Packet-Based Load Balancing in Data Center Networks

Yagiz Kaymak and Roberto Rojas-Cessa

Networking Research Laboratory, Department of Electrical and Computer Engineering, New Jersey Institute of Technology, Newark, NJ 07102
Email: {yk79; rojas}@njit.edu

Received 4 February 2016; Accepted 25 February 2016;
Publication XXX

Abstract

In this paper, we evaluate the performance of packet-based load balancing in data center networks (DCNs). Throughput and flow completion time are some of the main the metrics considered to evaluate the performance of the transport of flows over the presence of long flows in a DCN. Load balancing in a DCN may improve those metrics but it may also generate out-of-order packet forwarding. Therefore, we investigate the impact of out-of-order packet delivery on the throughput and flow completion time of long and short flows, respectively, in a DCN. We focus on per-packet load balancing. Our simulations show the presence of out-of-order packet delivery in a DCN using this load balancing approach. Simulation results also reveal that packet-based load balancing may yield smaller average flow completion time for short flows and larger average throughput for long flows than the single-path transport model used by Transmission Control Protocol (TCP), which prevents the presence of out-of-order packet delivery. Queueing diversity in the multi-path structure of DCNs promotes susceptibility of out-of-order delivery. As the delay difference between alternative paths decreases, the occurrence of out-of-order packet delivery in packet-based load balancing also decreases. Therefore, under the studied scenarios, the benefits of the packet-based load balancing seem to outweigh the out-of-order problem.

Journal of Cyber Security, Vol. 5, 1–18.
doi: 10.13052/jcsm2245-1439.511

Keywords: data center network, load balancing, multipath forwarding, flow completion time, TCP.

1 Introduction and Motivation

The interconnected servers of a data center run a wide variety of applications and services, which range from web search and social networking [11] to distributed file systems [12, 20]. Some of the underlying data distribution technologies are based on Hadoop [20] and Google File System [12]. While a web search entails small data transfers among servers of a data center, distributed file systems may require large data transfers [19]. Most applications/services in a data center may require multiple servers working in parallel to assemble a response for a requested task. For instance, a Hypertext Transfer Protocol (HTTP) request for a Facebook page requires a large number of servers, called workers, across the data center to be involved in fetching partial results to generate the web page [11].

The performance of data centers is directly related with how fast the user-initiated job requests are processed. Such a performance measure requires to generate responses to those requests as fast as possible [13]. This is where the intercommunication among servers impacts the achieved overall performance of a data center. The communication among workers takes place as flows, where a flow may be defined as a sequence of packets sent from a particular source to a particular destination [18]. Each worker is responsible for generating a small portion of the final result for a submitted task. The partial results are put together as a final result by an aggregator server. The distribution of the task portions to different workers and fetching the results requires a fast transmission of flows [5].

The traffic in a DCN can be coarsely divided into short and long flows. Short flows feature a size of a few kilobytes and they are associated with users' tasks that generate requests and responses between servers. Long flows, or background flows, have a size of several megabytes and they carry the data for the operation and maintenance of the data center. The time required to transmit a flow is referred to as the Flow Completion Time (FCT) [8, 10]. This parameter is a major metric for measuring the performance of the transport of short flows in data center networks (DCNs). The performance of the transport of long flows is measured by the achieved throughput. Therefore, long flows must be transferred with equal to or larger throughput than an acceptable minimum [19]. A high-performance DCN, in combination with the transport

protocol used in it, must simultaneously achieve small FCTs for short flows and high throughput for longs flows. In this paper, we focus on these performance metrics for both types of flows.

Congestion in the DCN is a factor that may impair achieving small FCT and high throughput. DCNs are generally provisioned with multiple paths between source-destination pairs to provide reliability and a large bisection bandwidth. Therefore, link or path congestion may be circumvented by efficiently using the available multiple paths of a DCN. However, if a DCN is not provisioned with a multipath routing/forwarding mechanism, alternative paths may be underused and the network may be underutilized. For instance, the use of the Transmission Control Protocol (TCP) and a shortest-path routing algorithm may not take full advantage of the existing multiple paths in a DCN and the utilization of the DCN may be low. Because all the packets of a TCP flow follows the same path, which might be congested because of the other flows using the same path.

This paper is motivated by the benefits of using a multipath routing/ forwarding mechanism on a DCN and the effect it may have on the FCT for short flows and the throughput for long flows. Based upon the potential benefits of multipath forwarding schemes, we implemented a high- granularity (i.e., packet-based), low-complexity packet-based load balancing scheme based on Random Packet Spraying (RPS) [9], using the NS-3 [1]. The load balancing scheme makes use of the multipath feature of a DCN to balance the traffic among alternative paths, decrease the FCT of short flows, and increase the throughput of long flows. In this paper, we also unveil how the FCT and throughput of flows are affected by background traffic on the DCN.

