Load Balancing in PFC-Enabled Datacenter Networks

Jinbin Hu[1,2], Chaoliang Zeng[1], Zilong Wang[1], Hong Xu[3], Jiawei Huang[4], Kai Chen[1]
Hong Kong University of Science & Technology[1], Changsha University of Science & Technology[2], Chinese University of Hong Kong[3], Central South University[4]

ABSTRACT

In Priority Flow Control (PFC) enabled datacenter networks (DCNs), PFC is inevitably triggered due to bursty traffic even with end-to-end congestion control. Load balancing as a complementary mechanism to transport protocols can make rerouting decisions in time to alleviate PFC's head-of-line (HoL) blocking problem. However, prior solutions designed for lossy DCNs do not work well in PFC-enabled networks, because the unreliable rerouting signals such as separate local queue length, round-trip time (RTT), explicit congestion notification (ECN), and link load cannot timely and correctly reflect PFC pausing.

We present a PFC-aware Load Balancer called PLB, which is resilient to hop-by-hop PFC pausing. At its heart, PLB leverages RTT-level signals (i.e., RTT and link utilization) and sub-RTT level signal (i.e., cumulative sojourn time) to react to PFC pausing timely and select the appropriate path rationally. This enables PLB to efficiently balance traffic and mitigate the victim flows suffering from serious HoL blocking. The NS-3 simulation results show that PLB handles PFC pausing well under realistic workloads and significantly reduces the average flow completion time by up to 24%, 30%, and 37% compared to CONGA, DRILL and Hermes, respectively.

CCS CONCEPTS

• **Networks** → **Network architectures**; **Data center networks**; **Routing protocols**;

KEYWORDS

Datacenter, Lossless Networks, Load Balancing

APNet'22, July 1-2, China

ACM ISBN 978-x-xxxx-xxxx-x/YY/MM…$15.00
https://doi.org/10.1145/nnnnnnn.nnnnnnn

ACM Reference Format:
Jinbin Hu[1,2], Chaoliang Zeng[1], Zilong Wang[1], Hong Xu[3], Jiawei Huang[4],, Kai Chen[1]. 2022. Load Balancing in PFC-Enabled Datacenter Networks. In *APNet'22: 6th Asia-Pacific Workshop on Networking, July 1-2, 2021, Fuzhou, China.* ACM, New York, NY, USA, 8 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 INTRODUCTION

Driven by the stringent demands for high throughput and low tail latency from datacenter applications, lossless networks are increasingly crucial in DCNs [1–5]. Production Ethernet-based DCNs rely on hop-by-hop PFC to guarantee no packet loss due to network congestion [6–10]. However, PFC mechanism has many well-known problems such as HoL blocking, congestion spreading and deadlock, which degrade the performance of individual flows dramatically [1–3, 7].

Recently, end-to-end transport protocols have been proposed to effectively reduce PFC triggering and in turn alleviate PFC's problems [1, 2, 5, 11–14]. Unfortunately, PFC is still triggered with these transmission schemes especially due to transient congestion caused by bursty traffic [2, 4]. In this paper, we do not intent to debate about whether it is better to deploy PFC for lossless networks or develop alternatives in lossy networks[1]. Rather, as a complementary mechanism to the already-deployed transport protocols in existing PFC-enabled DCNs, we design a data-plane load balancer in today's programmable switches, which can make rerouting decisions in time even during PFC pausing to mitigate HoL blocking.

Prior load balancing mechanisms designed for lossy DCNs do not work well in PFC-enabled lossless networks due to the unreliable rerouting signals (see §2.2 for details). First, local queue-based schemes such as DRILL [19] cannot react to remote PFC pausing, a local egress port with smaller queue length is not necessarily a better forwarding path. Second, schemes using end-to-end signals (i.e., RTT and ECN) to decide forwarding path such as Hermes [20] cannot react to PFC pausing timely due to the feedback loop of at least one RTT. This implies that a path with larger delay is not necessarily a worse forwarding path. Third, separate link

[1]Rather than continuing down the path of solving the side effects of PFC, some works [15–18] seek better methods to handle packet loss and disable PFC completely. This is also a direction worth studying.

load used in the schemes such as CONGA [21] and HULA [22] cannot reflect PFC pausing correctly. Thus, a path with lower link load due to PFC pausing may also be a worse path. We show that prior load balancing schemes can inflate FCT by up to 49% in PFC-enabled networks (§2.2).

We design a PFC-aware load balancing mechanism for lossless DCNs called PLB, which is resilient to the hop-by-hop PFC pausing (§3). On a high level, PLB relies on the RTT-level signals (i.e., RTT and link utilization) and sub-RTT level signal (i.e., cumulative sojourn time, meaning cumulative queueing time of a packet on different switches on the path) to choose the appropriate path and deal with PFC pausing rationally. In this way, PLB can efficiently balance traffic and reduce the tail latency of victim flows affected by PFC.

PLB first detects path conditions by probing RTT-level signals. To capture the characteristics of paths that are non-congested, congested, and unknown status with PFC pausing (which may be caused by real congestion or PFC backpressure diffusion), PLB uses RTT and link utilization to measure path state simultaneously. For non-congested paths, both RTT and link utilization are low; for congested paths, RTT is high due to queueing delay and the link is fully utilized; for paths with unknown status where PFC occurred, RTT is high due to queueing delay and the link utilization is less than 100% due to PFC pausing. Thus these two signals indicate the path state at a coarse granularity in PFC-enabled networks.

