Regional Congestion Mitigation in Lossless Datacenter Networks

Xiaoli Liu, Fan Yang, Yanan Jin, Zhan Wang, Zheng Cao, Ninghui Sun

Regional Congestion Mitigation in Lossless Datacenter Networks

Xiaoli Liu[1,2], Fan Yang[1,2], Yanan Jin[1,2], Zhan Wang[1], Zheng Cao[1], Ninghui Sun[1]
[1] State Key Laboratory of Computer Architecture, Institute of Computing Technology, Chinese Academy of Sciences
[2] Institute of Computer and Control Engineering, University of Chinese Academy of Sciences Beijing, China
{liuxiaoli, yangfan, jinyanan, wangzhan, cz, snh}@ncic.ac.cn

Abstract. To stop harmful congestion spreading, lossless network needs much faster congestion detection and reaction than the end-to-end approach. In this paper, we propose a switch-level regional congestion mitigation mechanism (RCM) that performs traffic management just at the congestion region edge. RCM moves the end-to-end congestion control to hop-by-hop switch level to lower the congested region's load as fast as possible. Meanwhile, to handle longer congestion, RCM detours the non-congestion flows to a light-loaded available path based on regional congestion degree to avoid the congestion region. Evaluation shows that the proposed RCM mechanism can perform timely congestion control over microburst flows, and achieve >10% improvement on mice flow's FCT and throughput than DCQCN, with rarely performance reduction on elephant flows.

Keywords: lossless datacenter network, congestion control, adaptive routing.

1 Introduction

RDMA (e.g. iWrap [1] and RoCEv2 [2]) has been increasingly deployed in datacenters with emerging artificial intelligence (AI) applications and distributed resource pooling systems (e.g. all-flash array with NVMe over fabric). RDMA technology relies on a lossless network to guarantee reliable transmission and high transmission performance. However, in current lossless network, the lossless link-level flow control (PFC: Priority Flow Control [3]) will back pressure the traffic, spread the congestion, and eventually build a congestion tree that saturates the whole network [4]. As shown in Figure 1, network congestion happening at an egress port will gradually form a global congestion tree (red lines) and block other flows transmitting to non-congested hosts. Such congestion tree is a unique issue in lossless network. It will become even worse in future ultra-high bandwidth network, e.g. 100Gb/s or 400Gb/s Ethernet, because the burst length that a switch buffer can hold is becoming shorter and shorter. Since congestion tree will cause global congestion spreading and leads to rapid degradation of the entire network performance, it is essential to provide high efficient congestion control in the lossless network.

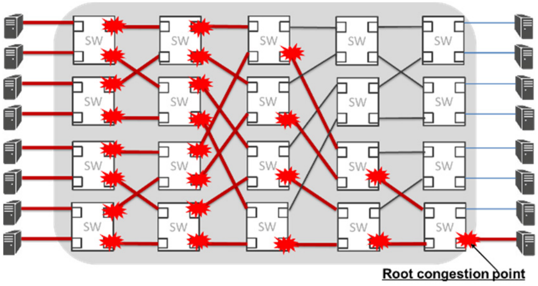


Fig. 1. Congestion spreading in lossless network

The key idea of mitigating congestion is to stop injecting non-admissible traffic into congested regions in time. Currently, end-to-end congestion control, such as ECN (Explicit Congestion Notification) [5] based DCTCP [6] and DCQCN [7], is main mechanism used to lower the injection rate of the source host. However, since most of congestion is caused by micro-burst traffics [8, 9], end-to-end approach may fail due to its long congestion notification latency (approaching to end-to-end round trip time). In addition, to stop congestion spreading, lossless network needs much faster congestion detection and reaction than the end-to-end approach. So, regarding the short-time congestion, we propose a regional congestion control mechanism that moves the end-to-end congestion control to hop-by-hop switch level to lower the congested region's load as fast as possible. Meanwhile, to handle longer congestion, we also propose a regional adaptive routing mechanism that detours the victim flows to avoid congestion region. Performance evaluation shows that our strategy can achieve better timeliness and fairness than the one with only end-to-end congestion control. Our key contributions are summarized as follows:

1) Fast Congestion Region Detection. We check both the status of the input queue and output queue. Once an intra-switch congestion is reported, the congestion notification together with IDs (e.g. TCP/IP five tuple) of harmful flows that are contributing to congestion will be sent to neighbor switches.

