PDLB: Path Diversity-aware Load Balancing with Adaptive Granularity in Data Center Network

weimin gao (✉ gwmhy@163.com)

Central South University <https://orcid.org/0000-0003-0912-9014>

huang jiawei

Central South University

Li zhaoyi

Central South University

zou shaojun

Central South University

wang jianxin

Central South University

Research Article

Keywords: Path diversity-aware, Adaptive granularity, Load balancing, Data center networks

Posted Date: January 12th, 2022

DOI: <https://doi.org/10.21203/rs.3.rs-819781/v1>

License: © ⓘ This work is licensed under a Creative Commons Attribution 4.0 International License.

[Read Full License](#)

PDLB: Path Diversity-aware Load Balancing with Adaptive Granularity in Data Center Network

Weimin Gao[§], Jiawei Huang[§], Zhaoyi Li[§] Shaojun Zou[§], Jianxin Wang[§]

[§]School of Computer Science and Engineering, Central South University, ChangSha, China 410083

[¶]Department of Computer Science and Engineering, Hunan Institute of Technology, HengYang, China 421002

E-mail: gwm@hnit.edu.cn, {jiawei Huang, lizhaoyi, zoushj, jxwang}@csu.edu.cn

Abstract—Modern data center topologies often take the form of a multi-rooted tree with rich parallel paths to provide high bandwidth. However, various path diversities caused by traffic dynamics, link failures and heterogeneous switching equipments widely exist in production data center network. Therefore, the multi-path load balancer in data center should be robust to these diversities. Although prior fine-grained schemes such as RPS and Presto make full use of available paths, they are prone to experience packet reordering problem under asymmetric topology. The coarse-grained solutions such as ECMP and LetFlow effectively avoid packet reordering, but easily lead to under-utilization of multiple paths. To cope with these inefficiencies, we propose a load balancing mechanism called PDLB, which adaptively adjusts flowcell granularity according to path diversity. PDLB increases flowcell granularity to alleviate packet reordering under large degrees of topology asymmetry, while reducing flowcell granularity to obtain high link utilization under small degrees of topology asymmetry. PDLB is only deployed on the sender without any modification on switch. We evaluate PDLB through large-scale NS2 simulations. The experimental results show that PDLB reduces the average flow completion time by up to ~11-53% over the state-of-the-art load balancing schemes.

Index Terms—Path diversity-aware, Adaptive granularity, Load balancing, Data center networks

I. INTRODUCTION

The data center network connects a large number of hosts through switches to provide large-scale computing and storage. In order to improve network transmission performance, new multi-rooted tree topologies such as Fat-tree [1], Clos [2], and VL2 [3] have appeared in the architecture design of data center network. These new network topologies provide multiple available transmission paths between the source host and the destination host. Parallel multi-path transmission can greatly improve the network throughput and reliability of data center.

The primary goal of load balancing is to evenly distribute traffic to each parallel path, improve network link utilization, and avoid network congestion caused by burst traffic. Equal Cost Multi-Path (ECMP) [4] is the most typical flow-level load balancing scheme in production datacenter, which uses flow hashing to transfer flows to available paths. LetFlow [5] and CONGA [6] reroute flowlets to avoid packet reordering. Random Packet Spraying (RPS) [7], Presto [8] and DRILL [9] flexibly split flows at a finer granularity to make full use of all available paths.

However, as the data center runs over time, traffic dynamics, topology asymmetry, and switch failures will appear [8, 10]. Under such uncertainty, the multiple paths become diverse or asymmetric. When these load balancing schemes scatter data packets on congested paths, some flows will inevitably experience unpredicted congestion and packet disordering. Increased transmission delay and decreased network throughput will lead to longer application response time and poor user experience. For example, Presto uses the maximum TCP segment offload (TSO) size (64 KB) as the flowcell granularity, allowing fine-grained load balancing at the network bandwidth of 10+ Gbps. Unfortunately, since Presto uses fixed-size packet clusters to split the traffic without considering the path diversity, it has the problems with out-of-order and packet loss in asymmetric topology.

In this paper, in order to ensure the robustness and flexibility of the above methods under asymmetric topology, we propose a load balancing mechanism called PDLB, which adaptively adjusts flowcell granularity to obtain both low packet reordering and high link utilization. To mitigate the impact of uncertainties under high path asymmetry, PDLB increases the size of flowcell to alleviate packet reordering. On the contrary, when the path asymmetry is low, PDLB reduces the size of the flowcell to achieve high link utilization. Moreover, PDLB is deployed on sender with negligible overhead, and making no modifications on switch.

In summary, our major contributions are:

- We conduct an extensive simulation-based study to analyze the impact of path asymmetry on the load balancing performance. We demonstrate experimentally and theoretically why controlling granularity of flowcell is effective in avoiding large tailed flow completion time (FCT) and packet reordering under large path asymmetry.
- We propose a flowcell-level load balancing scheme PDLB to spread flowcell across the multiple paths, which are adaptively selected according to path diversity. PDLB rationally adjusts the size of flowcell to improve link utilization under small path asymmetry and reduce tailed latency under large path asymmetry.
- By using large-scale NS2 simulations, we demonstrate that PDLB performs remarkably better than the state-of-the-art load balancing designs under different realistic traffic workloads. Especially, PDLB greatly reduces the

mean FCT up to $\sim 11\text{-}53\%$ over RPS, ECMP, LetFlow, and Presto.

