Sliding Mode Congestion Control for Data Center Ethernet Networks

Wanchun Jiang, Fengyuan Ren, *Member, IEEE*, Ran Shu, Yongwei Wu, and Chuang Lin, *Senior Member, IEEE*

Abstract—Recently, Ethernet is enhanced as the unified switch fabric of data centers, called data center Ethernet. One of the indispensable enhancements is end-to-end congestion management, and currently quantized congestion notification (QCN) has been ratified as the corresponding standard. However, our experiments show that QCN suffers from large oscillations of the queue length at the bottleneck link such that the buffer is emptied frequently and accordingly the link utilization degrades, with certain system parameters and network configurations. This phenomenon is corresponding to our theoretical analysis result that QCN fails to enter into the sliding mode motion (SMM) pattern with certain system parameters and network configurations. Knowing the drawbacks of QCN and realizing the advantage that congestion management system is insensitive to the changes of parameters and network configurations in the SMM pattern, we present sliding mode congestion control (SMCC), which can enter into the SMM pattern under any conditions. SMCC is simple, stable, fair, has short response time, and can be easily used to replace QCN because both of them follow the framework developed by the IEEE 802.1Qau work group. Experiments on the NetFPGA platform show that SMCC is superior to QCN, especially when traffic pattern and network states are variable.

Index Terms—Data center ethernet, sliding mode motion, quantized congestion notification

1 Introduction

Currently, there is substantial interest in enhancing Ethernet as the unified switch fabric for the TCP/IP networks, the storage area networks (SANs) and the high performance computing (HPC) networks in data centers [1], [2], [32]. The benefits are that deploying the unified switch fabric can decrease the number of redundant devices and simplify the design and management of data center networks (DCNs). The reasons for choosing Ethernet as the unified switch fabric are that Ethernet is cheap, simple, widely used and broadly compatible, and the speed of Ethernet is increasing rapidly. Recently, 10 Gbps Ethernet is ready for commercial application, and the standards for both 40 and 100 Gbps Ethernet have been ratified [3]. The disadvantage for Ethernet being the unified switch fabric is that it only provides best-effort service, which fails to satisfy the lossless and low latency requirements of SANs and HPC. The enhancements of Ethernet will fill the performance gap between the traditional Ethernet and the unified switch fabric. The IEEE 802.1 data center bridging task group [1] are standardizing these enhancements. The enhanced Ethernet is called data center bridging (DCB), data center Ethernet (DCE) or converged enhanced Ethernet (CEE). Meanwhile, techniques such as FCoE [9], [32] and RoCEE [10] make effort to accommodate traffics of both SANs and HPC to DCE.

Congestion occurs when links are oversubscribed and traffic is excessive. In DCNs, the link over-subscription ratio is large [4], and the traffic often occurs in burst [5], [6]. Hence, congestion naturally happens frequently in DCNs. Moreover, as the priority-based pause mechanism [7] is developed for DCE to prevent dropping packets, it also results in the saturation tree problem [8], where heavy congestion occurs. Therefore, the end-to-end congestion management is indispensable. Since kinds of traffics either don't involve any congestion management mechanism (such as UDP traffic) or have different congestion management mechanisms (such as TCP traffic and SAN traffic), it would be more economical and efficient to deploy a uniform congestion management scheme on the link layer for all the traffics in DCNs.

In Mar. 2010, the quantized congestion notification (QCN), developed by the IEEE 802.1Qau work group, has been ratified to be the standard for the end-to-end congestion management scheme of DCE [1]. Presently, QCN has been implemented in devices, such as Juniper QFX3500 [12] and FocalPoint FM6000 [13]. Historically, QCN is heuristically designed. Up to now, there are only a few theoretical results on QCN, because QCN involves the segmented nonlinearity, which is intractable through the classic linear analytical method. As a result, QCN has not been understood thoroughly. In this paper, we investigate the end-to-end congestion management scheme for DCE. More specifically, we make the following contributions.

We reveal drawbacks of QCN through experiments and theoretical analysis. Experiments show that the performance

- *W. Jiang is with the School of Information Science and Engineering, Central South University, Changsha, China and with the Tsinghua National Laboratory for Information Science and Technology, Department of Computer Science and Technology, Tsinghua University, Beijing, China. E-mail: web_191@163.com.*
- *F. Ren. R. Shu, Y. Wu and C. Lin are with the Tsinghua National Laboratory for Information Science and Technology, Department of Computer Science and Technology, Tsinghua University, Beijing, China. E-mail: {renfy, wuyw, chlin}@tsinghua.edu.cn, shur11@mails.tsinghua.edu.cn.*

Manuscript received 7 June 2013; revised 29 Sept. 2014; accepted 26 Oct. 2014. Date of publication 19 Nov. 2014; date of current version 12 Aug. 2015. Recommended for acceptance by Y. Yang.
For information on obtaining reprints of this article, please send e-mail to: reprints@ieee.org, and reference the Digital Object Identifier below.
Digital Object Identifier no. 10.1109/TC.2014.2366733

of QCN depends on both parameters settings and network configurations. With the changes of parameters or network configurations, QCN may suffer from large oscillations of the queue length at the bottleneck link such that the buffer is emptied frequently and accordingly the link utilization degrades. Meanwhile, theoretical analysis shows that QCN approaches to the stable state mainly through a special motion pattern, called sliding mode motion. However, whether QCN is able enter into the sliding mode motion pattern depends on both parameters settings and network configurations. Accordingly, QCN fails to reach the stable state via the sliding mode motion pattern with certain parameters and network configurations. This is corresponding to the bad performance of QCN in experiments.

Realizing the drawbacks of QCN and knowing the advantage that the performance of congestion management system is insensitive to changes of parameters and network configurations in the sliding mode motion pattern, we design the sliding mode congestion control (SMCC) scheme for DCE. SMCC follows the congestion management framework defined by the IEEE 802.1Qau work group, namely, QCN can be easily replaced by SMCC. Furthermore, SMCC is superior to QCN in the following aspects.

- Compared with QCN, which unconsciously utilizes the advantages of the sliding mode motion pattern, SMCC is able to enter into the sliding mode motion pattern under any conditions. Therefore, SMCC can always approaches to the stable state via the sliding mode motion pattern, and the performance of SMCC is insensitive with the changes of parameters settings and network configurations in the stable state. Namely, SMCC is stable and robust.
- SMCC is much simpler than QCN. Hence, the hardware implementation of SMCC is more economical.
- SMCC is more responsive than QCN. The core rate adjustment algorithm of QCN is originated from BIC-TCP [17]. It feeds back the weighted sum of the queue length and its derivative, and does a binary search for rate increase without using the feedback information. In SMCC, the queue length and its derivative are separately involved in the feedback packet. Hence, more feedback information about the congestion is obtained. With the feedback information for both rate decrease and rate increase, SMCC can approach to proper sending rate directly instead of searching several times as QCN does.

We evaluate SMCC on the NetFPGA platform [18] since both QCN and SMCC are designed for hardware implementation. Experimental results indicate that SMCC is stable, fair, has a short response time and can adapt to the changes of system parameters and network configurations in DCE. Moreover, SMCC outperforms QCN, especially in the condition that the traffic pattern and the network state are variable. We also conduct simulations with NS2, which confirm that SMCC works well in face of short-lived burst traffic and tree topology.

The rest of this paper is organized as follows: Section 2 introduces the background. And then the motivation behind the work is presented in Section 3. Subsequently, Section 4 describes the design of the SMCC scheme and Section 5 evaluates SMCC on the NetFPGA platform. Finally, Section 6 concludes this paper.

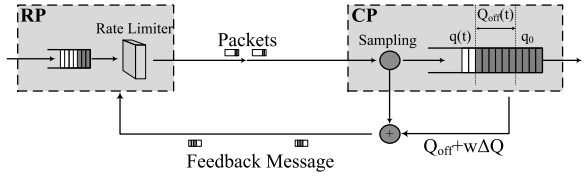


Fig. 1. Current framework of the congestion management mechanism developed by the IEEE 802.1Qau work group.