Then PLB further utilizes sub-RTT level signal, i.e., a packet's cumulative sojourn time (CST), to timely handle PFC pausing. Once a packet experiences PFC pausing, and the CST on the path is larger than the maximum acceptable delay compared to other paths, PLB removes this path from the available forwarding paths until its RTT and link utilization are updated. At the same time, the packet and subsequent ones of the corresponding flows on this path are rerouted to avoid HoL blocking.

In brief, PLB makes routing decisions based on both RTT- and sub-RTT level signals to timely and rationally react to PFC pausing. On the one hand, PLB flexibly chooses the best path among the available ones for each packet based on RTT-level signals. On the other hand, when packets are blocked due to PFC PAUSE messages, instead of switching path instantly, PLB rationally considers whether to remove the current path and reroute packets based on the sub-RTT level signal. Therefore, PLB is able to effectively balance traffic and reduce tail latency due to PFC pausing.

Our preliminary NS-3 simulation results show that PLB significantly outperforms prior schemes (§4). For example, under the web server workload [23], PLB reduces the average flow completion time (FCT) and 99^{th} percentile FCT by up to 24% and 30%, 37% and 23%, 28% and 34% compared to the CONGA, DRILL, and Hermes, respectively.

The rest of this paper is organized as follows. §2 introduces background and motivation. §3 describes the design of PLB in detail. §4 presents evaluation results, and §5 discusses related works before we conclude the paper in §6.

2 BACKGROUND AND MOTIVATION

2.1 PFC is Still Triggered

PFC mechanism. To guarantee lossless transmission, Converged Enhanced Ethernet (CEE) employs the hop-by-hop PFC mechanism, which is defined by IEEE 802.1Qbb [25]. Specifically, once the ingress queue length exceeds a specified threshold, the switch sends PFC PAUSE to upstream switch to stop data transmission until the pause duration expires or receiving PFC RESUME when the ingress queue drains below another threshold. Since PFC pausing is agnostic to the congested flows, the coarse reaction to congestion causes the well-known problems such as HoL blocking, congestion spreading, unfairness and even deadlock [1–4, 26]. This paper focuses on how to handle HoL blocking in a timely and better-later-than-never manner.

PFC is triggered under bursty traffic. Recent proposed end-to-end transport protocols can effectively control non-transient congestion and thus reduce PFC triggering [1, 2, 12–14]. However, it is hard for them to control the transient congestion due to the bursty and frequent short-lived flows that usually finish within one RTT, resulting in PFC triggering. Since the existing transport protocols are inadequate to prevent PFC triggering under transient congestion, an efficient load balancing scheme is necessary as a cooperation mechanism for congestion control to react to PFC pausing quickly. Load balancing in PFC-enabled networks is a non-trivial challenge. To cope with dynamic traffic and fast hop-by-hop PFC, an ideal solution needs to sense path congestion and PFC pausing timely and correctly, and thus it can be resilient to path state dynamics by flexibly rerouting. Although the existing load balancing schemes work well in lossy DCNs, they are insufficient to handle PFC pausing in lossless DCNs as we demonstrate in the following.

2.2 Prior Load Balancing Falls Short

To motivate our design, we discuss the drawbacks of three typical switching signals used by the existing load balancing schemes in PFC-enabled DCNs.

Simulation settings: We conduct NS-3 simulations for motivation cases in a common leaf-spine topology as shown in Fig. 1. There are two equal-cost paths between leaf switches $L1$ and $L3$, i.e., $Path1$ {$P1$/$L1$, $C1$, $L3$} and $Path2$ {$P2$/$L1$, $C2$, $L3$}. Each link capacity is 40Gbps and the link delay is 5μs. In the following tests, we deployed DCQCN [1] as the underlying transport protocol and enabled PFC. The background flows (in gray) are transmitting on the $Path1$ from the source

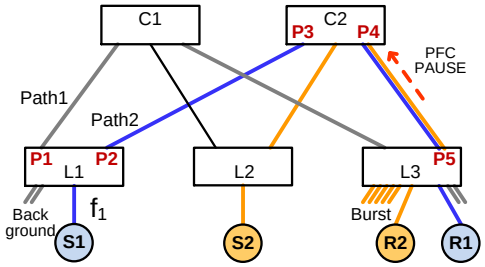
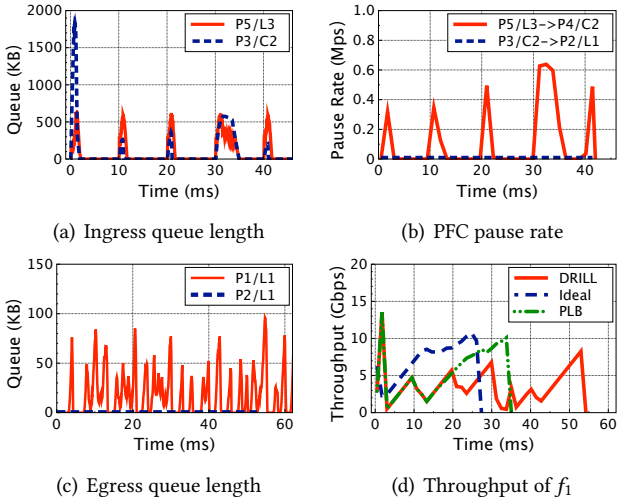


Figure 1: Typical network scenario.