TCP is a widely-used transport protocol on the Internet and DCNs [7, 16]. We adopt TCP as the transport mechanism in the load balancing scheme to keep the transport layer as simple as possible. However, because the use of packet-based multipath forwarding and the existence of possible delay differences between alternative paths, out-of-order packet delivery may occur at the receiver and TCP may react to these out-of-order packets as congestion indicators. Note that the preliminary results of this paper were presented in [15].

The remainder of this paper is organized as follows. In Section 2, we describe the operation of packet-based load balancing and give our simulation results. In Section 3, we present related works, and in Section 4, we conclude the paper.

2 Mechanism and Evaluation

In this section, we describe the operation of packet-based load balancing. We then present different simulation parameters used to analyze the performance of these two schemes and show the simulation results from a data center using a fat-tree-based DCN [2].

2.1 Mechanism

Packet-based load balancing works at each switch by randomly selecting one of the output ports to independently forward an incoming packet. The selection of the output port considers only equal-cost paths to the destination of the packet. If multiple equal-cost paths to the destination are available, one of them is randomly selected with uniform probability and the packet is forwarded to that specific port. The use of this technique requires calculation of equal-cost shortest paths in advance, so that the packet-based load balancing scheme distributes the traffic among them.

It is expected that equal-cost paths between a source-destination pair have similar queue build-ups and latencies [9]. Having almost equal latencies between these paths may not always be the case because the traffic pattern on the network has an impact on the latency difference between paths and the number of out-of-order packets. This claim is supported by our findings, as Figures 4 and 7 show.

We adopt a DCN using a fat-tree topology with four pods in our study. Each pod hosts four servers, two edge, and two aggregation switches. We use four core switches to connect these pods in the core layer of the DCN. Figure 1

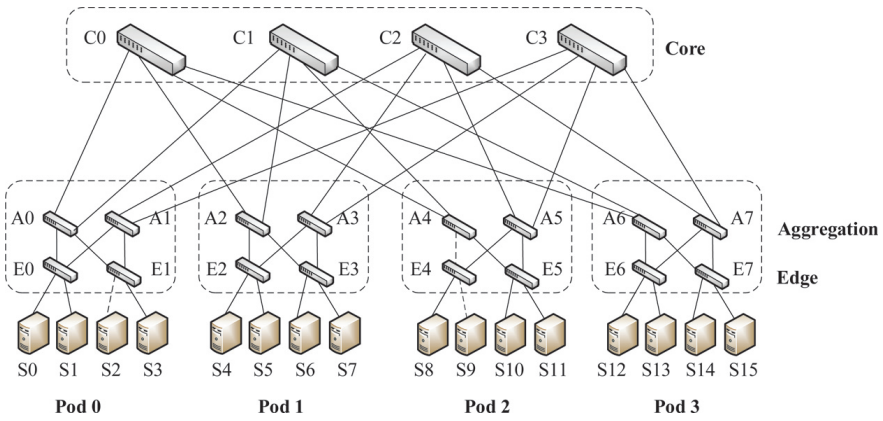

Figure 1 A Fat-tree DCN with four pods.

shows the fat-tree network used in our simulations. The letter on the label of each switch in this figure indicates the layer the switch belongs to. Edge, aggregation, and core switches are labeled by letters E, A, and C, respectively. The number following the capital letter indicates the switch number. Servers are labeled by the letter S and a number.

We select the fat-tree topology as the architecture of its multiple paths between any two servers, which makes multipath forwarding and load balancing suitable for DCNs. In general, a fat-tree network with k pods has $(k/2)^2$ shortest paths between any two hosts [2]. Moreover, a fat-tree network may provide full bisection bandwidth to clusters with a large number of servers [2].

2.2 Evaluation

We base our study on computer simulation. Each simulated switch in our topology represents a commodity switch with a shallow buffer. We select a buffer size equal to 1000 packets, where each packet is 1500-bytes long. Therefore, the switch's buffer capacity is set to 1.5 Mbytes, which is a suitable value to represent the buffer size of shallow-buffered switches [5]. Drop-tail is employed as the packet dropping mechanism in the switch. The DCN uses point-to-point (p2p) links with 1-Gbps full-duplex links. The delay of each link is set to 25 µs to represent a Round Trip Time (RTT) of 300 µs. Note that one-way delay of the longest path in terms of the number of links between a pair of servers in different pods is 150 µs when all the links carry no traffic [5]. Any two interfaces located each side of a p2p link have IP addresses from the same subnet. In other words, we created a different subnet for each p2p link that interconnects two nodes. To keep the routing complexity low at each switch, we implemented packet-based load balancing by enabling packet-based random routing as the routing function in NS-3. Packet-based random routing allows a switch to randomly select one of its output ports among the possible equal-cost shortest paths for each incoming packet. Therefore, random routing may enable each packet to follow a distinct shortest path. Our simulation considers hop count as the primary metric to calculate the shortest path, although different metrics, such as one-way delay or residual capacity of a path may also be used. As an example of how packet-based load balancing works, let's assume that a flow is created from S0 to S15, as Figure 1 shows. In this example, each packet is first forwarded by switch E0, where there are two possible next-hop switches, A0 and A1, with equal-cost shortest paths to the destination node, S15. Depending on the selection of the edge switch, A0 or A1, corresponding core switches, C0, C1, C2 or C3, can be selected.

