CCRP: Converging Credit-Based and Reactive Protocols in Datacenters

Yang Bai, Dinghuang Hu, Dezun Dong, Shan Huang, Xiangke Liao

CCRP: Converging Credit-based and Reactive Protocols in Datacenters

Yang Bai, Dinghuang Hu, Dezun Dong*, Shan Huang, Xiangke Liao
College of Computer, National University of Defense Technology
Changsha 410073, China
{baiyang14, hudinghuang19, dong, huangshang12, xkliao}@nudt.edu.cn

Abstract. As the link speed has grown steadily from 10 Gbps to 100 Gbps, high-speed data center networks (DCNs) require more efficient congestion management. Therefore, proactive transports, especially credit-based congestion control, nowadays have drawn much attention because of fast convergence, near-zero queueing and low latency. However, in real deployment scenarios, it is hard to guarantee one protocol to be deployed in every host at one time. Thus, when the credit-based protocols are deployed into DCNs incrementally, the network will convert to multi-protocol state and face the following fundamental challenges: (i) unfairness, (ii) non-convergence, and (iii) high buffer occupancy. In this paper, we propose a new protocol, called CCRP, aiming for converging credit-based and reactive protocols in data centers. Targeting the mostly deployed protocol, i.e. DCQCN based on explicit congestion notification (ECN), in DCNs, CCRP leverages the forward ECN to detect the network congestion in data queue and optimizes feedback control of the credit-based transports. Our experiment results show that this design can address the unfair link allocation and converge with reactive protocols rapidly. Furthermore, CCRP achieves high utilization and low buffer occupancy at the same time.

Keywords: Data Center, Credit-based and ECN-based Protocol, Multi-protocol Converging

1 INTRODUCTION

The data center networks (DCNs) are growing rapidly in size and link speed in recent years. A large data center uses a Clos network of shallow buffered switches to connect more than 100,000 computers. In the past decade, the link speed has steadily increased from 10 Gbps to 100 Gbps [2]. These evolutions have enabled low-latency and high-bandwidth communications in data centers. At the same time, it presents a series of challenges to congestion control [1].

Many reactive congestion control protocols [3–8] have been proposed to solve these challenges. Reactive protocols use congestion signals (e.g., packet loss, explicit congestion notification (ECN) and network delay) to make accurate responses after congestion occurs, which can maintain good average performance under long traffic conditions. However, due to the slow detection of network congestion, it is difficult for reactive protocols to achieve the ”correct” rate in each round, which is deciding for small flows and tail performance.

Therefore, the current line of work introduces a new method called proactive congestion control [9–13] , which has received much attention in recent years. Among these technologies, one of the most promising deployments in future data centers is ExpressPass [11]—a credit-based proactive congestion algorithm, which can provide zero data loss, fast convergence, low buffer occupancy, and high utilization.

* corresponding author.

Due to their attractive advantages, credit-based transports, especially ExpressPass, are highly recognized by academia [20, 21]. However, current DCNs, such as Google and Amazon, still mainly deploy ECN-based reactive protocols like DCTCP [6] and DCQCN [5]. Thus, the gradual deployment of ExpressPass into DCNs is a visible task in the future. Nonetheless, deploying credit-based protocols [14, 15] like ExpressPass in DCNs will bring many challenges to the fairness of bandwidth allocation, especially in multi-tenant DCNs [25,26] . We show in Section 2 that simply mixing ExpressPass with reactive protocols deployed in real DCNs will cause serious trouble.

Therefore, we will face severe challenges when incrementally deploying ExpressPass with existing reactive transports in the current DCNs. The root cause is due to the different ways of detecting network congestion. Reactive protocols detect network congestion based on those indirect and passive congestion signals used in the data queue, such as packet loss [22,23], ECN [5,6,24] and network delay [7,17]. Taking DCQCN as an example, when the queue length exceeds the ECN threshold in switch, the data packets will be marked with congestion experienced (CE) codepoint. Then, DCQCN can detect congestion by simply identifying whether the packets are ECN-marked at the end hosts. However, ExpressPass acquires congestion information from the credit queue. As shown in Fig.1, a clear physical isolation exists between data queue and credit queue under Expresspass. ExpressPass uses the credit loss rate as the congestion indicator. When congestion is detected, ExpressPass reduces the credit sending rate at the receiver.

Thus, if we mix Expresspass with DCQCN traffic in the network, Expresspass can not detect the network congestion in the data queue and will transmit packets at a full speed, even the queue length exceeds the buffer size in switch. In contrast, DCQCN will reduce its sending rate continually until its bandwidth occupancy approaches zero, since a great quantity of packets may be marked with the CE codepoint in the data queue. We conducted several multi-protocol experiments to prove the aggressiveness of ExpressPass when coexisting with other three different types of reactive algorithms: DCQCN (FECN-based), CUBIC [29] (drop-based) and Timely (delay-based). The result indicates that ExpressPass is too aggressive for all of them, thus, the unfairness caused by the physical isolation must be eliminated.