(a) Ingress queue length (b) PFC pause rate

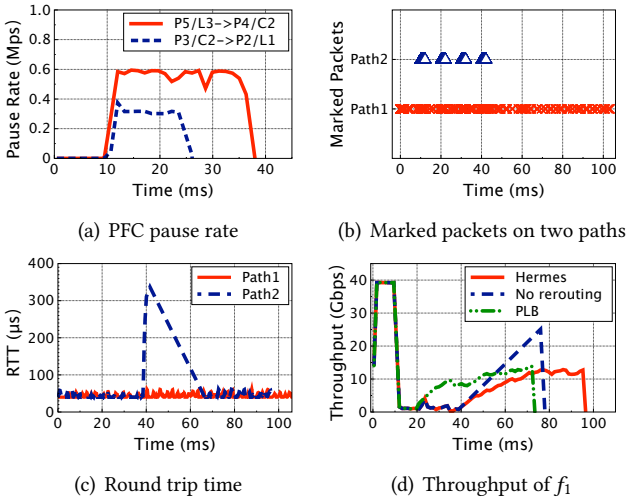
(c) Egress queue length (d) Throughput of f_1

Figure 2: Rerouting based on local queue (DRILL).

hosts under $L1$ to the destination hosts under $L3$. The victim flow f_1 (in blue) is sent from the sender $S1$ to the receiver $R1$. The bursty flows (in yellow) are sent intermittently from the sender $S2$ to the receiver $R2$, and 14 end-hosts under $L3$ also send bursty flows at line rate to the same receiver $R2$.

2.2.1 Local Queue cannot React to Remote PFC Pausing. The local queue-based switching is oblivious to remote PFC triggering on the same path, a local egress port with smaller queue is not necessarily a better choice. For a typical scheme DRILL [19], it relies on local queue length to make forwarding decisions to deal with microbursts quickly. Since there is no any coordination among switches, it cannot sense the remote PFC pausing, potentially leading to serious HoL blocking at the downstream switches and even inability to reroute. Next, we show a case study. At the beginning, f_1 with 20MB starts at line rate. At time 1ms, 26 bursty flows from $S2$ and 56 bursty flows under $L3$ with each flow size of 200KB are intermittently sent to $R2$, with a total of 5 bursts.

In Fig. 2(a) and Fig. 2(b), when each bursty traffic starts, the queue length of ingress port $P5/L3$ is increased and PFC PAUSE messages are sent from $P5/L3$ to pause the egress port $P4/C2$. In this process, since the queue length of $P3/C2$ does not exceed the PFC threshold, PFC pausing is not spread to the upstream switch $L1$. The queue length of $P2/L1$ is not



(a) PFC pause rate (b) Marked packets on two paths

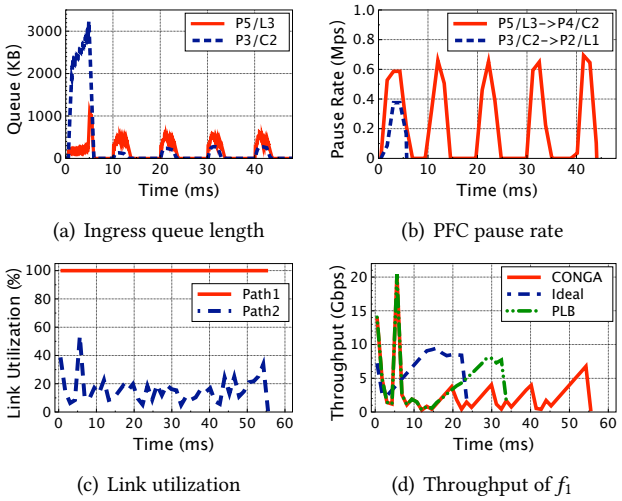
(c) Round trip time (d) Throughput of f_1

Figure 3: Rerouting Based on RTT and ECN (Hermes).

increased and is always lower than that of $P1/L1$ as shown in Fig. 2(c). Thus, based on the local queue to make routing decision, DRILL always chooses $P2/L1$ to forward packets of f_1 on $Path2$ {$P2/L1$, $C2$, $L3$}, resulting in f_1 suffering from PFC's HoL blocking and largest FCT, as shown in Fig. 2(d). To illustrate the potential performance improvement without HoL blocking, we employ an ideal routing solution, which can predict PFC triggering on $Path2$ and forward packets of f_1 to $Path1$ without any HoL blocking. Fig. 2(d) shows the ideal solution reduces FCT by up to 49%.

2.2.2 End-to-end Signals cannot React to PFC Pausing Timely. The end-to-end switching signals cannot react to PFC pausing timely due to the feedback loop of at least one RTT, and thus a path with larger delay is not necessarily a worse choice. For the typical mechanism Hermes [20], it leverages RTT and ECN to detect path congestion. Although these two signals capture the queueing delay, neither they can react to PFC pausing timely due to their stale property especially when PFC pausing lasts for a long time, nor identity whether the path with increased RTT is caused by real congestion or PFC pausing. In this test, f_1 is 120MB and each bursty flow is 200KB. At time 10ms, 45 bursty flows from $S2$ and 630 bursty flows under $L3$ are continuously sent to $R2$.