2 Background

2.1 Congestion Management in DCNs

There exists a rich history of design and control-theoretic analysis of congestion management scheme for the Internet [19], [20], [21]. However, the environment of congestion management in DCE differs from that in Internet. The main differences are in the following aspects:

- There is no per-packet ACK in the Ethernet. Hence, it is hard to estimate the round trip time, and the congestion control algorithm cannot be automatically self-clocked like TCP.
- The traffic is highly burst. e.g. traffic can potentially arrive at the speed of the network interface cards (NICs).
- The switch buffer size is much smaller than the router buffer size.
- The link lengths are small in data center (within 500 m), which implies that the propagation delay is small (within 3 μs). Therefore, the bandwidth delay product (BDP) is small in DCE.

Furthermore, the congestion management scheme faces additional requirements for DCE being the unified switch fabric in DCNs.

Simple. The congestion management algorithm should be simple enough to be implemented completely in hardware to handle traffics at the speed of 1 or 10 Gbps in DCE.

Losslessness. Links may be paused and packets cannot be dropped in data center, because upper layer protocols for storage traffic rely on no frame loss to achieve low processing overhead and high performance.

Low delay. To carry the HPC traffic over Ethernet, DCE should keep controllable low delay.

These environment and requirements specialize the design of the congestion management scheme in DCE.

The IEEE 802.1Qau work group [22] have worked on the end-to-end congestion management schemes of DCE for several years and four proposals have been released up to now. In these proposals, various methods used in traditional congestion management schemes, such as explicit feedback v.s. implicit feedback, and rate based load sensor v.s. queue based load sensor, are compared. At last, proper methods are embedded into the final framework. Now, QCN is adopted as the final standard.

2.2 Framework Developed by the IEEE 802.1Qau

The current framework of the congestion management scheme developed by the IEEE 802.1Qau work group is composed of two parts as shown in Fig. 1.

The switch, or the congestion point (CP). The task of CP is to detect congestion, generate feedback packets and send them back to sources.

The source, or the reaction point (RP). The goal of RP is to adjust the sending rate according to the feedback information.

Generally, CP monitors the queue length and "samples" incoming packets periodically with probability p to generate the feedback packets. The feedback information F_b, which represents the current congestion state, consists of two parts: the current offset of the queue length ($Q_{off} = q(t) - q_0$) and the variance of the queue length in a sampling interval ($\Delta Q = q(t) - q_{old}$), where q_0 is the target queue length and q_{old} is the queue length at the latest sampling. F_b is given by

$$F_b = -(Q_{off} + w * \Delta Q), \tag{1}$$

where w is the weight. RP receives the feedback packet and then adjusts the sending rate according to the feedback information involved in the packet. The rate adjustment is achieved by implementing the rate limiter at the edge switch or the NICs.

Historically, the first proposal BCN [23] developed by IEEE 802.1Qau work group uses the feedback information for both rate decrease and rate increase. There is also a CPID flag involved in the feedback packets, which univocally identify the congestion point. BCN responds to the congestion using a additive increase and multiplicative decrease (AIMD) like algorithm, which is defined as follows:

$$r \leftarrow \begin{cases} r(1 + G_d F_b) & when \quad F_b < 0 \\ r + G_i R_u F_b & when \quad F_b > 0. \end{cases} \tag{2}$$

In (2), G_d is a constant chosen such that $G_d |F_{bmax}| = \frac{1}{2}$, i.e., the sending rate is decreased no more than 50 percent each time, G_i is the factor of rate increase and R_u is the unit of rate increase. The rate decrease algorithm of QCN is the same as BCN. But QCN employs a self-increasing algorithm for rate increase, similar to BIC-TCP. Thus, QCN only needs to generate feedback packets involving negative feedback information and doesn't need to identify the congestion point. Let R denote the sending rate just before the arrival of the latest feedback packet. The rate increase algorithm of QCN is as follows:

Fast recovery. Immediately after the rate decrease, RP enters into the state of fast recovery. Fast recovery persists five cycles and the time length T of each cycle is set to be the time of sending 150 KB data. At the end of each cycle, R keeps unchanged and r is updated by

$$r \leftarrow \frac{1}{2}(r + R). \tag{3}$$

Obviously, QCN does binary search in the process of fast recovery and R is called the target sending rate.

Active increase. After the process of fast recovery, RP enters into the active increase state, probing for more available bandwidth. In the state of active increase, R and r are updated by

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

where R_{AI} is the constant unit of rate increase. The updating is executed at the end of transmitting 75 KB data. Namely the time length of each cycle in the active increase state is half of that in the fast recovery state.

We present only the core mechanisms of both BCN and QCN above. More details can be found in [22]. In general, QCN is more aggressive than BCN during rate increase, and generates less feedback packets than BCN. The fast recovery mechanism is borrowed from the BIC-TCP algorithm, which is an improvement to the AIMD algorithm for the network with high BDP. The active increase mechanism of QCN is a heuristic improvement of the fast recovery mechanism. QCN still has other add-on improvements: R_{AI} is replaced by a larger constant R_{HAI} when the sending rate at the state of active increase is really large, the extra fast recovery mechanism and the target rate reduction mechanism [24]. Thus, QCN is complex. Moreover, the rate adjustment algorithm of QCN is heuristically designed. Although proper parameters of QCN can be obtained for certain network configurations through experiments and simulations, the limited experiments and simulations can hardly cover the unlimited network configurations. As a result, parameters of QCN are hard to be chosen. In addition, QCN has been shown to be unfair in lots of literature [14], [15].

2.3 Sliding Mode Motion

Given an autonomous system described by differential equation $\ddot{u}(t) = f(u(t), \dot{u}(t))$, the trajectory of its solution can be drawn on the phase plane by connecting points $(u(t), \dot{u}(t))$ along the direction where time t increases. The phase trajectories of second order differential equations have been presented by lots of literature [26]. There are also a wealth of methods to draw the phase trajectories of all kind of second order differential equations, such as the isocline, the liénard plane and the delta method [26]. On the contrary, it's always difficult to find the analytical solution of differential equations explicitly. Therefore, drawing phase trajectory is superior to displaying $u(t)$ and $\dot{u}(t)$ against time t respectively.

Take the following differential equations as example:

$$\begin{cases} \frac{du(t)}{dt} = v(t) \\ \frac{dv(t)}{dt} = -bu(t) - av(t). \end{cases} \tag{5}$$

In the differential equations, parameters a and b are positive constants. The characteristic equation of (5) is

$$z^2 + az + b = 0 \tag{6}$$

and the corresponding characteristic roots are

$$\lambda_{1,2} = -\frac{-a \pm \sqrt{a^2 - 4b}}{2}. \tag{7}$$

When $a^2 < 4b$, both λ_1 and λ_2 are complex numbers, and the solution of differential equations (5) is

$$\begin{cases} u(t) = Ae^{\alpha t} cos(\beta t + \psi) \\ v(t) = A\alpha e^{\alpha t} cos(\beta t + \psi) - A\beta e^{\alpha t} sin(\beta t + \psi), \end{cases} \tag{8}$$

where $\alpha = -\frac{a}{2}$, $\beta = \frac{\sqrt{4b - a^2}}{2}$, and A, ψ are coefficients associated with parameters a, b and the initial value. Based on equation (8), there is obviously

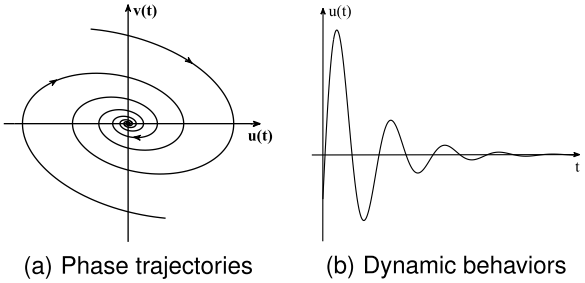


Fig. 2. The phase trajectories and dynamical behaviors of differential equations (5) when $a^2 < 4b$.

$$[v(t) - \alpha u(t)]^2 + \beta^2 u^2(t) = (A\beta)^2 e^{2\alpha t}. \quad (9)$$

Let $u(t)$ be the abscissa and let $v(t)$ be the ordinate, the phase trajectory of equation (9) can be drawn easily on the phase plane. As shown in Fig. 2a, the basic shape of the phase trajectories is spiral and the arrows point out the direction where the time t increases. The curvature of the spiral is associated with parameters a and b. Moreover, Fig. 2a also implies that $u(t)$ converges to 0 with the increase of time t, as shown in Fig. 2b. In reality, the values of A and ψ in equation (8) are usually hard to compute, while the phase trajectories shown in Fig. 2 is easy to draw.