These edge and core switch selections yield four different equal-cost shortest paths as $S0 \rightarrow E0 \rightarrow A0 \rightarrow C0 \rightarrow A6 \rightarrow E7 \rightarrow S15, S0 \rightarrow E0 \rightarrow A0 \rightarrow C1 \rightarrow A6 \rightarrow E7 \rightarrow S15, S0 \rightarrow E0 \rightarrow A1 \rightarrow C2 \rightarrow A7 \rightarrow E7 \rightarrow S15,$ and $S0 \rightarrow E0 \rightarrow A1 \rightarrow C3 \rightarrow A7 \rightarrow E7 \rightarrow S15$. Although core switches have only one downlink to forward each packet to the destination pod, packet-based load balancing at the edge and aggregation layers enable different packets to use almost completely disjoint paths to the destination node.

In our evaluations, we tested two different scenarios: static and randomized source-destination pairs for all flows. The static scenario consists of fixed source(s) and a destination server, and the randomized scenario consists of a randomized source and multiple destination servers. Specifically, in the static scenario, sources generate from 0 to 5 flows, each with a size of 10 Mbyte, towards S15. Source and the destination nodes for long flows are fixed. This means that the destination node is always S15, and each source node is selected among S0, S1, S2, S3, and S4, where each server contributes with one long flow. For instance, if five long flows are generated, the traffic of these five flows would present a pattern resembling a TCP incast scenario [21]. In addition to long flows, we generate a 200-Kbyte short flow from S0 to S15 to show how the FCT and the number of out-of-order packets for short flow are affected. We also analyze the average throughput for long flows change under the presence of long flows. We stress out the downlink queue at switch E7 by increasing the number of long flows and observe the change in performance. In the scenario with randomized source-destination pairs, we generated 20 short flows and a different number of long flows between randomly selected source-destination pairs. Except for the number of short flows and the randomized selection of source and destination servers, all other settings in the randomized source-destination pairs are similar to the static source-destination scenario. The experiment with randomized source-destination pairs is used to show how the packet-based load balancing mechanism affects the performance of flows under random background traffic. In the simulation, the transmission of all flows in both scenarios is ended when all flows are completely transmitted. TCP is used as the transport protocol for all flows. The transmission speed of all generated flows is equal to the line capacity, 1 Gbps. Note that TCP is the only control mechanism on the sending rate and sender's TCP adjusts the transmission rate by reducing the size of the congestion window, if TCP detects any congestion on the network. Table 1 summarizes all the parameters used in the simulations. Each test is run 10 times. Figure 2 shows the average throughput of long flows for a different number of them in the static scenario. As Figure 2 shows, packet-based load balancing provides a slightly larger

Table 1 Parameters used in simulations

Parameter	Value
NS-3 version	ns-3.23
Line rate	1 Gbps
Sending rate for each interface	1 Gbps
Delay for each link	25 µs
Long flow size	10 Mbyte
Short flow size	200 Kbyte
Packet size	1500 byte
Queue size at each switch	1.5 Mbyte

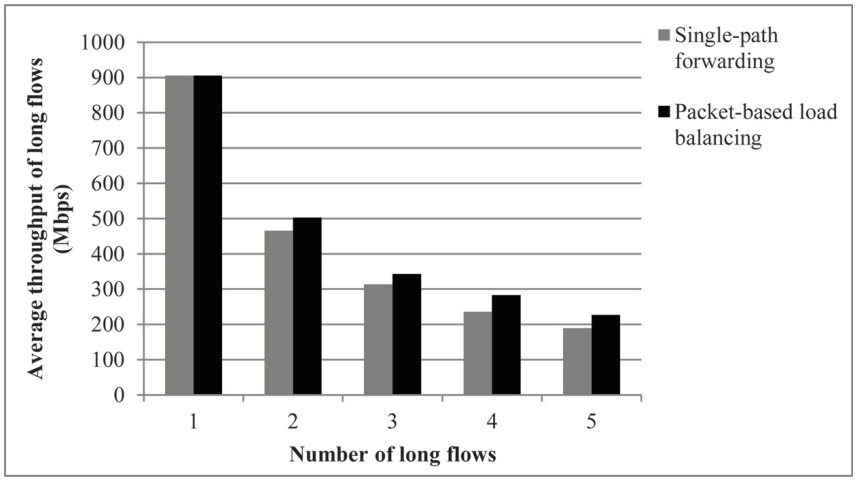

Figure 2 Average throughput of the long flows for a different number of long flows in the static source-destination pair case.

average throughput for long flows than that achieved by the single-path forwarding approach of TCP, which we refer to as single-path forwarding in the remaining of the paper. The achievable throughput of each flow is bounded by its fair share at the bottleneck link, E7-S15.

