This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

SRCC: Sub-RTT Congestion Control for Lossless Datacenter Networks

Jinbin Hu, *Member, IEEE*, Shuying Rao, Min Zhu, Jiawei Huang, Jianxin Wang, *Senior Member, IEEE*, and Jin Wang, *Senior Member, IEEE*

***Abstract*—To meet the stringent requirements of industrial applications, modern Ethernet datacenter networks widely deployed with remote direct memory access (RDMA) technology and priority-based flow control (PFC) scheme aim at providing low latency and high throughput transmission performance. However, the existing end-to-end congestion control cannot handle the transient congestion timely due to the round-trip-time (RTT) level control loop, inevitably resulting in PFC triggering. In this article, we propose a Sub-RTT congestion control mechanism called SRCC to alleviate bursty congestion timely. Specifically, SRCC identifies the congested flows accurately, notifies congestion directly from the hotspot to the corresponding source at the sub-RTT control loop and adjusts the sending rate to avoid PFC's head-of-line blocking. Compared to the state-of-the-art end-to-end transmission protocols, the evaluation results show that SRCC effectively reduces the average flow completion time (FCT) by up to 61%, 52%, 40%, and 24% over datacenter quantized congestion notification (DCQCN), Swift, high precision congestion control (HPCC), and photonic congestion notification (PCN), respectively.**

***Index Terms*—Congestion control, datacenter networks (DCNs), industrial applications, priority-based flow control (PFC), sub-round-trip-time (RTT).**

Received 22 November 2023; revised 8 February 2024 and 1 September 2024; accepted 29 October 2024. This work was supported in part by the National Natural Science Foundation of China under Grant 62473146, Grant 62472050, Grant 62102046, and Grant 62072056, in part by the Key Project of Natural Science Foundation of Hunan Province under Grant 2024JJ3017, in part by the Science and Technology Project of Hunan Provincial Department of Water Resources under Grant XSKJ2024064-36, and in part by New Generation Information Technology Innovation Project 2023 under Grant 2023IT271. Paper no. TII-23-4672. *(Corresponding author: Jin Wang.)*

Jinbin Hu and Shuying Rao are with the School of Computer and Communication Engineering, Changsha University of Science and Technology, Changsha 410114, China (e-mail: jinbinhu@csust.edu.cn; shuyingrao@stu.csust.edu.cn).

Min Zhu is with the College of Information Science and Technology, Zhejiang Shuren University, Hangzhou 310009, China (e-mail: zhumin@zjsru.edu.cn).

Jiawei Huang and Jianxin Wang are with Central South University, Changsha 410017, China (e-mail: jiaweihuang@csu.edu.cn; jxwang@csu.edu.cn).

Jin Wang is with the Sanya Institute, Hunan University of Science and Technology, Sanya 572024, China (e-mail: jinwang@hnust.edu.cn).

Color versions of one or more figures in this article are available at https://doi.org/10.1109/TII.2024.3495759.

Digital Object Identifier 10.1109/TII.2024.3495759

I. Introduction

WITH the continuous growth of modern industry, Industry 5.0 has been recognized as the next generation of industrialization, bringing about an increasing number of real time and critical applications, such as robot control, real-time monitoring, and predictive maintenance [1], [2], [3]. These emerging computing-intensive and data-oriented applications demand the increasing ultra-low latency and high-throughput [4], [5], [6], [7], [8], [9], [10], [11]. To meet these stringent requirements, remote direct memory access (RDMA) technology is widely deployed with RDMA over converged ethernet v2 protocol in the Ethernet datacenter networks (DCNs), which deploy the hop-by-hop priority-based flow control (PFC) to provide lossless reliable transmission [13], [14], [15], [16].

For the PFC mechanism [17], once the ingress queue length exceeds a specified PFC threshold, a PFC PAUSE frame is generated and sent to the upstream switches to pause the corresponding port or priority queue. The transmission is resumed when receiving a PFC RESUME frame or the PFC pause duration expiring [18]. Since the coarse-grained PFC pausing is performed on a port or a priority queue, it cannot distinguish between flows, resulting in serious head-of-line (HoL) blocking and congestion spreading, even tearing down the whole networks. Specifically, when an egress port is paused due to PFC PAUSE, in addition to the congested flows that are really responsible for the congestion, the victim flows that are not responsible for congestion are unreasonably blocked, this is the well-known HoL blocking problem [5], [11].

Recently, many excellent end-to-end transport protocols based on different congestion signals focus on end-to-end transmission control to alleviate congestion and reduce PFC triggering. End-to-end congestion control mechanisms, such as datacenter quantized congestion notification (DCQCN), transport informed by measurement of latency (TIMELY), Swift, high precision congestion control (HPCC), and photonic congestion notification (PCN) require at least one round-trip time (RTT) to effectively control congestion. However, as revealed by [19], approximately 60% to 90% of the short-lived flows can be completed in less than one RTT in the high-speed DCNs. From this, it can be seen that although end-to-end congestion control mechanisms perform well in controlling long-lived flows and long-term congestion, they are not sufficient to control short-lived flows and transient congestion. Therefore, despite significant efforts on end-to-end congestion control, PFC is inevitably triggered especially under bursty traffic scenarios.

In this article, we propose a sub-RTT congestion control mechanism named SRCC to alleviate congestion timely and

1551-3203 © 2024 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission.
See https://www.ieee.org/publications/rights/index.html for more information.

Authorized licensed use limited to: CHANGSHA UNIV OF SCIENCE AND TECHNOLOGY. Downloaded on January 13,2025 at 07:33:00 UTC from IEEE Xplore. Restrictions apply.

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

avoid PFC triggering. Specifically, SRCC identifies congested flows and calculates the congestion threshold dynamically based on the traffic arrival strength at the ingress port, the derivative of the egress queue length and the number of hops required for the congestion information to back to the end-hosts. Then, at the switches, once the egress queue length exceeds the congestion threshold, the congestion notification message (CNM) is directly sent back to the sender of the corresponding congested flow at the sub-RTT level delay for accurate sending rate adjustment.

In summary, our major contributions are as follows.

1) We conduct experiments to explore the key issues that lead to HoL blocking and congestion spreading under bursty scenarios: 1) PFC is inevitably triggered because the end-to-end congestion control cannot handle transient congestion timely. 2) The coarse-grained PFC mechanism cannot distinguish congested flows that really responsible for congestion, resulting in uncongested flows (i.e., victim flows) being blocked.
2) We propose SRCC, which identifies the congested flows and notifies the congestion directly from the switch to the source end-hosts once the queue length exceeds the congestion threshold, thereby effectively avoiding PFC triggering within the sub RTT delay. In order to prevent the victim flows suffering from HoL blocking problem, SRCC also modifies the sending rate of the congested flows which block them.
3) We conduct testbed experiments and NS-3 simulations with various realistic datacenter workloads to test the effectiveness of SRCC. The results show that SRCC successfully avoids PFC triggering and HoL blocking, and effectively reduces the average FCT by up to 61%, 52%, 40%, and 24% over DCQCN, Swift, HPCC, and PCN, respectively.

The rest of this article is organized as follows. Section II introduces the background and motivation. Section III describes the design overview and details, respectively. Section IV discusses the implementation. Section V shows testbed experiments and NS-3 simulations. Sections VI and VII present the discussion on research and the related works, respectively. Finally, Section VIII concludes this article.