2) Regional Congestion Control (RCC). We perform congestion control at the edge of a congestion region. Once a switch confirms a congestion region and gets the IDs of harmful flows, it dynamically increases non-congested flows' priority to bypass the congestion flow at the output port.

3) Switch-level Adaptive Routing (SAR). Once the congestion exceeds RCC's capability, the switch at the edge of congestion region will detour the victim flows to other light-loaded paths. Note that the end-to-end congestion control is still needed, the injection rate of the long-life harmful flow will be reduced by end-to-end congestion control mechanism eventually.

2 Background and Related Work

2.1 RDMA Deployment in Data Center

RDMA was first developed in HPC system to deliver high bandwidth and low latency network service [10]. It significantly reduces CPU overhead and overall latency by performing transport layer hardware offloading and OS-bypass data path between the NIC and applications. In recent years, big data analysis applications, including artificial intelligence, are becoming more and more popular. These applications can also be treated as a kind of HPC applications, since they require ultra-high computing and network performance. Therefore, more and more datacenters are trying to deploy RDMA at scale to provide better performance.

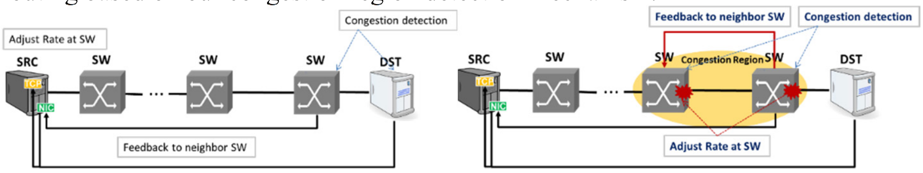
However, RDMA in HPC is deployed over lossless Layer 2 fabric (e.g. Infiniband [11]), while most data center networks are built over lossy Ethernet with the commodity switch. To enable RDMA over Ethernet, the RDMA over Converged Ethernet (RoCE and RoCEv2[2]) was proposed. In converged Ethernet CEE standard, the link-level flow control protocol PFC was adopted. PFC [3] is a kind of point-to-point On/Off flow control that can prevent switch or NIC from buffer overflow. However, once the network congestion occurs, it will produce backpressure to its upstream switch. If the congestion lasts for long enough, such backpressure will be spread hop by hop and eventually forms congestion region.

2.2 End-to-end Congestion Control

End-to-end congestion control is current main mechanism used to lower the injection rate of the source host. TCP with ECN enabled is the classic layer 3 end-to-end congestion control mechanism, and QCN (Quantized Congestion Notification [12]) is a layer 2 end-to-end congestion control mechanism proposed for lossless network. ECN-aware switch/router sets a congestion mark in the IP header to signal impending congestion. After the receiver gets the labeled packet, it feeds back to the sender to reduce its transmission rate. Unlike ECN just using a mark bit, QCN tries to quantify the congestion degree, which keeps tracking the status of switch's output queue and calculates the quantized degree from the queue's offset between enqueuing and dequeuing rate. The QCN-aware switch detected congestion directly sends out CNMs (Congestion Notification Message) carrying the flow ID and congestion degree to the flow's source host. The source NIC handles the CNM and performs rate control.

At the host side, new transport protocols, like DCTCP[6] and DCQCN[7], were proposed to work with the ECN or QCN. In DCTCP, the host counts the fraction of ECN-marked packets (*F*) and adjusts the window size based on the variable *F*. DCQCN combines ECN's congestion notification mechanism and QCN's rate adjustment mechanism at the host. To avoid the congestion tree in the lossless datacenter network, properly tuning the switch buffer's thresholds triggering ECN is rather difficult, not to mention tuning the thresholds of PFC and ECN at the same time. There are also some network measurement based end-to-end congestion control mechanisms, such as TIMELY [13].