The core problem of conflicting with reactive protocols is that the credit-based transports like Expresspass only detect network congestion through the credit queue, so a natural idea is to optimize the congestion detection mechanism for ExpressPass so that it can also detect congestion in the data queue. Therefore, we propose a new scheme called CCRP, which aims to achieve the symbiosis between credit-based protocols and reactive protocols. The key points of CCRP are as follows.

• CCRP breaks the isolation between the credit queue of credit-based protocols and the data queue by adding the ECN marking mechanism.

• We make a trade-off between the throughput and latency of the multi-protocol network by adjusting the threshold of the random early detection (RED) ECN marking scheme in the switch data queue.

• We replace the old feedback control of the credit-based protocols by a new ECN-based control algorithm, so that CCRP can also detect and deal with the network congestion occurs in the data queue.

Our new ECN-based feedback control in CCRP can detect and respond to network congestion in a few RTTs without affecting the performance of the reactive protocol stack. CCRP ensures fairness between different protocols with cross-protocol convergence time reaching milliseconds. The average buffer occupancy is also reduced to less than 170 KB under CCRP. Moreover, when CCRP coexists with few traffic under other congestion algorithms in the network, the transmission can be completed at high convergence speed

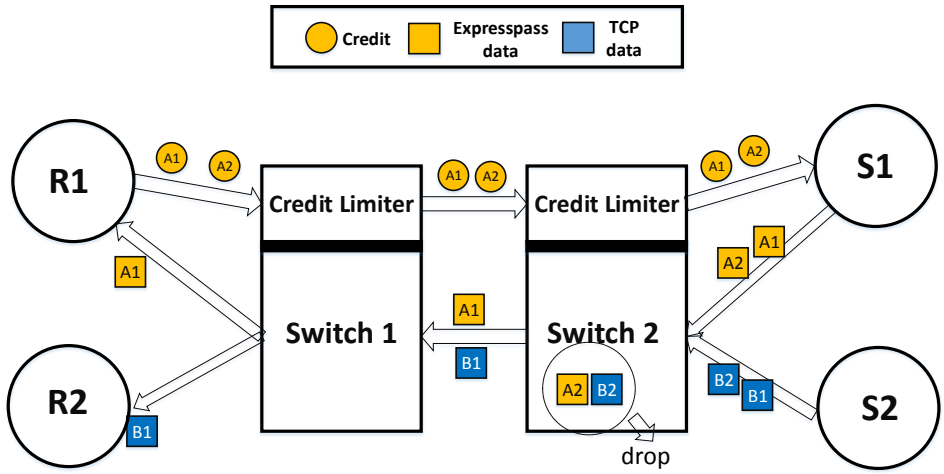


Fig. 1. The Dilemma of Deploying ExpressPass in DCN: ExpressPass gets network congestion information only in credit queue. Thus, when there is other traffic (ep. TCP flows) in the network and congestion occurs in the data queue, Expresspass will preempt the bandwidth, resulting in a large number of queues, even packet loss.

without introducing additional overhead. Simulating the current DCN common partitioning/aggregation mode shows that CCRP can greatly reduces the tail delay and also ensure that the average flow completion time (FCT) in a multi-protocol network will not be affected.

2 BACKGROUND AND MOTIVATION

2.1 DCN Needs the Credit-based Protocols

In the past decade, reactive protocols have played an important role, because they achieve high throughput and low buffer occupancy rate in DCNs. So far, many data centers are still deploying reactive protocols, such as DCQCN and DCTCP, to prevent the network from collapsing under heavy and sudden traffic.

However, due to the increased link speeds and shallow buffered commercial switches, the buffer size for per Gbps link speed is decreasing. For example, DCQCN result in unfairness due to the slow response time, which puts DCQCN in straits. In addition, the simulation result shows that when coexisting with a large number of concurrent flows, the instantaneous queue length is much larger than the maximum queue capacity under DCQCN, leading to low QOS for DCN.

Therefore, in recent years, credit-based proactive algorithms have attracted widespread attention. Unlike reactive protocols, credit-based proactive protocols like ExpressPass are characterized by the merits of high throughput, fast convergence and bounded-queue. ExpressPass is a hop-by-hop proactive congestion control algorithm. Before sending data packets, sender sends a credit request to the receiver. After receiving the request, receiver sends the credits to the sender on a per-flow basis in an end-to-end manner. The Switches then rate-limit the credits and decide the available bandwidth for packets which will flow in the reverse direction. As shown in Fig.1, credit packets throttled to 5% bandwidth ensure that traffic is transmitted within the link capacity. ExpressPass can achieve lossless

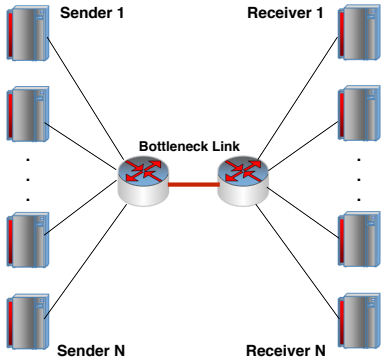


Fig. 2. Dumbbell topology: can the bottleneck resource be allocated in a fair way?