The remainder of this paper is structured as follows. In Section II, we describe our design motivation. We present the design overview and details of PDLB in Section III-IV. In Section V, we evaluate the performance of PDLB using NS2 simulation. In Section VI, we demonstrate existing approaches. Finally, we conclude the paper in section VII.

II. BACKGROUND AND MOTIVATION

In this section, we provide empirical research to show the path asymmetry is very common in the modern data center. Then, we analyze the impact of path asymmetry on load balancing performance and demonstrate that controlling the granularity of flowcell is effective in reducing latency under large path asymmetry.

A. Path Asymmetry in Production Data Center

Data center network traffic is known for its burstiness [12,13]. When the traffic bursts instantaneously, it is easy to cause congestion, resulting in path asymmetry. In addition, due to link failures and heterogeneity of network equipment, path asymmetry generally exists in data centers [5,6,20,21,24,27]. In the following, we use an example to show the wide existence of path asymmetry.

As shown in Fig. 1(a), a dynamic burst flow is transmitted by ECMP from leaf switch L_1 to leaf switch L_2 . Since the link between spine switch S_0 and leaf switch L_2 is blocked by the burst flow, the two paths from L_0 to L_2 become asymmetric. Fig. 1(b) shows the asymmetric topology caused by link failures. When the link between L_1 and S_1 breaks down, any traffic from L_1 to L_2 is forced to be transferred on path $L_1 \rightarrow S_1 \rightarrow L_2$. Consequently, traffic from leaf switch L_1 to L_2 will go through two paths with different latencies. Besides, switch failures such as random packet dropping and heterogeneous switches with different link speeds can also induce topology asymmetry [5].

Fig. 1: Topology Asymmetry.

B. The Granularity of Flowcell Affects The Performance of Load Balancing

In order to explore the impact of path asymmetry on load balancing performances, we use NS2 simulation to test the performance of Presto, which is the typical datacenter

load balancing design already implemented on the commodity switches [8,30]. The test topology is a leaf-spine topology [12] with 8 spine switches and 2 leaf switches in Fig. 2. The bandwidth of each path and buffer size of each switch are 1Gbps and 250 packets, respectively. Each sender sends a DCTCP flow to a single receiver via leaf switches with RPS scheme, which randomly spreads the arriving packets to all 8 paths. To produce the path asymmetry, we change the round trip propagation delay of each path. Randomly set the RTTs of the two congested paths to $300\mu\text{s}$ and $600\mu\text{s}$, and the RTTs of the other non-congested paths are set to $100\mu\text{s}$, respectively. At sender-host, we generate 10 flows with 10000 packets per-flow.

Fig. 2: Leaf-spine topology.

1) FCT under different granularity of flowcell:

Fig. 3 shows the average flow completion time (AFCT) and 99th percentile flow completion time with different granularity of flowcell. The RTTs of congested path is different. Fig. 3(a) shows that AFCT firstly declines and then arises with the increasing size of flowcells. In Fig. 3(b), the 99th percentile FCT shows the similar trend. This result shows that, smaller flowcells provide higher link utilization to reduce flow completion time under smaller path asymmetry. However, under large path asymmetry, larger flowcells easily increase the tailed delay and reduce link utilization, resulting in large AFCT and 99th percentile FCT.

Fig. 3: FCT under different granularity of flowcell.

2) Packet reordering and link utilization under different granularity of flowcell:

We further investigate the reasons of above results. Fig. 4(a) shows the ratio of 3-dupack events caused by out-of-order packets to all packets. As the flowcell is split into smaller, the more data packets are scattered on the bad path, and the out-of-order ratio of the data packet is higher. At the same

time, increasing the number of bad paths will cause the same trend. Fig. 4(b) shows the link utilization rate under symmetric and asymmetric paths. Under the symmetric path, the link utilization rate increases as the flowcell cutting granularity decreases; under the asymmetric path, the link utilization rate will show a convex curve as the flowcell cutting granularity increases.

Fig. 4: FCT under different granularity of flowcell.

C. Summary

Based on the above analysis, we draw the following conclusions that (i) smaller flowcells provide higher link utilization, but out-of-order is more likely to occur in path asymmetry, (ii) larger flowcells easily reduce out-of-order events but decrease the link utilization. These conclusions motivate us to design a novel load balancing scheme that adjusts the granularity of flowcell to achieve good tradeoff between the packet reordering and link utilization. In the rest of this paper, we present our PDLB scheme as well as a prototype implementation in NS2 simulation.

III. ADAPTIVELY ADJUSTING FLOWCELL GRANULARITY

In this section, we firstly describe the architecture of PDLB. Our goal is to design a load balancing mechanism adjusting flowcell granularity based on the latency difference among multiple paths to achieve the tradeoff between packet reordering and link utilization. Then, we present the details how to estimate path asymmetry at end-host. Moreover, we theoretically analyze how to obtain the optimal granularity of flowcell according to network congestion state. Finally we evaluate the accuracy of model by comparing the results of theoretical analyze and simulation test.

A. Design Overview

In Fig. 5, we plot the architecture of PDLB, which includes path asymmetry estimation module and flowcell size adjustment module. Firstly, PDLB implements the RTT measurement of the path at the sending end, which periodically sends the detection packet with the time stamp option to obtain the RTT sample information of each link in the network. then, the sampling information can be processed to obtain the real-time congestion states, and the paths are divided into congested paths and uncongested ones. With the path states, PDLB makes flowcell-level forwarding decisions for each arrival packet. Specifically, PDLB calculates the optimal

size of flowcells *gran* based on the path diversity. Then PDLB spreads flowcells with polling on all paths to balance the out-of-order ratio and link utilization.