Generally, a congestion management scheme deploy two different algorithms for rate increase and rate decrease, respectively. Both algorithms can be described by differential equations by building the flow-fluid model. Assume the rate increase area is defined by $u(t) + mv(t) < 0$, the rate decrease area is defined by $u(t) + mv(t) > 0$, the differential equation (5) describes the rate decrease algorithm, and the rate increase algorithm is described by the following differential equations,

$$\begin{cases} \frac{du(t)}{dt} = v(t) \\ \frac{dv(t)}{dt} = g - hv(t), \end{cases} \quad (10)$$

where parameters g and h are positive constants. As shown in Fig. 3, the basic shape of the phase trajectories defined by differential equations (10) is hyperbola. The shape of the phase trajectory is very important referring to the stability analysis. If the differential equations (5) is the only congestion management algorithm, the phase trajectory would converge to the origin once $a^2 < 4b$, referring to Fig. 2. Namely, the system would be stable once $a^2 < 4b$. Similarly, if differential equations (10) is the only congestion management algorithm, the congestion management system isn't stable, referring to Fig. 3. When both (5) and (10) take effects

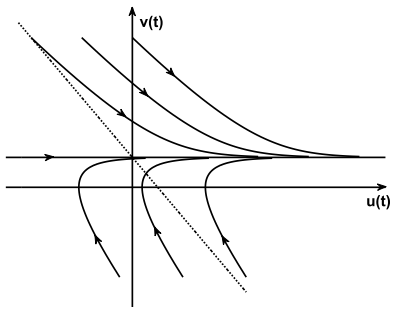


Fig. 3. Phase trajectories whose shape is hyperbola.

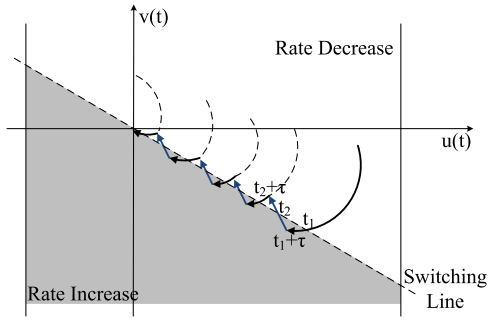


Fig. 4. Details of the sliding mode motion pattern.

as what we assumed before, the sliding mode motion pattern [16] may occur as we will show below.

Assume there exists a delay τ in the switching process. Starting from the rate decrease area, the phase trajectory of the congestion management system will reach the switching line $u(t) + mv(t) = 0$ at time t_1 along the spiral, as the black solid curve shown in Fig. 4. Were there no delay τ, the congestion management system would be controlled by (10) at time t_1, namely the shape of the phase trajectory would become hyperbola at time t_1. But, this will happen only at time $t_1 + \tau$ due to the delay. And during the time $[t_1, t_1 + \tau]$, the phase trajectory manages to "pierce" into the rate increase area along the spiral. Thus, at the time $t_1 + \tau$, the phase trajectory starts from one point in the rate increase area, rather than in the switching line, and then approaches to the switching line again along the hyperbola, as the blue solid curve shown Fig. 4. Assume it hits the switching line at time t_2. But the shape of the phase trajectory doesn't change until time $t_2 + \tau$, when the phase trajectory has already "trespassed" into the rate decrease area. At time $t_2 + \tau$, the shape of the phase trajectory changes and the phase trajectory returns to the rate increase area again, and so on. As a result, QCN keeps switching frequently between the rate increase area and the rate decrease area. As the solid line shown in Fig. 4, the phase trajectory moves "on the average" toward the origin along the switching line, while oscillating in this way.

The smaller τ, the smaller is the amplitude of the oscillations and the higher the frequency of the switching. When $\tau \to 0$, the phase trajectory will oscillate with infinitesimal amplitude and infinitely high frequency, so that the phase trajectory will move along the switching line, namely the congestion management system will be described by the differential equation

$$u(t) + mv(t) = 0. \quad (11)$$

This special motion pattern is called sliding mode motion, and the part of the switching line where the sliding mode motion pattern occurs is called sliding regime. According to [16], we have

Theorem 1. *The necessary and sufficient condition for the occurrence of the sliding mode motion pattern here is*

$$\lim_{\dot{u}(t)+m\dot{v}(t)\to 0^+} [u(t) + mv(t)][\dot{u}(t) + m\dot{v}(t)] \leq 0.$$

The inequality in Theorem 1 means that the phase trajectory moves towards the switching line defined by differential equations (11) from both sides of it.

The sliding mode motion pattern has three advantages. The first one is that the congestion management system is equivalent to a lower order system in the sliding mode motion pattern. In our example, the behaviors of the second order congestion management algorithms are described by the first order differential equation (11). The second advantage is that the performance of congestion management system becomes insensitive to the changes of system parameters and external disturbances in the sliding mode motion pattern. In our example, with the slight change of system parameters a, b, g and h, the slope of velocity vectors of the phase trajectory changes, but the congestion management system still keeps staying in the sliding mode motion pattern. The change of these parameters have no impacts on the differential equation (11), which describes the behaviors of the congestion management system. Similarly, a disturbance undoubtedly distorts the phase trajectories too. However, once the basic shape of the phase trajectories keeps unchanged, these phase trajectories point in the opposite directions in the neighborhood of the switching line, as shown in Fig. 4. Hence, the sliding mode motion pattern persists. In other word, the sliding mode motion pattern is invariant with respects to disturbance. The third advantage is that the response time can become as short as possible by setting system parameters a, b, g and h properly, because the performance of congestion management system is insensitive to them in the sliding mode motion pattern.

3 Motivation

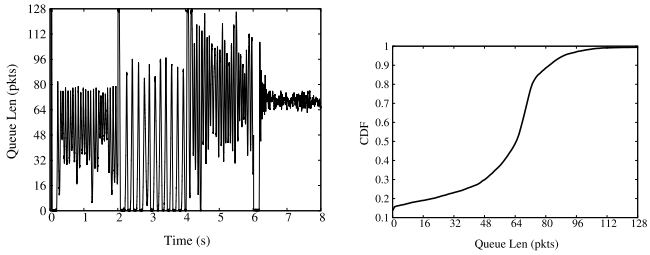
This section reports our empirical study of QCN by experimental observations and theoretical analysis. More precisely, we observe some interesting phenomena in experiments and obtain some insights through theoretical analysis. They serve as the motivation of designing the SMCC scheme for DCE.

3.1 Experimental Study on QCN

To explore the performance of QCN with the changes of system parameters and network configurations, we implement QCN on the NetFPGA [18] platform and conduct experiments on the dumbbell topology. The initial parameters setting of QCN is the same as [24] excepting that we set $q_0 = 64$ KB for the convenience of observation

- At CP: $w = 2, p = 0.01$.
- At RP: $G_d = \frac{1}{128}$, $R_{AI} = 1$ Mbps and T is the time of sending 150 KB data.

The buffer size is set to be 128 KB and the link capacity is 1 Gbps. At the beginning of the experiment, two RPs start flows at the speed of 1 Gbps. After initializing, the speed of these flows will be controlled by the rate adjustment rule of RPs. Then, an extra background flow of fixed sending speed 500 Mbps preempts the bandwidth of bottleneck link at the 2nd second. Subsequently, parameter T is reset to be the time of sending 30 KB data at the 4th second. The extra background flow stops at the 6th second.



