Fairness Improvement of Multiple-Bottleneck Flow in Data Center Networks

Kenta Matsushima, Yuki Tanisawa and Miki Yamamoto
Faculty of Engineering Science
Kansai University
3-3-35 Yamate-cho, Suita-shi, Osaka, Japan
Email: k506903, k284455, yama-m@kansai-u.ac.jp

***Abstract*—Quantized Congestion Notification (QCN), discussed in IEEE 802.1Qau, is one of the promising layer 2 congestion control methods for data center networks. Data center network fundamentally has symmetric structure and links are designed to have high link utilizations. So, data center flows probably pass through multiple bottleneck links. QCN reduces its transmission rate with each congestion notification feedback reception, which might cause excessive regulation of transmission rate. We have already proposed QCN with Bottleneck Selection (QCN/BS) for multicast communications in data center. QCN/BS is originally proposed for multicast communications, but it can also be applied to unicast communication with multiple bottleneck points. QCN/BS selects the worst congestion level and the transmission rate of the sending device is calculated exclusively according to feedback from the selected switch. In this paper, we preliminary evaluate QCN/BS in unicast communications with multiple bottleneck points. Our preliminary evaluation reveals that QCN/BS can resolve this excessive rate regulation problem but has new fairness problem for long-hop flow. To resolve this fairness problem, we integrates QCN/BS and our proposed Adaptive BC_LIMIT. In Adaptive BC_LIMIT, parameter BC_LIMIT is adaptively decided so that the time interval between QCN rate increase is independent of transmission rate. With rate increase interval independent of transmission rate defined in the original QCN as well as rate decrease dependent on it defined in our proposed Adaptive BC_LIMIT, convergence of fair rate allocation among flows sharing a bottleneck link is accelerated. Our simulation results show that our proposed integration of QCN/BS and Adaptive BC_LIMIT significantly improves fairness problem for unicast communications with multiple bottleneck points in data center networks.**

Keywords–data center; QCN; congestion control; fairness.

I. Introduction

Efficient networking is extremely important when a large number of servers are connected in a high-speed, high-capacity network, which is typical situation of data centers supporting cloud computing. Many large-scale data centers implement both a storage area network (SAN) and a local area network (LAN). A SAN generally uses a Fiber Channel for high reliability. Fiber Channel technology is more expensive than Ethernet technology, and the management cost of maintaining the two types of network is high. Thus, the integration of LANs and SANs with Ethernet technology has been under standardized by the IEEE 802.1 Data Center Bridging Task Group [1]. The convergence of data center networking is expected to yield low power consumption and simplify hard wiring [2].

With LAN/SAN integration, conventional SAN performance, i.e., high reliability with low frame loss, will be provided on Ethernet technology. A multi-hop Ethernet configuration is generally used to accommodate the large number of end systems (servers) in current data center networks. Consequently, heterogeneous traffic causes congestion in traffic hot spots. Quantized Congestion Notification (QCN) [3] [4] is one of the promising congestion control methods for data center networks. QCN can keep queue length at a low value because the sending device determines a transmission rate according to feedback from a switch.

Data center network generally has symmetry structure, e.g., Fat Tree [5] and VL2 [6]. Currently, traffic engineering techniques for data center networks, such as MicroTE [7] and Penalized Exponential Flow-spliTing algorithm (PEFT) in data center [8], have been proposed to improve oversubscription environment in data center networks. Purpose of these traffic engineering techniques is generally minimizing maximum link utilization, which might increase the number of highly utilized links in data center networks. Symmetric structure with highly utilized links introduces quite different traffic feature from current wide area Internet, multiple bottlenecks on a path. QCN is feedback-based congestion control and reduces transmission rate with each feedback reception. With multiple bottleneck links, QCN might cause excessive rate regulation due to feedback frames sent by multiple bottleneck points.