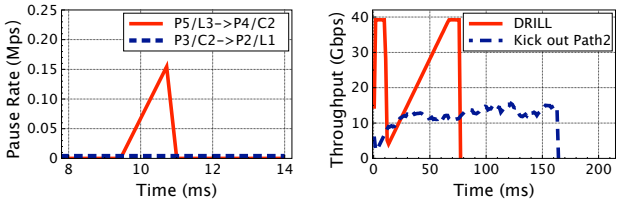
Fig. 3(a) shows that PFC PAUSE messages are sent from $P5/L3$ to $P4/C2$ continuously and spread from $P3/C2$ to $P2/L1$ during bursty congestion. Fig. 3(b) shows packets on $Path2$ are marked by ECN during PFC triggering, while packets on $Path1$ are marked continuously due to real congestion. In Fig. 3(c), the RTT of $Path2$ is increased significantly due to PFC pausing. Thus, Hermes preferentially selects $Path2$ with small RTT and ECN at the beginning, and it forwards packets to the $Path1$ once the RTT of $Path2$ is much larger than $Path1$, as shown in Fig. 3(d). However, such a late rerouting based on the staled switching signals after a long PFC pausing



(a) Ingress queue length (b) PFC pause rate

(c) Link utilization (d) Throughput of f_1

Figure 4: Rerouting Based on Link Load (CONGA).



(a) PFC pause rate (b) Throughput of f_1

Figure 5: Kick out the path with PFC pausing.

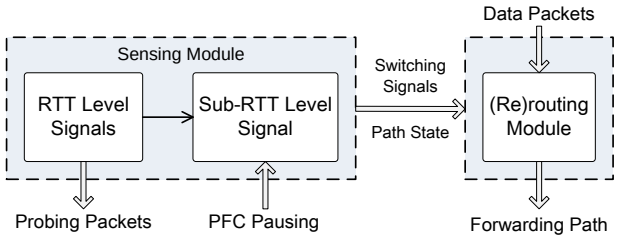


Figure 6: PLB overview.

does not bring performance benefits. On the contrary, FCT without rerouting is better than Hermes by 21%.

2.2.3 Link Load cannot React to PFC Pausing Correctly. Since the link is under-utilized when PFC is triggered, a path with lower link load is not necessarily a better choice. For the typical mechanisms CONGA [21] and HULA [22], they detect path conditions by measuring link utilization. However, the path experiencing PFC pausing has even lower link load, which cannot indicate path congestion correctly and will misguide rerouting. We further conduct simulations to illustrate this issue with CONGA. The victim flow f_1 is 20MB and bursty flow is 200KB. At time 1ms, 42 bursty flows from $S2$ and 588 bursty flows under $L3$ are sent to $R2$.

In Fig. 4(a) and Fig. 4(b), 5 bursty congestion occurred intermittently on $Path2$, and PFC is triggered on the ingress port $P5/L3$ and spread to $P3/C2$. Fig. 4(c) shows that the link utilization of $Path2$ is less than that of $Path1$ with real congestion. As shown in Fig. 4(d), CONGA always chooses $Path2$ with the lower measured link load to forward packets of f_1, resulting in serious HoL blocking. The ideal solution knows in advance that PFC will triggered multiple times on $Path2$ and does not choose $Path2$, achieving the lowest FCT.

2.2.4 Kick Out PFC Pausing Path Directly is Unwise. Intuitively, as long as the path where PFC is triggered, it should be kicked out directly. However, for a short PFC pausing, it is unwise not to choose this path arbitrarily. To illustrate this, we conduct simulations with DRILL in the same experiment settings as in §2.2.1. The flow f_1 is 250MB and bursty flow is 200KB. At time 9ms, 2 bursty flows from $S2$ and 28 bursty flows from 14 end-hosts under $L3$ are sent to $R2$.

Fig. 5(a) shows the duration of PFC pausing is very short. In this test, DRILL always chooses the path with the minimum local queue length, achieving smaller FCT, as shown in Fig. 5(b). On the contrary, if $Path2$ is kicked out directly and packets are forwarded on $Path1$, FCT is increased by 53% due to the real congestion on $Path1$. Therefore, it is unreasonable to kick out the path with PFC pausing directly, but it is necessary to carefully decide whether to reroute according to the actual path conditions.

3 DESIGN

3.1 Design Overview

The limitations of switching signals discussed in §2.2 highlight the properties required for load balancing solutions in PFC-enabled DCNs. Thus, PLB aims to achieve two design goals: (1) correctly detect path states and timely sense the duration of PFC pausing to guide load balancing decisions; (2) flexibly and rationally rerouting to balance traffic and avoid PFC's HoL blocking.

To this end, we propose PLB, a simple yet effective load balancer for PFC-enabled DCNs. Fig. 6 overviews the two main modules of PLB: (1) the sensing module that contains RTT level signals to correctly detect path states including non-congested path, real congested path and the undetermined path with PFC pausing, and sub-RTT level signal to timely sense the PFC pausing delay; (2) the rerouting module that is responsible for rationally making forwarding decisions for each data packet based on the above sensing state.

3.2 RTT Level Signals

In PFC-enabled networks, in addition to the congested and non-congested path states, there is also an undetermined path state, which showing ON/OFF transmission pattern due to PFC pausing/resuming with under-utilized link. It is insufficient to simultaneously reflect these path states

relying on the congestion signals alone used in the prior load balancing schemes. PLB detects path conditions through the combination of RTT and link utilization, which can well indicate the aforementioned path states in lossless networks.