(a) Evolution of queue length (b) CDF of queue length

Fig. 5. Queue length of QCN at the bottleneck link.

The evolution of queue length at the bottleneck link are shown in Fig. 5a. In the first two seconds, the queue oscillates around the stable point. When the bandwidth shared by RPs is preempted at the 2nd second, the aptitude of the oscillations increases such that the queue becomes empty frequently. Accordingly, the link utilization heavily degrades. After the parameter T is changed at the 4th second, the aptitude of the oscillations of queue length is reduces. After the extra flow stops at the 6th second, the aptitude of the oscillations decreases greatly. Note that, the network configuration in $[0, 2\,\mathrm{s}]$ is the same as that in $[6, 8\,\mathrm{s}]$ and the network configuration in $[2, 4\,\mathrm{s}]$ is the same as that in $[4, 6\,\mathrm{s}]$. Comparing the evolutions of the queue length between $[0, 2\,\mathrm{s}]$ and $[6, 8\,\mathrm{s}]$, or between $[2, 4\,\mathrm{s}]$ and $[4, 6\,\mathrm{s}]$, we can conclude that the performance of QCN depends on the value parameter T. The parameters setting of QCN in $[0, 2\,\mathrm{s}]$ is the same as that in $[2, 4\,\mathrm{s}]$ and the parameters setting of QCN in $[4, 6\,\mathrm{s}]$ is the same as that in $[6, 8\,\mathrm{s}]$. Comparing the evolutions of the queue length between $[0, 2\,\mathrm{s}]$ and $[2, 4\,\mathrm{s}]$, or between $[4, 6\,\mathrm{s}]$ and $[6, 8\,\mathrm{s}]$, we can conclude that the change of bandwidth also affects the performance of QCN. Similar results can be obtained when some other parameters or network configurations are changed.

Repeating this experiment eight times, we have also drawn the cumulative distribute function (CDF) graph of the queue length. As shown in Fig. 5b, the probability that the queue length becomes zero is larger than 10 percent. Obviously, the phenomena that queue length becomes zero implies the degradation of the link utilization. For example, corresponding to the frequently emptied buffer, the link utilization degrades to 96.8 percent in $[2, 4\,\mathrm{s}]$.

In total, QCN can hold the queue length around the state point tightly with proper parameters setting and network configuration. But its performance may degrade with the changes of system parameters and network configurations.

3.2 Theoretical Analysis on QCN

Concerning on the bottleneck link, we build a fluid-flow model for QCN. Let $r(t)$ denote the sending rate of each source, since sources are always homogeneous due to symmetrical network topologies such as Fat-Tree [4] and the special computing paradigm such as Map-Reduce in data centers. We neglect the delay because the propagation delay is relatively small in data center, and then have

$$\frac{dq(t)}{dt} = Nr(t) - C \tag{12}$$

and

$$\Delta Q = \Delta t \frac{dq(t)}{dt} = \frac{1}{pC}[Nr(t) - C] \tag{13}$$

at the bottleneck link. Consequently, equation (1) can be rewritten as

$$F_b(t) = -[q(t) - q_0] - \frac{w}{pC}[Nr(t) - C], \tag{14}$$

where N is the number of active flows sharing the bottleneck link, C denotes the capacity of the bottleneck link, and other variables are defined the same as that in Section 2.2. The rate adjustment algorithm of QCN can be modeled by

$$\begin{cases} \frac{dr(t)}{dt} = G_d F_b(t) r(t) & Rate\ Decrease \\ \frac{dr(t)}{dt} = -\frac{r(t)}{2T} + \frac{R(0)}{2T} & Fast\ Recovery \\ \frac{dr(t)}{dt} = -\frac{r(t)}{T} + \frac{R(0)}{T} + \frac{2R_{AI}t}{T^2} & Active\ Increase, \end{cases} \tag{15}$$

where $R(0)$ is the target sending rate in the state of fast recovery. For the sake of simplicity, we make a substitution

$$\begin{cases} x(t) = q(t) - q_0 \\ y(t) = Nr(t) - C. \end{cases} \tag{16}$$

In this way, $x(t)$ denotes the offset of the queue length towards the target point, $y(t)$ is associated with the sending rate, and the equilibrium point of the system is transferred to $(x(t), y(t)) = (0, 0)$. Combining (12), (14) and (15), the differential equations describing the rate decrease subsystem of QCN can be obtained

$$\begin{cases} \frac{dx(t)}{dt} = y(t) \\ \frac{dy(t)}{dt} = -G_d[x(t) + \frac{w}{pC}y(t)][y(t) + C]. \end{cases} \tag{17}$$

Note that equation (17) shows that the rate decrease subsystem of QCN is nonlinear, namely this multiple decrease method of QCN is different to that of the classic AIMD algorithm. The procedure of fast recovery in QCN can also be represented by

$$\begin{cases} \frac{dx(t)}{dt} = y(t) \\ \frac{dy(t)}{dt} = \frac{NR(0)-C}{2T} - \frac{y(t)}{2T}. \end{cases} \tag{18}$$

With the same method, the active increase procedure and the add-on improvements of QCN can also be described respectively by differential equations associated with $x(t)$ and $y(t)$, which are similar to (17) and (18). Hence, the behaviors of the QCN system is equivalent to the evolution of the queue length, $q(t)$ or $x(t)$, which are constrained by the corresponding differential equations. For the convenience of retrieval, we summarize the definition of variables into Table 1.

Lyapunov has shown that the stability and the behaviors of the nonlinear system, such as (17), in the neighborhood of a singular point can be found from the linearized version of nonlinear differential equations about the singular point [25]. The linearized version of (17) about the singular point, i.e. the equilibrium point $(x(t), y(t)) = (0, 0)$, is

TABLE 1
Definition of Parameters

Parameters	Definitions
C	Link capacity
N	Number of flows sharing bottleneck link
w	Weight of the derivative of queue length
p	Sampling probability
T	Interval of cycle of fast recovery
G_d	Factor for rate decrease
R_{AI}	Unit of active increase
q_0	Target queue length
F_b	Feedback information of queue length

$$\begin{cases} \frac{dx(t)}{dt} = y(t) \\ \frac{dy(t)}{dt} = -G_d C x(t) - \frac{G_d w}{p} y(t) \end{cases} \tag{19}$$

and its characteristic equation is

$$z^2 + \frac{G_d w}{p} z + G_d C = 0. \tag{20}$$

Since the forms of (19) and (20) are the same as that of (5) and (6) respectively, the shape of the phase trajectory defined by (19) is also spiral when $\frac{G_d^2 w^2}{p^2} < 4G_d C$ (i.e., $\sqrt{\frac{G_d}{4C}}\frac{w}{p} < 1$), as shown in Fig. 2. Similarly, comparing (18) with (10), we can know that the shape of the phase trajectory defined by (18) is hyperbola when $NR(0) - C > 0$, as shown in Fig. 3. Therefore, QCN may enter into the sliding mode motion pattern, as we have shown in Section 2.3.

Fig. 6 shows the combination of the phase trajectories of Figs. 2 and 3. When the QCN system is in the rate decrease area, the shape of the phase trajectory, namely the black solid curve in Fig. 6, is the same as the one shown in Fig. 2. With the increase of the time t, the sending rate decreases until there is no feedback packet received, namely $F_b \geq 0$. Subsequently, the phase trajectory will "pierce" the switching line $F_b = 0$, entering into the rate increase area. In the rate increase area, the shape of the phase trajectory, namely the blue solid curve in Fig. 6, becomes the same as the one shown in Fig. 3. Consequently, the phase trajectory will immediately "pierce" the switching line $F_b = 0$ again and return back to the rate decrease area.

