2019 International Conference on Vision Towards Emerging Trends in Communication and Networking (ViTECoN)

Dynamic Hard Timeout based Flow Table Management in Openflow enabled SDN

Abinas Panda
Computer Science and Engineering
National Institute of Technology
Rourkela,India
abinash.panda1987@gmail.com

Siddharth Shankar Samal
Computer Science and Engineering
National Institute of Technology
Rourkela,India
sidhuking07@gmail.com

Prof Ashok Kumar Turuk
Computer Science and Engineering
National Institute of Technology
Rourkela,India
akturuk@nitrkl.ac.in

Aliva Panda
Computer Science and Engineering
Center of Advanced Post Graduate Studies,BPUT
Rourkela, India
aliva.panda14@gmail.com

Vummadi Chetty Venkatesh
Computer Science and Engineering
National Institute of Technology
Rourkela,India
vummadichettyvenkatesh@gmail.com

***Abstract*—Software Defined Networking is the innovation in the network which allows us to manage the network dynamically in an efficient way. Ternary Content Addressable Memory (TCAM) is being used by the OpenFlow switches because it is faster look-up. However, it is very costly, and also the number of flow entries accommodated in it is less due to its limited size. So to take full advantage of the TCAM, we need to maximize the flow table utilization. To improve the TCAM allocation, This paper emphasizes managing the Hard Timeout of the flow table entries. We are proposing a Dynamic way to allocate the flow entry timeout so that the flow table can be used optimally by assigning different dynamic timeout considering predictable and unpredictable flows. The results show that Dynamically allotted hard timeout along with LRU as the eviction method performs better than the Statically allotted hard timeout along with LRU.**

***Index Terms*—SDN, Hard Timeout, TCAM, OpenFlow**

I. Introduction

A. Software Defined Networking

Software Defined Networks (SDN) [1] adopts an innovative network architecture that aims to directly program the network computing. SDN gives us provision to manage network behavior and control in a dynamic way which was not possible in case of traditional networks. SDN has separate data plane and control plane as in figure 1. It's control plane is in the controller while to interact with switches we have OpenFlow [2] protocol to take care of it. SDN controller stores the flow entries that are created in the SDN switches and it forwards traffic based on the flow entries made in it. So it basically decouples the data plane and control plane.

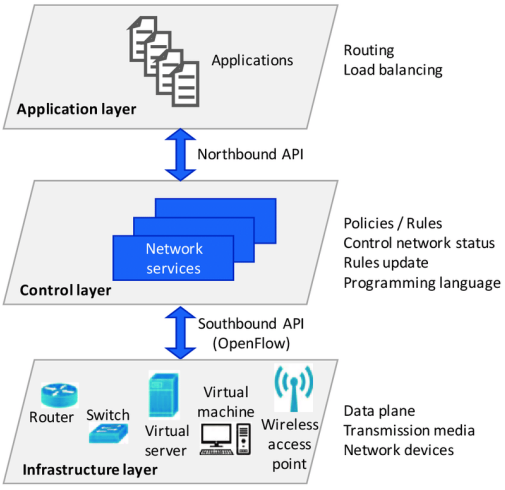

Fig. 1. SDN Architecture [3]

B. OpenFlow

OpenFlow is the standard protocol for SDN that is being used in the controllers of SDN. The controller can communicate with the switches via the OpenFlow Protocol. It can add the flow policies in the switch and set up paths. The controller can manage the different switches from different vendors using a single and open protocol i.e. OpenFlow. It decides the path using various characteristics such as the latency, minimum number of hops and others. It separates the data

978-1-5386-9353-7/19/$31.00 ©2019 IEEE

plane and control plane that helps us in having a programmable network. OpenFlow is no doubt a centralized network but it can improvise to adapt the fluctuating requirements of the network.

In OpenFlow switch specification 1.3.0, we are concerned over the flow table overflow problem. Since the TCAM (Ternary content addressable memory) [4] is very costly and has certain limitations on hardware to have very small number of policies saved, the flow table usually suffers the flow table overflow problem.