Specifically, based on RTT and link utilization, parallel paths can be divided into the following three categories: (1) non-congested paths with small RTT (i.e., less than RTT_{low}, which is set to 20-40μs plus the base RTT to ensure that the path is lightly loaded [20]) and small link utilization (less than or equal to 1); (2) real congested path with large RTT (i.e., greater than RTT_{low}) and the link utilization is equal to 1; (3) undetermined paths occurred PFC-pausing with RTT larger than RTT_{low} and the link utilization is less than 1.

3.3 Sub-RTT Level Signal

PLB further tracks per-packet CST at sub-RTT timescale and then makes a rational decision whether to switch path to react to the transient congestion timely. The high-level idea is that if the current path delay is less than other paths and PFC pausing is only transient, PLB prefers to tolerate the delay caused by PFC PAUSE rather than switching paths to risk larger delay. In contrast, PLB makes rerouting decision timely based on the idea of better-later-than-never to avoid continuous HoL blocking.

Algorithm 1: Sensing PFC by Sub-RTT Level CST

```
Input:
t_td: Tolerable delay; t_enqueue: The enqueue time;
t_avgST: The average sojourn time on a switch;
t'_avgST: The last average sojourn time on a switch;
1  for every packet do
2      Assume its forwarding path is p;
3      Predict the minimum queueing delay t_pd;
4      if PFC pausing then
5          if t_td < max(t_avgST, t_pd) then
6              Wait to be forwarded at time t_dequeue;
7              t_td = t_td - (t_dequeue - t_enqueue);
8              t_avgST = (1- α)*t_avgST + α*t'_avgST;
9          else
10             Wait to be forwarded & Trigger rerouting;
11             p_active = 0;   /* kick out path p */
12     return p_active
```

The Algorithm 1 illustrates the detailed process of sensing PFC pausing by using sub-RTT level signal CST. Specifically, each packet header carries a tolerable delay t_{td} spending on switches over the path p, which is the delay difference between the optimal and sub-optimal paths. After arriving the ingress port, the packet first predicts the minimum queuing delay t_{pd} based on the current destination egress port queue. Then, if t_{td} is less than the maximum value between the average sojourn time t_{avgST} and t_{pd}, the packet waits to be forwarded. The tolerable delay t_{td} is updated by subtracting the sojourn time at the current switch, and t_{avgST} maintained at each switch is also updated corresponding (lines 5-8). Otherwise, if the packet's CST exceeds t_{td}, PLB triggers rerouting and kicks out the path p (P_{active}=0) (lines 10-11) until its RTT and link utilization are updated (triggered by RTT-level signals).

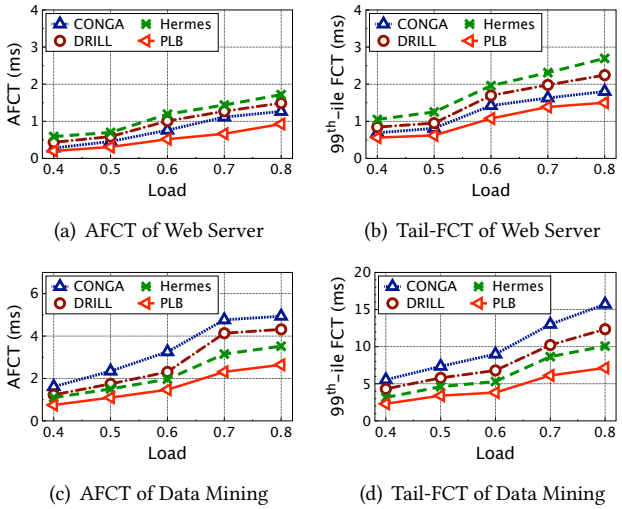
3.4 Rerouting

PLB uses RTT and sub-RTT level signals to guide routing simultaneously. Based on the RTT and link utilization measured on a coarse RTT level timescale, PLB selects forwarding paths from 3 types of paths (i.e., non-congested path, undetermined path, and real congested path) in order of priority. Since RTT level congestion signals is stale by at least one RTT, PLB further uses sub-RTT signal to guide rerouting. Although we cannot accurately predict the duration of PFC pausing, we can measure the elapsed time on the path, that is, the cumulative sojourn time spent on the path, to determine whether it is necessary to reroute. Based on CST, even under the bursty traffic scenario, PLB can react to PFC pausing timely by rational better-later-than-never rerouting.

Algorithm 2: Rationally Rerouting

```
Input:
t_RTT, u_link: Measured RTT and link load of a path;
1  for every packet do
2      Assume its corresponding flow is f and path is p;
3      if a new flow f || p is congested or kicked out then
4          {P'} = non-congested paths & p_active==1;
5          if {P'} ≠ ∅ then
6              p* = Argmin_{p∈{P'}}(p.t_RTT);
7          else
8              {P''} = paths occurred PFC & p_active==1;
9              if {P''} ≠ ∅ then
10                 p* = Argmax_{p∈{P''}}(p.u_link);
11             else
12                 {P'''} = congested paths & p_active==1;
13                 p* = Argmin_{p∈{P'''}}(p.t_RTT);
14     return p*   /* The new routing path p* */
```

The rerouting logic of PLB is illustrated in Algorithm 2, which is triggered for the cases of a new flow packet arriving, the current path is congested or kicked out. PLB first tries to select an available (p_{active}==1) non-congested path with the minimum RTT (lines 4-6). If it fails, PLB chooses an available undetermined path with the maximum link utilization (lines 8-10), which is likely resumed transmission quickly. The lowest priority choice of PLB is the path with the minimum RTT among the remaining real congested paths (lines 12-13).



