Congestion Control in Software Defined Data Center Networks Through Flow Rerouting

Masoumeh Gholami
The Faculty of Electrical and Computer Engineering
Tarbiat Modares University
Tehran, Iran
masoomeh.gholami@modares.ac.ir

Behzad Akbari
The Faculty of Electrical and Computer Engineering
Tarbiat Modares University
Tehran, Iran
b.akbari@modares.ac.ir

***Abstract*—nowadays, the performance of data center networks is very important due to hosting different cloud applications and services. The performance of such networks is affected by many factors, like network design and the congestion control mechanism used in the network. Today data center networks have high volume and dynamic traffic due to the deployment of cloud services that make it more challenging to control the congestion in an efficient manner. By the emergence of Software Defined Networking (SDN), a new horizon opened up for the new generation of data center networks through separation of control and data planes. In this paper we propose an efficient method to control the congestion in the software defined data center networks based on OpenFlow protocol. In our proposed method, congestion in the network links is detected by centrally monitoring the port statistics of the OpenFlow enabled switches and then some of the flows in any congested link are rerouted through paths with more free resources by the OpenFlow controller. We implemented our proposed method by Mininet. The experimental results show the efficiency of the proposed method in decreasing the congestion and improving the performance of the network.**

Keywords—data center; congestion control; software-defined network; OpenFlow.

I. INTRODUCTION

Recently, data centers services have been increasingly used in campus, enterprise, and companies for running a wide range of programs and applications. Many internet communications occur in data centers and a large number of concentrated programs related to data centers are hosted using the one-to-many and many-to-many communication patterns. Electricity consumption, cooling system, infrastructure maintenance, and administrative issues have a really high OPEX cost in a data center due to the large amount of communication traffic in it. Hence, it is necessary to develop an integrated network for data centers, which meets administrative capabilities, and efficiency needs such as different types of traffics. Through commercializing the Ethernet as a solution with most efficient cost for the creation of infrastructure, an integrated network with certain criteria such as scalability, efficiency, and integrity has been proposed. However, the proposed system still lacks a set of quality service requirements, especially Ethernet congestion management. Regarding the characteristics of data centers, Ethernet congestion management must consider issues such as low delay, low packet loss and high throughput of the network. Therefore, data centers need flexible methods for congestion control in a way that they would be able to provide low delay and packet loss, which consequently, lead to high throughput of the network.

IEEE Data Center Bridging (DCB) task group is the director of developing new schemes for congestion control in data center networks [1]. Here, IEEE802.3 standard provides a link surface current control mechanism known as PAUSE (802.3x) which can temporarily stop the link when the buffer is full. However, this procedure consequently results in congestion expansion. The IEEE 802.1Qau group has proposed a new two-layer End-to-End congestion control design known as Quantized Congestion Notification (QCN). In this design, whenever congestion occurs in a switch, a congestion message is transmitted to the source that caused congestion to reduce its data transmission rate. By delivery of such alarm, the related source decreases its transmission rate by activating the rate limiter [1, 2, 3, 4 and 5].

Software Defined Networking (SDN) [6] is an innovative network technology that offers high interoperability and cost-efficient ways for user control and network programmability in data center networks. The main difference between a traditional networking and SDN based networking is separation of data plane and control plane in SDN based networks. In SDN technology, the whole network is centrally managed by a dedicated controller which interacts with network switches using the OpenFlow protocol. A common OpenFlow network is composed of three components: the OpenFlow controller, OpenFlow switches, and hosts. Each of the switches maintains a flow table that contains routing information. The controller and switches communicate via OpenFlow messages. There are a series of actions that the OpenFlow controller can perform by sending messages to the switches, such as updating flow tables or probing switch statistics. By analysis of the reply messages from the switches, the OpenFlow controller can detect network congestion and reroute the network paths, by using OpenFlow protocol, in order to decrease the congestion [7, 8].

In this paper, an efficient method based on Software-Defined Networking is introduced for reduction of congestion in data center networks. The main characteristics of the proposed method are 1) the enhanced operational efficiency, 2) dynamic decision making about the congestion and 3) the efficiency of response. In this method, the congestion detection

978-1-4799-1972-7/15/$31.00

and routing component are modules of OpenFlow controller. The main property of our proposed method is the rerouting of some selected flows in the switches with the congested links. In order to evaluate our proposed congestion control method, we modeled a data center network with Fat-tree topology by using Mininet emulator [9] and Floodlight controller [10]. Our experimental results show that by deploying our proposed approach, the performance metrics, such as throughput and end-end-delay, are considerably improved.