PDLB design needs to solve several key challenges. Firstly, PDLB needs to periodically collect path delays with limited overhead to distinguish congested and non-congested paths. Secondly, the size of flowcell adjustment should quickly respond to the dynamic changes of the network. Finally, PDLB should be compatible with existing transport layer protocols and be easy to deploy in large-scale network.

Fig. 5: The architecture of PDLB.

B. Path asymmetry estimation

Load balancing schemes that require path congestion information, naturally, are much more complex [14]. Modern data center networks are usually organized in multi-rooted tree topologies, in which the load balancing schemes split traffic across multiple paths between the source leaf switch and the destination one. To obtain the path congestion state, PDLB should measure the end-to-end delay of the transmission path.

The path asymmetry can be estimated in advance at the sender based on the measurement of RTT [18]. Since PDLB mainly adjusts the granularity of flowcell adaptively according to the difference of the path, in order to realize the purpose of transferring the traffic from the congested path to the uncongested path. But the sender is unable to directly obtain the exact RTT for each path. Here, we utilize the TCP congestion control mechanism [11]. Specifically, when the sender receives the ACK packets, based on the corresponding RTT of each ACK, the bad path probability is calculated as the ratio of the number of ACK packets with large RTT to the total number of received ACK packets. For example,

assuming that 10 flowcells are sent at the sender, the number of flowcells should generally be consistent with the number of paths. Since the switch uses polling scattering to route the flowcells, the sender calculates the probability of bad path according to the path delay. For example, if the transmission times of two flowcells are obviously longer than that of other 8 flowcells, it is possible that these two flowcells are taking the bad path. Then, the probability of a bad path $P_b=0.2$. To limit the computing and memory overhead, the sender measures the path delay every 100 μ s.

PDLB brings about limited overhead, since it only measures the one-way-delay by using source and destination hosts. Moreover, to reduce overhead and enhance scalability, PDLB periodically uses few data and ACK packets to carry the delay information in the option field of TCP header. Thus, the path delay is collected with very small traffic and deployment overhead.

C. Flowcell Size Adjustment

The flowcell size affects both the TCP reordering probability and network utilization under different asymmetric degrees. We give the analysis on how to get the optimal value of flowcell size as following.

Let F_{size} and $gran$ denote the size of a TCP flow and flowcell granularity, respectively. Because the smallest granularity is a packet, and the largest granularity is 64KB (44 packets) [8,21], the value of $gran$ ranges from 1 to 44. Then the flow is cut into $\frac{F_{size}}{gran}$ flowcells according to $gran$ granularity.

When the flowcell are transferred on multiple paths, a flowcell is out-of-order only when at least one flowcell sent later arrives at the destination earlier. We assume that the n flowcells may select m parallel paths, which consist of N_b congested paths and N_g uncongested paths with the propagation delay D_b and D_g , respectively. Assuming that the ratio of the number of bad paths to the number of good paths is R , then R is equal to $\frac{N_b}{N_g}$. Let X denote the ratio of bad paths to good paths, then X is equal to $\frac{D_b}{D_g}$.

Due to the uncertainty of network traffic, the number of congested paths and non-congested paths will change with the diversity of paths. In our design, in order to avoid synchronization effect, each flowcell is randomly assigned to one of the available paths. According to the number of congested paths N_b that has been assumed before, if P_g and P_b are the probabilities that the flowcell selects the uncongested and congested paths, respectively, then we can calculate the probability of choosing a congested path P_b as $\frac{N_b}{m}$.

Therefore, we get the probability P_g that an uncongested path is selected as

$$P_g = 1 - \frac{N_b}{m}. \quad (1)$$

In this case, the out-of-order event occurs when one packet train is transmitted on congested path and at least one flowcell

sent later is transmitted on uncongested path. Therefore, the reordering probability P of n flowcells is calculated as

$$\begin{aligned} P &= 1 - (P_b^n + P_b^{n-1} \times P_g + P_b^{n-2} \times P_g^2 + \dots \\ &\quad + P_b \times P_g^{n-1} + P_g^n) \\ &= 1 - \sum_{i=0}^n P_b^{n-i} \times P_g^i. \end{aligned} \quad (2)$$

Substitute P_b and P_g into Eq.(2), we get the reordering probability P as

$$P = 1 - \sum_{i=0}^n (1 - \frac{N_b}{m})^i \times (\frac{N_b}{m})^{n-i}. \quad (3)$$

Assume that the largest window in each round of transmission on a path is $MaxW$, and the initial value of the network congestion window is W_0 . When detecting the out-of-order packet, the TCP sender reduces its congestion window by half. Thus, in each round of data transmission, if n_i represents the number of flowcells in i -th round, the i -th round window W_i and the out-of-order ratio of i -th round P_i are

$$\begin{cases} n_1 = f(W_0) = \min(m, \frac{W_0}{gran}) \\ P_1 = g(n_1) = 1 - \sum_{i=0}^{n_1} P_b^{n_1-i} \times P_g^i \\ W_1 = \delta(P_1) \\ \quad = \min(n_1 \times MaxW, (W_0 + 1) \times (1 - P_1) + \frac{W_0}{2} \times P_1) \\ \dots \\ n_i = f(W_i) = \min(m, \frac{W_{i-1}}{gran}) \\ P_i = g(n_i) = 1 - \sum_{i=0}^{n_i} P_b^{n_i-i} \times P_g^i \\ W_i = \delta(P_i) \\ \quad = \min(n_i \times MaxW, (W_{i-1} + 1) \times (1 - P_i) + \frac{W_{i-1}}{2} \times P_i) \end{cases}$$