C. TCAM

Ternary content-addressable memory (TCAM) is a CAM that has don't care condition in addition to the 1 or 0 state. This feature helps us to search look-up in the table with O(1) time complexity. For achieving the O(1) time complexity, the intense and costly hardware of TCAM comes into play. OpenFlow flow tables use TCAM for storing the flow entry rules. TCAM can fast up different processes that require flow table. The fact that it is very costly leads us to limited TCAM size. Normally the TCAM sizes are able to hold around 2k rules.

D. Overflow

Since the OpenFlow switch stores the flow entry in a flow table which has a limited capacity which is implemented in TCAM, the table gets full when number of different flows increases.

So when we get a new request and the flow comes for installation, instead of installing we first see if the rule is in the table or not. If it is not in the table then it is considered as a miss and another packet_in happens and one of the rules from the flow table is evicted so as to make space for the incoming packet_in flow.

E. Timeout

Every flow rule while getting installed in the Flow Table is assigned a timeout value. It is the time after which the flow is to be removed from the flow table if no packets of the flow is moving through.

Timeout can help us improve the flow table overflow problem as there must be vacant space for the incoming flows to install their rules. Each flow entry may have an idle timeout and or a hard timeout associated with it. Timeouts are of two types:

- Idle Timeout: The corresponding flow entry will be deleted after this time if no packets has been matched by the flow.
- Hard Timeout: The corresponding flow entry will be deleted after this time regardless of whether the packets are being matched or not.

Different controller and their idle and hard timeout:

Controller	**Idle Timeout (in seconds)**	**Hard Timeout (in seconds)**
OpenDaylight	0	0
Floodlight	5	0
Pox	10	30
Ryu	0	0
Beacon	5	0

F. Motivation

The timeout is a very sensitive parameter for the flow rules in the flow table. If we set low timeout values, the flow rules will be deleted soon even if the flows are still active. If we set the timeout high, there's a great chance that the rule will block the space for many incoming new rules. All the flow table management strategies that are currently being used normally use a fixed idle and hard timeout value [5]. They fix the value of timeout irrespective of the volume and duration of the flow. That means the flows are going to reside in the TCAM memory until the timeout period is over. This will be a overhead as their performance is acceptable if the Flow Table memory is quite sufficient. But in SDN where the Flow Tables are very limited, these methods fail to give good performance.

G. Objective

The main objective here is to improve the TCAM occupancy by utilizing proper timeouts for different flows in a dynamic way. The parameters are:

- Reduce the number of capacity misses
- Improve the hit ratio
- Flow Table Occupancy

These parameters will help us improve the TCAM occupancy and since the timeout values are given dynamically which means that we will have better vacancy in the Flow Table.

H. Organization of the paper

The remainder of this paper is structured as follows:Section II describes the related work due to timeout based eviction in SDN, in Section III we have the system design and then in Section IV we have the proposed work which is later evaluated using Ryu and OpenVSwich SDN platform in Section V. finally section VI concludes the paper.

II. Related Work

There has been lot of researches for optimizing of flow table in SDN due to its limited size storage in TCAM and cost.This optimization is achieved through various ways like compressed flow size and highly efficient data structures, timeout managements and some of use the factors that depend upon the characteristics and type of flow.

In [6] author optimizing the flow table by assign a suitable timeout based on flow charastetritics such as inter-arrival of packet by adding a cache module in controller and dynamically adjust the timeout value by seeing the current load in the SDN controller.

He has later proposed another way to tackle this problem by implementing a method called TimeoutX [5] that combines traffic characteristics, flow types and Flow table utilization ratio to decide the timeout of each entry.

In [7] author achieve scalability by considering fine grain policy and maximize the flow table utilization while optimizing the hard timeout value according to network condition. In [8] have developed a model which divides the flows table into multiple flow table as sub-flow table and further divide based on different field layer of the flow attribute.

In [9], they have proposed an efficient flow table management proposition through intelligent autonomous eviction mechanism .Instead of relying entirely on the expiry period alone of a flow entry, author propose and efficient data structure as multiple bloom filter(MBF) to decide the flow entry removal

In [8] proposed an algorithm which is to predict the number new flows and estimate the number remaining number of flows using Weibull distribution.Here by reducing the number of flow setup request made to controller .To achieve scalability they dynamically assign the timeout so as to have enough memory to store new flow entries.

While all these research have been better with time, there's still a scope for improvement. We aim to improve those methods using dynamic allocation for timeouts.