According to Theorem 1, the necessary and sufficient condition for the occurrence of the sliding mode motion pattern is

$$\begin{cases} \lim_{F_b(t)\to 0^+} \dot{F}_b(t) \leq 0 \\ \lim_{F_b(t)\to 0^-} \dot{F}_b(t) \geq 0. \end{cases} \tag{21}$$

Inequality $\lim_{F_b(t)\to 0^-} \dot{F}_b(t) \geq 0$ means that once QCN enters into the rate decrease area in the forth quadrant, it immediately moves back to the switching line and enters into state of fast recovery. According to the shape of the phase trajectories shown in Fig. 9, $\lim_{F_b(t)\to 0^-} \dot{F}_b(t) \geq 0$ always holds when $\sqrt{\frac{G_d}{4C}}\frac{w}{p} < 1$ and $x(t) > 0$. The derivation below proves it

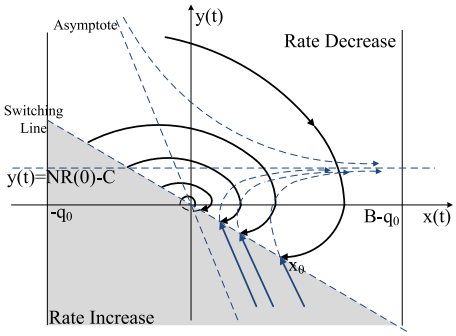


Fig. 6. The sliding mode motion pattern of QCN.

$$\begin{aligned}\lim_{F_b(t)\to 0^-}\dot{F}_b(t) &= \lim_{F_b(t)\to 0^-} -\dot{x}(t)-\frac{w}{pC}\dot{y}(t)\\ &= \lim_{F_b(t)\to 0^-}\frac{w}{pC}\left[G_dCx(t)+\frac{G_dw}{p}y(t)\right]-y(t)\\ &= \frac{pC}{w}x(t)+\lim_{F_b(t)\to 0^-}\frac{G_dw}{p}\left[x(t)+\frac{w}{pC}y(t)\right]\\ &= \frac{pC}{w}x(t)\\ &> 0.\end{aligned} \tag{22}$$

Inequality $\lim_{F_b(t)\to 0^+}\dot{F}_b(t)\le 0$ means that once QCN enters into the process of fast recovery, it immediately moves back to the switching line and enters into rate decrease area. Referring to equations (14) and (15), we have

$$\begin{aligned}\lim_{F_b(t)\to 0^+}\dot{F}_b(t) &= \lim_{F_b(t)\to 0^+}\left[-\dot{x}(t)-\frac{w}{pC}\dot{y}(t)\right]\\ &= \lim_{F_b(t)\to 0^+} -y(t)-\frac{w}{pC}\left[\frac{NR(0)-C}{2T}-\frac{y(t)}{2T}\right]\\ &= \left(\frac{pC}{w}-\frac{1}{2T}\right)x(t)-\frac{w[NR(0)-C]}{2TpC}.\end{aligned} \tag{23}$$

Hence, inequality $\lim_{F_b(t)\to 0^+}\dot{F}_b(t)\le 0$ is equivalent to

$$\left(2T-\frac{w}{pC}\right)x_0\le\frac{w^2}{p^2C^2}[NR(0)-C] \tag{24}$$

at the point (x_0, y_0) in the switching line. In summary, we have the following theorem:

Theorem 2. *The QCN system enters into the sliding mode motion pattern from the point (x_0, y_0) in the switching line if all of the following items are satisfied: (1) Inequality (24) holds, (2) $\sqrt{\frac{G_d}{4C}}\frac{w}{p}<1$ and (3) $x_0>0$.*

In Theorem 2, T, w, p are system parameters, C, N are related to the network configuration, and x_0, $R(0)$ reflect the current state of the QCN system. Therefore, whether the QCN system can enter into the sliding mode motion pattern depends on all of the system parameters, the current system state and the network configurations.

In practice, $\sqrt{\frac{G_d}{4C}}\frac{w}{p}<1$ always holds. Under this condition, Theorem 2 indicates that when x_0 is small, namely when the offset of the queue length to the target point is small, the QCN system probably enters into the sliding mode motion pattern. After entering into the sliding mode motion pattern, the QCN system will approach to the equilibrium point directly along the switching line, as we have shown in Figs. 4 and 6. However, inequality (24) may not be satisfied due to the parameters setting and network configuration. For example, according to Theorem 2, when T is large, the QCN system fails to enter into the sliding mode motion pattern. The experimental results above verify this theoretical analysis results.

In a word, QCN moves to the equilibrium point mainly via the sliding mode motion pattern. But QCN just unconsciously utilizes the advantages of the sliding mode motion pattern. The heuristic rate adjustment algorithm cannot guarantee that the QCN system enters into the sliding mode motion pattern under any conditions. Inspired by the advantage of the sliding mode motion pattern and the problems of QCN, we intend to deliberately design an elaborate rate adjustment algorithm such that the congestion management system can enter into the sliding mode motion pattern under any conditions.

Remark. There exists another theoretical work [27], which also focuses on the stability of QCN. In [27], the QCN system is approximated by a linear control system using the Lyapunovs linearization method, and then the classical frequency analysis method is used to analyze it, and finally QCN is shown to be stable. The Lyapunovs linearization method ensures that the stability of the QCN system is equivalent to the stability of its linearized system in the neighborhood of the equilibrium point. Unluckily, the range of the neighborhood is absent from [27]. Moreover, the classical frequency analysis method used in [27] cannot capture the dynamic behaviors such as the sliding mode motion pattern. In our theoretical analysis, Theorem 2 not only implicates that QCN can reach the stable state via the sliding mode motion pattern in the neighborhood of the equilibrium point, but aslo provides the explicit sufficient condition, which can be considered as the boundary of the neighborhood, for QCN entering into the sliding mode motion pattern.

4 The SMCC Scheme

4.1 Design Principles

The design principles of the SMCC scheme are as follows:

- SMCC should be able to enter into the sliding mode motion pattern starting from any initial states with any network configurations to utilize the advantages that the performance of congestion management system is insensitive to the changes of parameters and external disturbances in the sliding mode motion pattern.
- SMCC should stick to the framework of the congestion management schemes developed by the IEEE 802.1Qau work group. Because this framework integrates the existing experiences of designing congestion management schemes and has been tested for several years. In this way, it is also convenient to replace QCN by SMCC.
- To handle traffics at the speed of 1 or 10 Gbps, SMCC should be simple enough for the hardware implementation. QCN is relatively complex, as shown below.

- At RP, QCN needs to "remember" the current state for rate adjustment.
- In the process of fast recovery and active increase, the complex combination of timer and counter is used to identify time length of cycles and whether R_{HAI} should be used.
- Other add-on mechanisms, such as extra fast recovery and target rate reduction, are used to restore the performance of QCN in special environment.

- The controllable low queuing delay and no packets loss is required in DCE. Facing the burst traffic in data centers, the response time of SMCC is expected to be as short as possible.

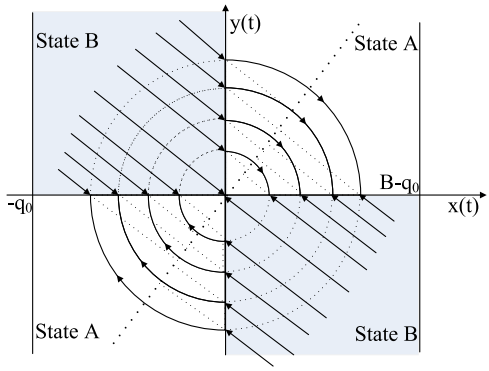


Fig. 7. The phase trajectories of SMCC when $a \geq 1$.

4.2 The SMCC Scheme

SMCC follows the framework developed by the IEEE 802.1Qau work group. In brief, the main differences between SMCC and QCN are as follows:

- At CP, the queue length and its derivative are separately involved in the feedback packet in SMCC. While the CP of QCN computes a weighted sum of them and embeds this sum into the feedback packet.
- At RP, SMCC uses feedback information for rate increase and needs to identify the congestion point similar to BCN, instead of self-increase as QCN.
- The rate adjustment algorithm of SMCC is different from that of QCN.