This excessive feedback problem typically occurs in multicast communications. We have already proposed Quantized Congestion Control with Bottleneck Selection (QCN/BS) [9] which resolves Loss Path Multiplicity (LPM) problem [10]. In QCN/BS, the sending device identifies the transmission rate of each bottleneck point and selects the lowest one. Our preliminary evaluation in this paper shows that QCN/BS can improve somewhat fairness problem caused by excessive rate regulation but we discovers curious fairness problem for long-hop flow. In multiple bottlenecks situation, frame departure process from a highly utilized bottleneck link (in QCN, link utilization generally grows up to 0.99 [3]), is groomed. This means inter-arrival time to a next bottleneck link of this flow is almost discrete (exactly it is not discrete due to 0.99 utilization and frame length fluctuation: with 1.0 utilization and fixed frame size, it is completely discrete). When discrete arrival is mixed with a fluctuated flow, i.e., flow encountering the first bottleneck point, discrete arrival sees slightly large queue length. This slight difference in measured queue length causes slightly more feedback reception in long-hop flow.

The original QCN rate calculation algorithm has a tendency of keeping unfair situation [11]. We proposed adaptive BC_LIMIT [11] where parameter BC_LIMIT is adaptively decided according to current transmission rate of flows (details of adaptive BC_LIMIT is explained in section III). This parameter setting accelerates convergence of fair rate allocation among flows sharing a bottleneck link. In this paper, we integrate our proposed two algorithms, QCN/BS and Adaptive BC_LIMIT, in order to resolve unfair problem suffered in long-hop flow. Our simulation results show that our proposed integrated algorithm significantly improves fairness among flows in multiple-bottleneck situation.

The paper is structured as follows. In Section II, we review the QCN and QCN/BS algorithm and show preliminary performance evaluation of QCN and QCN/BS for multiple-bottleneck flow. Section III gives detailed description of our proposed integrated QCN/BS. In Section IV, we present the simulation results. Section V concludes our paper.

II. QCN/BS for Multiple Bottleneck Points

A. QCN

QCN is a Layer 2 congestion control method for multi-hop Ethernet, and its standardization is discussed mainly in IEEE 802.1 Qau. The QCN algorithm is based on a congestion notification from a switch. A QCN switch observes current queue length and calculates a feedback value when a data frame arrives. When congestion occurs, as identified by the calculated feedback value, the switch sends a feedback frame with a certain probability to the source. When a feedback frame is received, the source decreases the transmission rate according to the feedback value. If no feedback frame is received, the transmission rate is increased according to the three-phase rate increase algorithm.

The QCN algorithm works at Reaction Points (RPs) and Congestion Points (CPs). RPs and CPs represent QCN source and switch dynamics, respectively. These behaviors have been described in greater detail in [12].

The CP Algorithm

The CP calculates its feedback value as follows:

$$F_b = (Q_{eq} - Q) - w(Q - Q_{old}), \tag{1}$$

where F_b is the feedback value, Q is the current queue length of the corresponding switch, Q_{eq} is the target queue length, Q_{old} is the queue length at the time of the previous feedback transmission, and w is a constant weight value.

The first component on the right side of 1 shows how much smaller the current queue length is than the target queue length, the second component shows how much the queue length has decreased.

When a CP receives a data frame with a negative feedback value, it returns the feedback value to the corresponding RP. To avoid incurring overhead, the CP returns the feedback value with a certain probability.

The RP Algorithm

The behavior of the RP is shown in Figure 1. The rate increase phase is divided into three sub-phases: Fast Recovery, Active Increase, and Hyper-Active Increase. Each RP maintains four

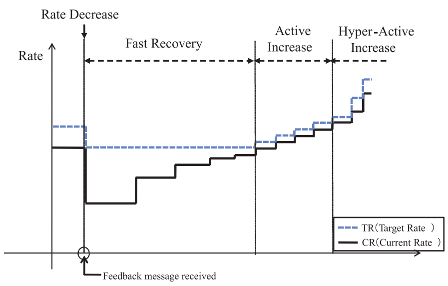


Figure 1. behavior of the RP.

variables for its rate control: Current Rate (CR), Target Rate (TR), Byte Counter, and Time Counter. CR is the current transmission rate, and TR is the transmission rate that is the goal for CR; it is always larger than or equal to CR. The Byte Counter is incremented by 1 whenever the RP sends a mixed byte value (denoted BC_LIMIT = 150 KB in pseudo QCN code [12]). The Time Counter is also incremented by 1 whenever a mixed amount of time has passed (15 ms in [12]). The behaviors of RPs are described below.