The congestion control in network can be considered as a basic "control system" with feedback loop. As shown in Figure 2(a), all these end-to-end mechanisms involve a long congestion notification procedure and the reaction procedure at the transport layer may take hundreds of microseconds or even milliseconds. Long congestion control loop will cause the end-to-end control fail in microburst scenarios, because the congestion may already widely spread before the end-to-end control works. Therefore, it is essential to perform timely congestion mitigation in hardware. Note that performing congestion control is not always the optimal congestion mitigating method, especially not for congestion caused by intermediate path collisions. Adaptive routing in HPC interconnection fabric [14, 15] is a candidate solution to such collision, which detours certain traffics to light-loaded path based on the switch's local port load. We borrow the idea and proposed a switch-level adaptive routing based on our congestion region detection mechanism.



(a) end-to-end control system (b) regional control system

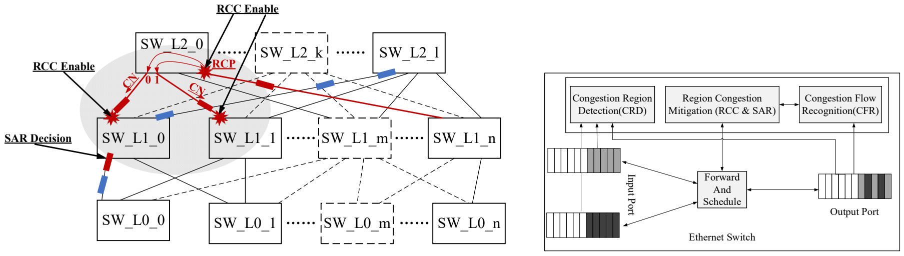
Fig. 2. Network congestion control: regional control system aims to shorten the feedback loop and perform the reaction as fast as possible compared to end-to-end control system

3 Regional Congestion Mitigation Mechanism

3.1 Design Philosophy

As shown in Figure 2(b), in our proposed RCM mechanism, a switch sends out its local congestion notification only to neighbors, and performs accurate congestion control locally based on the congestion condition it detects or receives. First, it will absorb the micro-burst congestion to a certain extent with certain free buffer, and then start the switch-level accurate reaction, which performs pacing to harmful flows at the output port. By doing this, the switch at the congested region edge can prevent the congestion spreading as fast as possible. When the reserved buffer is almost full or the switch gets neighbor's congestion notification again, it will try to detour non-harmful flows especially new flows to lightly loaded path.

Figure 3(a) illustrates the system architecture of our proposed RCM. As shown in Figure 3(a), the congestion region is to isolate congestion within a harmless region that starts from the root congestion point and includes switches that the harmful congestion flows have passed by. The switch in Figure 3(b) monitors the potential harmful flows that are most likely to cause congestion, and plays the role of both congestion detect and reaction point. The switch architecture includes CRD (Congestion Region Detection) module for fast congestion region detection, RCC (Regional Congestion Control) module for regional congestion control to micro-burst flows, SAR (Switch-level Adaptive Routing) module for re-routing flows suffering intermediate path collision, and CFR (Congestion Flow Recognition) module for congestion flow Recognition.



(a) system architecture

(b) switch architecture

Fig. 3. Architecture of Regional congestion mitigation

The switch port that first detects the congestion is called RCP (Root Congestion Point), e.g. the SW_L2_0 shown in Figure 3(a). Once a RCP is reported, it starts congestion control (RCC) to the flows recognized by CFR. Then, the input port0 and port1 of SW_L2_0 holding congestion flows (indicated by red flow) will detect the congestion, and send out congestion notification (CN) with congestion flows' IDs (e.g. TCP/IP five tuple) to their neighbor switch. The neighbor switch labels itself as the edge of the congestion region, and performs congestion control only on congestion flows indicated by the remote congestion notification it received, without interfering with victim flows. Correspondingly, the neighbor switch may also detect its local congestion condition and further send congestion notification with IDs to its neighbor switches. With such fine-grained hop-by-hop congestion control and congestion notification, the congestion will be back-pressured along the way as harmful flows passed by, and a harmless congestion region is formed, as the SW_L2_0, SW_L1_0 and SW_L1_1 formed the region in Figure 3(a).