In SMCC, the information involved in the feedback packet are Q_{off} and ΔQ, where Q_{off} is the offset of the queue length to the target point and ΔQ presents the variance of the queue length in a sampling interval. QCN simply computes the weighted sum of Q_{off} and ΔQ according to equation (1), and feeds the sum back. In this way, QCN loses valuable information. For example, QCN cannot identify whether the rate decrease is mainly caused by long queue length, i.e., large Q_{off}, or by a great variance of the queue length in a sampling interval, i.e., large ΔQ. In contrast, SMCC not only obtains much more feedback information, but also efficiently utilizes the feedback information. For example, SMCC uses the feedback information for both rate increase and rate decrease. Feedback information makes it possible to set proper sending rate at once. Whereas a search algorithm, such as the self-increase algorithm used in QCN, needs to try at least several times for the proper sending rate. Thus, the rate adjustment algorithm of SMCC can be more precise and rapid.

The SMCC system is divided into two states as shown in Fig. 7, where axises $x(t)$ and $y(t)$ are defined by equation (16). State A is that Q_{off} and ΔQ have the same signal. And State B is that Q_{off} and ΔQ have different signals. These states are differentiated by the feedback information, namely SMCC need not to "remember" these states as in QCN. Since the goal of the congestion management scheme is to hold the queue length around the target point, SMCC focuses on the degree that the current queue length deviates from the equilibrium point, instead of whether the sending rate is increased or decreased. In State A, the queue length is offset to the equilibrium point and will move away from the equilibrium point. Thus the sending rate should be adjusted in large amount in State A for short response time. Let State A stand alone can help the system to achieve this goal.

The linear rate adjustment algorithms of SMCC are

$$r \leftarrow \begin{cases} r - a * Q_{off} & in \quad State \quad A \\ r - b * \Delta Q & in \quad State \quad B \end{cases} \tag{25}$$

where a and b are positive parameters. When SMCC stays in the first quadrant of Fig. 7, the sending rate will be reduced in proposition to the offset of queue length; note that the queue length keeps increasing because $y(t) = Nr(t) - C > 0$ in the first quadrant. When SMCC stays in the third quadrant of Fig. 7, the sending rate will be increased in proposition to the offset of queue length; note that the queue length keeps decreasing because $y(t) = Nr(t) - C < 0$ in the third quadrant. Therefore, we let State A stand alone and set coefficient a large value to accelerate rate adjustment in both cases. On the other side, when SMCC stays in the second quadrant of Fig. 7, the queue length increases towards the target point q_0 because $y(t) = Nr(t) - C > 0$. When SMCC stays in the forth quadrant of Fig. 7, the queue length decreases towards the target point q_0 because $y(t) = Nr(t) - C < 0$. Therefore, we just slightly adjust the sending rate by setting coefficient b small value in State B. Differentiating the system states in this way, SMCC has two switching lines $x(t) = 0$ and $y(t) = 0$. In next section, we will theoretically show that SMCC can hit the sliding regime $y(t) = 0$ as shown in Fig. 7 starting from any initial states with any network configurations.

In addition, substituting equation (13) into equation (25), we can find that the rate adjustment algorithm of SMCC involves both addictive increase phase and multiple decrease phase (AIMD) essentially. Thus, SMCC is fair according to the theoretical analysis of [28].

4.3 Theoretical Analysis

In this section, the theoretical foundations for designing SMCC are presented. To exhibit the procedure we design SMCC, we present the sufficient condition that the following general system hits the sliding regime $y(t) = 0$ starting from any system states with any network configurations.

- The system is divided into State A and State B.
- The subsystems are linear, and the negative feedback is employed.

With the same method used in Section 3, the general system can be modeled by the following differential equations. When $x(t)$ and $y(t)$ have the same signal,

$$\begin{cases} \frac{dx(t)}{dt} = y(t) \\ \frac{dy(t)}{dt} = -a_1(t)x(t) - a_2(t)y(t) \end{cases} \tag{26}$$

and when $x(t)$ and $y(t)$ have different signals,

$$\begin{cases} \frac{dx(t)}{dt} = y(t) \\ \frac{dy(t)}{dt} = -b_1(t)x(t) - b_2(t)y(t), \end{cases} \tag{27}$$

where $a_1(t), a_2(t), b_1(t), b_2(t)$ are nonnegative coefficients. These coefficients are associated with time t because parameters such as link capacity C and the number of flows N may change with the time t.

We should firstly make sure that $y(t) = 0$ is the sliding regime, through which the system reaches the equilibrium point. Secondly, we should ensure the system hit the sliding regime starting from any system states with any network configurations. The necessary and sufficient condition for the switching line $y(t) = 0$ being the sliding regime is that the following inequalities hold [16]

$$\begin{cases} \lim_{y(t)\to 0^+} \dot{y}(t) < 0 & when \quad x(t) > 0 \\ \lim_{y(t)\to 0^+} \dot{y}(t) < 0 & when \quad x(t) < 0 \\ \lim_{y(t)\to 0^-} \dot{y}(t) \geq 0 & when \quad x(t) \leq 0 \\ \lim_{y(t)\to 0^-} \dot{y}(t) \geq 0 & when \quad x(t) \geq 0. \end{cases} \tag{28}$$

Inequalities (28) mean that the phase trajectory will "pierce" the switching line $y(t) = 0$ from both sides of line $y(t) = 0$. Referring to differential equations (26) and (27), inequalities (28) can be simplified as

$$\begin{cases} -a_1(t)x(t) < 0 & when \quad x(t) > 0 \\ -b_1(t)x(t) < 0 & when \quad x(t) < 0 \\ -a_1(t)x(t) \geq 0 & when \quad x(t) \leq 0 \\ -b_1(t)x(t) \geq 0 & when \quad x(t) \geq 0. \end{cases} \tag{29}$$

Obviously, inequalities (29) is equivalent to the following inequality

$$\min\{a_1(t)\} > 0 \quad and \quad \max\{b_1(t)\} \leq 0. \tag{30}$$

Therefore, once inequality (30) holds, the switching line $y(t) = 0$ is the sliding regime of the whole congestion control system.

The next step is to explore the sufficient condition such that the congestion management system hits the sliding regime starting from any initial states with any network configurations. When inequality (30) holds, there must be $b_1(t) = 0$ since $b_1(t)$ is nonnegative. Therefore, the phase trajectories of the differential equations (27) are straight lines in the second and forth quadrant, as shown in Fig. 7. In this condition, the phase trajectories of the differential equations (27) can hit the sliding regime $y(t) = 0$ starting from any initial states with any network configurations. On the other hand, because both $a_1(t)$ and $a_2(t)$ are nonnegative, the characteristic equation $z^2 + a_2(t)z + a_1(t) = 0$ of differential equations (26) has no nonnegative real roots. Therefore, according to [16], the phase trajectories of the differential equations (26) can hit the sliding regime $y(t) = 0$ starting from any initial states with any network configurations. In sum, we have the following Theorem.

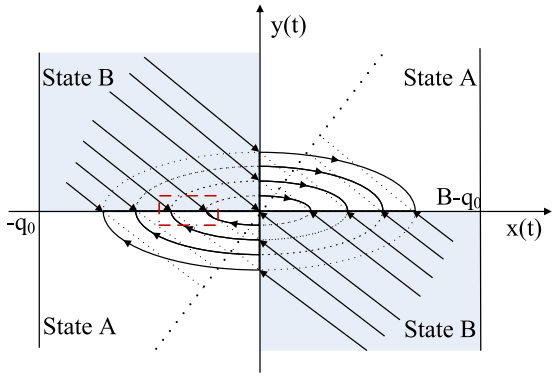
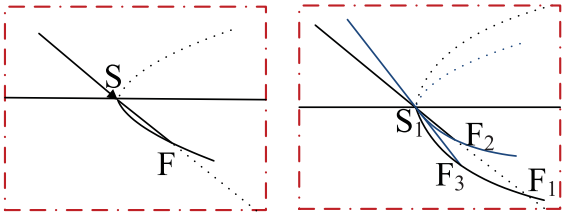


Fig. 8. The phase trajectories of SMCC when $a < 1$.