The rest of this paper is organized as follows: Section 2 presents our proposed method for congestion control. Evaluation of the proposed method is provided and discussed in sections 3 and finally, section 4 provides conclusions and some suggestions for further work.

II. THE PROPOSED CONGESTION CONTROL METHOD

The conventional topology applied in data center network is Fate-tree, in which top-of-rack (ToR) switches in the access layer provide the communication between the rack-mounted servers. In this system, each aggregation switch (AS) in aggregation layer (which is also known as distribution layer) routs traffic from several switches of access layer to the core layer. Each ToR switch is linked to several aggregation switches. The core layer provides the secure connection between the aggregation switches and core routers (CR) connected to internet [11].

Fig. 1 illustrates the topology of Fat-tree with k=4, which consists of K-port switches with k pod. Each pod is composed of two (aggregation and edge) layers of k/2 switches. In addition, each aggregation is composed of (k/2)2 core switches with one port attached to each k pod. The i^{th} port of each core switch is connected to i^{th} pod, so that the successive ports in the aggregation layer of each pod switch are attached to the core switches. Each edge switch is directly connected to the host k/2 and the remaining k/2 port of the edge switches is connected to a k/2 aggregation switch.

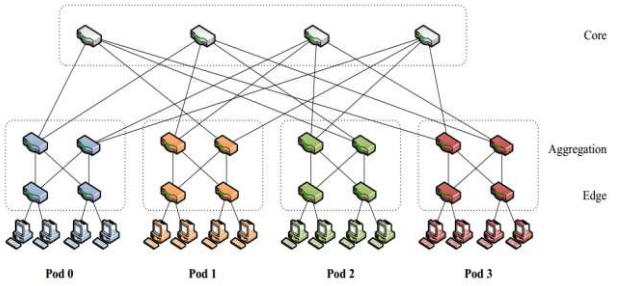

Fig. 1. The topology of Fat-tree with k=4.

A. Proposed method

Congestion control components of the OpenFlow controller has two modules: congestion detection module and control module. A congestion control prototype has been implemented in the Floodlight controller [10], which is a component based OpenFlow controller.

1) Congestion detection module

The aim of this module is to query, consolidate and store the statistics from all OpenFlow switches. These statistics are then used by the controlling module to compute the load on various links. The main idea behind it is to route the packets along the lightly loaded path. Hence, loads the links along the possible path must be specified. This module gathers statistics from each OpenFlow switch (per table, per flow and per port) by polling them at fixed intervals. Once the transmitted bytes are 70% higher than the link capacity, the congestion conditions occur in the port. The controller requests STATS_REQUEST message assigned to the PORT in the network as the statistics of each switch. Then, the switch responds to the controller with STATS_REPLY message. The overhead of monitoring port statistics is trivial. The sizes of messages for requesting and replying port statistics on each port are 8 bytes and 104 bytes respectively [12].

2) Controlling module

The controlling module is responsible for rerouting. It computes the minimum loaded shortest paths based on the statistics collected from the switches by the congestion detection module.

Whenever a congestion occurs in the system, the controller performs the rerouting process based on the network topology. For further rerouting, first, it computes all the possible shortest paths. Once the set of all shortest paths P is computed, the statistics from the congestion detection module will be used to compute the least loaded path. The total cost of each path $p_i \in P$ is defined as $y_i = a_i + b_i$.

$$a_i = \sum_k^N w_k \qquad (1)$$

$$b_i = \sum_k^N \left(\frac{L_k}{C_k}\right)$$

Where a_i is the total fixed costs of each link in the path $l_k \in p_i$ and b_i is the total variable costs of $l_k \in p_i$. Each l_k has a predetermined fixed weight w_k, a link load of L_k bps computed from the statistics, and link capacity C_k bps. Currently, all link weights are set to 1. The value of L_k is estimated from the change in byte count for all flow entries in a switch between the two most recent statistics from the congestion detection module. After computing paths y_i from all $p_i \in P$. The controller then picks the path that has the minimum y_i. If the amount of statistics collected is insufficient, a route is randomly picked from P. The controller through the OFP_FLOW_MOD message, the current table of the switches are updated [12].