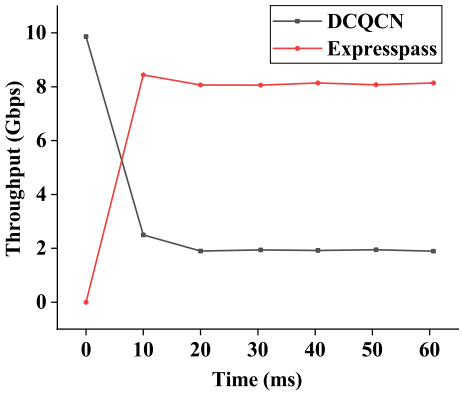


Fig. 3. The Problem of Multi-protocol in DCNs. Most bandwidth are occupied by Expresspass once it is injected into network.

transmission at only 5% bandwidth cost by using credits. However, Expresspass hasn't been deployed in DCNs. To deploy incrementally in data centers, ExpressPass must be optimized to coexist with the already widely deployed reactive protocols, such as DCQCN and DCTCP.

2.2 The Challenges of Deploying Credit-based Protocols in DCNs

Credit-based transports must coexist with other traditional protocols in DCNs, however, whether two kind of traffic can converge fairly is a main problem. In Fig.2, we assume that flow 1 to N are scheduled by two different protocols— ExpressPass and DCQCN. Then the question arises, if all the senders send data to the receivers simultaneously in these topologies, can the bottleneck link resource be allocated in a fair manner?

Fig.1 shows the dilemma of deploying ExpressPass in DCNs. The credit queue for Expresspass is isolated from the data queue for other protocols. Thus, when coexisting with other traffic in the network, the transmission rate of the ExpressPass traffic will not be reduced, since Expresspass detect network congestion only through the credit queue. In contrast, other "reactive" flow will be restricted because they can detect the network congestion in the data queue by using different congestion signals (ECN, RTT or packet loss). Consequently, ExpressPass preempts all resources of the bottleneck link aggressively, while the "reactive" flow can only wait.

To verify our analysis, we design an experiment based on the topology shown in Fig.2 by using OMNeT++ simulator. We let DCQCN run first, and then insert ExpressPass traffic into the network after 10ms. We use long flows so that they can be transmitted continuously. In addition, we set the link speed to 10Gbps and the propagation delay to 5μs . The result is shown in Fig.3, as Expresspass inserted into network, it preempts all resources of the bottleneck link. ExpressPass can always occupies the bandwidth quickly, while the other "reactive" flows could only be restricted and wait. Therefore, ExpressPass must be improved to coexist with reactive protocols, which is a quite a challenging job.

3 CCRP Design

3.1 Basic Idea

As we describe in Section 2, the main problem is that Expresspass can not receive any congestion information from the data queue due to the isolation between the credit queue and the data queue. Thus, we propose a new protocol called CCRP, which has achieved great success by using appropriate congestion signals for the credit-based proactive protocols, developing a new feedback control schemes, and determining reasonable network configurations. The key idea of CCRP is to optimize the feedback control and congestion detect scheme of the credit-based protocols, so that it can break the isolation between the data queue and the credit limiter to achieve great convergence when coexisting with other traffic. In CCRP, we choose FECN to deliver congestion information of data queue as it can be provided by commodity switches in DCN.

Specifically, when the network is converted to a congested state, low mode stage of DCQCN feedback control starts; when the network is not crowded, ExpressPass's credit feedback control algorithm will be fully effective. For ECN-based feedback control, once we deploy a reasonable ECN threshold on the switch, CCRP will automatically reduce the sending rate immediately after receiving ECN-marked packet, because only when there is some other non-ExpressPass traffic in the network , the queue length in switch will exceeds the ECN threshold. In addition, when ECN-marked packet is no longer received in multiple RTTs, CCRP will send credits more excessively.

3.2 ECN-based Feedback Control

As an effective technique to detect network congestion, using ECN is not a rare occurrence in reactive protocols. However, applying ECN to credit-based proactive protocols is a novel work. Our ECN-based feedback control focuses on two issues: (i) How to react appropriately to the different types of congestion information from credit queue and data queue. (ii) How to detect congestion more effectively.

First, we need to keep the good performance of the credit-based transports. When there is little "reactive" traffic in the network environment, credits can be sent excessively to achieve high convergence. To this end, in the initial few RTTs, CCRP enables traffic tend to send packets at the link capacity. After multiple RTTs, if not getting any packet marked with CE codepoint, we set the *DCQCN Timer* as true and assume that there is no other "reactive" traffic in the network and take a more aggressive approach.

Here, we must make a trade-off between convergence and packet loss. Before the network environment is determined, it's easy to cause packet loss when the sender sends credits at high speed in the manner of Algorithm1. We can design the phase to send packets by using slow start scheme, which will harm the convergence of Expresspass. There are the reasons for sending credit at high rate in CCRP:

• As the link speed has grown steadily from 10 Gbps to 100 Gbps, the transmission of data flow will be completed in much shorter time. Under 100Gbps network, the AvgFCT of 0-100KB small flows would be around 2 RTTs [16], which indicates that even one RTT is quite important for these small flows. Thus, If CCRP starts at a low speed, it may waste the bandwidth.