II. Background and Motivation

A. Background

1) Traffic Characteristics in Datacenters: The demand for high-speed data transmission between servers in DCNs is significantly increasing with the growing trend towards cloud computing and big data analysis. The link bandwidth in datacenters rapidly increases from 100 to 400 G and even continues to grow. Therefore, the microburst workloads with highly dynamic initiation and completion are hard to predict, more and more tiny flows can be finished within a few RTTs, as revealed by the recent measurement schemes [19]. However, the end-to-end transmission protocols have at least one RTT control loop, during which the packets of tiny flows have been sent. It is precisely because of this slow reaction to microbursts that the bulk of microbursts are the main culprit of PFC triggering in lossless datacenters, resulting in unexpected HoL blocking and congestion spreading.

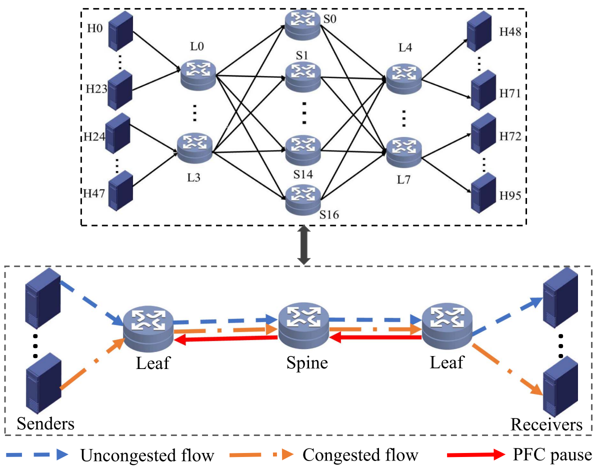


Fig. 1. HoL blocking.

2) PFC Mechanism and Issues: PFC is widely deployed in Ethernet-based lossless datacenters to prevent buffer overflow due to congestion. Specifically, once the ingress queue length exceeds the preset threshold, the switch sends PFC PAUSE frames to the corresponding upstream egress port or queue to stop data transmission temporarily until the queue drains to a certain threshold before resuming data transmission, thus guaranteeing lossless transmission. Although PFC can ensure the reliability of network transmission, its adverse effects of HoL congestion and congestion diffusion cannot be ignored. As shown in Fig. 1, the HoL blocking occurs when the congested flows that are really responsible for congestion and the uncongested flows that are unrelated to congestion share the same egress port, which is paused by PFC PAUSE frame. That is, the uncongested flows called victim flows are innocently blocked, leading to large tail delay. Even worse, PFC pausing is transmitted to the source in reverse hop by hop, resulting in a PFC storm that paralyzes the entire network.

3) End-to-End Congestion Control Mechanism in Lossless DCN: Based on network characteristics, in order to alleviate congestion and the negative impacts due to PFC pausing, several end-to-end congestion control schemes, such as DCQCN [11], HPCC [10], Swift [8], and PCN [5], have been proposed in recent years. DCQCN [11] detects congestion and adjusts the sending rate step by step based on the explicit congestion notification (ECN) signals with at least one RTT delay. HPCC [10] leverages in-network telemetry (INT) technology to obtain accurate network status information such as queue length and link load to control congestion in each update period. Swift [8] performs rate adjustment based on the end-to-end RTT signal considering the processing delay at the end-hosts. PCN [5] identifies congested flows based on ECN and only throttles their sending rate to prevent uncongested flows from being blocked. Although the abovementioned end-to-end transport protocols can effectively control congestion caused by long flows, they cannot timely react to transient congestion caused by bursty traffic.

B. Motivation

To more intuitively observe the impact of transient congestion caused by bursty traffic on lossless network performance, we conduct a series of NS-3 simulation tests. We use a multilevel

Authorized licensed use limited to: CHANGSHA UNIV OF SCIENCE AND TECHNOLOGY. Downloaded on January 13,2025 at 07:33:00 UTC from IEEE Xplore. Restrictions apply.

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

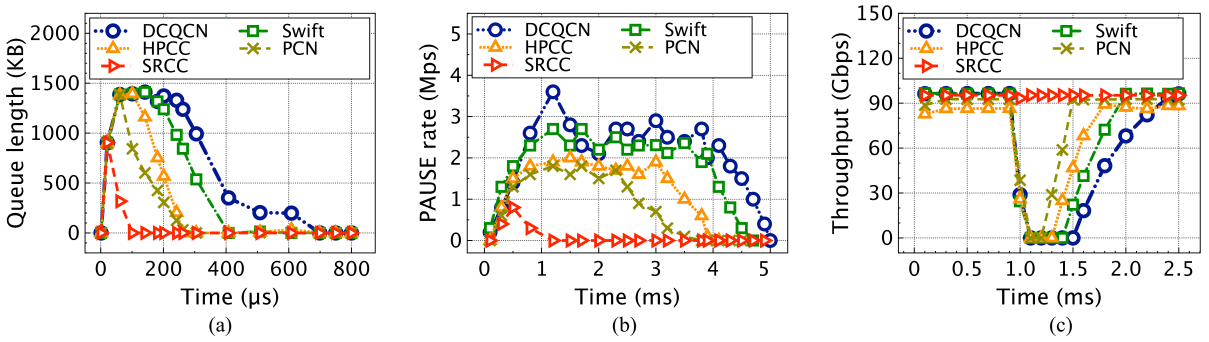


Fig. 2. Performance under different congestion control mechanisms. (a) Queue length. (b) PAUSE rate. (c) Throughput.

leaf-spine topology with 4 leaf and 8 core switches, where each leaf switch connects 16 end-hosts. The link bandwidth is 100 Gb/s, and the default latency for each link is 4 μs. To simulate traffic burst scenarios, we adopt a mixed traffic model of concurrent long-lived flows and burst mice flows. Specifically, we enable the senders H0–H23 to, respectively, send long-lived flows to the receivers H95. Meanwhile, H24–H47, respectively, send burst mice flows of 100 kb in size to the receiver H95, with a duration of 10 ms. The duration of each burst mice flow is less than one RTT, which is beyond the control of end-to-end congestion control mechanisms. We compare the performance of existing end-to-end congestion control mechanisms, including DCQCN, Swift, HPCC, and PCN under the burst scenarios. The test results are shown in Fig. 2.

As shown in Fig. 2(a), during a sharp increase in traffic, the queue length increases rapidly. In this process, because the end-to-end congestion control mechanism requires feedback duration and multiple RTTs for the rate evolution, PFC is triggered due to queue building up. When the PFC is triggered, the upstream switch will stop sending packets, and the queue at the downstream switch drains gradually. During the entire process from PFC triggering to congestion disappearance, we can find that PCN outperforms the other coarse-grained control schemes by distinguishing congested and uncongested flows to alleviate HoL blocking. However, PCN with the end-to-end congestion control still suffers from severe PFC pausing and throughput degradation.

We also test the PAUSE rate (the rate of transmitting PAUSE messages) of the path where the congested flows are transmitted. As shown in Fig. 2(b), the pausing duration of the different congestion control mechanisms varies, mainly distributed within 4–5 ms. PCN with the advantage of distinguishing congested flows and victim flows has the shortest duration of 3.5 ms (>17 RTT). DCQCN has the longest duration of 5 ms (>25R TT). This is because ECN marking needs to be processed in network, and its update speed depends on the processing speed of the slowest node in the network, which will result in extremely high latency. In short, under bursty traffic scenarios, PFC is inevitably triggered, resulting in the victim flows to be blocked.

To better demonstrate the negative impact of PFC mechanism on network performance, we also test the throughput of victim flows. In Fig. 2(c), it can be seen that before triggering PFC, the throughput and link bandwidth of the four congestion control mechanisms are almost the same, all maintained at 80%–100% of the link bandwidth. At 0.9 ms, they encounter PFC pausing, the throughput drops sharply and even close to 0. This is because the end-to-end mechanism cannot timely control transient congestion caused by burst flows. Once the congestion continues to spread, leading to PFC triggering. When the congested and uncongested flows share the same egress port, the victim flows are innocently blocked due to PFC pausing, resulting in a decrease in throughput. The root cause of the aforementioned problems is that the congestion feedback loop delay of the existing end-to-end transmission protocols is too large to control microbursts.

