대용량 네트워크 트래픽을 위한 SDN 플로우 테이블 캐쉬 대체 알고리즘 평가

부둑 티엡, 뉘엔트리 투안 힙, 김경백
전남대학교 전자컴퓨터공학부
e-mail : ductiep91@gmail.com, tuanhiep1232@gmail.com, kyungbaekkim@jnu.ac.kr

Evaluation of Replacement Algorithms of SDN Flow Table Cache for Heavy Network Traffics

VU DUC TIEP, NGUYEN TRI TUAN HIEP, KYUNGBAEK KIM
Department of Electronics and Computer Engineering,
Chonnam National University

요 약

Software Defined Networking is a dynamic architecture that allows fast, efficient network changes from a centralized management point. A flow table is a vital component to support the operation of SDN switches. However, it is very limited in size and easily filled up in heavy traffics, which results in significant performance degradation. The issues can be mitigated by using a flow table cache but it requires an effective flow replacement algorithm. Recently proposed replacement approaches are: Least Recently Used (LRU), packet rate based, and ranking based approaches. In heavy network traffics nowadays, the diverse packet rates might influence the performance of these algorithms significantly. In this paper, we evaluate the state-of-the-art replacement algorithms of SDN flow table cache for heavy network traffics. A SDN flow table cache module has been implemented with three different replacement algorithms such as LRU, packet rate based, and ranking based algorithms. The experiment was well-organized and carried out carefully. The results indicate that the ranking based algorithm has the best performance while LRU algorithm is the worst in heavy network traffic.

1. Introduction

Software Defined Networking (SDN) is an emerging technology that allows to directly program the network control and the underlying infrastructure [1]. A Flow table is a vital component that supports the forwarding function of SDN switches. The flow tables are build out of physical components called TCAMs [2], which are very expensive. Therefore, flow tables are limited in size and get filled up quickly in heavy network, causing significant performance degradation of SDN system.

One promising approach to resolve this issues is to use flow table cache [3][4][5], which is a component sit between the switch and controller. The flow table cache approach usually requires a smart flow replacement algorithm. In [3], a flow table cache used the Least Recently Used algorithm [6] as the flow replacement algorithm. Recently researchers have proposed a flow table cache with packet rate based [4], and ranking based [5] flow replacement algorithms. In heavy network traffics nowadays, the diverse packet rates might influence the performance of these algorithms significantly. Therefore, in this work we want to evaluate the state of the art replacement algorithms of SDN flow table cache in heavy network traffic.

The rest of the paper is organized as follows: in section 2, we present previous works, our observations and motivation for the evaluation. The evaluation scenario is described in section 3. In section 4, we present the obtained results. Finally, we conclude this paper in section 5.

2. Related Work

The flow table issues has been addressed and being resolved. One way is to reduce the flow entries in flow table by improving SDN flows and rules management through the use of wildcard and tag as proposed in [7],[8], and [9]. A more attractive approach is to use flow table cache to store flow entries when flow table is full as in [3], [4] and [5]. When a packet arrives at a switch, the switch looks up a matched flow entry in its flow table first. Then, if there is no matched flow entry, it will search in the flow table cache. A simple

flow table cache has no flow replacement strategy. Since the speed of the cache is much slower than the flow table, some flows should be stored in the flow table instead. Therefore, a smart flow replacement strategy is required to improve flow table cache performance.

In [3], when the flow table is full and SDN controller wants to add new flow entry, the flow table cache will use the Least Recently Used algorithm to choose which flow should be replaced by new entry. That is the old flow entry which has been unused for the longest time in the flow table. The performance of the algorithm is further increased by ensure the fairness between elephant flow and mice flow [3]. The elephant flows are large in size but few in numbers compared to mice flow. Therefore, they are usually evicted when flow table is full.

In [4], the authors take advantage of the packet rate of each flows into their replacement strategies. First, the flow table cache classifies flows into groups. Each group contains only flow entries that are dependent on each other. The, each flow entry is given a weight corresponding to its packet rate and a cost corresponding to the number of dependent flow entries. The flow entry which maximizes the total weight will replace the flow with lower weight in flow table.

Recently the ranking-based flow table cache was proposed in [5]. Unlike other approaches, the ranking based algorithm uses both packet rate and the size of the packet to classify network flows. Ranking calculation is simple, and fast. The influence of packet rate and packet size can be adjust through a weight. The flows that have the highest rank will replace the flow with lower rank in flow table.