Figure 3 shows the average FCT of the short flow for a different number of long flows in the case of static source-destination pairs. The results in this figure show that the average FCT of the short flow is smaller than that of single-path forwarding. The advantage becomes more noticeable as the FCT of the short flow decreases as the number of long flows increases.

This phenomenon occurs because the short flow is affected by the increasing queuing delay caused by the long flows sharing the path with it. This effect is mitigated when packet-based load balancing is employed. The load

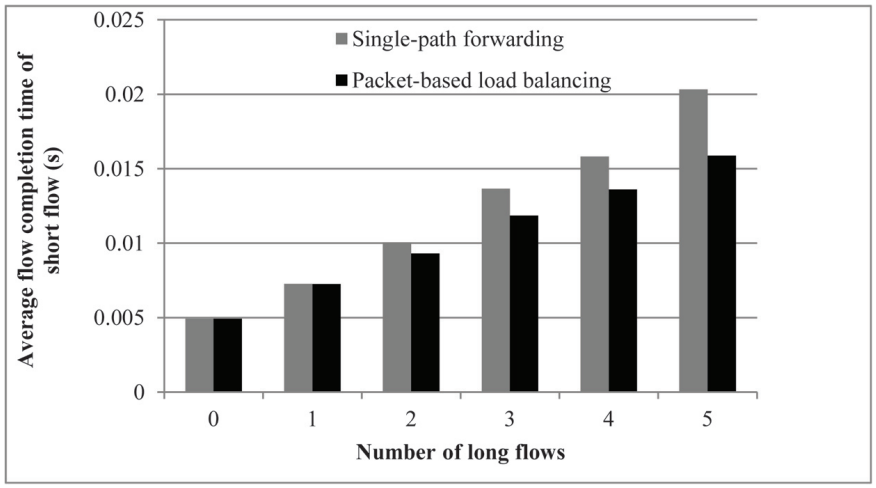

Figure 3 Average FCT of the short flow in the case of the static source-destination pairs.

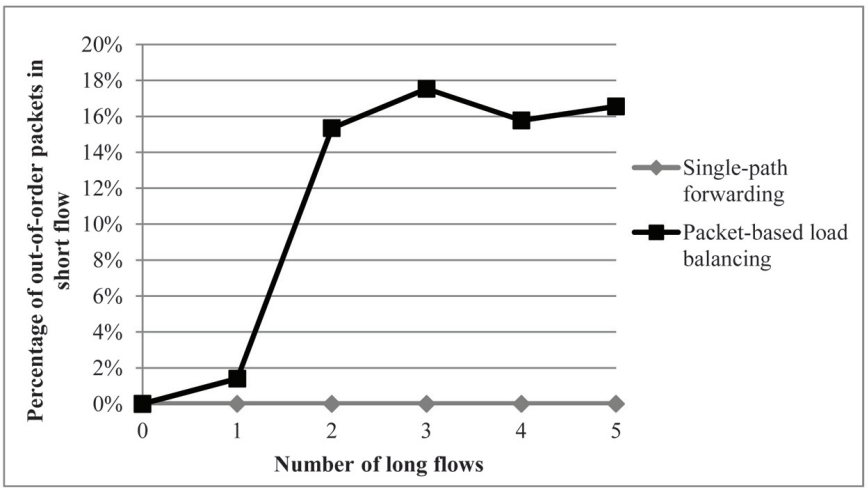

Figure 4 Percentage of out-of-order packets in short flow in the case of static source destination pairs.

distribution on multiple links decreases the queuing delays experienced by the packets of the short flow.

Figure 4 shows the percentage of out-of-order packets for the short flow over a different number of long flows when the source-destination pairs are static. This figure shows about the same percentage of out-of order packets

when there are two or more long flows sharing the path of the short flow. These results show that the number of out-of-order packets does not increases with the number of long flows, so that packet-based load balancing is advantageous in terms of FCT and throughput.

We also investigate the average throughput, average FCT, and the percentage of out-of-order packets when 20 short flows are generated between random source-destination pairs for a different number of long flows. Figure 5 shows how the average throughput for the long flows changes when the number of long flows increases. Packet-based load balancing in Figure 5 exhibits a larger throughput as compared to that of the single-path forwarding approach. The throughput gap between the two different approaches compared in this figure occurs because the random selection of the destination servers is un likely to create a bottleneck link. Therefore, the benefits of packet-based load balancing is more apparent than that in a TCP incast-like scenario.

Figure 6 shows the average FCT of the short flows under randomized background traffic for a different number of long flows. This figure shows a smaller FCT for packet-based load balancing, indicating that using the multipath feature of a DCN may benefit the FCT of short flows.