Rate Decrease. Whenever a feedback frame is received, the RP first activates the rate decrease phase, and both the Byte Counter and Time Counter are initialized, i.e., reset to 0. Just after receiving feedback, TR and CR are changed as follows:

$$\begin{cases} TR \leftarrow CR \\ CR \leftarrow CR(1 - G_d|F_b|), \end{cases} \tag{2}$$

where G_d is a constant that satisfies $G_d|F_{b_{max}}| = 0.5$.

Fast Recovery. In the Fast Recovery phase, CR is increased rapidly just after rate reduction if no feedback is received. Whenever the Byte Counter or Time Counter is incremented by 1 (after the RP sends data frames of BC_LIMIT KB or the timer spends 15 ms) CR is recovered as follows:

$$CR \leftarrow \frac{1}{2}(CR + TR). \tag{3}$$

Active Increase. When the Byte Counter or the Time Counter reaches 5, the Fast Recovery phase transits to the Active Increase phase. The rate increase in this phase is slower than that in the Fast Recovery phase because the transmission rate might be very close to the rate at which congestion occurred (and feedback was received). In this phase, whenever the Byte Counter or Time Counter is incremented by 1, TR and CR are changed as follows:

$$\begin{cases} TR \leftarrow TR + R_{AI} \\ CR \leftarrow \frac{1}{2}(CR + TR), \end{cases} \tag{4}$$

where R_{AI} is a fixed value (5 Mbps). During this phase, whenever the RP sends half of BC_LIMIT (75 KB) or the timer records 7.5 ms, the Byte Counter and Time Counter are incremented by 1.

Hyper-Active Increase. If no feedback is received after Fast Recovery and Active Increase phases, considerable bandwidth might be left unused. For example, this occurs, when one or more sessions that share a bottleneck switch are closed, and their unused bandwidth is available for other sessions. In this case, unused bandwidth should be rapidly consumed by existing session(s). In the Hyper-Active Increase phase, R_{AI} in 4 is replaced by a higher value (50 Mbps) to achieve the rapid growth of consumed bandwidth. Hyper-Active Increase starts when both Byte Counter and Time Counter reach 5.

B. QCN/BS

In multicast communications, multiple bottleneck points (switches) might be observed on tree-shaped multicast transmission path. Taking the ideal protocol for tracking the most congested path under changing network conditions discussed in the LPM problem paper [10] into consideration, we proposed modified QCN for a flow with multiple bottleneck points, QCN/BS [9].

In QCN/BS, the most congested switch is selected. The transmission rate of the sending device is calculated from

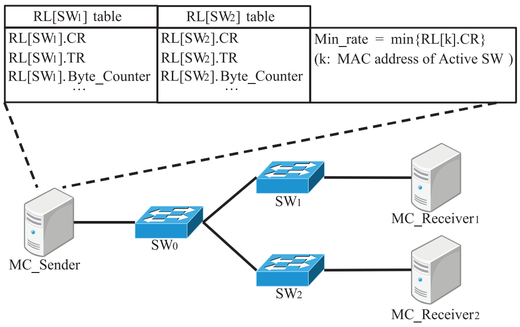


Figure 2. Overview of QCN/BS.

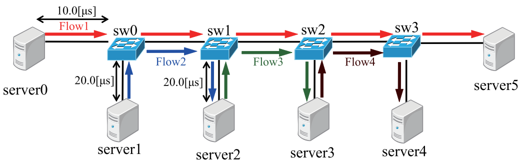


Figure 3. Multiplebottleneck Topology.

the feedback frames sent by the selected switch. The sending device stores the transmission rates for all switches on its multicast tree. To limit the number of transmission rates that the sending device has to manage, only the rates for congested switches should be stored. New transmission rates are calculated for all feedback frames received. The sending device selects the lowest transmission rate, which means it selects the most congested switch. Switches can be identified by their MAC address, which is stored in a feedback frame field.

Figure 2 is a schematic diagram of QCN/BS. In conventional QCN, the source (RP) maintains table based Rate Limiter (RL) for congestion control. The RL table maintains CR, TR, Byte Counter, and Timer Counter data. In QCN/BS, to determine the location of the worst switch congestion, the RL table also needs to maintain RL entries for all congested switches. Each switch is identified by a MAC address. The source (RP) can identify each congested switch and calculate its transmission rate independently.

