate ELPC in Mininet emulator and a testbed, and the experimental results show that information entropy can effectively increases the throughput of bisection bandwidth.

Key words. data center, congestion price, information entropy, TOPSIS, penflow.

1. Introduction

As the information infrastructure, the datacenter network has been widely deployed and implemented to provide computing, storage and interactive services for users and enterprises at present. Therefore the data center network is the core components of data center, not only provide interconnection service for tens of hundreds of thousands of host server,and provide efficiently network communication and data transmission for upper computing service[1,2].

There has been a lot of recent work on transport layer designs[3], especially for datacenters, that target different bandwidth allocation objectives. Many aim to

[1]Acknowledgment - The authors thank all the reviewers and editors for their valuable comments and works. This paper This work is supported by the Public welfare research project of Guangdong Province under Grant No. 2016KZ010102 and the Zhejiang Natural Science Foundation of China (Grant: LY18F010018)

[2]Workshop 1 - School of Electronic and Information, Guang Dong Polytechnic Normal University, Guangzhou 510665, China;email:530664230@qq.com

minimize rate of deadline and latency or minimize average flow completion time, while others target multi-tenant bandwidth allocation,while still others focus on sophisticated objectives like resource pooling,policy-based bandwidth allocation, or coflow scheduling . In effect, each design supports one point in the bandwidth allocation policy design space, but operators ideally want a transport that can be tuned for different points in the design space depending on workload requirements.

but these improved TCP is still calculating the link congestion price according to the proportion of the congestion link state such as queue length??queuing delay etc. and the proportion is invariable.And calculating the congestion pricing algorithm can be summarized as the following formula:

$$p = f(B, Q) \tag{1}$$

where B is proportion parameter vector Q is queuing length??queuing delay status vector etc.

In this paper, we will make full use of the character that the controller of SDN architecture can grasp the global resource informatio, put forward a kind of two layer network global congestion price calculating(ELPC) algorithm based on optimization theory framework of distributed network. The upper algorithm deploy in the central controller and the controller give proportion parameter vector B according to the global resource information(including network topology link state??queue length).The lower algorithm deployment in Openflow switch, compute the link congestion price according proportion parameter vector B given by the upper algorithm.

2. Congestion price algorithm based on information entropy

2.1. Upper layer algorithm in central controller

1)Global link congestion state analysis

The upper layer algorithm in central controller send regularly the packet of link layer discovery protocol(LLDP) to all switch equipment and collect the connection information of all switch equipment. Then opendaylight in the central controller can build the global network topology. By through Openflow channels between the central controller and switch, we modify exchange of three kinds of exchange packet include controller-to-switch,asynchronous and symmetric packets and add new exchange information to the data structure of these three kinds of exchange packets. The above modified information of data structure include the link load,proportion parameter vector B for link congestion price calculating algorithm and practical running effect of switch in data center. The key information in the modified data structure is the mapping relationship between global network congestion state and the proportion parameter vector B of calculated congestion price parameter vector.

2)Algorithm initialization

At the beginning of the algorithm running, according to the openness and programming of the SDN framework,the mapping relationship between the proportion parameter vector B of calculated congestion price parameter vector and global net-

work congestion state are set by the network administrator according to experience. When part link state are not heavy ,the upper layer algorithm in central controller give less conservation proportion parameter vector B .However when all link state are heavy load, ,the central control will give the no conservation proportion parameter vector B and raise the ling congestion price to supress the send rate of the TCP source and to avoid network congestion.

3) Calculating the information entropy of link load in datacenter network

If the information entropy of link load is bigger, indicating that each link load of the data center network are basically in the same status the global network link load status can be judged according to any one of link load case.If the information entropy of link load is smaller, indicating that each of the link load is different fully, so it is necessary to adopt different calculating strategy of link price for different link load status. If the link load is heavier , the link price will be increased greatly. If the link load is lower,the link price must be brought down.

Specific algorithm is described as follows:

Step 1:For each link load, normalizing the link load According to the formula 2:

$$s_{i,t}=\frac{s_{i.t}}{\sum_{i=1}^{n} s_{i.t}}, t = 1, ..., t_h \tag{2}$$

Step 2 calculating the information entropy of the link load in the datacenter network

$$E_t = -\sum_{i=1}^{n}(s_{i,t} \ln s_{i,t}), t = 1, 2, ...t_h \tag{3}$$

the datacenter network. And the biggest value is $E_{\max}=\ln n$.

Step 3 : If the entropy value greater then $0.8 \bullet E_{\max}$ then indicating that each link load of the data center network are basically in the same status the global network link load status can be judged according to any one of link load case .And set the vector B of the formula(1) with $p_{i,\max} = B_{\max} \cdot (1/(1 + \exp(\frac{1}{t_h} \sum_{t=1}^{t_h} s_i)))$

Technique for Order Preference by Similarity to an Ideal Solution

Where $B_{\max}$ is the maximum link price

And the program is end.