Therefore, the congestion window for each round W_i is given by:

$$W_i = \begin{cases} m \times MaxW & i = 0; \\ \min(n_i \times MaxW, (W_{i-1} + 1) \times (1 - P_i) + \frac{W_{i-1}}{2} \times P_i) & i > 0. \end{cases} \quad (4)$$

Let r and W_s represent the number of rounds required to transmit n flowcells and sum of congested windows respectively, then W_s can be expressed as

$$W_s = \sum_{i=0}^r W_i. \quad (5)$$

When W_s is firstly greater than or equal to F_{size} , the subscript of W_i corresponds to the number of transmission rounds r , then we get the average congestion window $\bar{W}$ as

$$\bar{W} = \frac{W_s}{r}. \quad (6)$$

Though the small size of flowcell leads to large packet reordering probability, the small flowcell could utilize more paths, increasing the total utilized bandwidth. Given the link bandwidth C for each path, since each flowcell randomly picks its transmission path, the total bandwidth for n flowcell is $n \times C$.

Typically, the end-to-end latency mainly consists of the queueing and propagation delay. Given the average congestion window $\bar{W}$, we obtain the average end-to-end round-trip time $\overline{RTT}$ as

$$\overline{RTT} = (1 - (\frac{N_b}{m})\bar{W}) \times D_g + (\frac{N_b}{m})\bar{W} \times D_b + \frac{\bar{W}}{n \times C}. \quad (7)$$

Substitute D_b and $\bar{W}$ into Eq.(7), we can get the $\overline{RTT}$ as

$$\overline{RTT} = ((X - 1) \times (\frac{N_b}{m})\bar{W} + 1) \times D_g + \frac{W_s}{r \times n \times C}. \quad (8)$$

When the number of rounds of transmission is known, the end-to-end latency of each round of the data flow can also be calculated, the completion time of the flow (FCT) can be expressed as

$$\begin{aligned} FCT &= \frac{F_{size}}{\bar{W}} \times \overline{RTT} \\ &= \frac{F_{size}}{\bar{W}} \times ((X - 1) \times (\frac{N_b}{m})\bar{W} + 1) \times D_g + \frac{F_{size}}{n \times C} \\ &= \frac{F_{size}}{\bar{W}} \times ((X - 1) \times (\frac{N_b}{m})\bar{W} + 1) \times D_g + \frac{gran}{C}. \end{aligned} \quad (9)$$

wherein F_{size} , C , N_b , m , D_g , X are constant coefficients. There is a quadratic function relationship between the variable $\bar{W}$ and the independent variable $gran$. Finally, according to Eq.9 we can get the optimal granularity $gran$ to obtain the minimum value of average flow completion time FCT as

$$gran = \arg \min_{gran_i \in [1, 44]} \|FCT(gran_i)\|. \quad (10)$$

D. Model Verification

We evaluate the correctness of the numeric analysis by NS2 simulations. As shown in Fig. 6, we use DCTCP [19] as the underlying transport protocol, and the experimental topology shown is Fig. 2. Besides, the flow size, the delay of two bad paths (D_b), and D_g are 10000 packets, 300 μ s, 600 μ s and 100 μ s, respectively. Other parameters are the same as the experimental scenario in Sect. II-B.

When the granularity of flowcell increases from 1 to 44, we calculate the theoretical average flow completion time for ten flows with size of 100000 packets. The value of numeric analysis is consistent with the varying trend in NS2 simulation test.

Fig. 6: Model verification.

IV. ADAPTIVE GRANULARITY ALGORITHM

Based on the above theory and verification experiment analysis, we can get the optimal granularity of flowcell. However, in the production data center network, with thousands of servers, complex topology and frequent burst traffic, it is obviously unreasonable to use a fixed flowcell granularity for transmission. Therefore, we have to adopt certain adjustment strategies as shown in Algorithm 1 to quickly respond to network changes. Adaptive granularity algorithm consists of path asymmetry estimation module and flowcell size adjustment module.

Algorithm 1: Adaptive Granularity Algorithm

Input: m /*The number of paths in DCN*/
Output: $gran$ /*The granularity of flowcell will be selected*/

```

1 Initialization;
2  $p_i.mode \leftarrow \emptyset$ ;  $path[] \leftarrow allpaths$ ;
3  $m \leftarrow 8$ ;  $i \leftarrow 1$ ;
4  $fct_1 = f(gran_1)$ ;  $fct_2 = f(gran_2)$ ;
5  $\Delta fct = fct_2 - fct_1$ ;
6 /* path asymmetry estimation module */;
7 if The timer is timeout then
8    $D_b = +\infty$ ;  $D_g = 0$ ;  $Sum = 0$ ;
9   for each path  $p_i \in path[]$  do
10      $D_b = MIN(D_b, p_i.RTT)$ ;
11      $D_g = MAX(D_g, p_i.RTT)$ ;
12      $Sum += p_i.RTT$ ;
13    $D_{avg} = \frac{Sum}{m}$ ;
14    $X = \frac{D_b}{D_g}$ ;
15   for each path  $p_i \in path[]$  do
16     if  $p_i.RTT > 2 \times D_{avg}$  then
17        $p_i.mode = congested$ ;
18     else
19        $p_i.mode = uncongested$ ;
20 /* Flowcell size adjustment module */;
21 if the path state changes then
22   while  $\Delta fct \leq 0$  and  $i \geq 3$  and  $i \leq 44$  do
23      $fct = f(gran_i)$ ;
24      $fct_1 = fct_2$ ;
25      $fct_2 = fct$ ;
26      $\Delta fct = fct_2 - fct_1$ ;
27      $i \leftarrow i + 1$ ;
28     if  $\Delta fct \geq 0$  then
29        $gran = i$ ;
30 return  $gran$ ;