Figure 7 shows the percentage of out-of-order packets of short flows under randomized background traffic. This figure shows that although about 20% of packets are delivered in out-of-order, packet-based load balancing may still benefit the FCT and throughput of short and long flows, respectively.

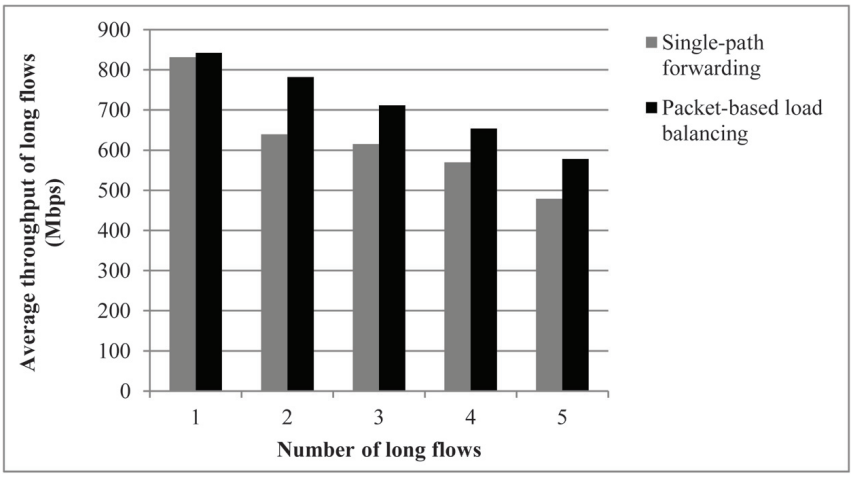

Figure 5 Average throughput of the long flows in the case of randomized source and destination nodes.

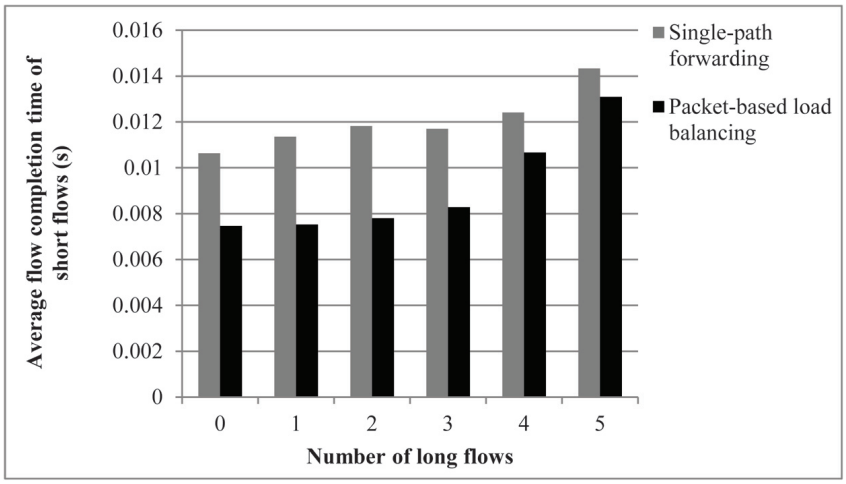

Figure 6 Average flow completion time of the short flows in the case of randomized source-destination pairs.

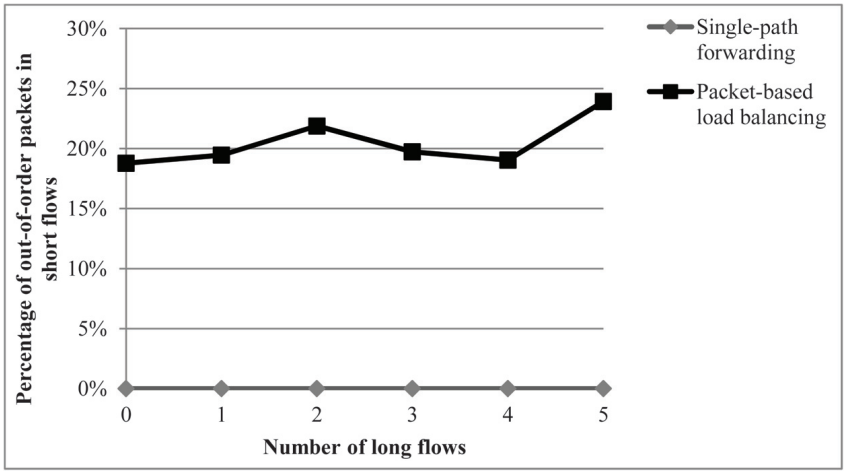

Figure 7 Percentage of out-of-order packets in short flows in the case of randomized source-destination pairs.

Figure 8 shows the average throughput of long flows when the number of short flows increases from 10 flows to 20. As this figure shows, the average throughput of long flows slightly decreases when the number of short flows increases from 10 to 20 because of the slight contribution of extra 10 short flows to the traffic.