Based on RCC, traffics from congestion flows will be absorbed to a certain extent, but it may still generate back-pressure to the source node and cause congestion spreading. To provide more burst-absorption capability and achieve better load balancing, switches within the congestion region can perform adaptive routing to non-congestion flows or new flows. As the blue flow shown in Figure 3(a), switch SW_L1_0 in the congestion region will detour it to other light-loaded paths based on the regional congestion degree.

3.2 Fast Congestion Region Detection

As shown in Figure 3(a), the congestion region starts from the root congestion point, and includes switches that the harmful congestion flows passed through. To mark the edge of the congestion region, each switch port maintains three parameters: *congestion degree, local congestion flag and remote congestion flag*. The switch point with only the *local congestion flag* set will be marked as root congestion point and the congestion flows recognition will be conducted for following accurate congestion control. The switch point with only the *remote congestion flag* set will be marked as the edge of the congestion region, while the switch point with both *local congestion flag and remote congestion flag* set will be marked as inner congestion point inside the congestion region. The harmful congestion flow IDs at the edge and inner congestion points will be replaced by the ones getting from root congestion point.

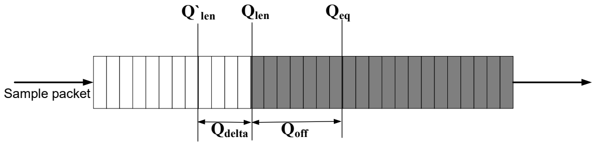


Fig. 4. Congestion detection in both input queue and output queue

Because of the link-level lossless flow control, back-pressure at the input port of switch may occur before the congestion happens at the output port. Therefore, the congestion detection is conducted on both input queue and output queue to improve the timeliness of detection, as shown in Figure 4. Regarding each queue's congestion detection, we use similar CP algorithm introduced in QCN. The RCD module (in Figure3 (b)) samples incoming packets with a predefined sampling interval depending on the degree of congestion. Within each sampling phase, a congestion measure F_b is computed by two factors: one is the offset of current queue length (Q_{len}) exceeding a predefined equilibrium length (Q_{eq}), represented as Q_{off}, and the other one is the differential of queue length between the instantaneous Q_{len} and the Q'_{len} when the last packet was sampled, in another word, the differential of the enqueuing and dequeuing rates, represented as Q_{delta}. Then F_b is given by the following formula:

$$F_b = Q_{off} + w_q \times Q_{delta} \tag{1}$$

$F_b > 0$ means that the queue length is exceeding the equilibrium length or the packets are accumulating, indicating a more likely congested state. Once a congestion state ($F_b > 0$) is firstly detected in an input queue, the port will set its *local congestion flag* (root congestion point) and send a feedback message containing quantified F_b to the neighbor switch. The root congestion point conducts Regional Congestion Control to the flows (harmful flows) recognized by CFR. The recognition algorithm will be introduced in Section 3.3.

Actually, the real F_b' sent out to the upstream neighbor will take the downstream neighbor's congestion degree F_{br} into consideration. If there is a valid congestion notification getting from the neighbor, the F_b' will be updated as:

$$F_b' \leftarrow (1-w) \times F_b + w F_{br} \tag{2}$$

Where the parameter w depicts the weight that the received remote *congestion degree* F_{br} taken in the contribution to congestion degree.

3.3 Congestion Flows Recognition

The proposed mechanism will determine the harmful flows that are contributing to the congestion happening at the root congestion point, to make sure that congestion control only performs on harmful flows, not on victim flows.

As shown in Figure 3 (b), the congestion flow recognition is implemented by CFR at switch's output ports. The flows with relatively high injection rate in a monitor window can be treated as potential congestion flows. The rate measurement for each flow is impractical as there may be thousands of flows. A sophisticated method adopted by the proposed mechanism is to do a periodic sampling on the packets passing through the output queue. Based on flow's statistical information, the flow with relatively higher bandwidth will be sampled with higher probability.