Else execute the step 4-7

Step 4 Set the positive and negative Ideal Solution of each link load.

$$s_t^+ = \max_{i=1,n} s_{i,t},\ s_t^- = \min_{i=1,n} s_{i,t}\ t = 1, 2, ..., t_h$$

Step 5 Calculating the euclidean metric between link load and the positive and negative Ideal Solution.

$$d_i^+ = \sqrt{\sum_{t=1}^{t_h}(s_{i,t} - s_t^+)^2} \tag{4}$$

$$d_i^- = \sqrt{\sum_{t=1}^{t_h}(s_{i,t} - s_t^-)^2} \tag{5}$$

Step 6 Calculating the relative closeness coefficient of each link load:

$$G_i = \frac{d_i^-}{d_i^- + d_i^+} \tag{6}$$

Step 7 Set the vector B in the formula(1)

$$p_{i,\max} = B_{\max} \cdot (1/(1 + \exp(G_i))) \tag{7}$$

2.2. Lower layer algorithm in Openflow switch

1)At regular intervals, the map relation between the current link load and the vector B will be feedback to the central controller and collected as the samples training data for BP neural network.

2)Based on the vector B given by the upper layer algorithm, according to the $p = f(B, Q)$,calculate the congestion price of the each link, where Q is the current link congestion status.

3)Standard TCP NEW Reno is used as the baseline of our evaluation.The initial window is 12KB,and Openflow switches use RED queues and FCFS scheduling. The RED congestion price is calculated as follows:

$$p_i = p_{\max}(\frac{s_i - b_{\min}}{b_{\max} - b_{\min}}) \tag{8}$$

the calculation result is shown in Table 1.

Table 1. the map relation between the load and the parameter

Num.	s1	s2	s3	s4	load	$p$max	$b$min	$b$max
1	0.2	0.3	0.2	0.3	lighter	0.02	0.06	0.6
2	0.4	0.5	0.7	0.6	light	0.08	0.08	0.6
3	0.7	0.6	0.8	0.5	low heavy	0.12	0.1	0.7
4	0.8	0.9	0.7	0.6	heavy	0.1	0.1	0.7

3. Schemes compared

1)TCP :standard TCP NEW Reno is used as the baseline of our evaluation. The initial window is 12KB,and switches use Drop tail queues and FCFS scheduling. These are standard settings used in many studies.

2)ECMP:we use standard ECMP algorithm,in which controller choose the next

hop for each packet by hashing the five-tuple of source/destination IP address. source/ destination port number and the protocol of transport layer.TCP New Reno is employed in servers with the same parameters as TCP.

3)ELPC: Our design as described in section 2 ,we still utilize TCP New Reno in server as above mentioned. We still utilize TCP New Reno in servers as above mentioned. Besides.

4. Experimental on mininet

In our experiment,we select one-to-one communication, where our server randomly communicates with another server at one time. The experiment results are demonstrated in fig.1 and fig.2.

Throughput: fig.1 displays the throughput of TCP,ECMP and ELPC. Because only one path is available, the throughput of tcp is rather poor, and just make use of 51.2% of the bisection bandwidth in the web search workload. ECMP's random hashing results in many collisions, thus the throughput is less than 69.8%.In contrast, ELPC achieves 89%,which significantly increases the throughput 73.8%,27% comparing to TCP,ECMP respectively. In summary, ELPC efficiently improves the throughput.

Fig. 1. Throughput of per-flow in vl2 under the web search workloads

Efficiency:fig.6 shows the per-flow bandwidth utilization in TCP,ECMP and ELPC.We can learn that TCP,ECMP have poor performance in improving network efficiency. Specifically, in TCP,more than 50% of flow's.

5. Experimental on testbed

In this section, we deploy ELPC in real network testbed and test its performance. Because the ELPC does not modify the server operating system .The ELPC program is deployed in Opendaylight and can easy be run without any modify. We fully

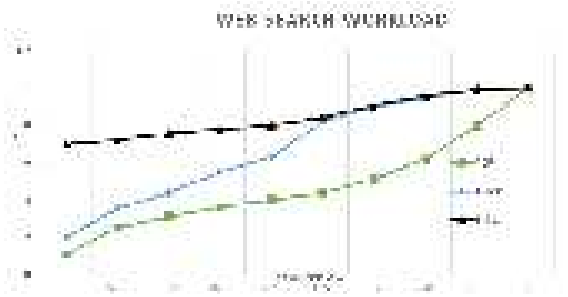

Fig. 2. CDF(cumulative distribution function) of per-flow

utilized the web search workload as our test benchmark and adopt the all-to-all communication model. The experiment results are shown in Fig.3 and Fig.4.

Throughput: fig.3 illustrates the throughput of TCP,ECMP and ELPC under the web search workload. It is manifest the ELPC can efficiently improve the network throughput, which achieves more the 90% of bisection bandwidth and increased by 156% and 79% compared to TCP,ECMP.