```

path asymmetry estimation module: The path congestion estimation of PDLB at the sender side periodically (e.g. 100 μ s) sends probe flowcells to measure the congestion state. When the timer expires after the timeout T , the PDLB first estimates the delay of all paths between the sender and the destination

host. Then the PDLB updates the number of congested paths with a delay greater than $2X$ the average RTT [16,18]. According to the path delay information, PDLB calculates the difference of path asymmetry.

Flowcell size adjustment module: When the path state changes, PDLB measures the difference of paths and iteratively calculates the optimal flowcell granularity.

Since the granularity of the flowcell takes a value in the range of $[1,44]$, the completion time of the flow has a concave curve relationship with the granularity of the flowcell. We use the sequential search method to continuously iterate until we can find the optimal granularity of flowcell. The time complexity of the adaptive granularity algorithm for path diversity-aware is $O(\text{gran})$. We enforce the optimal granularity to all active flows in network. When the number of arriving packets from a given flow is less than the optimal granularity, the packets follow the path of the same flowcell by using the same five-tuple of packet header.

V. SIMULATION EVALUATION

In this section, we conduct the NS2 simulation tests to evaluate the performance of PDLB. We firstly test the basic performance of PDLB by adjusting the number of congested paths and the asymmetry of the path respectively, and then compare PDLB with the state-of-the-art schemes under data-center workloads in large-scale test. We use flow completion time (FCT), link utilization and out-of-order ratio as the primary performance metrics.

A. Basic Performance

In this section, we test the basic performance of PDLB. We compare the packet reordering, link utilization and flow completion time of RPS, ECMP, Letflow and Presto.

1) Adjusting the number of congested paths

The experimental topology is shown in Fig. 2. There are 8 available paths from the one leaf switch to another leaf switch. The RTT of each path are $100\mu\text{s}$. The downlink and uplink bandwidth of leaf switch are 1 Gbps. In this test, 10 end-hosts are connected to the source leaf switch, and each end-host under one leaf switch sends a flow with 5000 packets to a destination host under another leaf switch. The hotspot path with latency of $300\mu\text{s}$, then we increase the number of hot paths from 1 to 4, we compare the performances of PDLB and others methods.

Fig. 7 shows the basic performance with different load balancing mechanisms. Fig. 7(a) shows that out-of-order ratio of RPS, ECMP, Letflow, Presto and PDLB under path asymmetry. Since RPS forwards packets of each flow to all paths in a packet-level spraying manner, with the increasing of bad paths, the packets scattered on the bad paths will experience a long transmission delay, so there will be a large number of out-of-order packets. ECMP and LetFlow are coarse-grained scheduling schemes, which can completely avoid packet reordering, while fail to fully utilize all parallel paths.

Fig. 7: The basic performance with different load balancing mechanisms.

Fig. 7(b) and Fig. 7(c) show the similar trend. Smaller flowcells provide higher link utilization to reduce flow completion time under small path asymmetry. However, smaller flowcells easily increase the tailed delay under large path asymmetry. As RPS experiences more disorder, its average flow completion time is the worst. ECMP's performance is also relatively poor, because ECMP has a higher probability of hashing flows to the same hot path. LetFlow can effectively avoid packet reordering, while fail to fully utilize all parallel paths. PDLB still outperforms Presto. The reason is that PDLB uses path asymmetry-aware switch granularity adjustment. Therefore, PDLB always maintains the highest total throughput and the smallest FCT compared to the other schemes.

2) Adjusting the asymmetry degree of the path

The experimental topology used in the same as above. We randomly choose 4 parallel paths as the bad paths, while the remaining 4 paths are the good paths. Each link of the good paths is with 1Gbps bandwidth, and thus the round trip propagation delay of good paths is $200\mu\text{s}$. For the bad paths, we gradually increase the propagation link delay to enlarge the degree of topology asymmetry. Therefore, the ratio of bad paths' RTT to good paths' RTT varies from 1.5 to 3.5. In our tests, The senders generates 100 DCTCP flows based on the data mining workload according to the Poisson process. The threshold of flowlet using in LetFlow is set as $500\mu\text{s}$ [5,9]. We evaluate the performances of various schemes in terms of the average flow completion times (AFCTs) of short flows ($<100\text{KB}$), the total throughput of long flows ($\geq 100\text{KB}$) [19, 20], the ratio of retransmission packets caused by packet reordering.

Fig. 8(a) compares the retransmission ratio of short flows under different schemes. The rising of asymmetric degree causes RPS to experience increasingly serious packet reorder-

Fig. 8: The performances of short flows.

ing, generating much higher retransmission ratio compared to the other schemes. On the contrary, PDLB can effectively control packet reordering, and obtain high link utilizations, thus achieving the shortest average and 99th percentile flow completion times across all cases, as shown in Fig. 8(b) and Fig. 8(c).