For QCN/BS, the source (RP) has been modified as follows.

- When a feedback frame arrives

A new transmission rate is calculated for the congested switch, identified by its MAC address. The initial value of the QCN/BS RL table is set as initial value of the convention QCN RL table [12].

- When a data frame is transmitted

In QCN/BS, the source updates transmitted bytes for all entries (*RL*[*i*].*Byte_Counter* for all congested switches *i*); when the Byte Counter reaches a certain value, the transmission rate is increased.

In QCN/BS, the sending device does not need to identify whether a flow is multicast or unicast because QCN/BS only selects a single bottleneck. This means that QCN/BS can be applied to multiple bottlenecks on a unicast path.

TABLE I. Simulation Parameters.

Simulator	ns-2.33
Bandwidth	10[Gbps]
the number of Seeds	20
RTT	100[μs]
Packets size	1500[byte]
Queue length	100[pkts]
Qeq	22[pkts]

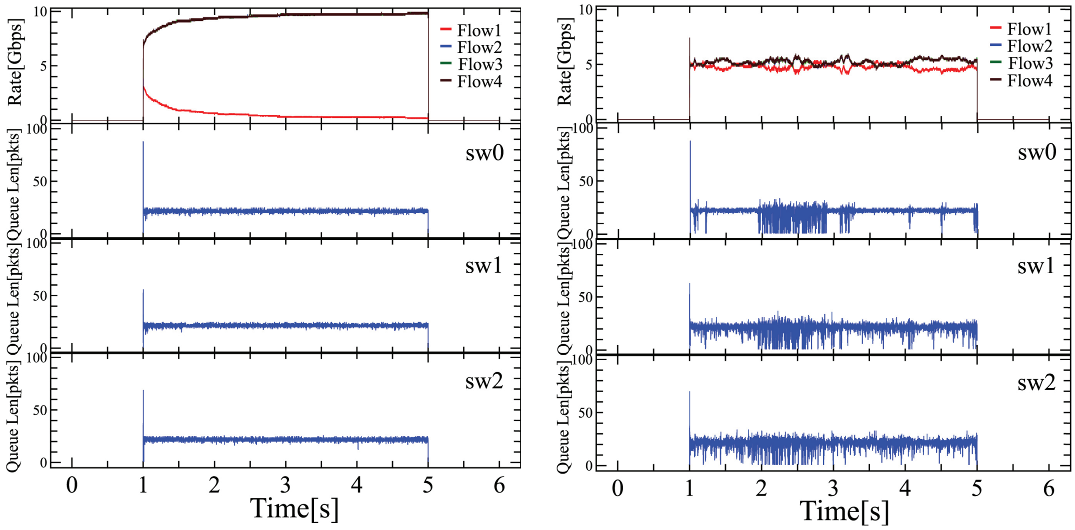
C. *Preliminary Evaluation*

In this section, we preliminary evaluate QCN and QCN/BS in unicast communications with multiple bottleneck points. We use ns2 [13] as the simulation tool. Figure 3 shows the simulation model. Flow 1 passes through 3 bottleneck links each of which is respectively shared with flow2, 3 and 4. All links have identical bandwidth and operate 10Gbps. sw0, sw1 and sw2 are independent bottleneck switches and the CP implemented in these three switches return feedback to the RP of the corresponding sending devices (server0, 1, 2 and 3). RTT of all flows is 100μs and the propagation delay of each link is depicted in this figure. Other simulation parameters are listed in Table I. We use greedy model where transmission rate of the sending device is exactly defined by the QCN algorithm. For the detailed parameters of QCN, we used the parameters recommended in [12].

Figure 4 (a) and (b) show the transmission rate characteristics of the original QCN and QCN/BS, respectively. These figures also show the queue length characteristics of three bottleneck switches. In the original QCN, transmission rate of Flow 1 (red line) is suppressed just after flow is initiated (1.0[sec]). Other flows (Flow 2, 3 and 4) can obtain remained bandwidth, i.e., their transmission rates are significantly higher than Flow1 (Flow 2, 3 and 4 has almost the same transmission rate and these curves in this figure are eventually overlapped). In QCN/BS, transmission rate of Flow 1 is generally close to other flows, which means link bandwidth is fairly shared between one-hop flow and three-hop flow having multiple bottlenecks on its path. Queue length of QCN/BS is slightly fluctuated but no significant overshoot is observed. From congestion control viewpoint, it is good behavior because queue length is kept low. As shown in Table II, link utilization of QCN/BS is generally high, which means QCN/BS does not over-regulate the transmission rate.