III. System Design

A. Flow and Rules in SDN

SDN is a rule based system for forwarding of packet, where rule means a unique characteristics of a flow ,which give how to process a packet in the network.Whenever a packet arrive to the network through openflow switch it check for any previous matching flow associated with packet is present or not, if yes process the packet accordingly or else install a new flow rule in the switch memory directed from controller through a packet-in event and packet-out event. Here, we have taken flow as five attribute such as IP source, IP destination, source port, destination port and protocol to define a flow which make size of rule is 256 bit. A flow table consists of flow entries.

Match Fields	Priority	Counters	Instructions	Timeouts	Cookie

These are the main components of a flow entry in a flow table, where match fields are the fields which selects the flow entry matching with these match fields in the TCAM. Priority is the precedence at which the flow entry is being matched and counters are incremented when the match fields are matched. There is instruction set to modify any action for that flow entry. Timeout is given to the flow entry while installing the rule and it's the time after which the flow entry is to be removed from the flow table. Cookies are there to get insights of the flow and it's value is set by the controller.

B. Notation

r	Average inter-arrival time of last N packets
t	Timeout value
t_{max}	Maximum Timeout value
tt_{tcam}	Threshold value for TCAM occupancy after which eviction starts

C. Idle Timeout Based Eviction

Eviction will be done once the TCAM occupancy reaches 90%. The eviction will be done according to the following criteria. If there are flows having the timeout values as Max idle Timeout, then among them we will evict the least recently used, i.e., LRU.

IV. Proposed Method

A. Problem Formulation

Generally, we have two different type of flows. It can be predictable and unpredictable type of flows. Predictable flows are the normal flows which reside for long time. It is the traffic from deterministic network services and periodic in nature while unpredictable flows are spontaneous in nature and are not periodic. We should keep the unpredictable flow rules for less time in the TCAM while we should increase the availability for the predictable flows in the TCAM.

- Hit ratio: The hit ratio is defined as such, let the number of packets that go through the switches using the flow rules already present in the flow table is c and the number of packets that undergo *PACKET_IN* condition as it could not match any existing flow rules in the flow table be pc then we define:

$$HitRatio = \frac{c}{c + pc}$$

- Capacity Miss Rate: First of all, capacity miss is the flow table miss that is caused by the filling of the flow table and not having enough storage to accommodate the new incoming flow entry. The miss rate is the ratio of the packets that miss the flow table due to capacity miss (cm) upon the total number of miss (tm).

$$CapacityMissRate = \frac{cm}{tm} * 100$$

- Flow table occupancy : Flow table occupancy is defined as the percentage of flow table that are occupied by the existing flow rule entries.

B. Dynamic Hard Timeout Allocation

We propose a dynamic method to allocate the timeout such that we will have maximum utilization of the flow table. In this method we will first look after the last N packets of the flow to get their average inter-arrival time of the packets. Instead of directly installing the packet, we wait for small amount of time to let pass few packets and then we get r: rate of similar flows arriving at a particular second.
Now for the first time when the flow is being installed, we

set the hard timeout value to 1s [10]. As we know that almost 75% of the flows are having hard timeout value less than 1s. After that with each N packet arrival, we update the timeout as such :

$$t = max\{t * 2^r, t_{max}\}$$

Maximum timeout here has been set as 11s [11] as we know that nearly 90% of the flows have timeout value less than 11s. At any given time if the TCAM occupancy overshoots 90% then we start evicting entries with MaxTimeout values in a LRU form.

C. Algorithm Description

We are proposing this algorithm for the dynamic hard timeout allocation. The hard timeout deletes the flow entry from the table after the given time irrespective of whether the packets of the flow has been matched or not. The algorithm for same follows as:

Algorithm 1 Dynamic Hard Timeout Allocation Algorithm

procedure DHTA($packets, N$)
  $packet\_counter \leftarrow 0$
  while $packet\_counter < N$ **do**
    $checkPacketArrival(N)$
  $t \leftarrow DEFAULT$
  $r \leftarrow AvgInterArrivalTime(packets, N)$
  if packet is matched **then**
    if flow is unpredictable **then**
      $t \leftarrow max\{t * 2^r, t_{max}\}$
  return t