III. EVALUATION

The Mininet emulator is utilized to carry out simulation process in the proposed method. Mininet creates virtual networks (in a few minutes) including core, switch, user codes and staff like that just on a single machine (virtual machine, cloud, etc.). Because of the simple communing and customizing the most of components, this emulator can be applied in various development, training, and research projects.

Floodlight controller is an open-source project with modular and flexible controller platform at its core. This OpenFlow controller is implemented purely in software and is included in its own Java Virtual Machine (JVM). As such, it can be deployed on any hardware and operating system

platform that supports Java. Floodlight is a community for the promotion and/or proposing standardization of SDN Northbound APIs, so that services that use an OpenFlow controller can be written quickly and efficiently.

To generate traffic on the network, the highly precise Ipref instrument [13], which suits analytical and industrial purposes, was used in this work. Ipref is an open source code, which can be used for the evaluation of the traffics generated through TCP and UDP. This instrument reports different types of statistical scales such as throughput delay dispersion, and datagram removal. Using the Ipref instrument, the packets with determined size are transmitted in a given time and a specific share of load proportional to the linkage rate. The applied values are as follows:

Packet size: 64 byte

Load amount: 10% to 99%

Duration: 1 minute

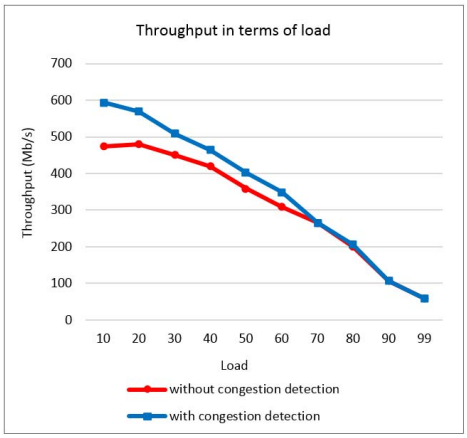

Fig. 2. Throughput considering different amounts of load on the network.

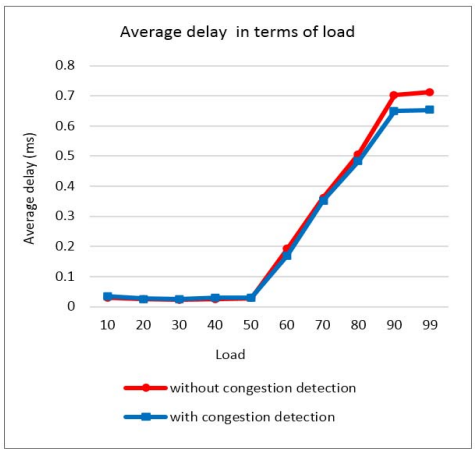

Fig. 3. Average delay with load size.

Fig. 2 and 3 show throughput (Mb/s) and average delay (ms), respectively. It must be noted that the computed delay is not two-way and just the time difference between the source and destination is considered. In addition, since characteristics of the links are considered as the same, the average is measured as point-to-point for the entire delays.

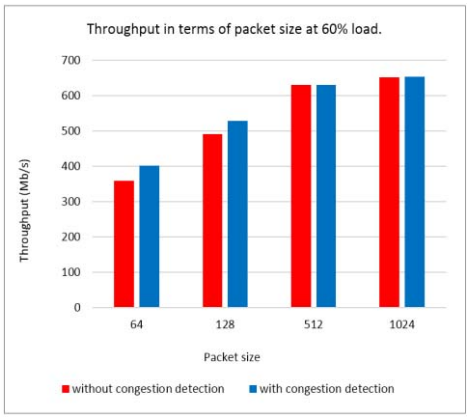

Fig. 4. Throughput for different packet sizes at 60% load.

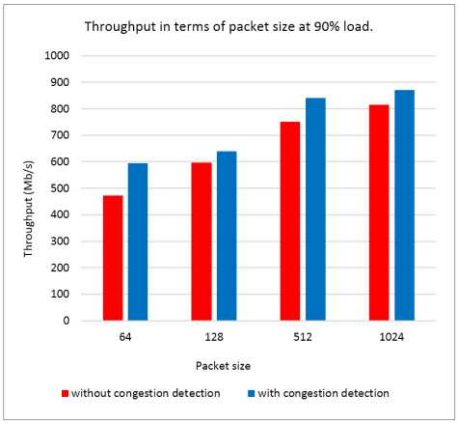

Fig. 5. Throughput for different packet sizes at 99% load.