• Starting at high rate can help CCRP quickly detect whether there is other "reactive" flows in DCNs.

• When there is few other "reactive" flows in the network, the characteristic of high of convergence can be guaranteed for credit-based protocols.

Algorithm 1: ECN-based Feedback Control on Receiver

$ECN\_\alpha \leftarrow 1, \omega \leftarrow 0, cur\_rate \leftarrow initial\_rate, DCQCN\_Timer \leftarrow FALSE$
repeat
  $credit\_Loss = \#\_dropped\_credit/\#\_sum\_credit$;
  $ECN\_ratio = \#\_ECN\_packet/\#\_sum\_packet$;
  if $ECN\_ratio \geqslant target\_ECN\_ratio$ **then**
    $ECN\_\alpha = (1-g) * ECN\_\alpha + g * ECN\_ratio$;
    $tmp\_rate = cur\_rate * (1 - (target\_ECN\_ratio + ECN\_\alpha)/2)$;
    if $DCQCN\_Timer = TRUE$ **then**
      Use ECN-based feedback control of DCQCN;
  end if
  if $credit\_Loss \leqslant target\_loss$ **then**
    ▷ (increasing phase)
    $\omega = (\omega + \omega_{max})/2$;
    $cur\_rate = (1-\omega) * tmp\_rate + \omega * max\_rate$;
  else
    ▷ (decreasing phase)
    $cur\_rate = tmp\_rate * (1 - credit\_loss\_rate)$;
    $\omega = max(\omega_{min}, \omega/2)$;
  end if
until End of flow

In addition, we use ECN to detect the network congestion of data queue. When Expresspass coexist with other "reactive" traffic in the network, some packets will be marked with CE codepoint and the low mode of DCQCN feedback control will be triggered. After serval increasing and decreasing stages, the ECN ratio will converge to the target_ECN_ratio recursively.

Finally, in CCRP, we also design a variant version of the credit-based increase and decrease phase to make full use of the link capacity information provided by the credit queue. The variable ω that we added to these phases floats between ω_{min} and ω_{max}, and can achieve slow self-increase and rapid decrease. Based on this, we have designed the ECN-based feedback control algorithm, which is shown in Algorithm 1. With this algorithm, when coexisting with other "reactive" traffic in the network, CCRP can allocate the bottleneck link bandwidth in a fair manner instead of disrupting other traffic aggressively. Also, CCRP can keep the advantages of bounded queue and high convergence of credit-based protocols through the acceleration algorithm of DCQCN. Our experiments show that the new version of credit-based feedback control is strongly suitable for multi-protocol networks.

3.3 Parameter Choice

Target_Loss: Since the credit-based protocols like Expresspass can disrupt other traffic roughly, we need to set a lower target_loss in the variant version of credit-based control phase to reduce the aggressiveness in the early stages of deploying CCRP. Thus, we set 0 for the target_loss to adapt to the current multi-protocol network environment.

Target_ECN_Ratio: In the current shallow-buffer switch environment, target_ECN_ratio should be set as 0, which will ensure that the average queue length in switch is around the ECN threshold K. We also provide an interface by defining the target_ECN_ratio to improve feedforward compatibility of DCN, so that we can control the switch buffer usage by modifying the parameters at the host instead of modifying the threshold at the switch.

ω_{max}: The aggression factor ω is particularly significant in CCRP feedback control algorithm. Based on our experiments, we find in most cases, the credit_loss is less than the target_loss since CCRP does not always utilize all the bandwidth. Thus, ω is tend to get closer to ω_{max} in multi-protocol network. In order to reduce the aggressiveness of CCRP, we choose a smaller ω_{max}. To choose an appropriate value for ω_{max}, we did a parallel experiments between CCRP and DCQCN and finally found that 0.04-0.07 is suitable for ω_{max} to ensure fairness. In our experiments, ω_{max} is set to 0.06 and the evaluation of the range of ω_{max} is shown in Section 4.

3.4 Endhost and Switch Mechanism

Marking at CP: Based on the random early detection scheme, we use instantaneous queue length to detect network congestion. DCTCP recommends to set Kmin and Kmax to the same value K, so that congestion can be detected and handled quickly. DCQCN advocates to use three parameters, Kmin, Kmax, and Pmax to mark packets. However, it's not efficient because in most cases, there is no ingress/egress traffic and queue occupancy is close to zero. Therefore, we set Kmin as 5KB and Kmax as 200KB in CCRP.

Controller at RP: Different from other protocols, the ECN-based feedback control algorithm of CCRP is on the receiving side. The controller of the receiver can provide great convenience, since we can detect congestion accurately by checking whether the data packet is marked with CE codepoint. Compared with traditional method of using ACK to notify the sender to cut the congestion window, CCRP can also reduce the load of the link by reducing the credit sending rate at the receiver.