Similarly, Figure 9 presents the average FCT of short flows when the number of short flows increases from 10 flows to 20. Increase in the average FCT of short flows is expected due to the presence of new short flows in the system.

Figures 8 and 9 show how the packet-based load balancing scheme scales up when the number of short flows increases in the network.

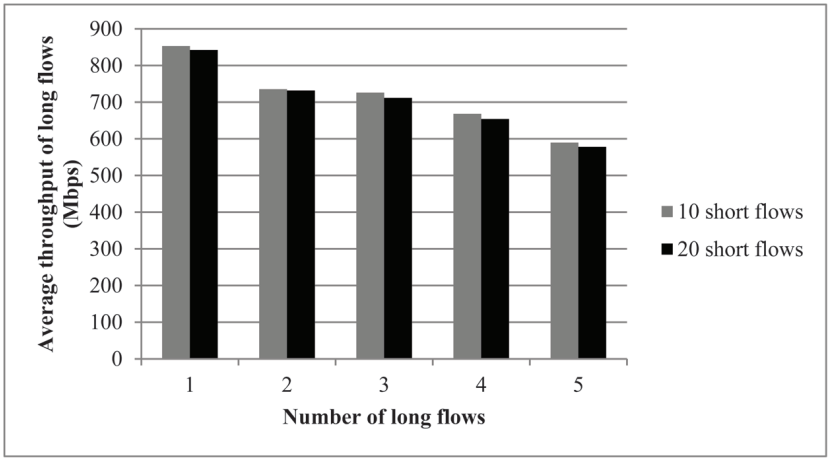

Figure 8 Average throughput comparison of long flows when 10 and 20 short flows are generated in the case of randomized source and destination nodes for packet-based load balancing.

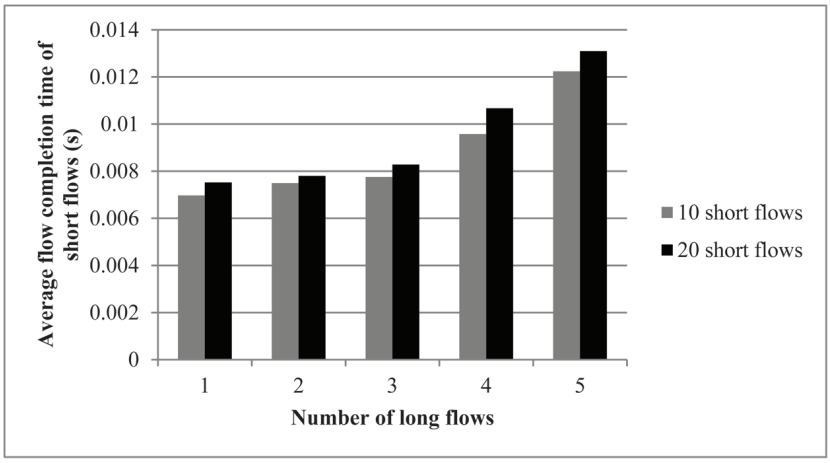

Figure 9 Average flow completion time comparison for 10 and 20 short flows in the case of randomized source-destination pairs for packet-based load balancing.

3 Related Work

Equal-Cost Multiple-Path (ECMP) forwarding is a flow-based load balancing mechanism which is used to balance the load among equal-cost shortest paths to the destination. ECMP hashes the five-tuple (i.e., source and destination IP addresses and the port numbers, and the protocol number) of a packet to decide which output port at the switch should be selected to forward the packet. If the received packet is the first packet of a flow, a hash function is executed to determine the output port and the remaining packets of that flow are forwarded to the same output port. Because ECMP is a flow-based load balancing mechanism and it does not differentiate short from long flows, the same output port is selected to forward all these long flows. This means that large queuing delays or congestion may be experienced by the delay-sensitive short flows if they are coincidentally forwarded to the same output port to that used by the long flows [19].

To improve the performance of a DCN, multipath forwarding schemes, such as DeTail [22], Hedera [3], Congestion-Aware Load Balancing (CONGA) [4], Multipath TCP (MPTCP) [17] and, Random Packet, and Spraying (RPS) [9] have been recently proposed.

DeTail is a cross-layer scheme aimed at reducing long tail FCT of short flows in DCNs. DeTail proposes to use packet-based adaptive load balancing at network layer and a reorder-resistant transport to prevent TCP from reacting out-of-order packets as an indication of congestion. However, DeTail requires complex modifications on more than one layer of the network stack, hence it diverges from keeping the implementation complexity low.

Hedera is a central flow scheduling scheme that aims to relocate long flows if they occupy at least 10% of the link capacity. Because Hedera uses a central scheduler, it is necessary to frequently run this central algorithm to relocate the long flows, which may make Hedera slow to react to dynamically changing traffic in a DCN as compared to distributed multipath forwarding schemes.