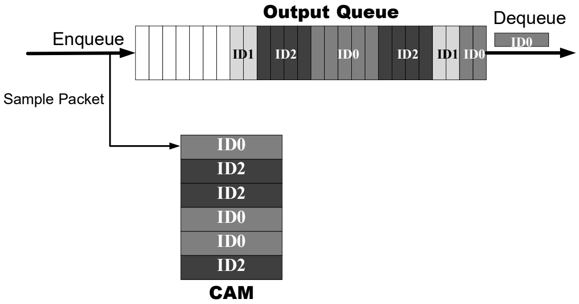


Fig. 5. Schematic Diagram of CFR

Figure 5 shows the schematic diagram of CFR. For each output queue, we implement a CAM (content addressable memory) to store the IDs (here TCP/IP five-tuple) of packets which has been sampled. The CAM is written in a cyclic way, the depth of CAM decides the sampling granularity of congested flows. Statistically, the IDs of flows that contribution to the congestion are more likely to be sampled and stored in the CAM than the victim flows. Hence, we can treat the flows whose IDs are being stored in the CAM as the harmful congestion flows.

3.4 RCM: Regional Congestion Mitigation

RCC: Regional Congestion Control

To perform efficient and fair reaction to the congestion, our RCC introduces a window-based hop-by-hop congestion control. "Window" implies the amount of data can be sent during a period. The send window is running periodically, that's to say, when the window-limited data has been sent out, we can start another *window with period W*. The window is comprised of two sub-windows:

$$W = W_c + W_{nc} \tag{3}$$

Wherein, W_c *is the congestion flow window,* indicating the data amount of harmful congestion flows has been sent out, while W_{nc} is the *non-congestion flow window,* indicating the data amount of non-congestion flows has been sent out. The W_{nc} and W

are estimated by the corresponding data amount sent in a time period from a given start when a congestion detected to the time when packets belong to all the congestion flow has been sent. We adjust the rate of congestion flows by controlling the W_c proportion in one window period W.

When one port detects output congestion state or receives a congestion notification message, it extracts the degree F_b from the notification message, and decreases its sending rate based on the following formula:

$$R(t) \leftarrow R(t) \times (1 - F_b \times G_d) \tag{4}$$

Where the constant G_d is chosen to satisfy $F_{bmax} * G_d = 1/2$, indicating the rate can be decreased by at most 50%. With formula (4), we can compute a new *"window":*

$$W_c \leftarrow W_c(1 - F_b \times G_d) \tag{5}$$

$$W_{nc} \leftarrow W - W_c \tag{6}$$

Then, the switch output schedules packets from various input ports according to the "*Window*". Once the amount of data that congestion flow has been sent reaches its window limit W_c, it cannot continue to send until the whole window W has been sent. By doing this, the congestion flow's rate can be decreased fleetly and back-pressure notifications will be generated along the path that the congestion flow passed by. Once the congestion state disappears, the window-based congestion control will quit.

SAR: Switch-level Adaptive Routing

Our proposed SAR is a switch-level adaptive routing based on the local output ports' load and their corresponding regional estimated congestion degree. We assume the equivalent path information has been pre-configured in date center network.

When the switch in a congestion region (as shown in Figure 3(a)) has received several server congestion notifications from the neighbor switch or itself is undergoing congestion, it makes adaptive routing decision for non-congested or new flows to select the next available output port with minimum *regional congestion degree*. As depicted in Formula (2), the switch port calculates *regional congestion degree* with its local congestion degree F_b and neighbors' congestion degree F_{br}. To limit the congestion spreading, the harmful congestion flows should not be detoured by SAR. SAR will check whether the flow hits in the CAM as depicted in Section 3.3. If the flow is not the congestion flow indicated by the CAM, it can be detour to another path to avoid the congestion region. However, if there are no more available paths, the flow not hitting in CAM still will not be detoured.