Now after the timeout being set, there will be situations where the TCAM occupancy will be more than tt_{tcam} which is 90

Algorithm 2 Eviction Condition

procedure EVICTION CONDITION($TCAM_{occupancy}$)
  if $TCAM_{occupancy} > tt_{tcam}$ **then**
    Select all flow entry having timeout $t = t_{max}$
    Delete the flow entry based on LRU
  return $deleted\_flow\_entry$

V. Evaluation

A. System Setting

We have evaluated our proposed work compared to existing work by taking Ryu as SDN controller and Mininet as network emulator while taking OpenFlow protocol version 1.3 due to its supported by maximum no of hardware device.We have taken switch as software switch as OpenVswitch for experimental evaluation and TCAM utilization by taking their no of rule to be kept varying from 500 to 3000.We have used two dataset name UNIV1 and UNIV2 for packet processing and rule installation.We have extensively simulated for 10 min

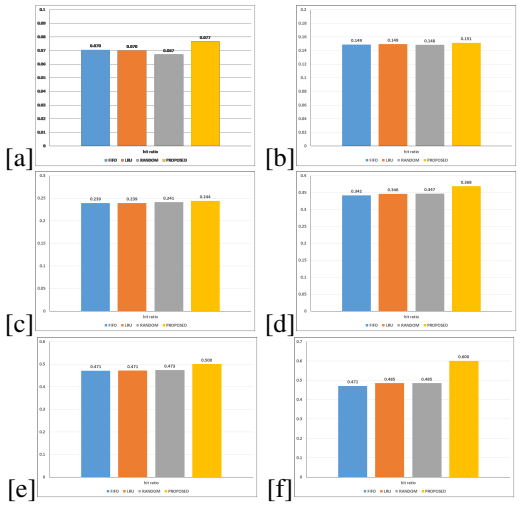

Fig. 2. (a), (f) Various hit ratio for different TCAM size such as 500,1000,1500,2000,2500,3000 with respect to different static eviction policy and our proposed dynamic hard timeout allocation policy.

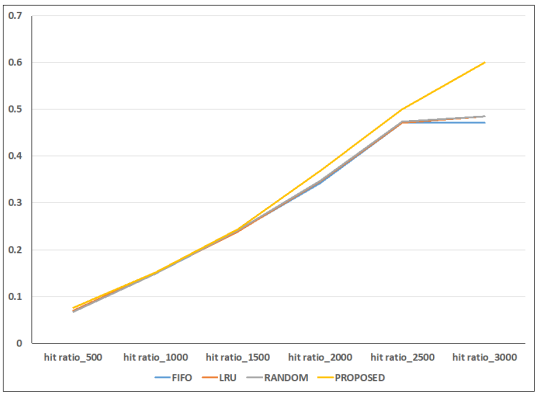

Fig. 3. Hit ratio vs Different policy for different TCAM size

and used the traffic pattern as in dataset. We have used static timeout policy such as FIFO, LRU, RANDOM for comparison of our work.

TCAM size	500,1000,1500,2000,2500,3000
Idle Timeout	Default value of Controller
Switch	OpenVSwitch [12]
Controller	Ryu [13]
Simulator	Mininet [14]
Dataset	UNIV1, UNIV2 [15]
Eviction Type	LRU
Simulation Time	10 min
Traffic Distribution	Logarithmic, Exponential
OpenFlow version	1.3

B. Parameter for Evaluation

The parameters for evaluation here are :

1) Hit Ratio: The hit ratio is the number of packets that doesn't need rule installation while it comes to the switch

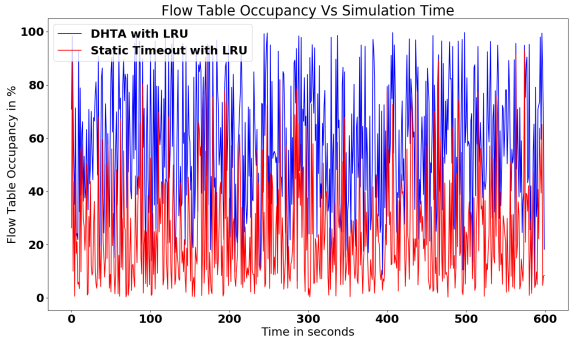

Fig. 4. Flow table occupancy percentage vs simulation time

upon all the packets.