CONGA is a distributed, congestion-aware, in-network load-balancing scheme. CONGA is specifically designed for a two-layer DCN architecture, which is called leaf-spine, to balance the load among equal-cost shortest paths by splitting the flows into trains of packets, called flowlets. CONGA collects the congestion information using piggybacked feedback messages from a destination leaf (i.e., edge) switch, stores them, and selects the best output port, in terms of congestion extent at the source leaf switch, to forward the flowlets. Despite CONGA's distributed nature, it requires modifications at

the switches and the necessary space requirements to store the congestion information at the switches makes it costly to implement.

Presto is a load balancing mechanism that breaks each flow into discrete units of packets, called flowcells, to near-optimally balance the intra-DCN in a 2-tier (i.e., leaf-spine) network topology [14]. A centralized controller is used to partition the network into a set of disjoint spanning trees consisting of leaf and spine vSwitches as the nodes. Once the spanning trees are created, load balancing information are disseminated to the corresponding edge vSwitches. Load balancing is performed at each leaf vSwitch by sending the flowcells of a flow through the constructed spanning trees. Because flowcells are sent through different spanning trees to the destination, packet reordering may arise. Therefore, Presto uses a reordering buffer that works in a layer, called Generic Receive Offload, located in the hypervisor operating system to mitigate the impairing effect of packet reordering on TCP. Note that Presto requires a centralized controller in addition to some modifications at the switches, which increase the complexity of the load balancing scheme.

MPTCP is a multipath forwarding scheme which splits a TCP flow into several sub-flows at the sender and transmits these sub-flows on different paths using ECMP. MPTCP shifts the complexity from switches to end hosts where each end host requires a more complex transport protocol than regular TCP.

Digit-Reversal Bouncing (DRB) is a packet-based round-robin based routing technique that balances the load in the network to fully utilize network band- width for both Fat-tree [2] and VL2 [13] data center networks [6]. Routing decision in DRB is performed by the sender server. For each outgoing packet between a source-destination pair, the source sends the packet to a highest level switch, called bouncing switch, by digit-reversing its ID. Then, the receiving switch bounces the packet to its destination because there is only one down-path to reach the destination. Despite the simplicity of the proposed technique, DRB is only proposed for Fat-tree and VL2 DCNs and it is not generalized for every type of DCN topologies.

RPS is a packet-based forwarding technique that forwards packets of flows through different equal-cost shortest paths to their destinations, where a path is randomly selected with uniform probability. We take RPS as our model since it is simple and it does not require any modification at switches or servers.

4 Conclusions

In this paper, we analyzed the random packet spraying load balancing scheme and evaluated the performance of this scheme on short and long flows. We

considered the FCT of short flows and the throughput of long flows as performance metrics. We simulated the two schemes on a data center network using a fat-tree architecture comprising four pods and 16 servers. We also presented the percentage of out-of-order packets at the receiver node and showed that this percentage remains almost constant as the number of long flows increases. Our results showed that implemented random packet spraying scheme achieves smaller average FCT for short flows and higher average throughput for long flows over the single path used by TCP. We observed that out-of-order packet delivery does not have a significant impact on the average FCT of short flows and the average throughput of long flows under the tested conditions.

References

[1] The ns-3 network simulator. http://www.nsnam.org/.

[2] Al-Fares, M., Loukissas, A., and Vahdat, A. (2008). A scalable, commodity data center network architecture. *ACM SIGCOMM Comput. Commun. Rev.*, 38, 63–74.

[3] Al-Fares, M., Radhakrishnan, S., Raghavan, B., Huang, N., and Vahdat, A. (2010). "Hedera: dynamic flow scheduling for data center networks." In *NSDI*.

[4] Alizadeh, M., Edsall, T., Dharmapurikar, S., Vaidyanathan, R., Chu, K., Fingerhut, A., et al. (2014). "CONGA: distributed congestion-aware load balancing for data centers," in *SIGCOMM'14 Proceedings of the 2014 ACM conference on SIGCOMM* (New York, NY: ACM).

[5] Alizadeh, M., Greenberg, A., Maltz, D. A., Padhye, J., Patel, P., Prabhakar, B., et al. (2011). DCTCP. *ACM SIGCOMM comput. Commun. Rev.*, 41, 63–74.

[6] Cao, J., Xia, R., Yang, P., Guo, C., Lu, G., Yuan, L., et al. (2013). "Per-packet load-balanced, low-latency routing for close-based data center networks," in *Proceedings of the ninth ACM conference on Emerging networking experiments and technologies* (New york, NY: ACM), 49–60.

[7] Cerf, V. G., and Icahn, R. E. (2005). A protocol for packet network intercommunication. *ACM SIGCOMM Comput. Commun. Rev*., 35, 71–82.

[8] Ding C., and Rojas-Cessa, R. (2014). "Daq: deadline-aware queue scheme for scheduling service flows in data centers," in *Proceedings of the 2014 IEEE International Conference on Communications (ICC),* 2989–2994.