Though the previous works have evaluated their algorithms extensively with many flows, they did not consider the diversity of flows carefully. For example, in a large scale network, heavy network traffics create many flows with very different packet rates and packet sizes. The influence of diverse packet rates of traffic flows might cause significant change in the performance of each flow replacement algorithms. Under the circumstance, the algorithm that considers packet rates of flows such as packet rate based and ranking based might perform better than the LRU. In order to confirm the assumption, we conducted the experiment to evaluate the flow replacement algorithms for heavy network traffics with really diverse packet rates.

3. Evaluation Scenario

3.1 Methodology

The performance of replacement algorithms is indicated through two factors: the average packet drop ratio and the average packet delay time when each algorithm is applied at a flow table cache. We will measure these two factors in heavy network traffics.

3.2 Topology Setup

The topology of experiment is depicted in figure 1. The network consists of a SDN controller, 3 switches and 40 hosts. The switches are connected together in a chain, and they are all connected to controller. Hosts are divided into two same size groups: 20 hosts in group 1 connect to switch 1 and other 20 hosts in the group 2 connect to switch 3.

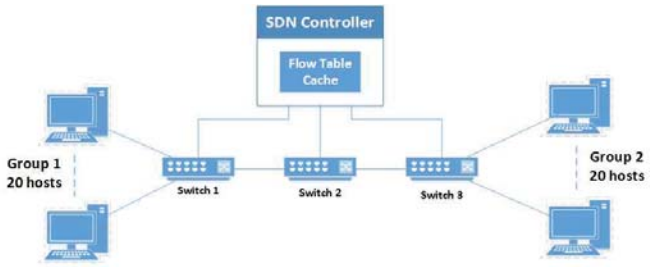

Figure 1. Topology of the experiment.

We run OpenDaylight Controller [10] as the SDN controller and mininet [11] is used to emulate switches and hosts. OpenDaylight is run on computer which has 2 3.4GHz CPUs and 8GB memory. Mininet runs on a different computer with 2 3.3GHz CPUs and 8GB memory. A SDN Flow Table Cache module is implemented with three different replacement algorithms: LRU, packet rate based and ranking based algorithms.

3.3 Experiment Scenario

First, in every switch, the size of a flow table supports only 11 traffic flows. We create 20 flows by exchanging network traffics between 20 hosts of group 1 and other 20 hosts of group 2. The packet size of every packet is 1024 bytes. The packet rate of each flow is varied in linearly increasing manner. As shown in figure 2, we have totally 5 packet rate profiles with different scale for the created flows. Each profile is applied one by one while we evaluate each algorithm.

4. Evaluation Results

The experimental average packet drop ratio and average packet delay time are shown in figure 3 and figure 4 respectively. As can be seen in figure 3, the

performance of LRU algorithm degrades dramatically at packet rate 200 packet/s with 62% packet dropped, and up to nearly 100% when the packet rate increases. From 200 to 400 packet/s, no replacement, packet rate based and ranking-based perform equivalently with almost 0% packet drop. At packet rate of 800 packet/s, the performance of no replacement and packet based algorithm degrade significantly with packet drop rate of 71% and 31% respectively. However, at that point, only 5% packet being dropped by the ranking-based flow table cache, which is 66% and 26% less than no replacement and packet rate based methods respectively. From 1200 up to 1600 packet/s, the ranking based method also decreases performance significantly but the packet drop rate is still 10% lower than no replacement and packet rate based methods.