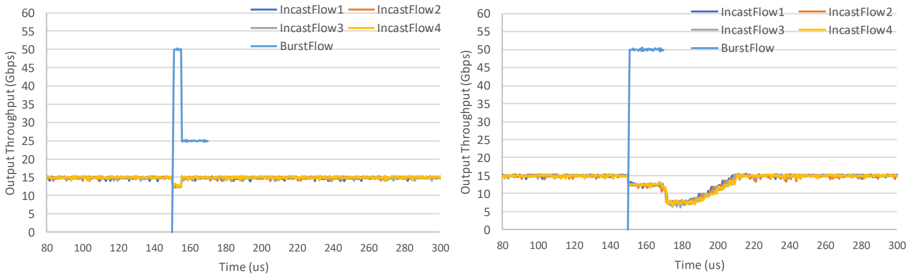
4 Evaluation

4.1 Simulation Models

We perform the packet-level network simulation based on OMNeT++ platform, with the modeling of PFC link-level flow control, the proposed RCM mechanism, ECMP routing and DCQCN. We choose the {12-port, 3-layer} fat tree topology with 432 hosts. Each link is configured to 100Gb/s. The switch is input-and-output queuing architecture, with virtual input queues (VOQ) at the input port and one output queue (OQ) at the output port. Each queue in the switch is 100KB in size. The parameters of DCQCN are configured with the default values used for Mellanox 100G adapter [17].

4.2 Performance with micro-benchmarks

Incast with burst flow: we begin with the experiment with Incast communication patterns that are common in datacenters, and evaluate the RCM's timeliness and efficiency. We use four hosts generating flows (IncastFlow) to the same destination host at the beginning of the simulation. And a micro-burst flow (BurstFlow) to the same destination is injected into the network at the time 150μs and lasts for 20μs. The four flows (IncastFlow in Figure 6) are injected into the last switch from the same port, while the micro-burst flow (BurstFlow in Figure 6) is injected from another port.

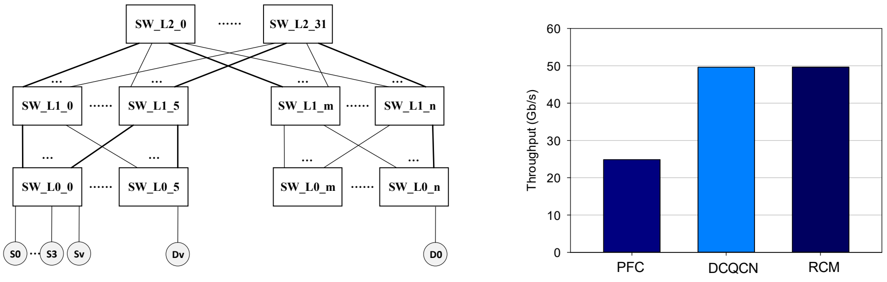


(a) Using RCM (b) Using DCQCN

Fig. 6. Throughput of flows at the last switch's output port

Figure 6 shows each flow's instant throughput at the last switch's output port. The plots confirm that RCM can absorb the micro-burst flows with rarely impacting the rate of background flows (IncastFlow). As shown in Figure 6(a), because of the fair scheduling, the micro-burst flow first takes 50Gb/s, the half bandwidth of output port. Meanwhile, each IncastFlow decreases to about 12.5Gb/s. Once the congestion is detected, the RCM reduces the micro-burst flow's rate to 25Gb/s sharply at the switch and the throughput of four IncastFlows recovers to their maximum bandwidth quickly. While in DCQCN, as shown in Figure 6(b), the throughput of micro-burst flow maintains at 50Gbps all the time. This is because even the micro-burst flow has already finished, the ECN congestion notifications are still on the way to source hosts. What is worse, the four IncastFlows are also tagged with ECN marks, which causes unnecessary rate control at the host.

Incast with victim flow: We now focus on the RCM's fairness. Considering Figure 7(a), four hosts (*S1-S4*) send data to the same destination (*D*0). In addition, a "victim flow" from source *Sv* is sent to destination *Dv*. ECMP maps flows from *S1-S4* equally to SW_L1_0 and SW_L1_5, while victim flow from *Sv* is mapped to SW_L1_0. All hosts use the same priority. As *D0* is the bottleneck of *S1-S4* Incast, it back-pressures its incoming links with PFC pause frame, limiting 50Gbps for each upstream port. This in turn leads to all the switches that Incast flows passed by to pause their incoming links. And eventually, SW_L0_0 is forced to pause the source hosts (*S1-S4*). Ideally, each Incast flow will get 25Gbps throughput and *Sv* gets 50Gbps throughput. However, we find that *Sv* only gets about 25Gbps because of HOL blocking issue. As shown in Figure 7(b), RCM can achieve fairness, since the victim flow can get a fair share nearly 50Gbps throughput.