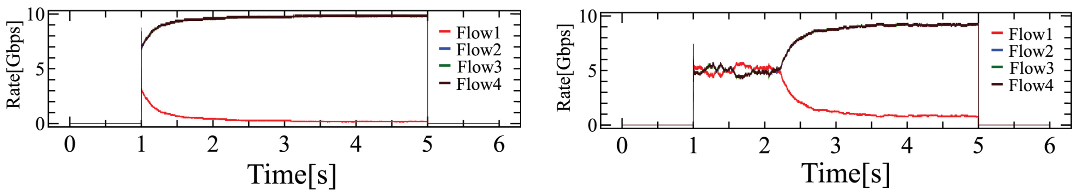
We evaluate the original QCN and QCN/BS for 20 different seeds. For the original QCN, transmission rate of Flow 1 is significantly regulated in all 20 seeds, i.e., all other 19 seeds have almost the same results as Figure 5(a). QCN/BS has good shape (close to other flows) in 7 seeds. However, in our simulation results for QCN/BS, other 13 seeds show too much regulated transmission rate of Flow 1. Figure 5 shows transmission rate characteristics for one of these 13 seeds. So, QCN/BS occasionally improves multi-bottleneck fairness issues in QCN but is a little far from the satisfied solution.

Table III shows the number of received feedback frames for each flow(these results are obtained for the same simulation seed as Figure 5). Flow 1 receives more feedbacks than other flows even though its transmission rate is the lowest (as shown in Figure 5, Flow 1 has the lowest rate). Figure 6 shows CDF of queue length of QCN/BS at each congested switch observed by each flow. At switch 0, flow 1 and 2 observes almost the same queue length. However, at switch 1 and 2, flow 1 observes



(a) originalQCN (b) QCN/BS

Figure 4. Characteristic originalQCN and QCN/BS queue lengths and transmission rate.



(a) originalQCN (b) QCN/BS

Figure 5. Characteristic originalQCN and QCN/BS.

TABLE II. Bottleneck link utilization.

	QCN	QCN/BS
Link Utilization	0.999726	0.999338

slightly larger queue length than one-hop flow (flow 3 and 4 at sw1 and 2, respectively). This curious situation of observing larger queue length is the reason for unfair bandwidth sharing. We try to explain why this situation happens only for Flow 1 by using Figure 7. QCN stabilizes queue length of bottleneck link around pre-defined target queue and enables high link utilization, such as over 0.99. Output process of this highly utilized link (over 0.99) is spaced out with fixed frame length. For a multi-hop flow, an output process of a link is an input process of the subsequent switch. So, input process of flow 1 in sw2 and sw3 is normalized with fixed frame length. Transmission rate of flow 3 and 4 in Figure 5 is defined by QCN rate control and is slightly fluctuated. When their transmission rate is slightly smaller than flow 1, flow 1 might see slightly larger queue size as shown in Figure 7. Flow 1 might see slightly shorter queue size when their (flow 3 or 4's) transmission rate is slightly larger. So, all flows have a possibility to see slightly larger queue length. QCN/BS selects the worst congested switch and adjusts its transmission rate as a calculated rate for feedbacks sent from this selected switch. This behavior causes unfair condition for long-hop flows when compared with short-hop flows even though their observed queue length has similar tendency. For a long-hop flow, eventual decrease of queue length at one switch cannot enforce increase of its transmission rate when another switch is selected and dominates QCN transmission. However, a short-hop flow, such as flow 3 and 4 can increase its transmission rate when it observes decreased queue length. This difference for rate increase behavior brings unfair condition for long-hop flow.