C. Summary

Based on experimental observations, we have come to the following conclusions: 1) The end-to-end congestion control mechanisms with multiple RTTs feedback delay cannot timely control the transient congestion caused by microbursts, potentially leading to the triggering of PFC. 2) The port or queue-based PFC cannot distinguish congested flows and victim flows, it causes serious congestion spreading and HoL blocking, resulting in great throughput loss and large flow completion time. To solve these problems, we need to explore an efficient congestion control mechanism for non-HoL-blocking lossless DCNs.

III. SRCC Design

A. SRCC Overview

SRCC is a sub-RTT congestion control framework, including congestion and notification points at the switches and reaction points at the source end-hosts. The key point of SRCC is to directly feed back congestion information from the hotspots to the corresponding sources. Compared with the end-to-end congestion control mechanisms, SRCC can react to congestion timely and avoid PFC triggering effectively within the sub-RTT delay, especially under the bursty traffic scenarios. The reason is that the end-to-end solutions cannot control the short-lived bursty flows in time and then PFC is inevitably triggered, while SRCC can adjust the sending rate to the target rate quickly to prevent from PFC triggering. As shown in Fig. 3, the framework of SRCC consists of the following four aspects.

1) Identify congested flows based on congestion thresholds and further identify congested flows that share the ingress port with uncongested flows.
2) Calculating congestion threshold dynamically based on the traffic strength at the ingress port, the derivative of egress queue length and the time to live (TTL) value of the congested flows.

Authorized licensed use limited to: CHANGSHA UNIV OF SCIENCE AND TECHNOLOGY. Downloaded on January 13,2025 at 07:33:00 UTC from IEEE Xplore. Restrictions apply.

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

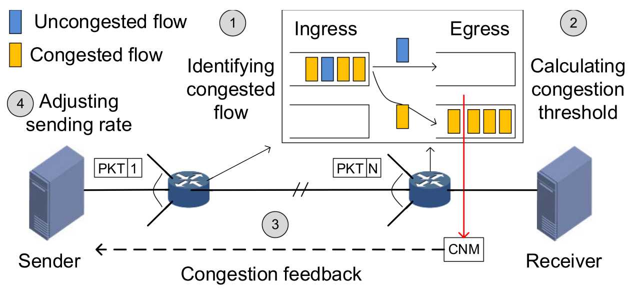


Fig. 3. Overview of SRCC.

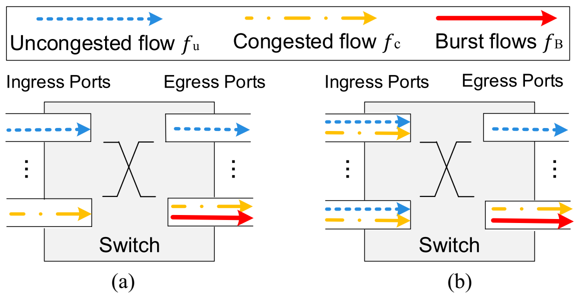


Fig. 4. Identifying the real congested flow. (a) f_u and f_c do not share ingress ports. (b) f_u and f_c share ingress ports.

3) Feeding back congestion directly from the switches to the end-hosts by leveraging CNM to significantly reducing control loop delays, after identifying congested flows.
4) Adjusting the sending rate of congested flows sharing the ingress port with uncongested flows to the target rate based on the congestion notification from the switch to avoid triggering PFC.

B. Identifying Congested Flows

The primary goal of SRCC is to prevent the victim flows suffering from HoL blocking problem. Thus, one challenge of SRCC is to identify the congested flows that potentially block the uncongested flows. First, the flows sent to the congested egress port whose queue length exceeds the congestion threshold Q_{cth} are considered as the congested flows. The threshold Q_{cth} is calculated dynamically in (see Section III-C). Then, SRCC further identifies which congested flows share the ingress port with uncongested flows.

Specifically, at the switch, SRCC records the arrival time t_{uf} of the last uncongested flow's packet for each ingress port. If the time interval between the current arrival time t_{cur} of the congested flow's packet and the t_{uf} is less than a specified time window T, it is considered that the congested flow is not share ingress port with uncongested flows. Otherwise, it means that the congested flow shares ingress port with victim flows. SRCC only performs rate adjustment on this congested flow that potentially affects the uncongested flows. For example, in Fig. 3, SRCC identifies the yellow flow as the congested flow that shares ingress port with the blue uncongested flow.

Notably, although PFC is triggered at the ingress port that only holds the congested flows, there are no victim flows blocked and no HoL blocking occurs. In this case, there is no need to quickly reduce the sending rate of the congested flow. As shown in Fig. 4(a), the uncongested flow f_u exclusively occupies the ingress and egress port, which is independent of the congested ports and, therefore, is not suffering from HoL blocking even if the congested flow f_c triggers PFC pausing. In Fig. 4(b), the uncongested flow f_u shares the ingress port with the congested flow f_c, which is potentially blocked due to PFC pausing. In this case, SRCC will identified f_c as the real culprit of congested flow and directly control the sending rate of f_c.

C. Calculating Congestion Threshold

Another challenge of SRCC is when to trigger congestion feedback, that is, the calculation of congestion threshold Q_{cth}. Inspired by [20] and [21], we establish a random flow model. We theoretically analyze the value of Q_{cth} under the worst congestion scenario, where m flows with the arrival rate of $\lambda_i(t)$ are sent to M egress port at time t at the switch. The link capacity and base delay are C and d, respectively. To avoid PFC being triggered at the ingress queue, the egress queue length Q_t at time t should be less than $\frac{Q_{\text{PFC}}}{M}$, where Q_{PFC} is the PFC triggering threshold.

We assume that congestion feedback is triggered at time t_c when the egress queue length reaches Q_{t_c} at switches. The value of TTL N_{TTL} between the current switch to the end-host of congested flow is obtained from the data packet header. Within the congestion feedback duration T_c (i.e., $N_{\text{TTL}} \times d$), the queue is still building up. Specifically, there are n congested flows out of m flows. The traffic of n congested flows is

$$\text{CT} = \sum_{i=1}^{n} \int_{t_c}^{t_c+T_c} \lambda_i(t)dt. \tag{1}$$

The traffic of m–n uncongested flows is

$$\text{UCT} = \sum_{i=1}^{m-n} \int_{t_c}^{t_c+T_c} \lambda_i(t)dt. \tag{2}$$

When the congestion feedback takes effect at the end-hosts, the maximum egress queue length is

$$\begin{aligned} Q(t_c+T_c) &= Q_{\text{cth}} + \text{CT} + \text{UCT} - N_{\text{TTL}} \times d \times C \\ &= Q_{\text{cth}} + \sum_{i=1}^{m} \int_{t_c}^{t_c+T_c} \lambda_i(t)dt - N_{\text{TTL}} \times d \times C. \end{aligned} \tag{3}$$

To ensure that PFC is not triggered, the following conditions need to be satisfied:

$$Q(t_c+T_c) < \frac{Q_{\text{PFC}}}{M}. \tag{4}$$

Therefore, based on (3), the congestion threshold Q_{cth} should meet the following condition:

$$Q_{\text{cth}} < \frac{Q_{\text{PFC}}}{M} - \sum_{i=1}^{m} \int_{t_c}^{t_c+T_c} \lambda_i(t)dt + N_{\text{TTL}} \times d \times C. \tag{5}$$

To meet the requirements of not triggering PFC and ensuring high throughput simultaneously, SRCC sets a conservative value of Q_{cth} as

$$Q_{\text{cth}} = \frac{Q_{\text{PFC}}}{M} - N_{\text{TTL}} \times d \times C \times (m-1). \tag{6}$$

D. Congestion Feedback

SRCC employs CNMs to provide direct congestion feedback from the switch to the end-hosts, significantly reducing control

Authorized licensed use limited to: CHANGSHA UNIV OF SCIENCE AND TECHNOLOGY. Downloaded on January 13,2025 at 07:33:00 UTC from IEEE Xplore. Restrictions apply.