The performance of long flows is shown in Fig. 9. RPS can distribute load in the most balanced way, but possess high retransmission ratio. Thus, under RPS, the total throughput of long flows decreases with the increasing of asymmetric degree. ECMP and LetFlow can completely avoid packet reordering, while failing to fully utilize all parallel paths. Fortunately, PDLB can adjust the granularity adaptively according to path conditions, and effectively avoids packet reordering. Therefore, PDLB is not affected by the rising of asymmetric degree, and always maintains the highest total throughput for long flows compared to the other schemes.

Fig. 9: The performances of long flows.

3) Adaptive Granularity

In this section, we compare the fixed granularity scheme and the adaptive granularity scheme in three aspects: the out-of-order ratio of the data packet, the link utilization rate and whether the granularity can be adjusted adaptively. We use the

Leaf-Spine topology with 4 paths. The round trip propagation delay is $200\mu s$ and the link bandwidth is 1Gbps. The buffer size of each switch is 100 packets. We generate 2 TCP flows with size of 30MB. There is a 500Mbps UDP background flow on each path to induce congestion. Meanwhile, to produce topology asymmetry, we increase the sending rate of UDP flow to 800Mbps on one randomly selected path from $100\mu s$ to $200\mu s$. We set the sample interval as $500\mu s$.

Fig. 10(a) shows the flowcell granularity of PDLB. When the multiple paths are symmetric before $100\mu s$ or after $200\mu s$, PDLB sends flowcells with smaller granularity to improve link utilization. Although the asymmetric degree of multiple paths becomes large, PDLB increases the granularity of flowcell to alleviate packet reordering.

Fig. 10(b) shows the ratio of 3-dupack events caused by out-of-order packets in all packets. Since the asymmetry of the path occurs in $[100, 200]\mu s$, PDLB experiences a little more packet reordering due to its random scheduling pattern. Fortunately, it can estimate the path asymmetric, and adjust the granularity of flowcell adaptively. Therefore, the packet reordering ratio of PDLB is still acceptable.

Fig. 10(c) shows the overall network utilization. The network utilization of fixed granularity schemes is less than 50%, because 2 flows respectively almost fully utilize 2 paths at a time, while the other 2 paths are unused. PDLB achieves high network utilization, because it adaptively adjusts the granularity of flowcell based on path asymmetry to make a tradeoff between the packet reordering and link utilization.

B. Large-scale Simulation Test

In this part, we construct a large-scale leaf-spine network in which 256 hosts are connected via 8 leaf switches, 8 spine switches, and many 1Gbps links. The switch buffer size is set to 250KB. There are 8 equal cost paths with the propagation delay as $100\mu s$ between any pair of hosts. In order to generate path asymmetry, we randomly selects one of the paths and set its round-trip propagation delay to $300\mu s$.

Besides, to make a comprehensive comparison, we use two representative data center workloads such as Web Server and Web Search [20,25]. Specifically, Web Server has the most short flows (86%), but its number of long flows is the smallest (14%). Besides, the data sizes of its long flows are all smaller than 1MB. Web Search is just the opposite for having the most long flows (38%), and thus its number of short flows is the smallest (62%). Overall, across two workloads, the ratios of short flows are always higher than those of long flows, following the heavy-tail distribution of data center traffic [17,18, 19, 21]. Flows are generated between random pairs of hosts following a Poisson process with load varying from 0.1 to 0.8. We evaluate the performance of PDLB by comparing with the state-of-the-art load balancing schemes, such as RPS [7], LetFlow [5], ECMP [4], and Presto [8].

The performance test results of Web Search and Web Server applications are shown in Fig. 11 and Fig. 12, respectively. Fig. 11(a) and Fig. 12(a) show the average flow completion time of short flows, which increases with larger traffic load.

Fig. 10: Comparison of fixed granularity and adaptive granularity schemes.

We observe that RPS performs poorly for small flows because of it scatters the data flow to all paths at packet granularity. In the case of bad paths, the long flows will block the short flows, thus affecting the completion time of the short flows. For ECMP, because it forwards packets to paths in a flow hashing method, data flows are prone to collisions, so the average flow completion time of short flows is only better than RPS. Presto splits each flow into a fixed granularity (i.e. 64KB). Though mitigating the issues of low link utilization and packet disordering to a certain extent, the fixed granularity of Presto is not adaptable to all asymmetric degrees. Under LetFlow, the switch automatically senses path congestion by using the elasticity of flowlet size. LetFlow sets a time interval threshold (i.e. 500 μ s) for packet clusters, resulting in no out-of-order and smaller AFCT for short flows. Due to the adaptive switching granularity, PDLB reduces the average FCT of short flows by up to 28%.

Fig. 12: FCT for the Web Server workload in the asymmetric topology

Web Search and Web Server workload, respectively. Note that under Web Server workload, since the flow sizes of 80% flows are less than 100KB [3], the average FCT of short flows are lower than that of Web Search workload. Meanwhile, as the long flow size in web server is larger, the short flows' 99th FCT are higher than that of Web Search workload.

Fig. 11(c) and Fig. 12(c) show the average FCT of long flows under different traffic loads. Since the Web Search workload contains more long flows, Letflow can well control the out-of-order. The size of the flowlet will automatically change according to the degree of path congestion, and the flowlet will be randomly sent to all available paths on the switch. However, due to the longer time interval for LetFlow to adjust the granularity, it has obtained sub-optimal performance. Meanwhile, Presto sends flowcells at a fixed granularity, under asymmetric topology, thereby obtaining worse performance than LetFlow. Compared with the other schemes, PDLB alleviates the impacts of large queueing delay and out-of-order problem by adaptively adjusting the flowcell granularity of long flows according to path congestion states.