Theorem 3. *When inequality* (30) *holds, the congestion management system described by differential equations* (26) *and* (27) *can enter into the sliding mode motion pattern under any conditions.*

SMCC is a special case of the general system described by differential equations (26) and (27), which chooses parameters $a = \frac{a_1(t)}{N} > 0$, $a_2(t) = 0$, $b_1(t) = 0$ and $b = \frac{pCb_2(t)}{N} > 0$. According to Theorem 3, the SMCC system can hit the sliding regime $y(t) = 0$ under any conditions. In State B, the phase trajectories of the SMCC system are straight lines, whose slopes are $-b$. In State A, the phase trajectories of the SMCC system are ellipses elongated along the vertical axis when $a > 1$, or ellipses elongated along the horizontal axis when $a < 1$, or circles when $a = 1$. Therefore, the phase trajectories of the whole SMCC system are shown in Figs. 7 and 8, respectively.

Remark 1. In reality, a deal-breaker phenomena may occur in the SMCC system. For example, the phase trajectories of SMCC may cross the switching line and trespass' into State A from State B, reach the point F, and then return back to point S along the ellipse, as shown in Fig. 9a, which is the enlargement of the red box in Fig. 8. In this condition, the SMCC system chatters between point S and point F rather than around the origin. Obviously, the occurrence of this special phenomena depends on the slope or curvature of the phase trajectories in State B or State A, respectively. For example, assume that the phase trajectory cant reach point F_1 from point S_1, as the solid black line shown in Fig. 9b. When parameter a is set smaller and correspondingly the curvature of the phase trajectories in State A increases, or when parameter b is large and correspondingly the absolute value of the slop of straight lines in State B is large, the phase trajectory may reach point F_2 or F_3 from point S_1, respectively. As a result, the deal-breaker phenomena probably occurs, as the blue lines shown in Fig. 9b. Moreover, we can also know from Fig. 9b that parameter a or the curvature of the phase trajectories in State A has greater impact than parameter b on the occurrence of the deal-breaker phenomena. Our subsequent



(a) Enlargement of the red box in $Fig.8$ (b) Impacts of Parameters a and b

Fig. 9. Details of deal-breaker phenomena.

experiments also indicate that parameter a dominates the occurrence of the deal-breaker phenomena.

Note that the impacts of "deal-breaker" phenomena on the performance of SMCC is not significant, because the bottleneck link is fully utilized even though the queue stays at some point where the queue length is smaller than the target value.

Remark 2. We can also set $a_2(t) \neq 0$ for SMCC. In this case, the ellipses in the first and third quadrant of Fig. 7 will be replaced by spirals. Obviously, the new system can also hit the sliding regime $y(t) = 0$ with any network configurations. In addition, when $b_1(t) < 0$, the straight lines in the second and forth quadrant of Fig. 7 will be replaced by hyperbolas with asymptotes $y(t) = g_1(t)x(t)$ and $y(t) = g_2(t)x(t)$, where $g_1(t)$ and $g_2(t)$ are roots of the characteristic equation $z^2 + b_2(t)z + b_1(t) = 0$ of differential equations (27). In this condition, the system can also hit the sliding regime $y(t) = 0$ with any network configurations. However, we straightforwardly fix $a_2(t) = 0$ and $b_1(t) = 0$ to simplify the hardware implementation of SMCC.

4.4 Parameters Settings

In the hardware implementation, CP can obtain Q_{off} and ΔQ by monitoring the instantaneous queue length. Thus, we set parameters a and b rather than parameters $a_1(t)$ and $b_2(t)$ for SMCC.

Although the performance of congestion management system is theoretically insensitive to the change of parameters in the sliding mode motion pattern, the parameters has large impact on the dynamical behaviors of SMCC. Generally, large a and large b are expected for short response time, large a and small b are needed to avoid the occurrence of the "deal-breaker" phenomena.

Moreover, the granularity of the rate adjustment should be considered in practice. For example, parameters a and b may be so large that the sending rate is adjusted once by an amount larger than the available bandwidth, and correspondingly the queue length may oscillate in the stable state. Naturally, large a and large b are required for large available bandwidth, and so forth. Unfortunately, the available bandwidth shared by RPs is time-varying. Hence, parameters a and b should adapt to the change of available bandwidth. In SMCC, parameter $b = \frac{pCb_2(t)}{N}$ is already associated with the available bandwidth. On the other hand, the change of available bandwidth can be inferred by RPs through the feedback information ΔQ. This fact makes it possible to adjust parameter a with the change of the available bandwidth. Therefore, we present the following two stages way to set parameter a for SMCC. In State A

- When $|\Delta Q| > T_1$, parameter a is set to be a large constant value a_{large}.
- Or else, parameter a is set to be a small constant value a_{small},

where T_1 is empirical threshold. In this way, the SMCC system can keep short response time when the traffics is excessive, avoid the occurrence of the "deal-breaker" phenomena and achieve fine-grained granularity of the rate adjustment in the other conditions. Of cause, let parameter a change with the available bandwidth will increase the complexity of SMCC, and thus this two stages way is optional.

4.5 Overhead

As both SMCC and QCN follow the same framework, the cost on deploying them would be close to each other. The main difference on the overhead is that SMCC generates more feedback packets and uses more bits to represent the information of queue length, because SMCC separates Q_{off} and ΔQ, and uses feedback information for both rate increase and rate decrease. However, the number of feedback packets is much smaller than the number of per packet ACKs, because one feedback packet is generated until 100 packets passing switch (assume the sampling probability $p = 0.01$). Therefore, this overhead is acceptable. On the other hand, the rate adjustment rule of SMCC is obviously simpler than that of QCN. Thus, the overhead of implementing SMCC would be smaller than the overhead of implementing QCN. The overhead of our hardware implementation shown in Section 5.1 also confirms this inference.

4.6 Discussion

Here we try to explain why BCN should be improved to be QCN, and now from QCN to SMCC. For the convenience of discussion, we define four states of the congestion management system: State (1) $Q_{off} > 0$ and $Nr(t) < C$, State (2) $Q_{off} < 0$ and $Nr(t) < C$, State (3) $Q_{off} < 0$ and $Nr(t) > C$, State (4) $Q_{off} > 0$ and $Nr(t) > C$. Naturally, the queue length keeps decreasing until $Nr(t) \geq C$ in State (1). If the sending rate is increased slowly as in BCN, Q_{off} becomes small than zero before $Nr(t) \geq C$. Subsequently, the BCN system will enter into State (2). In this condition, the queue length (Q_{off}) oscillates severely and the response time is large. Therefore, BCN is unsuitable for the burst traffic in data centers. That is why BCN was replaced by QCN historically. The QCN system employs the fast recovery algorithm for State (1). Hence, the accelerated speed of the sending rate is decided by the target rate for fast recovery in State (1). When the target rate is large enough, the sending rate is increased fast enough. The system will enter into the sliding mode motion pattern. Or else, QCN experiences the same sequence of the system states as BCN. Thus, the performance of QCN depends on the system state. In contrast, the SMCC system enters into the sliding mode motion pattern after State (1), and then switches either between State (1) and State (2) or between State (1) and State (3), as shown in Fig. 7. As a result, the SMCC system approaches to the equilibrium point directly along the switching line.

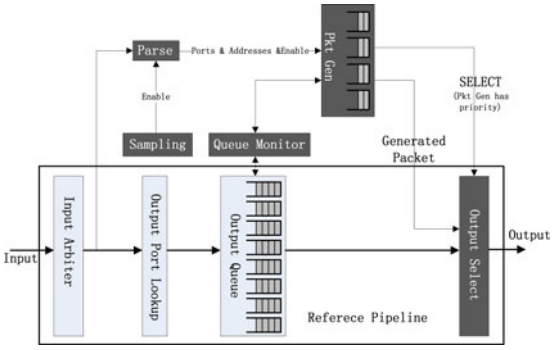


Fig. 10. Structure of the CPs of QCN and SMCC.