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

loop delays. After identifying the congested flows, once the egress queue length exceeds the calculated congestion threshold, a CNM is generated and directly fed back to the corresponding source end-host. The format of CNM packet is consistent with the congestion signal in the existing quantized congestion notification (QCN) mechanism [22], which is commonly supported in commodity switches [23].

However, another challenge is how to send CNM hop-by-hop to the source end-hosts in the internet protocol (IP) routed networks, since the original CNM in QCN does not preserve the source IP header and is forwarded based on media access control (MAC) address. SRCC records the flow identification (ID) and the corresponding source MAC address of the last hop in the flow table at each switch, and then CNM can be sent hop-by-hop according to the MAC address by looking up the flow table.

In details, the main field information encapsulated by CNM includes 48-b source MAC, 48-b destination MAC, 8-b congested flow ID, and 32-bits target sending rate. The target rate is calculated as the average rate of the congested flows at the congested egress port and normalized to the bottleneck bandwidth. Thus, SRCC utilizes CNM carrying the congested flow information to directly signal congestion from switches to the corresponding sender.

E. Adjusting Sending Rate

The sending rate adjustment is critical for congestion control. SRCC seeks to circumvent PFC triggering by adjusting the sending rate to the target rate quickly. Specifically, SRCC resets the target sending rate once receiving CNM at the sender to control congestion timely instead of the slow evolution-based rate adjustment like the end-to-end transmission protocols. The target rate is derived by normalizing the bottleneck link bandwidth based on the average rate of flows shared the same egress port. Thus, SRCC is able to accurately control congestion especially for bursty scenarios.

Please note that, if a congested flow's sender receives multiple CNMs from different switches, the sending rate is set to the minimal bottleneck target rate to avoid PFC triggering. In addition, SRCC can be friendly compatible with the end-to-end congestion control to alleviate long-term congestion when the egress queue length does not reach the congestion threshold Q_{cth} set by SRCC.

IV. Implementation

We implement SRCC prototype on programming protocol-independent packet processors (P4) hardware switch and end-hosts employed data plane development kit network card.

Switch implementation: We implement the congested flow ID and congestion notification of SRCC in the programmable switch Wedge 100BF-32X. The pipeline processing process of data packets on the data plane is described in P4 language. Notably, a pipeline supports up to 12 data processing units. The processing unit abstracts the data processing process into a process of matching and executing a match-action table. Multiple match-action tables without dependencies are arranged into on data processing unit, and vice versa, between different data processing units. The match-action tables within a data processing unit can be executed in parallel, but the match-action

TABLE I
Flow Size Distribution of Realistic Workloads

	Web server	Cache follower	Web search	Hadoop
0–100 KB	81%	53%	62%	60%
100 KB–1 MB	19%	18%	18%	38%
$>$ 1 MB	0	29%	20%	2%
Average flow size	64 KB	701 KB	1.6 MB	1.04 MB

tables from different data processing units can only be executed in series. The most challenging task of implementing SRCC is to access the hardware resources exclusively in the pipeline.

Specifically, the primary match-action tables of SRCC at the switch includes reading queue length from static random access memory, matching congested port and confirming congested flows correspondingly, then taking different actions such as generating and forwarding congestion notifications in the next stage. In the congested flow table, each entry consists of a 48-b MAC address and a 16-b flow ID, which is calculated by using a cyclic redundancy check (CRC) function (CRC16) based on the five tuples of the packet header.

Host implementation: We implement the rate adjustment at the end-hosts based on DPDK technology. At the sender, the application layer starts data transmission by calling the send() function. The sending rate is adjusted accordingly based on the received CNM or CNP, and then the data packets are transmitted to the network card by calling `rte_eth_tx_buffer_flush()` function. At the receiver, data packets are received from the network card buffer by calling the `rte_eth_rx_burst()` function and forwarded to the application layer by calling receive() function.

V. Performance Evaluation

We conduct testbed experiments and NS-3 simulations to evaluate the performance of SRCC compared with the state-of-the-art end-to-end congestion control schemes. Our evaluation seeks to answer the following questions.

1) *How does SRCC perform in practice?* Testbed experiments (see Section V-B) show that SRCC effectively controls congestion and suppresses PFC triggering. SRCC reduces the average FCT by up to 61% and cuts the 99th percentile FCT by up to 72% with varying load under realistic workloads compared with the state-of-the-art end-to-end congestion control schemes.
2) *How does SRCC perform in large-scale DCNs?* Large-scale NS-3 simulations (see Section V-C) demonstrate that SRCC significantly reduces the PAUSE rate by up to 91% and 94% under Web Search and Hadoop workloads, respectively. SRCC still achieves the lowest average FCT and 99th percentile FCT under a wide range of traffic load, and obtains the highest goodput under Incast scenario.

A. Evaluation Setup

Choices of baseline: We choose four end-to-end congestion control solutions, such as DCQCN [11], HPCC [10], Swift [8], and PCN [5], as the baseline transport protocols. DCQCN,

Authorized licensed use limited to: CHANGSHA UNIV OF SCIENCE AND TECHNOLOGY. Downloaded on January 13,2025 at 07:33:00 UTC from IEEE Xplore. Restrictions apply.

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

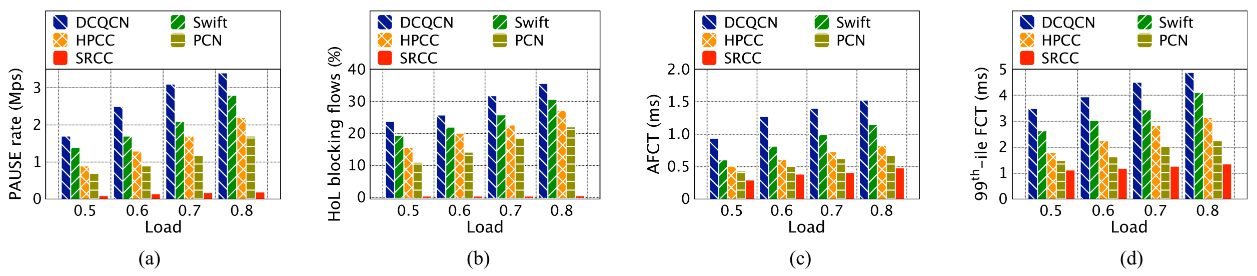


Fig. 5. Performance under web server workload. (a) PAUSE rate. (b) Ratio of HoL blocking flows. (c) AFCT. (d) 99th-ile FCT.