Fig. 11: FCT for the Web Search workload in the asymmetric topology

Fig. 11(b) and Fig. 12(b) show the 99th percentile flow completion time of short flows under different traffic loads. Compared with the other four scheme, PDLB significantly reduces the tail FCT of around 8%-40% and 20%-43% under

Therefore, PDLB reduces the average FCT of long flows by up to 8%-53% over other schemes.

Fig.11(d) and Fig.12(d) show the average FCT of overall flows in Web Search and Web Server workloads, respectively. PDLB has about 18%-53% and 11%-40% mean FCT improvements compared with other four schemes under Web Search and Web Server workload, respectively. Note that under Web Server workload, since the flowcell sizes of 86% flows are less than 100KB [2], the overall average FCT is lower than that of Web Search workload.

Overall, from the results of the two workloads shown in Fig.11 and Fig.12, RPS has the worst overall performance in the case of path asymmetry. This is mainly caused by plenty of retransmission packets since in the asymmetric network, both short and long flows are very prone to experience retransmissions. For PDLB, it can dynamically adjust the granularity of data flows according to the asymmetry of the path in the network, and adaptively control the optimal rate on the basis of balancing low disorder and high link utilization. Therefore, its performance is always better than other solutions under different load intensities.

VI. RELATED WORKS

In recent years, researchers have proposed various load balancing mechanisms to facilitate parallel data transmission across multiple paths, thereby further gaining performance enhancements.

LetFlow [5] uses the natural attributes between data packets to automatically sense path congestion. When a flow encounters congestion on a certain path, the interval between data packets will increase, naturally forming a packet cluster with a time interval. LetFlow distinguishes the packet clusters according to the time interval threshold, and randomly sends the packet clusters to other paths. Since the interval threshold is generally greater than the maximum path difference, LetFlow can avoid disorder and effectively combat asymmetry problems. However, due to the randomness of LetFlow scheduling, the optimal load balancing performance cannot be achieved.

Presto [8] is a typical flowcell-based mechanism which sprays the flowcells to all available paths in a round-robin style. Presto uses a software switch to split flows into 64KB data slices. Presto uses the central controller to collect topology information. The software switch on the host sends each packet cluster of equal size (64KB) to all available paths in round-robin style. The receiving end modifies the GRO mechanism to reduce out-of-order.

Flowtune [28] allocates an optimal transmission rate for each flowlet by using a centralized allocator. But at the endpoint updates congestion-control parameters, it is easy to trigger the herd effect.

TLB [29] proposes a load balancing method with adaptive path switching granularity in data center networks. The switch distinguishes between long and short flows according to the amount of sent data of the data flow, adjust the switching granularity of the long flow according to the load intensity of the short flow. When the load intensity of the short flow is

high, the long flow will be rerouted with a larger granularity, thereby leaving more light congestion paths for the short flow. On the contrary, when the load intensity of the short flow is low, the long flow is switched with a smaller granularity to improve the link utilization, and the short flow selects the path at the packet level to flexibly utilize the light-congestion path. TLB reduces the average completion time of the short flow and improves the throughput of the long flow. However, there is a problem of out-of-order in asymmetric topology.

Luopan [30] is a sample-based load balancing scheme. Luopan uses a fixed-size packet cluster as the scheduling unit, and the packet cluster size is set as the maximum TSO size of the server limit of 64KB. For the existence of multiple equivalents between source and destination switches paths, Luopan periodically samples part of the paths, and then forwards the packet cluster directly to the path with the smallest queue length to achieve load balancing.

VII. CONCLUSION

We proposed a novel asymmetry-aware load balancing scheme PDLB that reduces flow completion time and simultaneously improves link utilization. Specifically, PDLB performs path asymmetry estimation by periodically sending detection flowcells at the sender, and then adaptively adjusts the granularity according to the path diversity. Moreover, PDLB make full use of the powerful computing ability and storage resources of the end host, while avoiding the modification of the switch hardware.

ACKNOWLEDGMENT

This work is supported by the National Natural Science Foundation of China (No. 61872387), and Project of Foreign Cultural and Educational Expert (No. G20190018003), Scientific Research Project of Hunan Provincial Department of Education (No. 20A144).

REFERENCES

- [1] Huang Q, Jin X, Lee PPC, et al. "Sketchvisor: Robust network measurement for software packet processing", in Proc. ACM Special Interest Group on Data Communication, 2015, pp.113-126.
- [2] Bredel M, Bozakov Z, Barczyk A, et al. "Flow-based load balancing in multipathed layer-2 networks using OpenFlow and multipath-TCP", in Proc. Hot topics in software defined networking. 2014, pp.213-214.
- [3] Greenberg A, Hamilton JR, Jain N, Kandula S, Kim C, Lahiri P, Maltz DA, Patel P, Sengupta S. "VL2: A scalable and flexible data center network", In: Proc. of the ACM Special Interest Group on Data Communication (SIGCOMM). New York: ACM, 2009, pp.51-62.
- [4] C. Hopps. "Analysis of an Equal-Cost Multi-Path Algorithm", in RFC 2992.
- [5] Vanini E, Pan R, Alizadeh M, et al. "Let it flow: Resilient asymmetric load balancing with flowlet switching", in Proc. USENIX Symposium on Networked Systems Design and Implementation, 2017, pp.407-420.
- [6] Alizadeh M, Edsall T, Dharmapurikar S, et al. "CONGA: Distributed congestion-aware load balancing for datacenters", in Proc. ACM conference on SIGCOMM, 2014, pp.503-514.
- [7] A. Dixit, P. Prakash, Y. C. Hu, R. R. Kompella. "On the Impact of Packet Spraying in Data Center Networks", in Proc. IEEE INFOCOM, 2013, pp.2130-2138.
- [8] K. He, E. Rozner, K. Agarwal, W. Felter, J. Carter, A. Akella. "Presto: Edge-based Load Balancing for Fast Datacenter Networks", in Proc. ACM SIGCOMM, 2015, pp.465-478..