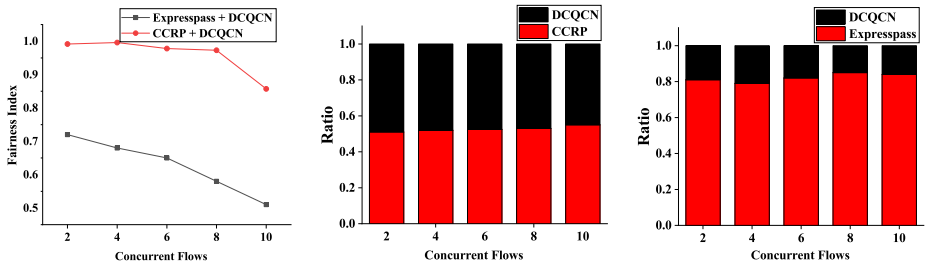
Bandwidth Allocation of Credit/Packet Channel: CCRP doesn't change the design of the credit packet size and the separate queue on the switch. An 84B size Ethernet frame credit can trigger the sender to send a 1538B size Ethernet frame packet. Therefore, 5% of the link capacity is used for credit transmit, while the remaining 95% of the link capacity is allocated for data packets. Even if the credit channel is not fully utilized in multi-protocol networks, the design remains unchanged since the traffic of DCNs is changing rapidly.

4 Evaluation

In this section, we measure the performance of CCRP from five perspectives: (i) utilization of the bandwidth (ii) fairness (iii) convergence speed (iv) flow completion time (v) queue length. All the experiments are completed with OMNeT++ simulator [27, 28, 30] and the ratio of CCRP/ExpressPass to DCQCN is 1:1.

Utilization of the Bandwidth: Firstly, we measure utilization since one of the benefits of using ECN is high utilization. As we described in Section 2, Expresspass must provide 5% of the link bandwidth to transmit credits, so that the utilization of CCRP and Expresspass is near to 95%. We compare the performance of CCRP, Expresspass and DCQCN in the same multi-protocol network environment, and the results show that all the three protocols can make full use of network bandwidth. The utilization rate of the network in which the credit-based protocol and the ECN-based protocol coexist is 95%, while the utilization of only ECN-based protocol is close to 100%.

Fairness: CCRP aims to optimize the feedback control algorithm and reduce the aggressiveness of credit-based proactive protocols so that they can coexist with other tradition "reactive" traffic in the network. Thus, fairness is the most significant metric in our evaluation. We mainly did two expreriments to measure the fairness of CCRP in multi-protocol networks. We calculate the average bandwidth of each flow in a 100 millisecond



(a) Jain's Fairness Index (b) CCRP + DCQCN (c) ExpressPass + DCQCN

Fig. 4. Fairness measurement. CCRP can achieive fairness greatly due to the ECN-based congestion control, while ExpressPass cannot guarantee fairness.

interval by using the Jain's fairness index and the result is shown in Fig.4a. Due to the aggressiveness of credit-based protocols, fairness between Expresspass and DCQCN is poor. Besides, as more concurrent traffic being injected into network, Expresspass will suffer from packet loss and fairness will deteriorate further. On the contrary, as shown in Fig.4a, CCRP performs much better than Expresspass. We attribute this improvement to the ECN-based feedback control in CCRP. As shown in Fig.4b and 4c, the fairness between the two types of traffic is evaluated by the ratio of CCRP/Expresspass to DCQCN. This measurement only focuses on the unfairness between different traffic and ignores the internal unfairness caused by packet loss. The result also shows that CCRP can ensure fairness when coexising with other "reactive" traffic.

Convergence: Convergence is also a highlight of our work. We did a series of experiments to simulate the convergence of CCRP/ExpressPass and DCQCN when they coexist in the network. Specifically, there are four types of hosts (A to D). Type A machine (running ExpressPass or CCRP, sender) connect type C machine (running ExpressPass or CCRP feedback control, receiver) and type B machine (running DCQCN, sender) connect to type D machine (running DCQCN, receiver) with 10 Gbps links via ECN-enable switch. A to C and B to D are through the same path. The switch creates two connections to simultaneously fetch two large flows from sender A and D. Since ω_{max} is one of the most important variable of ECN-based feedback control algorithm in CCRP, in our experiments, we range ω_{max} from 0.03-0.07.

As seen from Fig.5b, 5c, 5d, 5e , when ω_{max} ranges from 0.04 to 0.07, CCRP can greatly reduce the aggressiveness of Expresspass and converge with DCQCN perfectly. When the aggression factor is set to 0.07 as shown in Fig.5e, the convergence speed is only 100 ms slower than that of DCQCN as shown in Fig.5g. In contrast, Fig.5f shows that ExpressPass with DCQCN cannot achieve convergence in the multi-protocol network. However, when ω_{max} is too small (ω_{max}=0.03), as shown in Fig.5a, although CCRP can also help DCQCN seize the bandwidth, the result of convergence is not ideal. Finally, Fig.5h shows that both CCRP and Expresspass have the same performance with high convergence.