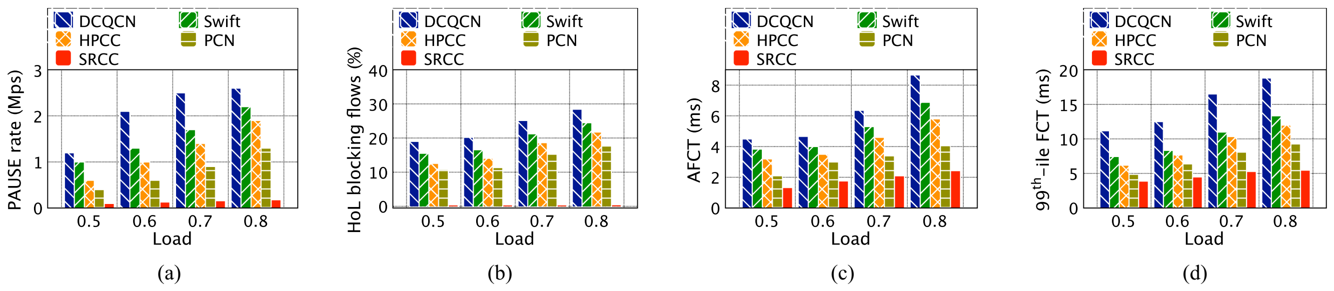


Fig. 6. Performance under cache follower workload. (a) PAUSE rate. (b) Ratio of HoL blocking flows. (c) AFCT. (d) 99th-ile FCT.

HPCC, and Swift are deployed in Microsoft DCNs, Alibaba DCNs, and Google DCNs, respectively.

Testbed setup: In the testbed evaluations, we use the same leaf-spine topology adopted in [5], which consists of 16 end-hosts (Dell PRECISION TOWER 5820 desktop) connected to 2 leaf switches (Wedge 100BF-32X hardware programmable switch) by using 100 Gb/s links. Each Dell desktop is equipped with 10 cores Intel Xeon W-2255 CPU, 64 GB memory, Ubuntu 20.04.1 (Linux version 5.4.0-42-generic), Mellanox ConnectX-5 100GbE network interface controllers (NICs) supported DPDK. Each switch enabled PFC at the ingress ports has 22 MB shared buffer and 32 full duplex 100 Gb/s ports.

Realistic workloads: We generate four typical realistic workloads including web server, cache follower, web search, and Hadoop [19]. The specific flow distribution is shown in Table I. The flows in these workloads are generated following a Poisson process between random pairs of senders and receivers, and the traffic load varies from 0.5 to 0.8.

B. Testbed Experiments

1) Basic Performance: We first conduct testbed experiments to evaluate the effectiveness of SRCC, which is implemented on P4 hardware switch [24] and end-hosts employed DPDK 20.08. We choose web server and cache follower realistic workloads, the distribution are shown in Table I.

Figs. 5 and 6 show the PAUSE rate, ratio of HoL blocking flows, average FCT and 99th percentile FCT under web server and cache follower workloads, respectively. As shown in Figs. 5(a), (b) and 6(a), (b), SRCC effectively reduces the PFC PAUSE rate with varying traffic load and prevents the uncongested flows from HoL blocking due to PFC pausing. The reason is that the sub-RTT level SRCC feeds back the congestion message directly from the switch to the corresponding sender for rate adjustment with less than one RTT control loop. For the other end-to-end transmission protocols, although they alleviate congestion by adjusting sending rate, they cannot avoid PFC triggering under the transient congestion due to requiring at least one RTT control loop. In addition, the ratio of blocking flows in web server with more bursty traffic is higher than that in cache follower. This is due to the fact that the burst traffic in web server is stronger and more tiny flows than in cache follower. The uncontrollable bursty tiny flows lead to frequent PFC triggering, resulting in severe congestion spreading and HoL blocking flows in the end-to-end congestion control solutions.

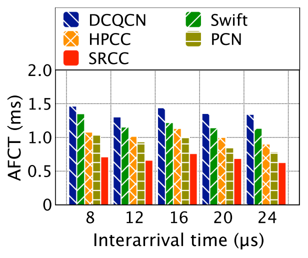
Figs. 5(c) and 6(c) show that SRCC achieves the lowest average FCT under both realistic workloads. For example, in web server application under 0.8 traffic load, SRCC reduces the average FCT by up to 61%, 52%, 40%, and 24% compared with DCQCN, Swift, HPCC, and PCN, respectively. The reason is that SRCC successfully avoids PFC triggering and resets the target sending rate once receiving congestion message without convergence evolution. While the other end-to-end transport protocols adjust the sending rate step by step according to the RTT-level congestion feedback. PCN employs the receiver-driven rate adjustment method and then obtains better latency performance compared with the other schemes.

As shown in Figs. 5(d) and 6(d), SRCC effectively cuts the 99th percentile tail FCT with the increasing traffic load. Specifically, in cache follower application under 0.8 traffic load, SRCC reduces the 99th percentile FCT by up to 72%, 58%, 42%, and 37% over DCQCN, Swift, HPCC, and PCN, respectively. This is because SRCC can rapidly detect connected flows that shared the ingress port with victim flows and regulates the rate of congested flows before triggering PFC, thus, there are no victim flows experience HoL blocking. Compared with the end-to-end congestion control schemes, SRCC working at the sub-RTT granularity improves the overall FCT by timely control the bursty congestion.

2) Performance Under Various Bursty Strength: To further evaluate the effectiveness of SRCC under the bursty scenario with different traffic intensities, we conduct experiments by

Authorized licensed use limited to: CHANGSHA UNIV OF SCIENCE AND TECHNOLOGY. Downloaded on January 13,2025 at 07:33:00 UTC from IEEE Xplore. Restrictions apply.

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.


(a)

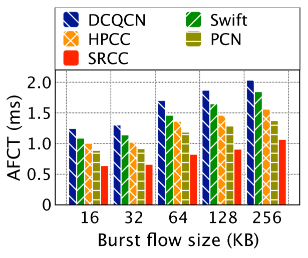
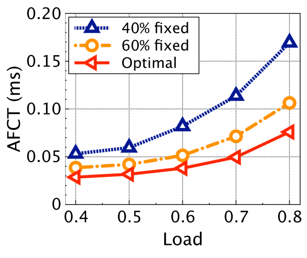

(b)

Fig. 7. AFCT with varying burst intensities. (a) Varying burst interarrival time. (b) Varying burst flow size.


(a)

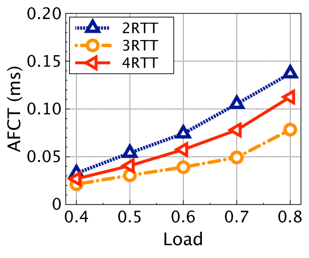

(b)

Fig. 8. Sensitivity to parameters. (a) Impact of congestion threshold. (b) Impact of time window.

changing the burst interarrival time and burst flow size, respectively. In the aforementioned testbed topology, 10 senders simultaneously transmit burst flows to a receiver, which together with the traffic sent by other end-hosts form the web server workload. By default, the burst flows are sent every 16 μs interval and each flow is 64 KB. In the first case, the burst traffic interarrival time varies from 8 to 24 μs. In the second scenario, the flow size is increased from 16 to 256 KB. We measure the average FCT shown in Fig. 7.

In Fig. 7(a), as the arrival interval of burst traffic decreases, the burst intensity increases, leading to an increase in average FCT for all transmission mechanisms. However, compared with the other three end-to-end congestion control schemes, SRCC is able to directly notify the detected congestion from the hotspot to the corresponding source and accurately adjust the rate of the identified congested flows. This effectively alleviates transient congestion and queue building up, avoids PFC triggering, and the uncongested flows do not suffering from HoL blocking. For example, at the interval time of 20 μs, SRCC reduces AFCT by up to 49% compared with other schemes.

As shown in Fig. 7(b), when the burst flow size increases, the transient congestion duration also increases, resulting in an increase in average FCT across all congestion control solutions. However, as the flows become longer, they are more tolerant of congestion control loop delay, resulting in better congestion control over long flows for all transmission protocols. Specifically, when the burst flow sizes are 32 KB and 128 KB, SRCC enhances the latency performance by 49%, 42%, 35%, 28% and 51%, 45%, 38%, 29% compared to DCQCN, Swift, HPCC, and PCN, respectively. Among them, SRCC obtains the lowest average FCT.