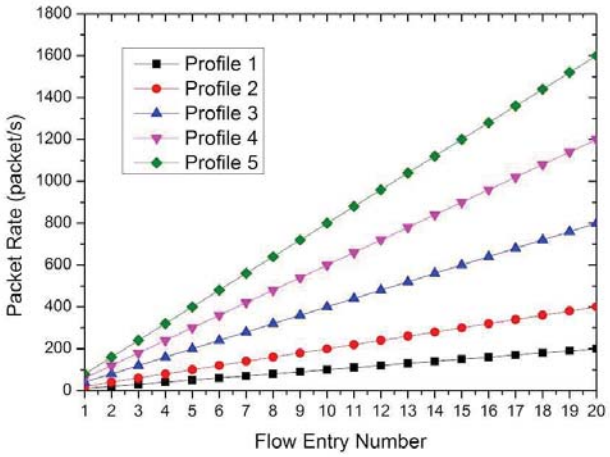

Figure 2. Flow's packet rate arrangement of traffic profiles

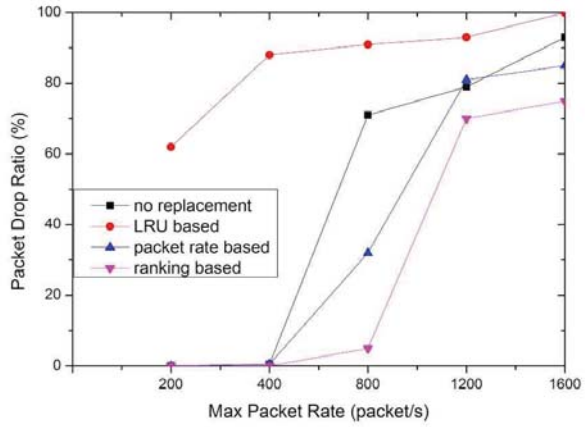

Figure 3. Average packet drop ratio

As shown in figure 4, from 200 to 1200 packet/s, no replacement, packet rate based, and ranking based have almost equivalent packet delay time. At 1600 packet/s, the packet delay goes up dramatically for all methods, but the ranking based still has packet delay time smaller than packet rate based and no replacement 1200 ms and 3600 ms respectively.

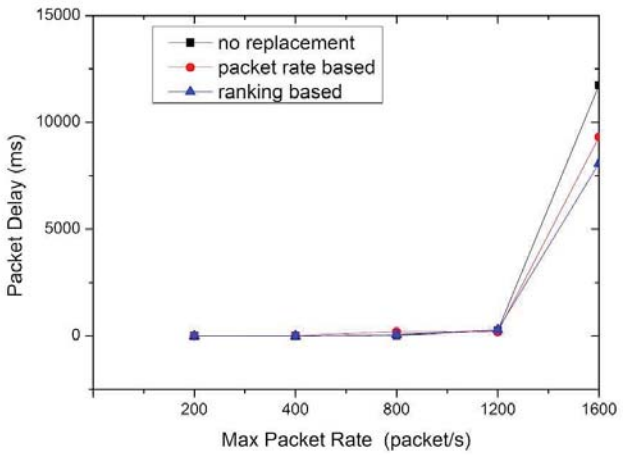

Figure 4. Average packet delay time

5. Conclusion

A smart flow replacement algorithm of a SDN flow table cache is a challenging issue for researchers, though some algorithms have been proposed. This paper was driven by the goal of realizing the impact of diverse packet rates in heavy network traffics to performance of replacement algorithms of SDN flow table cache. In such case, we predict that ranking based and packet based algorithms would be more efficient than LRU algorithm. Our experiment was conducted carefully and the results prove that our prediction is correct.

Acknowledgements

This work was supported by the National Research Foundation of Korea Grant funded by the korean Government(NRF-2014R1A1A1007734).

References

[1] "Software-Defined Networking (SDN) Definition", Available at: https://www.opennetworking.org.

[2] "TCAMs and OpenFlow - What Every SDN Practitioner Must Know", Available at: https://www.sdxcentral.com, 2012.

[3] B.S. Lee, R. Kanagavelu, and K. Aung, "An efficient flow cache algorithm with improved fairness in software-defined data center networks", Cloud Networking (CloudNet), 2013 IEEE 2nd International Conference on, pp.18 - 24, Nov 2013.

[4] N. Katta, O. Alipourfard, J. Rexford, and D. Walker,

"Infinite cacheflow in software-defined networks", Proceedings of the Third Workshop on Hot Topics in Software Defined Networking, HotSDN '14, New York, NY, USA, pp.175 - 180, ACM, 2014.
[5] N.T.T Hiep, "Ranking-based Flow Table Cache for Scalable Software Defined Network", Master Thesis, Department of Electronics and Computer Engineering, Chonnam Nation University, July 2015.
[6] "The least recently used page replacement algorithm", Accessed: 2015-09-1
[7] A.R. Curtis, J.C. Mogul, J. Tourrilhes, P. Yalagandula, P. Sharma, and S. Banerjee, "Devoflow: Scaling flow management for high performance networks", Proceedings of the ACM SIGCOMM 2011 Conference, SIGCOMM '11, New York, NY, USA, pp.254 - 265, ACM, 2011.
[8] S. Banerjee and K. Kannan, "Tag-in-tag: Efficient flow table management in sdn switches.," CNSM, ed. D. Raz, M. Nogueira, E.R.M. Madeira, B. Jennings, L.Z. Granville, and L.P. Gaspary, pp.109 - 117, IEEE, 2014.
[9] M. Yu, J. Rexford, M.J. Freedman, and J. Wang, "Scalable Flow-Based Networking with DIFANE", SIGCOMM Comput. Commun. Rev., vol.40, no.4, pp.351 - 362, Aug. 2010.
[10] OpenDaylight, "https://www.opendaylight.org", Accessed: 2015-09-01.
[11] "http://mininet.org", Accessed: 2015-09-01.