(a) AFCT of Web Server (b) Tail-FCT of Web Server

(c) AFCT of Data Mining (d) Tail-FCT of Data Mining

Figure 7: FCT under realistic workloads.

4 EVALUATION

We run NS-3 simulations to evaluate the performance of PLB preliminarily. Same as [20], we use a 8×8 leaf-spine topology with 128 hosts connected by 40Gbps links. The per-link propagation delay is 5μs. The switch buffer size is set to 9MB and PFC is enabled. We employ DCQCN for transmission protocol. Fig. 7 shows the average FCT (AFCT) and 99^{th} percentile FCT under Web Server and Data Mining workloads. For Web Server, as the traffic load increases, PLB outperforms CONGA, DRILL and Hermes by up to 24%, 30% and 37% for AFCT, and 23%, 28% and 34% for 99^{th} percentile FCT, respectively. For Data Mining, PLB reduces AFCT by up to 41% compared with CONGA at 0.8 load.

Result analysis: PLB achieves the lowest FCT at all levels of loads compared with CONGA, DRILL and Hermes. This indicates that PLB utilizes the RTT and sub-RTT level signals effectively to sense and react to congestion in PFC-enabled networks. On the one hand, PLB detects different types of paths correctly by using RTT and link utilization to guide load balancing decisions. On the other hand, PLB considers whether to switch path carefully through CST even though experiencing PFC pausing, rather than rerouting arbitrarily or stay on the paused path all the time. In addition, we observe that CONGA and Hermes behave differently under Web Server and Data Mining workloads. The reasons are that PFC pausing time is relatively short under the Web Server workload, where all flows are less than 1MB [34], and Hermes uses stale end-to-end congestion signals to guide routing without remedial scheme. These two factors result in many unnecessary rerouting when applying Hermes in the Web Server workload. The Data Mining workload contains more large flows, once a long PFC pausing occurs, the low link utilization misguides CONGA to always choose the paused path without timely rerouting, resulting in poor performance. For DRILL, although it cannot sense the remote PFC pausing, when the PFC PAUSE is backpressed to the local switch, there is a chance to reroute. In comparison, PLB can reroute flows rationally and timely under different traffic scenarios once sensing congestion in PFC-enabled networks.

5 RELATED WORK

In recent years, several RDMA congestion control mechanisms [1, 2, 5, 6, 10–14, 17, 18, 35] have been proposed. QCN [11] controls congestion at Layer 2. DCQCN [1] and DCQCN+ [6] rely on ECN marking to reduce PFC triggering. TIMELY [12] and Swift [13] leverage RTT as the congestion signal to reduce queuing delay. MP-RDMA [5] proposes a multi-path ACK-clocking scheme to reduce FCT. PCN [2] only regulates the rate of identified congested flows. TCD [4] detects congested flows for existing transports. BFC [17] designs a per-hop per-flow control to reduce HoL blocking. Dart [10] proposes a divide-and-specialize approach to reduce receiver and in-network congestion. The above efforts effectively reduce congestion, but they still cannot avoid PFC pausing especially under bursty scenario. IRN [18] explores loss recover scheme for RDMA without PFC.

To alleviate congestion by using multiple paths, there is a large body of load balancing schemes [19–22, 36–47] originally designed for lossy DCNs. CONGA [21] and HULA [22] use link utilization to sense path congestion. DRILL [19] forwards packets based on the local queue length. Hermes [20], CAPS [38], Clove [39] and FlowBender [40] make routing decision based on end-to-end signals such as RTT and ECN. LetFlow [41], Presto [43] and RPS [44] choose forwarding path randomly. TLB [45] is a traffic-aware load balancer. Hedera [46] and MicroTE [47] schedule flows by using network controller. However, these schemes do not work well in PFC-enabled networks due to unreliable congestion signals.

6 CONCLUSION AND FUTURE WORK

This paper presented PLB, a load balancing scheme for PFC-enabled networks, which leverages RTT level signals (i.e., RTT and link utilization) and sub- RTT level signal (i.e., cumulative sojourn time) to detect path conditions correctly and make rerouting decisions rationally in a better-later-than-never way. In this way, PLB can react to PFC pausing timely and reduce head-of-line blocking. The preliminary simulation results show that, PLB effectively reduces FCT by up to 41% at high load under realistic workloads compared with the existing load balancing solutions.

In the next, we will analysis the competitive ratio of PLB as an online decision-making algorithm. Then, we will implement PLB on the hardware programmable switches in a real testbed environment and conduct more experiments to evaluate that PLB can work well in the PFC-enable networks.

ACKNOWLEDGMENT

This work is supported in part by the Key-Area Research and Development Program of Guangdong Province (2021B0101400001), the Hong Kong RGC TRS T41-603/20-R, GRF 16213621 and GRF 16215119, and the National Natural Science Foundation of China (62132022, 62102046, 61872387).

REFERENCES

[1] Y. Zhu, H. Eran, D. Firestone, C. Guo, M. Lipshteyn, Y. Liron, J. Padhye, S. Raindel, M. H. Yahia, and M. Zhang. Congestion Control for Large-Scale RDMA Deployments. In Proc. ACM SIGCOMM, 2015.

[2] W. Cheng, K. Qian, W. Jiang, T. Zhang, and F. Ren. Re-architecting Congestion Management in Lossless Ethernet. In Proc. USENIX NSDI, 2020.