In figure(1) (a), (b), and (c) shows that the proposed algorithm having TCAM sizes as 500,1000,1500 having the highest hit ratio followed by FIFO, LRU and Random respectively. (d), (e), and (f) shows that the proposed algorithm having TCAM sizes as 2000, 2500, 3000 having the highest hit ratio followed by Random, LRU and FIFO respectively.

Figure(2) shows Hit Ratio of different eviction policies (Proposed algorithm, FIFO, LRU, Random Algorithms) for different TCAM sizes(500, 1000, 1500, 2000, 2500, 3000) which results that the proposed algorithm performs better with increasing hit ratio between TCAM sizes 2500-3000 than other eviction algorithms.

2) Flow table occupancy: Flow table occupancy rate is the number of rule or unique flow to be placed in TCAM in openflow switch for forwarding of packet with respect to total number of incoming flows.

Figure(3) shows the flow table occupancy percentage w.r.t simulation time for Dynamic Hard Timeout Allocation policy with LRU and Static Timeout with LRU which results that the proposed DHTA with LRU having the more Flow Table Occupancy rather than using Static Timeout with LRU all the time.

3) Capacity miss rate: Capacity misses are the miss which occurs due to no space for rule installation in the TCAM. Since there is no space so we have to discard the flow until we evict and make space for the rule to be installed.Capacity miss rate is the percentage of miss with respect to total number flows.

In figure(4) (a) shows that the proposed algorithm having TCAM size as 500 having the lowest capacity miss ratio followed by FIFO, Random, LRU respectively. (b) shows that the proposed algorithm having TCAM size as 1000 having the lowest capacity miss ratio followed by LRU, FIFO, Random respectively. (c) shows that the proposed algorithm having TCAM size as 1500 having the lowest capacity miss ratio followed by Random, FIFO, LRU respectively. (d), (e), and (f) shows that the proposed algorithm having TCAM sizes as 2000, 2500, 3000 having the lowest capacity miss ratio followed by Random, LRU, FIFO respectively.

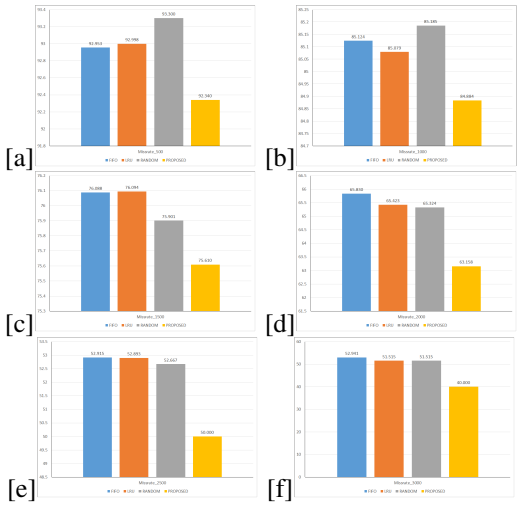

Fig. 5. (a), (f) Various capacity miss ratio for different TCAM size such as 500, 1000, 1500, 2000, 2500, 3000 with respect to different static eviction policy and our proposed dynamic hard timeout allocation policy.

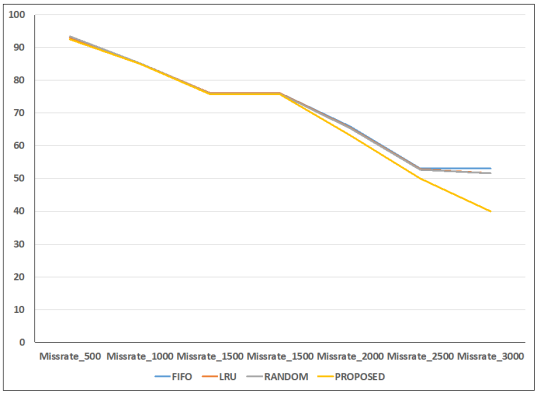

Fig. 6. Capacity miss rate vs Different policy for different TCAM size

Figure(5) shows Capacity miss ratio of different eviction policies (Proposed algorithm, FIFO, LRU, Random Algorithms) for different TCAM sizes (500, 1000, 1500, 2000, 2500, 3000) which results that the proposed algorithm performs better having low capacity miss from TCAM size 2500 to TCAM size 3000.