[9] Dixit, A., Prakash, P., Hu, Y. C., and Kompella, R. R. (2013). "On the impact of packet spraying in data center networks," in *Proceedings of the IEEEINFOCOM, 2013,* 2130–2138.

[10] Dukkipati, N., and McKeown, N. (2006). Why flow-completion time is the right metric for congestion control. *ACM SIGCOMM Comput. Commun. Rev.* 36, 59–62.

[11] Farrington, N., and Andreyev, A. (2013). "Facebook's data center network architecture," in *2013 IEEE Proceedings of the Optical Interconnects Conference*, 49–50.

[12] Ghemawat, S., Gobioff, H., and Leung S.-T. (2003). "The google file system," in *ACM SIGOPS operating systems review*, Vol. 37 (New York, NY: ACM), 29–43.

[13] Greenberg, A., Hamilton, J. R., Jain, N., Kandula, S., Kim, C., Lahiri, P., Maltz, D. A., Patel, P., and Sengupta, S. (2009). "VL2: a scalable and flexible data center network," in *ACM SIGCOMM computer communication review* (New York, NY: ACM), Vol. 39, 51–62.

[14] He, K. Rozner, E., Agarwal, K., Felter, W., Carter, J., and Akella. A. (2015). "Presto: edge-based load balancing for fast datacenter networks," in *Proceedings of the 2015 ACM Conference on Special Interest Group on Data Communication* (New york, NY: ACM), 465–478.

[15] Kaymak, Y., and Rojas-Cessa, R. (2015). "Per-packet load balancing in data center networks," in *Proceedings of 2015 36th IEEE Sarnoff Symposium*, 140–144.

[16] Prakash P. (2013). *Impact of network protocols on data center applications*. PhD thesis.

[17] Raiciu, C., Barre, S., Pluntke, C., Greenhalgh, A., Wischik, D., and Handley, M. (2011) Improving data center performance and robustness with multipath TCP. In *ACM SIGCOMM Computer Communication Review*, Vol. 41 (New york, NY: ACM), 266–277.

[18] Rajahalme, J., Amante, S., Jiang, S., and Carpenter B. (2011). Ipv6 flow label specification.

[19] Rojas-Cessa, R., Kaymak, Y., and Dong, Z. (2015). Schemes for fast transmission of flows in data center networks. *IEEE Commun. Surv. Tutor*. 17, 1391–1422. doi: 10.1109/COMST.2015.2427199

[20] Shvachko, K., Kuang, H., Radia, S., and Chansler, R. (2010). "The Hadoop distributed file system," in *IEEE 26th Symposium on Mass Storage Systems and Technologies (MSST), 2010*, 1–10.

[21] Wu, H., Feng, Z., Guo, C., and Zhang, Y. (2013). ICTCP: incast congestion control for TCP in data center networks. *IEEE/ACM Trans. Netw.*, 21, 345–358.

[22] Zats, D., Das, T., Mohan, P., Borthakur, D., and Katz, R. (2012). Detail: reducing the flow completion time tail in datacenter networks. *ACM SIGCOMM Comput Commun. Rev.*, 42, 139–150.

Biographies

Y. Kaymak received his B.S. degree in Mathematics from Celal Bayar University, Turkey, in 2003 and his M.S. degree in Computer Science from Ege University, Turkey, in 2011. He is currently working toward the Ph.D. degree in Computer Engineering at New Jersey Institute of Technology (NJIT), Newark, NJ, USA. He is a teaching assistant and a member of Networking Research Laboratory in the Department of Electrical and Computer Engineering, NJIT. His research interests include data center networking, peer-to-peer video streaming and distributed systems.

R. Rojas-Cessa received the Ph.D. degree in Electrical Engineering from Polytechnic Institute of New York University, Brooklyn, NY. Currently, he is an Associate Professor in the Department of Electrical and Computer Engineering, New Jersey Institute of Technology. He has been involved in design and implementation of application-specific integrated circuits (ASIC) for biomedical applications and high-speed computer communications, and

in the development of high-performance and scalable packet switches and reliable switches. He was part of a team designing a 40 Tb/s core router in Coree, Inc, in Tinton Falls, NJ. His research interests include data center networks, high-speed switching and routing, fault tolerance, quality-of-service networks, network measurements, and distributed systems. He was an Invited Fellow of the Japanese Society for the Advancement of Science in 2009. He visited the University of Electro-Communications, Japan. He was a Visiting Professor in Thammasat University, Thailand. He is a co-author of the book Advanced Internet Protocols, Services, and Applications, Wiley and Sons, 2012. His research has been funded by U.S. National Science Foundation and private companies. He has served in technical committees for numerous IEEE conferences, as a reviewer for several IEEE journals, and as a reviewer and panelist for U.S. National Science Foundation and U.S. Department of Energy. He is the recipient of the Excellence in Teaching Award 2013 from the Newark College of Engineering. He is a recipient of New Jersey Inventors Hall of Fame – Innovators Award in 2013. He is a Senior Member of IEEE.