[3] K. Qian, W. Cheng, T. Zhang, and F. Ren. Gentle Flow Control: Avoiding Deadlock in Lossless Networks. In Proc. ACM SIGCOMM, 2019.

[4] Y. Zhang, Y. Liu, Q. Meng, F. Ren. Congestion Detection in Lossless Networks. In Proc. ACM SIGCOMM, 2021.

[5] Y. Lu, G. Chen, B. Li, K. Tan, Y. Xiong, P. Cheng, J. Zhang, E. Chen, and Thomas Moscibroda. Multipath Transport for RDMA in Datacenters. In Proc. USENIX NSDI, 2018.

[6] Y. Zhu, M. Ghobadi, V. Misra, and J. Padhye. ECN or Delay: Lessons Learnt from Analysis of DCQCN and TIMELY. In Proc. ACM CoNEXT, 2016.

[7] C. Guo, H. Wu, Z. Deng, G. Soni, J. Ye, J. Padhye, and M. Lipshteyn. RDMA over Commodity Ethernet at Scale. In Proc. ACM SIGCOMM, 2016.

[8] H. Lim, W. Bai, Y. Zhu, Y. Jung, D. Han. Towards Timeout-less Transport in Commodity Datacenter Networks. In Proc. ACM EuroSys, 2021.

[9] C. Tian, B. Li, L. Qin, J. Zheng, J. Yang, W. Wang, G. Chen, and W. Dou. P-PFC: Reducing Tail Latency with Predictive PFC in Lossless Data Center Networks. IEEE Transactions on Parallel and Distributed Systems, 31(6):1447-1459, 2020.

[10] J. Xue, M. U. Chaudhry, B. Vamanan, T. N. Vijaykumar, and M. Thottethodi. Dart: Divide and Specialize for Fast Response to Congestion in RDMA-based Datacenter Networks. IEEE/ACM Transactions on Networking, 28(1):322-335, 2020.

[11] IEEE. 802.1Qau – Congestion Notification. http://www.ieee802.org/1/pages/802.1au.html.

[12] R. Mittal, V. T. Lam, N. Dukkipati, E. Blem, H. Wassel, M. Ghobadi, A. Vahdat, Y. Wang, D. Wetherall, and D. Zats. TIMELY: RTT-based Congestion Control for the Datacenter. In Proc. ACM SIGCOMM, 2015.

[13] G. Kumar, N. Dukkipati, K. Jang, et al. Swift: Delay is Simple and Effective for Congestion Control in the Datacenter. In Proc. ACM SIGCOMM, 2020.

[14] Y. Li, R. Miao, H. H. Liu, Y. Zhuang, F. Feng, L. Tang, Z. Cao, M. Zhang, F. Kelly, M. Alizadeh, M. Yu. HPCC: High Precision Congestion Control. In Proc. ACM SIGCOMM, 2019.

[15] A. Singhvi, A. Akella, D. Gibson, T. F. Wenisch, M. Wong-Chan, S. Clark, M. M. K. Martin, M. McLaren, P. Chandra, R. Cauble, H. M. G. Wassel, B. Montazeri, S. L. Sabato, J. Scherpelz, and A. Vahdat. 1RMA: Re-envisioning Remote Memory Access for Multi-tenant Datacenters. In Proc. ACM SIGCOMM, 2020.

[16] R. Mittal, A. Shpiner, A. Panda, E. Zahavi, A. Krishnamurthy, S. Ratnasamy, and S. Shenker. Revisiting Network Support for RDMA. In Proc. ACM SIGCOMM, 2018.

[17] P. Goyal, P. Shah, K. Zhao, G. Nikolaidis, M. Alizadeh, and T. E. Anderson. Backpressure Flow Control. In Proc. USENIX NSDI, 2022.

[18] R. Mittal, A. Shpiner, A. Panda, E. Zahavi, A. Krishnamurthy, S. Ratnasamy, and S. Shenker. Revisiting Network Support for RDMA. In Proc. ACM SIGCOMM, 2018.

[19] S. Ghorbani, Z. Yang, P. Godfrey, Y. Ganjali, and A. Firoozshahian. DRILL: Micro Load Balancing for Low-latency Data Center Networks. In Proc. ACM SIGCOMM, 2017.

[20] H. Zhang, J. Zhang, W. Bai, K. Chen, and M. Chowdhury. Resilient Datacenter Load Balancing in the Wild. In Proc. ACM SIGCOMM, 2017.

[21] M. Alizadeh, T. Edsall, S. Dharmapurikar, R. Vaidyanathan, K. Chu, A. Fingerhut, V. T. Lam, F. Matus, R. Pan, N. Yadav, G. Varghese. CONGA: Distributed Congestion-aware Load Balancing for Datacenters. In Proc. ACM SIGCOMM, 2014.

[22] N. Katta, M. Hiray, C. Kimz, A. Sivaraman and J. Rexford. Hula: Scalable Load Balancing Using Programmable Data Planes. In Proc. ACM SOSR, 2016.

[23] M. Alizadeh, A. Greenberg, D. A. Maltz, J. Padhye, P. Patel, B. Prabhakar, S. Sengupta, and M. Sridharan. Data Center TCP (DCTCP). In Proc. ACM SIGCOMM, 2010.

[24] Albert Greenberg, James R. Hamilton, Navendu Jain, Srikanth Kandula, Changhoon Kim, Parantap Lahiri, David A. Maltz, Parveen Patel, and Sudipta Sengupta. Vl2: A Scalable and Flexible Data Center Network. In Proc. ACM SIGCOMM, 2009.