3) Parameter Sensitivity: We further conduct simulations to explore the impact of the different parameters on the flow completion time of SRCC. Specifically, we test different congestion feedback thresholds and the time windows for identifying congested flows in the above large-scale leaf-spine topology with web server workload at 0.8 traffic load.

Fig. 8(a) shows the average FCT of SRCC under the dynamic optimal congestion feedback threshold and the fixed congestion feedback thresholds including 40% fixed buffer size and 60% fixed buffer size. With small fixed congestion feedback threshold, PFC is difficult to be triggered, but the sending rate is adjusted frequently, resulting high FCT. On the contrary, with large fixed congestion feedback threshold, PFC is easy to be triggered especially under high traffic load, resulting in serious HoL blocking. The test results show that SRCC achieves the lowest FCT under the optimal threshold calculated dynamically due to timely congestion control.

In Fig. 8(b), the time window is set to 2RTT, 3RTT, and 4RTT under varying traffic load. Under small time window, the congested flow is potentially mistaken as not sharing ingress port with the uncongested flows. However, in fact, the uncongested flows have not finished yet, and the congested flow will not be notified to throttle rate, resulting in HoL blocking. In contrary, with large time window, the uncongested flows that have finished are also possible to be mistaken as that has not completed. Thus, the congested flows that is mistaken for sharing the ingress port with uncongested flows decrease sending rate aggressively, resulting in increased flow completion time. As shown in Fig. 8(b), when the time window is set to a reasonable empirical value of 3RTT, SRCC obtains the lowest FCT.

C. Large-Scale NS-3 Simulations

1) Performance Under Realistic Workloads: We further conduct large-scale NS-3 simulations to evaluate the performance of SRCC under the typical datacenter applications. We choose Web Search and Hadoop workloads as shown in Table I. We use a two-level leaf-spine topology with 8 leaf and 16 core switches in the large-scale simulations. Each leaf switch connects to 24 end-hosts. The default delay of each link is 4 μ s and the link bandwidth is 100 Gb/s.

As described in Fig. 2 (see Section II-B), SRCC can effectively control the queue building up, reduce PAUSE frame and avoid HoL blocking, resulting in no throughput degradation compared to other schemes. Figs. 9 and 10 show the queue length, PAUSE rate, average FCT, and 99th percentile FCT under Web Search and Hadoop workloads, respectively. As shown in Figs. 9(a) and 10(a), the convergence speed of SRCC is far higher than other congestion control mechanisms after the PFC is triggered under bursty traffic. The reason is that SRCC has two major design advantages: timely congestion feedback with less than one RTT and timely rate adjustment for congested flows. This makes the queue drain quickly and allows SRCC to maintain the stable low queue length even under transient congestion. For other congestion control mechanisms, they cannot alleviate congestion by regulating rate in a timely and accurate manner simultaneously due to the congestion feedback requiring multiple RTTs. For example, compared to DCQCN, Swift, HPCC, and PCN, SRCC reduces convergence time by up to 91%, 87%, 78%, and 58% under 0.8 traffic load in web search, respectively.

Authorized licensed use limited to: CHANGSHA UNIV OF SCIENCE AND TECHNOLOGY. Downloaded on January 13,2025 at 07:33:00 UTC from IEEE Xplore. Restrictions apply.

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

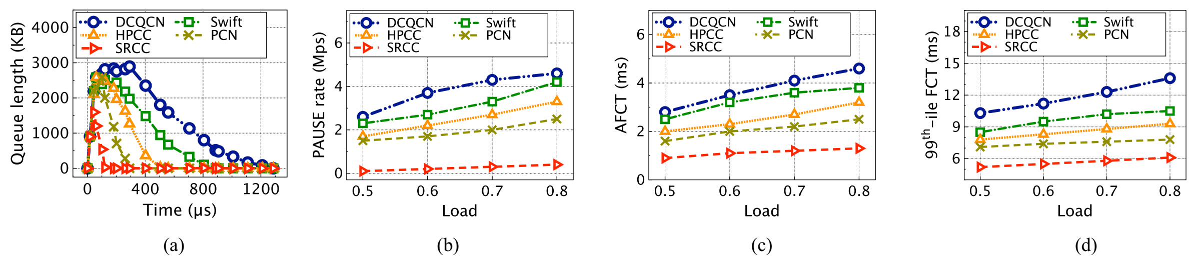


Fig. 9. Performance under web search workload. (a) Queue length. (b) PAUSE rate. (c) AFCT. (d) 99th-ile FCT.

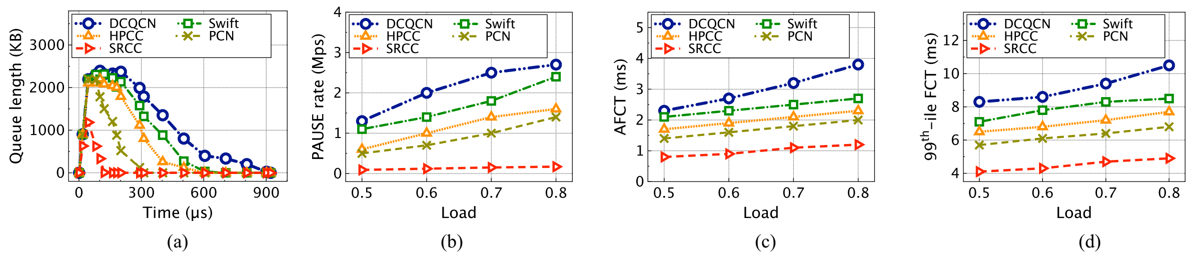


Fig. 10. Performance under Hadoop workload. (a) Queue length. (b) PAUSE rate. (c) AFCT. (d) 99th-ile FCT.

As shown in Figs. 9(b) and 10(b), with the increasing of traffic load, SRCC always maintains the lowest PFC PAUSE rate. For example, compared to DCQCN, Swift, HPCC, and PCN, the PAUSE rate of SRCC decreased by 91%, 90%, 88%, and 84% under 0.8 traffic load in web search, respectively. The reason is that although the end-to-end congestion control mechanisms perform well in controlling persistent congestion due to long-term traffic, they cannot prevent the queue building up within one RTT under transient congestion due to bursty traffic, resulting in PFC is inevitably triggered. However, SRCC can quickly respond to congestion by notifying congestion directly from the hotspot, and then SRCC can effectively reduce pausing rate.

Figs. 9(c) and 10(c) indicate that SRCC significantly reduces the average FCT compared to the state-of-the-art end-to-end congestion control schemes. This is because SRCC can effectively reduce the PFC PAUSE rate and the probability of PFC triggering. Even deploying the end-to-end congestion control mechanisms, burst mice flows can easily cause frequent triggering of PFC. In this case, the victim flows are innocently blocked, that is, they suffer from the well-known HoL blocking, resulting in the sharp increase of average FCT. Among them, DCQCN is most affected by instantaneous congestion because DCQCN lacks a mechanism to distinguish congested and uncongested flows and the rate evolution process is slow. Specifically, compared to DCQCN, Swift, HPCC, and PCN, the average FCT of SRCC is decreased by 65%–68%, 56%–62%, 48%–53%, and 39%–44% in Hadoop, respectively.