VI. Conclusion and Future Work

We can see the results show that the hard timeout allotted by the DHTA along with the LRU eviction method works well compared to other. This shows that we can optimize the Flow Table utilization by having proper hard timeout allotment. Having a dynamic timeout helps us analyze the traffic properly and setting up the timeout unlike setting it

fixed.

In future, we can now explore similar dynamic approach for finding out optimal combination of timeout values for both idle and hard timeout, at which we can get the maximum benefit of the TCAM without exhausting it. It should help us optimizing the flow table even more.

REFERENCES

[1] D. Kreutz, F. M. Ramos, P. E. Verissimo, C. E. Rothenberg, S. Azodolmolky, and S. Uhlig, "Software-defined networking: A comprehensive survey," *Proceedings of the IEEE*, vol. 103, no. 1, pp. 14–76, 2015.

[2] N. McKeown, T. Anderson, H. Balakrishnan, G. Parulkar, L. Peterson, J. Rexford, S. Shenker, and J. Turner, "Openflow: enabling innovation in campus networks," *ACM SIGCOMM Computer Communication Review*, vol. 38, no. 2, pp. 69–74, 2008.

[3] Z. Bojovic, "Implementing Software Defined Networking in Enterprise Networks," no. April, 2018.

[4] B. Agrawal and T. Sherwood, "Ternary cam power and delay model: Extensions and uses," *IEEE transactions on very large scale integration (VLSI) systems*, vol. 16, no. 5, pp. 554–564, 2008.

[5] L. Zhang, S. Wang, S. Xu, R. Lin, and H. Yu, "TimeoutX: An adaptive flow table management method in software defined networks," *2015 IEEE Global Communications Conference, GLOBECOM 2015*, 2015.

[6] H. Zhu, H. Fan, X. Luo, and Y. Jin, "Intelligent timeout master: Dynamic timeout for sdn-based data centers," in *2015 IFIP/IEEE International Symposium on Integrated Network Management (IM)*. IEEE, 2015, pp. 734–737.

[7] Z. Guo, R. Liu, Y. Xu, A. Gushchin, A. Walid, and H. J. Chao, "Star: Preventing flow-table overflow in software-defined networks," *Computer Networks*, vol. 125, pp. 15–25, 2017.

[8] T. Kim, K. Lee, J. Lee, S. Park, Y. Hwa Kim, and B. Lee, "A Dynamic Timeout Control Algorithm in Software Defined Networks," *International Journal of Future Computer and Communication*, vol. 3, no. 5, pp. 331–336, 2014. [Online]. Available: http://www.ijfcc.org/index.php?m=content&c=index&a=show&catid=49&id=611

[9] D. Wang, Q. Li, L. Wang, R. O. Sinnott, and Y. Jiang, "A hybrid-timeout mechanism to handle rule dependencies in software defined networks," *2017 IEEE Conference on Computer Communications Workshops, INFOCOM WKSHPS 2017*, pp. 241–246, 2017.

[10] G. Zhao, H. Xu, S. Chen, L. Huang, and P. Wang, "Joint Optimization of Flow Table and Group Table for Default Paths in SDNs," *IEEE/ACM Transactions on Networking*, vol. 26, no. 4, pp. 1837–1850, 2018. [Online]. Available: https://ieeexplore.ieee.org/document/8424423/

[11] L. Wang, Q. Li, R. Sinnott, Y. Jiang, and J. Wu, "An intelligent rule management scheme for Software Defined Networking," *Computer Networks*, vol. 144, pp. 77–88, 2018.

[12] B. Pfaff, J. Pettit, T. Koponen, E. J. Jackson, A. Zhou, J. Rajahalme, J. Gross, A. Wang, J. Stringer, P. Shelar *et al.*, "The design and implementation of open vswitch." in *NSDI*, vol. 15, 2015, pp. 117–130.

[13] A. A. L. Mantas, "Consistent and fault-tolerant sdn controller," Ph.D. dissertation, 2016.

[14] M. Team, "Mininet," 2014.

[15] T. Benson and A. Akella, "Data set for imc 2010 data center measurement," 2010.