[25] IEEE 802.1 Qbb - Priority-based Flow Control. https://1.ieee802.org/dcb/802-1qbb/.

[26] S. Hu, Y. Zhu, P. Cheng, C. Guo, K. Tan, J. Padhye, and K. Chen. Tagger: Practical PFC Deadlock Prevention in Data Center Networks. In Proc. ACM CoNEXT, 2017.

[27] B. Tian, J. Gao, M. Liu, E. Zhai, Y. Chen, Y. Zhou, L. Dai, F. Yan, M. Ma, M. Tang, J. Lu, X. Wei, H. H. Liu, M. Zhang, C. Tian, M. Yu. Aquila: A Practically Usable Verification System for Production-Scale Programmable Data Planes. In Proc. ACM SIGCOMM, 2021.

[28] T. Benson, A. Akella, and D. Maltz. Network Traffic Characteristics of Data Centers in the Wild. In Proc. IMC, 2010.

[29] A. Roy, H. Zeng, J. Bagga, G. Porter, and A. C. Snoeren. Inside the Social Network's (datacenter) Network. In Proc. ACM SIGCOMM, 2015.

[30] J. Zerwas, K. Aykurt, S. Schmid, A. Blenk. Network Traffic Characteristics of Machine Learning Frameworks Under the Microscope. In Proc. CNSM, 2021.

[31] M. Noormohammadpour and C. S. Raghavendra. Datacenter Traffic Control: Understanding Techniques and Tradeoffs. IEEE Communications Surveys & Tutorials, 20(2):1492–1525, 2018.

[32] S. Yan, X. Wang, X. Zheng, Y. Xia, D. Liu, W. Deng. ACC: Automatic ECN Tuning for High-Speed Datacenter Networks. In Proc. ACM SIGCOMM, 2021.

[33] S. Hu, K. Chen, H. Wu, W. Bai, C. Lan, H. Wang, H. Zhao, and C. Guo. Explicit Path Control in Commodity Data Centers: Design and Applications. In Proc. USENIX NSDI, 2015.

[34] S. Hu, W. Bai, G. Zeng, Z. Wang, B. Qiao, K. Chen, K. Tan, and Y. Wang. Aeolus: A Building Block for Proactive Transport in Datacenters. In Proc. ACM SIGCOMM, 2020.

[35] B. Yi, J. Xia, L. Chen, and K. Chen. Towards Zero Copy Dataflows Using RDMA. In Proc. ACM SIGCOMM Posters and Demos, 2017.

[36] J. Hu, J. Huang, W. Lv, Y. Zhou, J. Wang and T. He. CAPS: Coding-based Adaptive Packet Spraying to Reduce Flow Completion Time in Data Center. In Proc. IEEE INFOCOM, 2018.

[37] A. Elwalid, C. Jin, S. Low, I. Widjaja. MATE: MPLS Adaptive Traffic Engineering. In Proc. IEEE INFOCOM, 2001.

[38] J. Hu, J. Huang, W. Lv, Y. Zhou, J. Wang and T. He. CAPS: Coding-based Adaptive Packet Spraying to Reduce Flow Completion Time in Data Center. IEEE/ACM Transactions on Networking, 27(6): 2338-2353, 2019.

[39] N. Katta, A. Ghag, M. Hira, I. Keslassy, A. Bergman, C. Kim, and J. Rexford. Clove: Congestion-Aware Load Balancing at the Virtual Edges.

In Proc. ACM CoNEXT, 2017.

[40] A. Kabbani, B. Vamanan, J. Hasan, and F. Duchene. FlowBender: Flow-level Adaptive Routing for Improved Latency and Throughput in Datacenter Networks. In Proc. ACM CoNEXT, 2014.

[41] E. Vanini, R. Pan, M. Alizadeh, P. Taheri and T. Edsall. Let It Flow: Resilient Asymmetric Load Balancing with Flowlet Switching. In Proc. USENIX NSDI, 2017.

[42] J. Hu, J. Huang, W. Lv, W. Li, J. Wang and T. He. TLB: Traffic-aware Load Balancing with Adaptive Granularity in Data Center Networks. In Proc. ACM ICPP, 2019.

[43] K. He, E. Rozner, K. Agarwal, W. Felter, J. Carter and A. Akellay. Presto: Edge-based Load Balancing for Fast Datacenter Networks. In Proc. ACM SIGCOMM, 2015.

[44] A. Dixit, P. Prakash, Y. C. Hu, and R. R. Kompella. On the Impact of Packet Spraying in Data Center Networks. In Proc. of IEEE INFOCOM, 2013.

[45] J. Hu, J. Huang, W. Lv, W. Li, Z. Li, W. Jiang, J. Wang and T. He. Adjusting Switching Granularity of Load Balancing for Heterogeneous Datacenter Traffic. IEEE/ACM Transactions on Networking, 29(5): 2367-2384, 2021.

[46] M. Al-Fares, S. Radhakrishnan, B. Raghavan, N. Huang, and A. Vahdat. Hedera: Dynamic Flow Scheduling for Data Center Networks. In Proc. USENIX NSDI, 2010.

[47] T. Benson, A. Anand, A. Akella, and M. Zhang. MicroTE: Fine Grained Traffic Engineering for Data Centers. In Proc. ACM CoNEXT, 2011.