Flow Completion Time: Since the FCT is one of the most important performance metrics of network congestion protocols, we also use FCT as a comparison metric. We plot the FCT distributions in Fig.6 and the Avg/99th/99.9th FCT in Fig.7. All the simulation results show that, CCRP achieves much better FCT than Expresspass. We believe that our new ECN-based feedback control breaks the isolation between the data queue and credit queue, helping CCRP deal with the network congestion in data queue and improve the

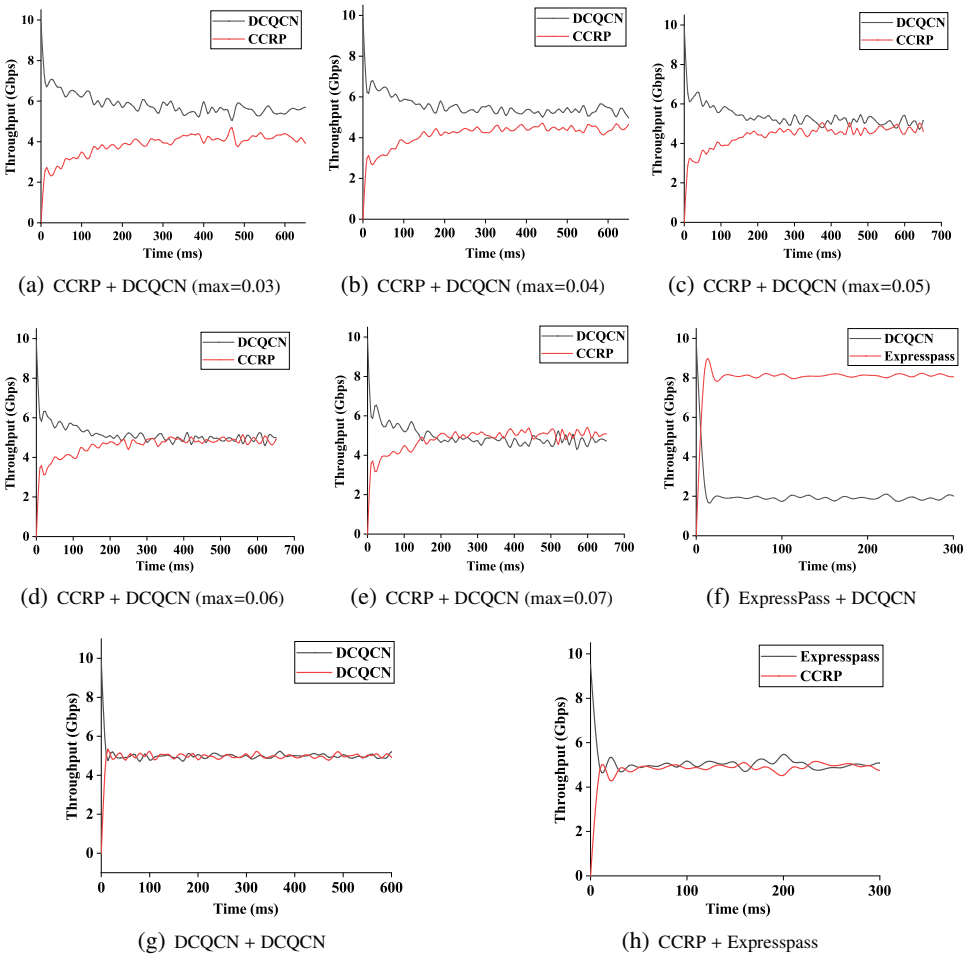
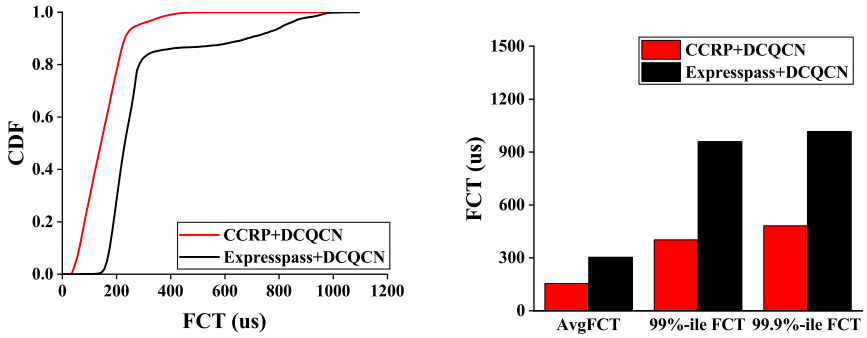


Fig. 5. Convergence Mearsurement. When ω_{max} ranges from 0.04-0.07, CCRP can converge to fairness with DCQCN, while ExpressPass cannot converge to fairness.

FCT performance. However, Expresspass cannot detect the network congestion in the data queue and send packets aggressively, causing severe packet loss and the increase of FCT.

Queue Length: The last major indicator is the buffer occupancy. Due to the shallow buffered switches, low buffer occupancy is also the performance that we pursue in our work. Thus, we measured the average queue length of the switch, which characterizes the queue under general circumstances. As shown in Fig.7, compared with Expresspass, CCRP can achieve much smaller queue length when coexist with DCQCN in the network.

Summary: After the detailed measurement and analysis, we conclude that CCRP can greatly narrow the competitiveness difference between credit-based proactive protocols like ExpressPass and traditional reactive protocols like DCQCN. Compared with Expresspass, CCRP can converge with DCQCN perfectly and achieve high Utilization, fairness, small FCT and low buffer occupancy at the same time.