As shown in Figs. 9(d) and 10(d), even though the traffic load intensity continues to increase, SRCC still reduces the 99th percentile FCT effectively. This is because SRCC adjusts the rate of congested flows accurately and timely based on the congestion notification from the congestion point within one RTT. In this way, SRCC greatly reduces the feedback delay, shortens the control loop and accelerates the rate convergence process. SRCC is more adaptable to burst scenario and effectively prevents the victim flows from HoL blocking. In both web search and Hadoop workloads, SRCC achieves the lowest 99th percentile FCT. For example, under 0.8 traffic load, SRCC reduces the 99th percentile FCT by 55%, 42%, 34%, 22% and 53%, 42%, 36%, 28% in web search and Hadoop applications over DCQCN, Swift, HPCC, and PCN, respectively.

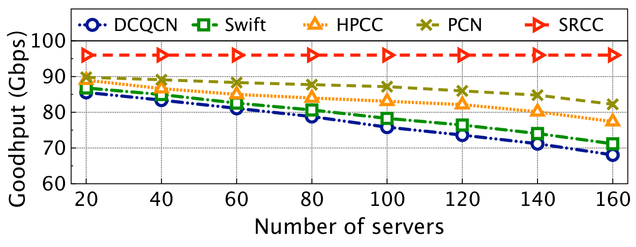


Fig. 11. Throughput of SRCC under incast scenario.

2) Performance Under Incast Traffic: Next, we evaluate the performance of SRCC under incast scenario, which is the most common many-to-one communication mode in DCNs. For example, during the distributed training process of larger language models, the massive model parameters need to be updated concurrently and a large number of incoming queries from different paths traverse multiple hops synchronously competing for a single bottleneck link. These large amounts of traffic arriving simultaneously cause instantaneous queue building up, which can potentially trigger PFC, leading to continuous PFC Pause/Resume and the subsequent HoL blocking.

In this test, a client initiates requests to multiple servers simultaneously to fetch the responses. We increase the incast degree by changing the number of involved servers from 20 to 160, and maintain the total incast traffic unchanged at 50 MB in each incast initiation. The bottleneck link capacity is 100 Gb/s. Fig. 11 shows the goodput measured at the client.

In Fig. 11, SRCC obtains higher goodput than other end-to-end congestion control solutions and the goodput is maintained about 100 Gb/s with the increasing degree of incast. In the incast communication pattern, all responses are sent back to the client concurrently, potentially resulting in continuous PFC pausing. SRCC directly feedbacks congestion messages from the switch to the corresponding transmitter for rate adjustment,

Authorized licensed use limited to: CHANGSHA UNIV OF SCIENCE AND TECHNOLOGY. Downloaded on January 13,2025 at 07:33:00 UTC from IEEE Xplore. Restrictions apply.

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

thereby achieving sub-RTT level congestion control, making it less than one RTT control loop. Therefore, SRCC can quickly feedback congestion information and reset the target sending rate for congested flows, resulting in no HoL blocking and high link utilization simultaneously. However, for other end-to-end transmission schemes, although they alleviate congestion by adjusting the transmission rate, they cannot avoid triggering PFC due to the need for at least one RTT control loop. As the degree of casting continues to increase, its Goodput will decrease due to more PFC PAUSE frames and slow sending rate adjustment. For example, SRCC improves the goodput by up to 42%, 34%, 25%, and 20% under 160 involved servers compared with DCQCN, Swift, HPCC, and PCN, respectively.

VI. Discussion

Effectiveness of protecting uncongested flows: Compared with existing congestion mechanisms, distinguishing congested flows, and adjusting their sending rate is the key to avoiding HoL problems, especially when traffic bursts cause PFC to be frequently triggered. Therefore, under the cache follower workload where micro bursts occur, SRCC performs better than other solutions [see Fig. 6(b)]. At the same time, in web server scenarios with stronger and smaller burst traffic, the proportion of SRCC blocking traffic is much lower than other mechanisms, and the performance effect is more significant [see Fig. 5(b)].

Stability and tolerance of network traffic: For stability, existing congestion control mechanisms have proven that as long as congestion information is reasonably obtained and speed regulation is carried out, stable performance can be achieved in controlling long-lived flows and long-term congestion. As for the tolerance of burst flows, SRCC only needs the feedback delay of sub-RTT to feedback congestion and handle micro bursts. Therefore, compared to existing mechanisms, SRCC can more effectively maintain network stability and handle uncertainty, because: 1) SRCC achieves sub-RTT feedback delay; 2) SRCC achieves protection for uncongested flows.

Optimization of Switch Resource Utilization: To reasonably distinguish congested flows, SRCC needs to obtain a lot of network information in real time within the switch and calculate congestion thresholds. In small-scale experiments, switch resources can still meet the computing and storage needs of SRCC. However, compared to other logically simple congestion control mechanisms, in large-scale experiments, SRCC will impose a significant burden on the storage and computing resources of the switch. Therefore, we believe that comprehensive optimization of information acquisition and computing programs is an important future work.

VII. Related Work

In recent years, congestion control for RDMA networking has been paid close attention by the academia and industrial. There are many end-to-end schemes have been proposed for lossless DCNs [5], [7], [8], [9], [10], [11], [12], [14], and some works devoted to congestion control for lossy DCN without PFC [15].

ECN-based: DCQCN [11] employs a fine-grained end-to-end congestion control based on ECN and adjusts the sending rate at the source to avoid PFC triggering persistently. PCN [5] rearchitects the congestion management for lossless DCNs. PCN can identify the real congested flows based on ECN and only throttle their rate to protect innocent flows from blocking.

RTT-based: TIMELY [9] adjusts the sending rate dynamically based on the gradient change of RTT. As an evolution of TIMELY, Swift [8] adapts rates according to the simplest end-to-end delay while taking into account the processing delay at the end-hosts, achieving consistently low tail completion time.

INT-based: As a next-generation congestion control for high-speed DCNs, HPCC [10] leverages INT technology to obtain historical information to control congestion, resulting in ultralow latency and high throughput.

Queue length-based: P-PFC [18] controls the buffer occupation by monitoring the change rate of queue length and triggers PFC pausing actively to cut the tail latency. TCD [7] defines ternary states of switch ports based on the queue length to detect congestion port accurately and identifies the congested flows that really contributing congestion.

ACK-driven: MP-RDMA [14] utilizes an ACK-clocking mechanism to distribute traffic among parallel paths, controls out-of-order degree with an reordering aware path selection scheme, and uses a synchronize mechanism to guarantee in-order memory updating, effectively improving the overall network utilization.

VIII. Conclusion

In this article, we proposed SRCC, a sub-RTT congestion control mechanism for lossless Ethernet DCNs. SRCC identifies the congested flows and calculates the congestion threshold dynamically at the switches. Then, the CNM carrying the congested flow ID and the target sending rate is sent back directly to the corresponding source end-hosts for the sending rate adjustment. The evaluation results show that SRCC effectively avoids PFC triggering and HoL blocking. SRCC outperforms the state-of-the-art solutions in terms of flow completion time and goodput especially under bursty congestion scenarios. Specifically, SRCC effectively suppresses PFC triggering and reduces the average FCT by up to 61%, 52%, 40%, and 24% over DCQCN, Swift, HPCC, and PCN, respectively.

References

[1] B. Cao et al., “Multiobjective 3-D topology optimization of next-generation wireless data center network,” *IEEE Trans. Ind. Informat.*, vol. 16, no. 5, pp. 3597–3605, May 2020.

[2] W. Zhang, R. Yadav, Y. C. Tian, S. K. S. Tyagi, I. A. Elgendy, and O. Kaiwartya, “Two-phase industrial manufacturing service management for energy efficiency of data centers,” *IEEE Trans. Ind. Informat.*, vol. 18, no. 11, pp. 7525–7536, Nov. 2022.

[3] P. K. R. Maddikunta et al., “Industry 5.0: A survey on enabling technologies and potential applications,” *J. Ind. Inf. Integration*, vol. 26, 2022, Art. no. 100257.