Fig. 4 and 5 indicate the throughput for different packet sizes where load percentages are maintained at 60 and 99%, respectively, while the packet size varies in the range of 64 to 1024 byte.

To achieve an overall trend resulting in an acceptable decision-making on the efficiency and evaluation of the proposed method, a considerable number of tests were conducted, some of which were presented in the previous sections. It must be noted that these tests are carried out considering different values of the applied loads out of the total possible capacity of the links.

The most evident result of this work is that once the minimum congestion does not exist, a random processing is performed on the packets within the equal intervals, which can potentially increase the delay values. However, according to the performed tests and assuming that all switches can support the OpenFlow, the overall overload does not differ by examining only one condition (a single if command) or its minimum value is so small that it does not show any difference even with a low measuring precision.

One of the most interesting results of this work is the effect of the proposed method for the cases where the line space is considered rather independent from the packet size, where the increase in packet size results in less congestion. Thus, the effect of congestion control procedure is less noticed.

Nevertheless, for packets with larger sizes it is obvious that the proposed method not only does not reduce the network efficiency, but also indicates a considerable rise in load peak.

Another issue discussed in this work is the congestion delay, where by the increased packet size, the overall delay is reduced and more delayed congestion occurs. Thus, the enhanced efficiency of the network is observed using the proposed method at higher percentages of load insertion on the network.

IV. Conclusion and Future Work

The most important achievement of this article is the efficiency enhancement of data center networks by means of the OpenFlow enabled switches. Using the proposed method, a significant improvement was observed in the efficiency of the data center, particularly in the throughput enhancement and average packet delay reduction. Another key point of the proposed method is its higher generalization potential, which originated from the minimum possible assumption on the problem conditions. This capability of the system results in the simple integration of the proposed method to the data centers SDN platforms. Future work will be dedicated to exploit machine learning methods for improvement of the detection and management components of our scheme.

References

[1] IEEE. 802.1 - Data Center Bridging Task Group. Available at http://www.ieee802.org/1/pages/dcbridges.html.

[2] G. Smith et al, "Converged Enhanced EthernetGood for iSCSI SANs" NetApp White Paper, Blade Network Technologies., 2008.

[3] J.R. Santos, Y. Turner, and G. Janakiraman, "End-to-end congestion control for InfiniBand," in *INFOCOM: Twenty-Second Annual Joint Conference of the IEEE Computer and Communications,* IEEE 2003, Societies, 2003. pp. 1123–1133.

[4] IEEE. 802.1Q-2005 Virtual Bridged Local Area Networks. [Online] Availabl: http://www.ieee802.org/1/pages/802.1Q.html.

[5] Alizadeh. Mohammad et al, "Data center transport mechanisms: Congestion control theory and IEEE standardization," in *Communication Control and Computing: 46th Annual Allerton Conference on*, IEEE 2008, 2008, pp. 1270-1277

[6] N. McKeown et al, "OpenFlow: Enabling Innovation in Campus Networks," in *SIGCOMM Computer. Communication. Reveiw.*, vol. 38, no. 2, pp. 69–74, Mar. 2008.

[7] H. Kim and N. Feamster, "Improving network management with software defined networking," *Communications Magazine, IEEE*, vol. 51, no. 2, pp. 114–119, 2013.

[8] Y. Jarraya, T. Madi, and M. Debbabi, "A survey and a layered taxonomy of software-defined networking," *Communications Surveys Tutorials, IEEE,* 2014.

[9] Mininet. [Online] Availabl: http://mininet.org.

[10] Floodlight. [Online] Availabl: http://www.projectfloodlight.org/floodlight.

[11] M. Al-Fares, A. Loukissas, and A. Vahdat, "A Scalable, Commodity Data Center Network Architecture," in *Proc. ACM SIGCOMM,* August 2008.

[12] Open networking foundation, "OpenFlow Switch Specification," version 1.0.

[13] D. A. S. T. D. at the National Laboratory for Applied Network Research (NLANR). Iperf. [Online] Availabl: http://iperf.sourceforge.net, Mar 2012.