Efficiency :fig.4 demonstrates the per-flow bandwidth utilization in TCP,ECMP and ELPC. ELPC achieves a better load balancing than other two mechanisms. Specially ,all of the flows occupy more than 83% of bisection bandwidth in ELPC ,while more than 66% of flows bandwidth utilization is less than 49% in TCP. For EMPC, there is still more than 35% of flows bandwidth utilization is less than 50% of bisection bandwidth. Therefore, ELPC can efficiently improve the throughput and load balancing in our testbed.

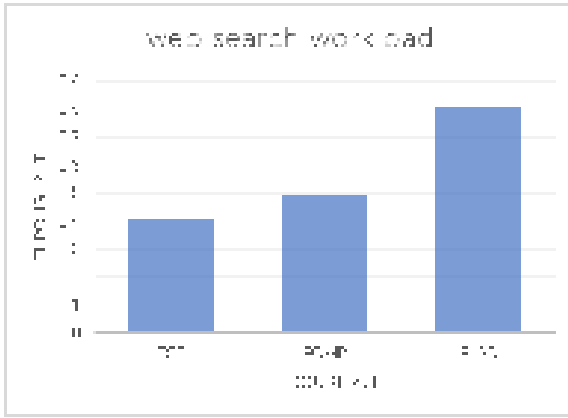

Fig. 3. Throughput of per-flow in vl2 under the web search workloads

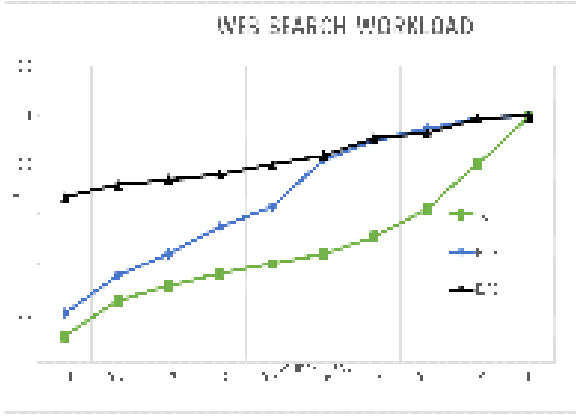

Fig. 4. CDF(cumulative distribution function) of per-flow in vl2 under the web search workloads

6. Conclusion

We present the design a kind of two layer network global congestion price calculating(ELPC) algorithm.The upper algorithm is deployed in the central controller such as Opendaylight and the proportion parameter vector B according to the global resource information(including network topology link state queuing length) is given by the upper algorithm with the Bp neural network machine learning algorithm and Fuzzy C-means Clustering Algorithm. The lower algorithm is deployed in Openflow switch and the link congestion price is calculated according proportion parameter vector B given by the upper algorithm. We evaluate ELPC in Mininet emulator and a testbed,and the experimental results show that ELPC can effectively increases the throughput of bisection bandwidth. In future work, we will implement ELPC in real datacentre networks for further research.

References

[1] H. W. Huang, S. Guo, P. Li: *Cost minimization for rule caching in software defined networking.* IEEE Transactions on parallel and distributed systems *27* (2016), No. 4, 1007–1016.

[2] W. Zhang, Y. Song, L. Ruan: *Resource management in internet-oriented data centers.* Ruanjian Xuebao/Journal of Software *23* (2012), No. 2, 179–199.

[3] Q. Y. Zuo, M. Chen: *Research on openflow-based SDN technologies.* Journal of software *24* (2013), No. 5, 1078–1082.

[4] K. Jacobsson, L. L. Andrew, A. Tang: *An improved link model for window flow control and its application to FAST TCP.* IEEE Transactions on Automatic Control *54* (2009), No. 3, 551–564.

[5] H. Kim, N. Feamster: *Improving network management with software defined networking.* IEEE Communications magazine *51* (2013), No. 2, 114–119.

[6] M. Alizadeh, A. Greenberg, D. A. Maltz: *Data center tcp (dctcp).* ACM SIGCOMM computer communication review *40* (2010) 63–74.

[7] B. Vamanan , J. Hasan, T. N. Vijaykumar: *Deadline-aware datacenter tcp.* ACM SIGCOMM computer communication review *42* (2012), No. 4, 115–126.

[8] H. Wu , Z. Feng , C. Guo: *ICTCP: Incast congestion control for TCP in data-center networks.* IEEE/ACM transactions on networking *21* (2008), No. 2, 345–358.

[9] C. Wilson, H. Ballani, T. Karagiannis: *Better never than late: Meeting deadlines in datacenter networks.* ACM SIGCOMM Computer Communication Review *41* (2011), No. 4, 50–61.

[10] C. Y. Hong, M. Caesar, P. Godfrey: *Finishing flows quickly with preemptive scheduling[C],Proceedings of the ACM SIGCOMM 2012 conference on Applications.* technologies (2012), 127–138.

Received November 16, 2017