(a) FCT Distributions (b) Avg/99th/99.9th FCT

Fig. 6. FCT Performance of CCRP+DCQN and Expresspass+DCQCN. CCRP achieves much better FCT than Expresspass.

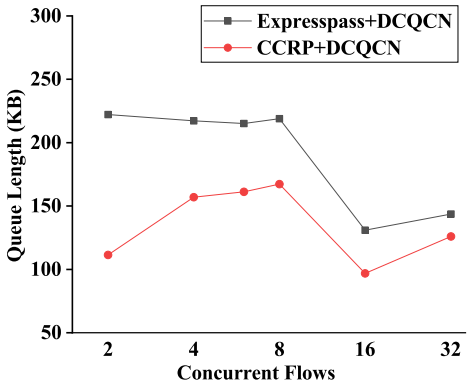


Fig. 7. Average Queue Length. Compared with Expresspass, CCRP avoids bursty traffic and reduces the buffer occupancy.

5 Related Work

RTT-based Protocols: For those congestion control algorithms (Timely [7] and DX [17]) which are based on delay, RTT is a very important congestion signal. They do not require any information feedback from the switches. Only the continuous record of delay of packets at the host can determine whether congestion happens. Timely belongs to a different class of algorithms that use delay measurements to detect congestion. Unlike TCP Vegas [17], which is window-based and maintain a queue close to the minimum RTT, Timely is a rate-based algorithm that employs a gradient approach and does not rely on measuring the minimum RTT. It works well with NIC support, despite infrequent RTT signals. Compared to DCTCP, Timely can significantly reduce queuing delay. Reducing CPU utilization of end hosts is not a goal for Timely. Different from Timely, DX implements accurate latency measurements using a DPDK driver for the NIC and the congestion control algorithm is within the Linux TCP stack, which is similar to the conventional window-based proposals.

Credit-based Feedback Control: Credit-based congestion control in data centers is inspired by credit-based flow control [19] for other interconnected systems. ExpressPass uses a similar idea like TVA [31] that performs rate-limit requests at the router and CCRP

inherite that method. Furthermore, in high-performance networks, proactive congestion control uses grants for congestion control. Unlike CCRP, those schemes use speculative packets on a grant-based basis to avoid wasting preparing the data transmission. Although they are difficult to implement and have extra preprocessing time overhead, and those trade-offs are hard to balance, they provide an idea for the credit-based protocols in the DCNs. We look forward to finding a reasonable compatibility solution for other credit-based transports. Moreover, compared with CCRP, end-to-end credit-scheduled congestion control focus on incast problems based on the receiver. Those transmission control algorithms add an extra control layer to make sure senders only transmit according to some quota assigned to them.

6 Conclusion

In this paper, we propose a new protocol called CCRP, aiming for incrementally deploying the credit-based congestion control in current data centers. CCRP breaks the isolation between the data queue and the credit rate limiter by using ECN as the congestion signal. The new efficient ECN-based feedback control algorithm that we use to control the credit sending rate can guarantee high performance of CCRP without interfering with other traffic in the network. The evaluation results show that CCRP can greatly narrow the competitiveness difference between credit-based proactive protocols like ExpressPass and traditional reactive protocols like DCQCN. Compared with Expresspass, CCRP can converge with DCQCN perfectly and achieve high Utilization, fairness, small FCT and low buffer occupancy at the same time. Therefore, CCRP has high applicability in the incremental deployment of credit-based congestion control in data centers.

Acknowledgment

We would like to thank the anonymous reviewers for their insightful comments. We gratefully acknowledge members of Tianhe interconnect group at NUDT for many inspiring conversations. The work was supported by the National Key RD Program of China under Grant No. 2018YFB0204300.

References

1. A.Singh et al., “Jupiter rising: A decade of clos topologies and centralized control in google’s datacenter network,” *Communications of the ACM 2016*, pp. 188-197. Available: https://doi.org/10.1145/2785956.2787508
2. L.Jose et al., “High speed networks need proactive congestion control,” *in Proceedings of HotNets 2015*, pp. 1-7. Available: https://doi.org/10.1145/2834050.2834096
3. C. Wilson et al., “Better never than late: meeting deadlines in datacenter networks,” *in Proceedings of SIGCOMM 2011*, pp. 50-61. Available: https://doi.org/10.1145/2018436.2018443
4. H. Wu et al., “ICTCP: Incast Congestion Control for TCP in Data-Center Networks,” *in Proceedings of CoNEXT 2010*, pp. 1-12. Available: https://doi.org/10.1145/1921168.1921186
5. H. Eran et al., “Congestion control for large-scale RDMA deployments,” *in Proceedings of SIGCOMM 2015*, pp. 523–536. Available: https://doi.org/10.1145/2785956.2787484
6. M. Alizadeh et al., “Data center TCP (DCTCP),” *in Proceedings of SIGCOMM 2010*, pp. 63–74. Available: https://doi.org/10.1145/1851182.1851192
7. R. Mittal et al., “Timely: RTT-based congestion control for the datacenter,” *in Proceedings of SIGCOMM 2015*, pp. 537–550. Available: https://doi.org/10.1145/2785956.2787510