- [9] S. Ghorbani, Z. Yang, P. B. Godfrey, Y. Ganjali, A. Firoozshahian. "DRILL: Micro Load Balancing for Low-latency Data Center Networks", in Proc. ACM SIGCOMM, 2017, pp.225-238.
- [10] Zhang H, Zhang J, Bai W, et al. "Resilient datacenter load balancing in the wild", in Proc. ACM Special Interest Group on Data Communication. 2017, pp.253-266.
- [11] Mittal R, Lam V T, Dukkipati N, et al. "TIMELY: RTT-based Congestion Control for the Datacenter", ACM SIGCOMM Computer Communication Review, 2015, vol.45, no.4, pp.537-550.
- [12] Kabbani A, Sharif M. "Flier: Flow-level congestion-aware routing for direct-connect data centers", in Proc. IEEE INFOCOM, 2017, pp.1-9.
- [13] Noormohammadpour M, Raghavendra C S. "Datacenter traffic control: Understanding techniques and tradeoffs", IEEE Communications Surveys & Tutorials, 2017, vol.20, no.2, pp.1492-1525.
- [14] Floyd S, Jacobson V. "Random early detection gateways for congestion avoidance", IEEE/ACM Transactions on networking, 1993, vol.1, no.4, pp.397-413.
- [15] Zou S, Huang J, Jiang W, et al. "Achieving high utilization of flowlet-based load balancing in data center networks", Future Generation Computer Systems, 2020, vol.108, pp.546-559.
- [16] Huang J, Li W, Li Q, et al. "Tuning high flow concurrency for MPTCP in data center networks", Journal of Cloud Computing, 2020, vol.9, no.1, pp.1-15.
- [17] Alizadeh M, Greenberg A, Maltz D A, et al. "Data center tcp (dctcp)", in Proc. ACM SIGCOMM, 2010, pp.63-74.
- [18] Hu J, Huang J, Lv W, et al. "CAPS: Coding-based adaptive packet spraying to reduce flow completion time in data center", IEEE/ACM Transactions on Networking, 2019, vol.27, no.6, pp.2338-2353.
- [19] Zhangy W, Lingy D, Zhangy Y, et al. "Achieving optimal edge-based congestion-aware load balancing in data center networks", in Proc. IEEE Networking Conference, 2020, pp.109-117.
- [20] Zou S, Huang J, Wang J, et al. "Improving TCP Robustness over Asymmetry with Reordering Marking and Coding in Data Centers", In Proc. International Conference on Distributed Computing Systems, 2019, pp.57-67.
- [21] Liu J, Huang J, Li W, et al. "AG: Adaptive Switching Granularity for Load Balancing with Asymmetric Topology in Data Center Network", in Proc. International Conference on Network Protocols, 2019, pp.1-11.
- [22] L. Poutievski, A. Singh, and A. Vahdat. "Wcmp:weighted cost multipathing for improved fairness in data centers", in Proc. ACM EuroSys, 2014.
- [23] Zhang T, Lei Y, Zhang Q , et al. "Fine-grained Load Balancing with Traffic-aware Rerouting in DataCenter Networks", Journal of Cloud Computing, 2021, vol.10, no.1.
- [24] G. Kumar, N. Dukkipati, K. Jang, H. M .G. Wassel, X. Wu, B. Montazeri, Y. Wang, K. Springborn, C. Alfeld, M. Ryan, D. Wetherall, A. Vahdat, "Swift: Delay is Simple and Effective for Congestion Control in the Datacenter", in Proc. ACM SIGCOMM, 2020, pp. 514-528.
- [25] Abbasloo, S. , Y. Xu , and H. J. Chao . "To schedule or not to schedule: When no-scheduling can beat the best-known flow scheduling algorithm in datacenter networks", Computer Networks 2020, pp.107-177.
- [26] Olmedilla C, Escudero-Sahuquillo J , Garcia P J , et al. "Optimizing Packet Dropping by Efficient Congesting-Flow Isolation in Lossy Data-Center Networks", 2020 IEEE Symposium on High-Performance Interconnects (HOTI) IEEE, 2020.
- [27] Gao, W. Huang J , Zou S , et al. "AAC: Adaptively Adjusting Concurrency by Exploiting Path Diversity in Datacenter Networks", Journal of Network and Systems Management, 2021, vol.29, no.3.
- [28] J. Perry, H. Balakrishnan, D. Shah, "Flowtune: flowlet control for datacenter networks", in Proc. USENIX NSDI, 2017, pp. 421-435.
- [29] Hu J, Huang J, Lv W, et al. "TLB: Traffic-aware Load Balancing with Adaptive Granularity in Data Center Networks", ICPP 2019: Proceedings of the 48th International Conference on Parallel Processing, 2019, pp.1-10.
- [30] Wang P, Trimponias G, Xu H, Geng YH. "Luopan: Sampling based load balancing in data center networks", IEEE Trans on Parallel and Distributed Systems, 2019, vol.30, no.1, pp.133-145.