[4] W. Bai et al., “Empowering azure storage with 100$\times$100 RDMA,” in *Proc. 20th USENIX Symp. Netw. Syst. Des. Implementation*, 2023, pp. 49–67.

[5] W. Cheng, K. Qian, W. Jiang, T. Zhang, and F. Ren, “Re-architecting congestion management in lossless ethernet,” in *Proc. 17th USENIX Symp. Netw. Syst. Des. Implementation*, 2020, pp. 19–36.

[6] J. Hu et al., “Load balancing with multi-level signals for lossless datacenter networks,” *IEEE/ACM Trans. Netw.*, vol. 32, no. 3, pp. 2736–2748, Jun. 2024.

[7] Y. Zhang, Y. Liu, Q. Meng, and F. Ren, “Congestion detection in lossless networks,” in *Proc. ACM SIGCOMM Conf.*, 2021, pp. 370–383.

[8] G. Kumar et al., “Swift: Delay is simple and effective for congestion control in the datacenter,” in *Proc. Annu. Conf. ACM Special Int. Group Data Commun. Appl., Technol., Archit., Protoc. comput. Commun.*, 2020, pp. 514–528.

Authorized licensed use limited to: CHANGSHA UNIV OF SCIENCE AND TECHNOLOGY. Downloaded on January 13,2025 at 07:33:00 UTC from IEEE Xplore. Restrictions apply.

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

[9] R. Mittal et al., "TIMELY: RTT-based congestion control for the datacenter," *ACM SIGCOMM Comput. Commun. Rev.*, vol. 45, no. 4, pp. 537–550, 2015.

[10] Y. Li et al., "HPCC: High precision congestion control," in *Proc. ACM Special Int. Group Data Commun.*, 2019, pp. 44–58.

[11] Y. Zhu et al., "Congestion control for large-scale RDMA deployments," *ACM SIGCOMM Comput. Commun. Rev.*, vol. 45, no. 4, pp. 523–536, 2015.

[12] J. Hu, H. Shen, X. Liu, and J. Wang, "RDMA transports in datacenter networks: Survey," *IEEE Netw.*, vol. 38, no. 6, pp. 380–387, Nov. 2024, doi: 10.1109/MNET.2024.3397781.

[13] C. Guo et al., "RDMA over commodity ethernet at scale," in *Proc. ACM SIGCOMM Conf.*, 2016, pp. 202–215.

[14] Y. Lu et al., "Multipath transport for RDMA in datacenters," in *Proc. 15th USENIX Symp. Netw. Syst. Des. Implementation*, 2018, pp. 357–371.

[15] Z. Wang et al., "S-RDMA: A scalable architecture for RDMA NICs," in *Proc. 20th USENIX Symp. Netw. Syst. Des. Implementation*, 2023, pp. 1–14.

[16] J. Hu, Y. He, W. Luo, J. Huang, and J. Wang., "Enhancing load balancing with in-network recirculation to prevent packet reordering in lossless data centers," *IEEE/ACM Trans. Netw.*, vol. 32, no. 5, pp. 4114–4127, Oct. 2024.

[17] "IEEE 802.1Qbb-Priority-based Flow Control," 2010. [Online]. Available: https://1.ieee802.org/dcb/802-1qbb/

[18] C. Tian et al., "P-PFC: Reducing tail latency with predictive PFC in lossless data center networks," *IEEE Trans. Parallel Distrib. Syst.*, vol. 31, no. 6, pp. 1447–1459, Jun. 2020.

[19] S. Hu et al., "Aeolus: A. building block for proactive transport in datacenters," in *Proc. Annu. Conf. ACM Special Int. Group Data Commun. Appl., Technol., Archit., Protoc. Comput. Commun.*, 2020, pp. 422–434.

[20] Y. Yang, B. Zhang, D. Guo, R. Xu, C. Su, and W. Wang, "Age of information optimization for privacy-preserving mobile crowdsensing," *IEEE Trans. Emerg. Topics Comput.*, vol. 12, no. 1, pp. 281–292, Jan./Mar. 2024.

[21] Y. Yang et al., "Semantic sensing performance analysis: Assessing keyword coverage in text data," *IEEE Trans. Veh. Technol.*, vol. 72, no. 11, pp. 15133–15137, Nov. 2023.

[22] "IEEE. 802.1Qau-Congestion notification," 2009. [Online]. Available: http://www.ieee802.org/1/pages/802.1au.html

[23] A. Saeed et al., "Annulus: A dual congestion control loop for datacenter and WAN traffic aggregates," in *Proc. Annu. Conf. ACM Special Int. Group Data Commun. Appl., Technol., Archit., Protoc. Comput. Commun.*, 2020, pp. 735–749.

[24] Edge-core Networks, 2018. [Online]. Available: https://www.edge-core.com/productsInfo.php?cls=1&cls2=180&cls3=181&id=335

Jinbin Hu (Member, IEEE) received the B.E. degree in electronic science and technology and the M.E. degree in microelectronics and solid state electronics from Beijing Jiao Tong University, Beijing, China, in 2008 and 2011, respectively, and the Ph.D. degree in computer science from Central South University, Changsha, China, in 2020.

She is currently an Associate Professor with the Changsha University of Science and Technology, Changsha, China, and a Postdoctoral Researcher with the Hong Kong University of Science and Technology, Hong Kong. Her current research interests are in the area of datacenter networks, RDMA networking, and learning-based network systems.

Shuying Rao received the M.E. degree in computer science from the School of Computer and Communication Engineering, Changsha University of Science and Technology, Changsha, China, in 2024.

Her research interests are in the area of datacenter networks and network security.

Min Zhu received the B.S. degree from the Nanjing University of Posts and Telecommunications, Nanjing, China, in 2002, the M.S. degree from the Beijing University of Posts and Telecommunications, Beijing, China, in 2005, and the Ph.D. degree from the Nanjing University of Posts and Telecommunications in 2018, all in computer science.

She is currently with the College of Information Science and Technology, Zhejiang Shuren University, Hangzhou, China, as a Lecturer. Her research interests mainly include routing protocol and optimization algorithm design.

Jiawei Huang received the bachelor's degree from the School of Computer Science, Hunan University, Changsha, China, in 1999, and the Ph.D. and master's degrees from the School of Computer Science and Engineering, Central South University, Changsha, China, in 2008 and 2004, respectively, all in computer science.

He is currently a Professor with the School of Computer Science and Engineering, Central South University. His research interests include performance modeling, analysis, and optimization for wireless networks and data center networks.

Jianxin Wang (Senior Member, IEEE) received the B.E. and M.E. degrees in computer engineering and the Ph.D. degree in computer science from Central South University, Changsha, China, in 1992, 1996, and 2001, respectively.

He is currently the Chair and a Professor with the School of Computer Science and Engineering, Central South University. His current research interests include algorithm analysis and optimization, parameraized algorithm, bioinformatics, and computer network.

Jin Wang (Senior Member, IEEE) received the B.S and M.S. degrees from the Nanjing University of Posts and Telecommunications, Nanjing, China, in 2002 and 2005, respectively, and the Ph.D. degree from Kyung Hee University Korea, Seoul, South Korea, in 2010, all in computer science.

He is currently a Professor with the Hunan University of Science and Technology, Xiangtan, China. He has authored or coauthored more than 400 international journal and conference papers. His research interests mainly include wireless ad hoc and sensor network, network performance analysis and optimization etc.

Dr. Wang is a Fellow of IET.

Authorized licensed use limited to: CHANGSHA UNIV OF SCIENCE AND TECHNOLOGY. Downloaded on January 13,2025 at 07:33:00 UTC from IEEE Xplore. Restrictions apply.