8. C. Hong et al. “Finishing Flows Quickly with Preemptive Scheduling,” *in Proceedings of SIGCOMM 2012*, pp. 127–138. Available: https://doi.org/10.1145/2377677.2377710
9. P. Gao et al. “phost: Distributed near-optimal datacenter transport over commodity network fabric,” *in Proceedings of CoNEXT 2015*, pp. 1–12. Available: https://doi.org/10.1145/2716281.2836086
10. J. Perry et al., “Fastpass: A centralized ”zero-queue” datacenter network,” *in Proceedings of SIGCOMM 2014*, pp. 307– 318. Available: https://doi.org/10.1145/2619239.2626309
11. I. Cho et al., “Credit-scheduled delay-bounded congestion control for datacenters,” *in Proceedings of SIGCOMM 2017*, pp. 239–252. Available: https://doi.org/10.1145/3098822.3098840
12. N. Jiang et al., “Network congestion avoidance through speculative reservation,” *in Proceedings of HPCA 2012*, pp. 1–12. Available: https://doi.org/10.1109/HPCA.2012.6169047
13. B. Montazeri et al., “Homa: A receiver-driven low-latency transport protocol using network priorities,” *in Proceedings of SIGCOMM 2018*, pp. 221–235. Available: https://doi.org/10.1145/3230543.3230564
14. G. Michelogiannakis et al., “Channel reservation protocol for over-subscribed channels and destinations,” *in Proceedings of HPCA 2013*, pp. 52:1–52:12. Available: https://doi.org/10.1145/2503210.2503213
15. J. Nan et al., “Network endpoint congestion control for fine-grained communication,” *in Proceedings of SC 2015*, pp. 35:1–35:12. Available: https://doi.org/10.1145/2807591.2807600
16. S. Hu et al., “Augmenting proactive congestion control with aeolus,” *in Proceedings of APNet 2018*, pp. 22–28. Available: https://doi.org/10.1145/3232565.3232567
17. C. Lee et al. “Accurate Latency-based Congestion Feedback for Datacenters,” *in Proceedings of USENIX ATC 2015*, pp. 403–415. Available: https://doi.org/10.1109/TNET.2016.2587286
18. L. Brakmo et al. “TCP Vegas: new techniques for congestion detection and avoidance,” *in Proceedings of SIGCOMM 1994*, pp. 24–35. Available: https://doi.org/10.1145/190314.190317
19. H. Kung et al., “Credit-based flow control for ATM networks: Credit update protocol, adaptive credit allocation, and statistical multiplexing,” *in Proceedings of SIGCOMM 1994*, pp. 101–114. Available: https://doi.org/10.1145/190314.190324
20. Y. Zhang et al., “BDS: a centralized near-optimal overlay network for inter-datacenter data replication,” *in Proceedings of EuroSys 2018*, pp.1-14. Available: https://doi.org/10.1145/3190508.3190519
21. R. Mittal et al., “Revisiting network support for RDMA,” *in Proceedings of SIGCOMM 2018*, pp. 313–326. Available: https://doi.org/10.1145/3230543.3230557
22. M. Alizadeh et al., “pFabric: minimal near-optimal datacenter transport,” *in Proceedings of SIGCOMM 2013*, pp. 435–446. Available: https://doi.org/10.1145/2486001.2486031
23. K. Fall et al., “Simulation-based comparisons of (tahoe, reno and sack tcp),” *in Proceedings of SIGCOMM 1996*, pp. 5–21. Available: https://doi.org/10.1145/235160.235162
24. D. Zats et al., “DeTail: reducing the flow completion time tail in datacenter networks,” *in Proceedings of SIGCOMM 2012*, pp. 139–150. Available: https://doi.org/10.1145/2377677.2377711
25. G. Judd et al., “Attaining the promise and avoiding the pitfalls of TCP in the datacenter,” *in Proceedings of NSDI 2015*, pp. 145–157. Available: https://doi.org/10.5555/2789770.2789781
26. K. He et al., “AC/DC TCP: Virtual congestion control enforcement for datacenter networks,” *in Proceedings of SIGCOMM 2016*, pp. 244–257. Available: https://doi.org/10.1145/2934872.2934903
27. http://omnetpp.org/.
28. https://inet.omnetpp.org/.
29. S. Ha et al., “CUBIC: a new TCP-friendly high-speed TCP variant,” *ACM SIGOPS Operating Systems Review, 2008*, pp. 64–74. Available: https://doi.org/10.1145/1400097.1400105.
30. A. Varga et al., “An overview of the OMNeT++ simulation environment,” *in Proceedings of SIMUTools 2008*, pp. 1–10. Available: https://doi.org/10.1145/1416222.1416290
31. X. Yang et al., “A dos-limiting network architecture,” *in Proceedings of SIGCOMM 2005*, pp. 241–252. Available: https://doi.org/10.1145/1080091.1080120