(a) Topology

(b) Throughput of the victim flow

Fig. 7. Fairness of RCM

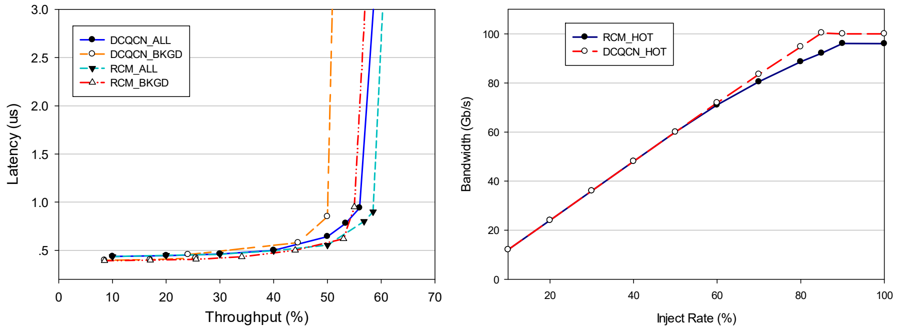


Fig. 8. Throughput and latency under hot-spot traffic pattern

Hot-spot traffic pattern: We now focus on its network performance with hot-spot traffic pattern. In this experiment, each host node sends its 85% traffic to uniform-randomly chosen destination node, as background flows, and the left 15% traffic is sent to the specified 12.5% nodes of the network, as hot flows. Such pattern coincides with traffic characteristics of data centers [6][8][16]. Figure 8 shows the throughput and latency of the background, hot and all flows respectively. As shown in Figure 8 (a), the background flows in RCM get 12.2% higher bandwidth than DCQCN, because of its accurate congestion control on hot flows and adaptive routing for background flows. Although the hot flows get a little lower performance shown in Figure 8 (b), the overall network performance of RCM still are still better than DCQCN, as there are a large proportion of background flows.

4.3 Performance with application traffic patterns

This section will show experiments with realistic workloads based on two empirically observed traffic patterns in deployed datacenters [18]. The first distribution is derived from packet traces represents a large enterprise workload. The second distribution is from a large cluster running data mining jobs.

Traffic patterns: As depicted in [18], both distributions are heavy-tailed: A small fraction of the flows contribute most of the data. We develop an application program to generate the traffic and divide the entire 432 nodes of the simulated fat tree network into 12 groups, each group with 36 nodes. Each node sends flows according to a Poisson process from randomly chosen nodes that are belong to the same group. The injection rate is chosen to 60% and the flow sizes are sampled from one of the above distributions. The flow distribution is shown as the pie charts in Figure 9. We use the flow completion time (FCT) as the main performance metric.

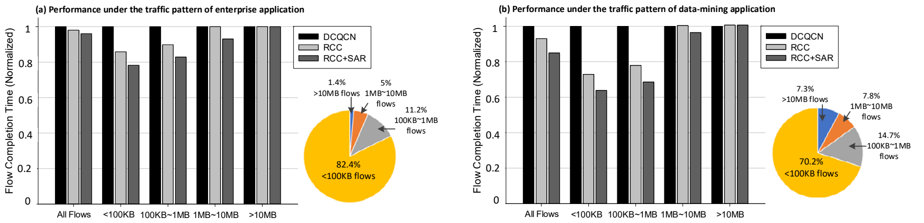


Fig. 9. Average flow completion time of enterprise and date-mining workloads

Result Analysis: Figures 9 shows the results for the two workloads. The proposed RCM has lower overall average FCT than DCQCN in both traffic patterns. This is achieved by RCC's timely reaction to congestion avoiding its harmful spreading. For heavy congestion, the SAR further detour non-congestion flows to other available light-load paths, providing better load balance. As shown in Figure 9, we find a degradation in FCT for mice flows, which confirms that the proposed RCM benefits for mice flows (<1MB), while it has little impact on the elephant flow (>10MB). Figure 9(b) shows better performance promotion for data-mining workload than enterprise workload as shown in Figure 9 (a). This is because data-mining workload is more "heavily congested" than the enterprise workload. In addition, data-mining workload has more elephant flows than enterprise workload. So, there are more path collisions happen and the RCM with SAR can achieve better performance promotion for data-mining workload from the load balancing perspective.