III. Integrated QCN/BS

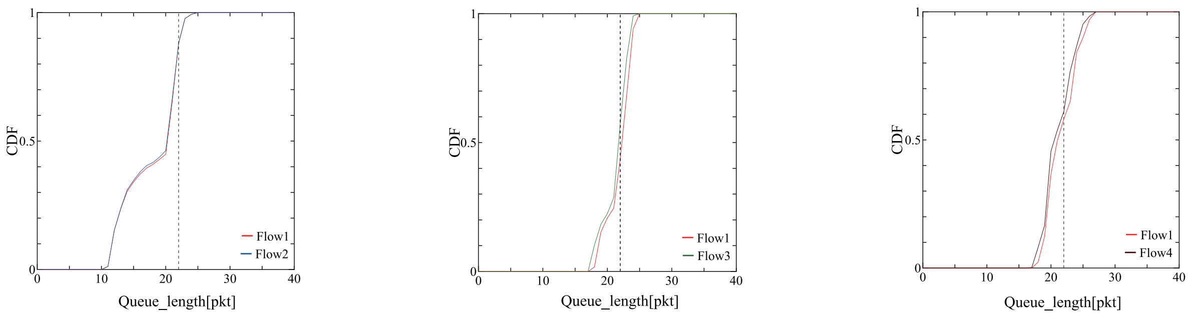
In this section, we propose QCN/BS integrated with Adaptive BC_LIMIT for resolving fairness issues for long hop flow in multiple bottleneck situation. First, we explain Adaptive BC_LIMIT.

A. *Adaptive BC_LIMIT*

When QCN occasionally falls into flow imbalance (unfair) situation, original QCN algorithm tends to maintain a state of this flow imbalance [11]. This is because the increase and decrease in its transmission rate depends on the flow's transmission rate. In all three increase phases, the transmission rates in 3 and 4 are increased when the counters incremented. Byte Counter is incremented whenever a fixed byte amount is transmitted. Therefore, an RP whose current rate is high tend to have a small interval for rate increases, and an RP with a low transmission rate has a long interval. Thus, RPs having a high transmission rate increase their transmission rate more rapidly than those with a low rate, and an unfair flow state might be maintained.

To improve this undesirable rate increase behavior, we proposed Adaptive BC_LIMIT [11]. In Adaptive BC_LIMIT,

Copyright (c) IARIA, 2014. ISBN: 978-1-61208-365-0



(a) sw0 (b) sw1 (c) sw2

Figure 6. CDF of queue length observed by each flow.

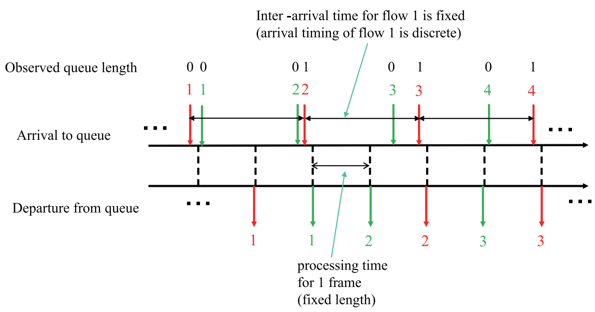


Figure 7. Discrete arrival sees slightly larger queue length.

TABLE III. The number of received feedback frames.

	flow1	flow2	flow3	flow4
sw0	6	13	0	0
sw1	27	0	14	0
sw2	26	0	0	14

the value of BC_LIMIT is defined as follows,

$$BC\_LIMIT = K * Current\_Rate, \quad (5)$$

where parameter K is set to $2.4*10^{-4}$ as defined in our former paper [11].

In Adaptive BC_LIMIT, the time interval between rate increases is independent of the current rate of a flow. This means the opportunities for rate increases are almost the same for all flows, even if their transmission rates (throughput) differ. Flows with a high transmission rate have more opportunities to receive feedback frames leading to rate decreases. This is because a feedback frame is transmitted (with a certain probability) back to the source when a data frame arrives at the CP. Two features of Adaptive BC_LIMIT, i.e., rate increases independent of transmission rate and rate decreases dependent on it, accelerate convergence to fair rate allocation among flows sharing a bottleneck link.

B. QCN/BS Integrated with Adaptive BC_LIMIT

When multiple bottleneck points are located on unicast path, long-hop flow might see slightly larger queue length and suffers throughput degradation. And QCN algorithm itself tends to keep unfair situation as explained in the previous subsection. When QCN algorithm is replaced to Adaptive BC_LIMIT, unfair condition might be instantly improved to fair condition.

Algorithm 1 QCN/BS integrated with Adaptive BC_LIMIT algorithm

Rate Increase Phase
When data transmitted
$BC\_LIMIT[SW_i] = BC\_LIMIT[SW_i] - transmitted\_data$
if $BC\_LIMIT <= 0$ **then**
  Byte Counter is incremented by 1
  $BC\_LIMIT[SW_i] = K * RL[SW_i].CR$ (Adaptive BC_LIMIT)
end if
When timer is expired
Time Counter is incremented by 1
When Byte Counter or Time Counter is incremented
if Byte Counter < 5 && Time Counter < 5 **then**
  In Fast Recovery
else
  if (Byte Counter => 5 && Time Counter < 5) || (Byte Counter < 5 && Time Counter => 5) **then**
    In Active Increase
  else
    if Byte Counter => 5 && Time Counter => 5 **then**
      In Hyper Active Increase
    end if
  end if
end if

Rate Decrease Phase
When RP receives feedback from SW_i
if SW_i already has table $RL[SW_i]$ **then**
  $RL[SW_i].Byte\_Counter$ and $RL[SW_i].Time\_Counter$ are initialized
  $BC\_LIMIT[SW_i] = K * RL[SW_i].CR$ (Adaptive BC_LIMIT)
  $RL[SW_i].CR$ decreases as in the conventional QCN
else
  RP creates a new table $RL[SW_i]$
end if
for all SW_i **do**
  if $Min\_rate > RL[SW_i].CR$ **then**
    $Min\_rate = RL[SW_i].CR$
  end if
end for
$Transmission\_rate = Min\_rate$

Inspired by this idea, we propose, in this paper, integration of QCN/BS and Adaptive BC_LIMIT to resolve fairness problem of long-hop flow in multiple bottleneck condition. Algorithm 1 shows our proposed QCN/BS integrated with Adaptive BC_LIMIT. As shown in this algorithm, transmission rate for each switch sending feedback frame(s) is managed at

 ISBN: 978-1-61208-365-0

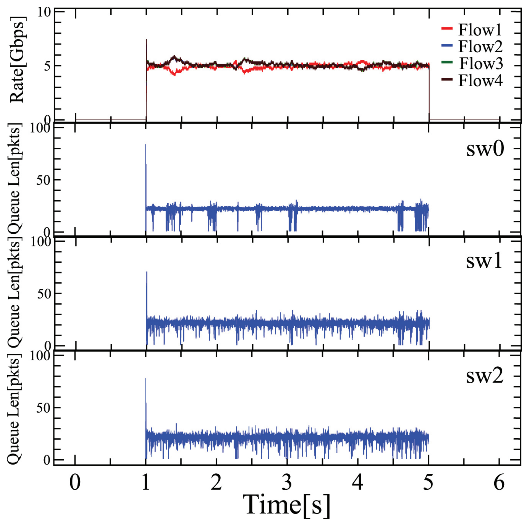


Figure 8. Characteristic QCN/BS integrated with Adaptive BC_LIMIT queue lengths and transmission rate.

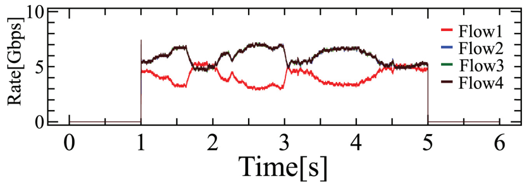


Figure 9. Characteristic QCN/BS integrated with Adaptive BC_LIMIT transmission rate.

the source (RP). In transmission increase phase, transmission rate is increased every time transmitted bytes reach BC_LIMIT (this BC_LIMIT is adaptively computed as shown in (5)). The transmission rate of the sender (RP) is defined as the lowest transmission rate among its managed rates (for all congested switches).