5 Conclusion

We propose a regional congestion mitigation solution in the lossless datacenter network, which identifies the congestion region and performs traffic management just at the congestion region edge. With the timely traffic management, our solution can achieve >10% improvement over DCQCN, regarding both mice flow's FCT and throughput. However, the impact of switch's micro-architecture has not been fully considered yet. As the preliminary phase, we have not performed in-depth study on the performance impact of reserved buffer size and the ratio of out-of-order packets introduced by SAR. All these will be remained as our future work.

References

1. RDMA Consortium. iWARP protocol specification. http://www.rdmaconsortium.org/
2. Infiniband Trade Association. RoCEv2. https://cw.infinibandta.org/document/dl/7781, September 2014
3. IEEE 802.802.1Qbb - Priority-based Flow Control. http://www.ieee802.org/1/pages/802.1bb.html, 2011
4. C. Guo, H. Wu, Z. Deng, G. Soni, J. Ye, J. Padhye, and M. Lipshteyn, "RDMA over Commodity Ethernet at Scale," in Proceedings of the 2016 ACM SIGCOMM Conference, 2016, pp. 202-215.
5. IETF.org. RFC 3168: The Addition of Explicit Congestion Notification (ECN) to IP. https://www.ietf.org/rfc/rfc3168.txt, 2001
6. M. Alizadeh, A. Greenberg, D. Maltz, J. Padhye, P. Patel, B. Prabhakar, S. Sengupta, and M. Sridharan, "Data Center TCP (DCTCP)," in Proceedings of the 2010 ACM SIGCOMM Conference, 2010.
7. Y. Zhu, H. Eran, D. Firestone, C. Guo, M. Lipshteyn, Y. Liron, J. Padhye, S. Raindel, M. H. Yahia, and M. Zhang, "Congestion Control for Large-Scale RDMA deployments," in Proceedings of the 2015 ACM SIGCOMM Conference, 2015, pp. 523-536.
8. T. Benson, A. Akella, and D. A. Maltz, "Network traffic characteristics of data centers in the wild," in Proceedings of the 10th ACM SIGCOMM conference on Internet measurement, 2010, pp. 267-280.
9. R. Kapoor, A. C. Snoeren, G. M. Voelker, and G. Porter, "A Study of NIC Burst Behavior at Microsecond Timescales " in Proceedings of the ninth ACM conference on Emerging networking experiments and technologies, 2013.
10. T. S. Woodall, G. M. Shipman, G. Bosilca, and A. B. Maccabe, "High Performance RDMA Protocols in HPC," in 13th European PVM/MPI User's Group Meeting, 2006.
11. Infiniband Trade Association. The InfiniBand™ Architecture [Online].
12. IEEE. 802.11Qau. Congestion notification, 2010.
13. R. Mittal∗, V. T. Lam, N. Dukkipati, E. Blem, H. Wassel, M. Ghobadi∗, A. Vahdat, Y. Wang, D. Wetherall, and D. Zats, "TIMELY: RTT-based Congestion Control for the Datacenter," in Proceedings of the 2015 ACM SIGCOMM Conference, 2015.
14. F. Petrini, W.-c. Feng, A. Hoisie, S. Coll, and E. Frachtenberg, "The Quadrics Network (QsNet): High-Performance Clustering Technology," in Proceedings of the 9th IEEE Hot Interconnects, Palo Alto, California, August 2001, pp. 125-130.
15. A. Ran and O. Sela, "Intel Omni-Path Architecture Technology Overview," in 21st Annual Symposium on High-Performance Interconnects, 2013.
16. T. Benson, A. Anand, and A. Akella, "Understanding data center traffic characteristics," in Proceedings of the 1st ACM workshop on Research on enterprise networking, 2009, pp. 65-72.
17. DC-QCN Parameters, https://community.mellanox.com/docs/DOC-2790
18. Speeding Applications in Data Center Networks. http://miercom.com/pdf/reports/20160210.pdf