IV. Performance Evaluation

In this section, we evaluate our proposed QCN/BS integrated with Adaptive BC_LIMIT. We use the same simulation model and parameters as section II. Figure 8 shows transmission rate and queue length characteristics of QCN/BS integrated with Adaptive BC_LIMIT (simulation seed is the same one as Figure 5). With integration with Adaptive BC_LIMIT, transmission rate of Flow1 is improved towards fair bandwidth allocation. Adaptive BC_LIMIT stabilizes all flows transmission rate towards fair situation, which improves fair bandwidth allocation to Flow 1. Among 20 simulation seeds, we only find 3 seeds which has slight worse situation. Figure 9 shows the worst case among these 3 seeds. As shown in this figure, even though imbalance between Flow1 and other flows can be observed in short time period, it can be regained towards fair condition. Other 17 seeds give quite good fair condition as shown in Figure 8.

V. Conclusion

QCN/BS is layer 2 congestion control applicable to multiple bottleneck points. Originally it has been proposed for multicast communications where multiple bottlenecks are generally observed on multicast tree, but QCN/BS can be applied to unicast communications with multiple bottleneck points. In the paper, we preliminary evaluate QCN and QCN/BS for unicast communications in multiple-bottleneck situation. Our evaluation results show that QCN/BS occasionally improves fairness issues but cannot resolve unfair condition to long-hop flow completely. We reveal that the reason for this fairness issue is that long-hop flow observes slightly longer queue and receives more congestion feedbacks. To resolve this unfair situation, we integrate QCN/BS with Adaptive BC_LIMIT, which accelerates stabilization towards fair condition. Our simulation results show that our proposed integrated QCN/BS can significantly improve fairness issues for long-hop flow in multiple bottleneck situation.

References

[1] "IEEE 802.1 Data Center Bridging Task Group." http://www.ieee802.org/1/pages/dcbridges.html. [retrieved : may 2014]

[2] R. Hays and H. Frasier, "40G Ethernet Market Potential," IEEE 802.3 HSSG Interim Meeting, Apr. 2007.

[3] M. Alizadeh, et al., "Data Center Transport Mechanisms: Congestion Control Theory and IEEE Standardization," in Proc. Annual Allerton Conference, Sep. 2008, pp. 1270-1277.

[4] "IEEE802.1 Qau." http://www.ieee802.org/1/pages/802.1au.html. [retrieved : may 2014]

[5] M. Al-Fares, A. Loukissas, and A. Vahdat, " A Scalable, Commodity Data Center Network Architecture," in Proc. ACM SIGCOMM, Seattle, WA, USA, Aug. 2008, pp. 63-74.

[6] A. Greenberg, et al., "VL2: a Scalable and Flexible Data Center Network, " in Proc. ACM SIGCOMM, Barcelona, Spain, Aug. 2009, pp. 51-62.

[7] T. Benson, A. Anand, A. Akella, and M. Zhang, "MicroTE: Fine Grained Traffic Engineering for Data Centers," in Proc. ACM CoNEXT 2011, Tokyo, Japan, Dec. 2011, pp. 6-9.

[8] F. Tso and D. Pezaros, " Improving Data Center Network Utilization Using Near-Optimal Traffic Engineering," IEEE Trans. on Parallel and Distributed Systems, June 2013, Vol.24, No.6, pp. 1139-1148.

[9] Y. Tanisawa, Y. Hayashi, and M. Yamamoto, " Quantized Congestion Notification for Multicast in Data Center Networks," in Proc. International Conference on Cloud Networking(IEEE CLOUDNET 2012), Paris, France, Nov. 2012, pp. 51-56.

[10] S. Bhattacharyya, D. Towsley, and J. Kurose, "The Loss Path Multiplicity Problem in Multicast Congestion Control," in Proc. IEEE INFOCOM, Mar. 1999, pp. 856-863.

[11] Y. Hayashi, H. Itsumi, and M. Yamamoto, " Improving Fairness of Quantized Congestion Notification for Data Center Ethernet Networks," DCPerf 2011(The 1st international Workshop on Data Center Performance), Minneapolis, USA, June. 2011, pp. 20-25.

[12] R. Pan, "QCN Pseude Code Version 2.1." http://www.ieee802.org/1/files/public/docs2008/au-pan-qcn-serial-hai-2-1-0408.zip. [retrieved : may 2014]

[13] NS2, Networks Simulator, 1991. Available at http://www.isi.edu/ns-nam/ns/ [retrieved : may 2014]