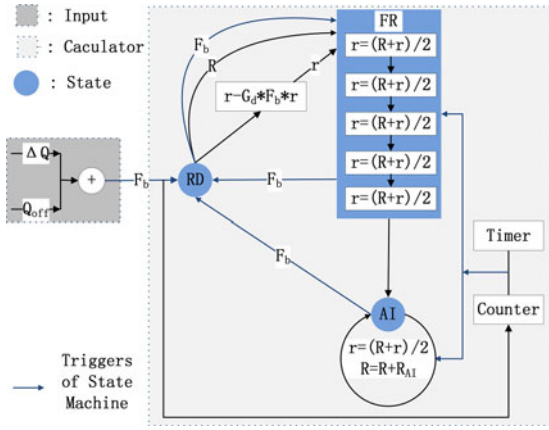


Fig. 11. Structure of the Calculator module of QCN.

5 Implementation and Evaluation

5.1 Implementation

NetFPGA [18] is a programmable hardware platform for fast prototyping. It consists of a Xilinx Virtex-II Pro FPGA, four 1 Gbps Ethernet ports and 4 MB SRAM.

The CPs of QCN and SMCC are almost the same except for the format of the feedback packets. They are implemented by extending the Reference Switch project [18]. As shown in Fig. 10, the black blocks are added for the implementation of the CPs of QCN and SMCC. When the Sampling module hits its target, it sends an Enable signal to the Parse module. The Parse module keeps parsing the incoming packets to obtain the corresponding ports and addresses. When it receives the Enable signal, the Parse module sends the current ports and addresses out to the Pkt Gen module. Asking the Queue Monitor module for the current queue length and using the ports and addresses, the Pkt Gen module generates feedback packets and sends them out by setting the SELECT signal. The feedback information is quantized into as many bit as possible and located in the Ethernet Padding region in our implementation. The Output Select module chooses to send the feedback packet when the SELECT signal is set to be true.

The RP is implemented by extending the Packet Generator project [18]. The Packet Generator can send packets at a given speed and capture the received packets. What we need to do is to add a module parsing the incoming packets and another module calculating the sending rate for the next step. The main difference between QCN and SMCC is the Calculator module, as shown in Figs. 11 and 12 respectively. We just implement the main rate adjustment rules of QCN, including rate decrease, fast recovery and active increase. Obviously, QCN needs to maintain its current status, e.g., how many times is fast recovery executed, with the help of special timer and counter. In other words, the Calculator module of QCN is composed by a state machine, and either the arrival of feedback packets or the hit of Timer and Counter triggers the state transition, as shown in Fig. 11. While SMCC only distinguishes its states directly by taking XOR of the signs of feedback information. It implies that the Calculator module of SMCC may be implemented by pure combinatorial logic, and thus simpler than that of QCN. In our hardware implementation, the number of slice registers used by the RP of SMCC is about 700 less than that used by the RP of QCN.

5.2 Experiment Setup

Our experiments are conducted on three scenarios. Scenario I uses the 3-sources dumbbell topology. In Scenario II, the parking-lot topology is configured. As shown in Figs. 13 and 14, the destinations of the flows started by S1, S2 and S3 are R1 and R2 and R3, respectively. And C1, C2 and C3 are Switches. The parameters setting of QCN is the same as that in [24] unless declared explicitly. Parameters a, b, a_{large} and a_{small} of SMCC are chosen such that the maximum amount of the rate adjustment in a sampling interval is R_a, R_b, R_{large} and R_{small} Mbps, respectively. In all the experiments, $q_0 = 64$ KB and the buffer size is set to be 128 KB. All packets are of length 1 KB. We sample the instantaneous queue length at the bottleneck switch every milliseconds as the experimental results.

Two types of flow are used in experiments. One is named controlled flows, whose sending rate is adjusted by the RPs of QCN and SMCC excepting the explicitly given initial sending rate. The other one is background flows, whose sending rate is always fixed and explicitly pointed out.

5.3 Scenario I

5.3.1 General Performance

We firstly let three RPs start controlled flows at the speed of 1 Gbps in Scenario I. With the change of the parameters of SMCC, the evolutions of the queue length at the bottleneck link are shown in Fig. 15. When $R_a = 256$, the queue length chatters around the target point after the initial process. According to the theoretical analysis in Section 2.3, the sliding mode motion pattern is characterized by the chattering around the target point. Thus, Fig. 15 means that SMCC enters

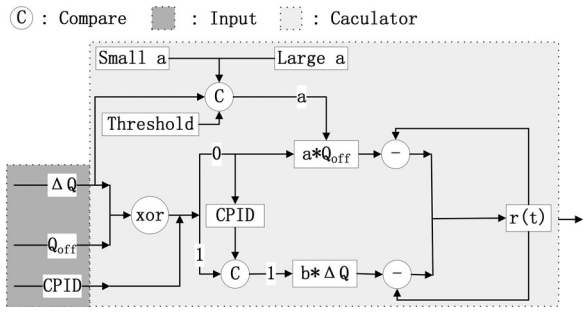


Fig. 12. Structure of the Calculator module of SMCC.

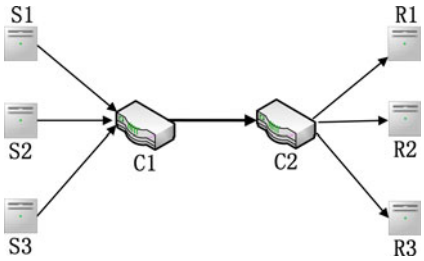


Fig. 13. Dumbbell topology.

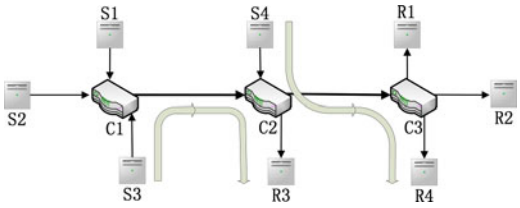


Fig. 14. Parking-lot topology.

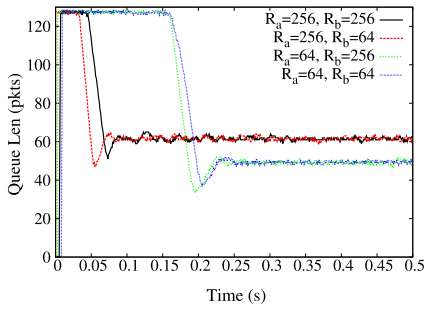


Fig. 15. Evolutions of the queue length of SMCC when parameters change.

into the sliding mode motion pattern after the initial process. Besides, when $R_a = 64$ is small, SMCC keeps stable with the queue length $q(t) = 48\,\mathrm{KB} < \mathrm{q_0}$, corresponding to the "deal-breaker" phenomena discussed in Remark 1 of Section 4. Obviously, parameter a has larger impacts than parameter b on the occurrence of the deal-breaker phenomena.

Comparing Fig. 15 with the $[0, 2\,\mathrm{s}]$ of Fig. 5a, where QCN is configured with the standard parameters setting, we know that SMCC has a more stable queue. In fact, Fig. 15 is similar to the $[6, 8\,\mathrm{s}]$ of Fig. 5a, where T is set to be the time of sending 30 KB data and thus QCN probably enters into the sliding mode motion pattern according to Theorem 2. In Fig. 5a, QCN responses to the big overshoot with a great amount of rate decrease, which results in empty buffer for a long period to time. While SMCC can response to the big overshoot properly and the queue length keeps larger than 0, as shown in Fig. 15. Moreover, SMCC takes less response time than QCN and Fig. 5a also indicates parameter a dominates the response time of SMCC.

We repeat the experiment ten times when the parameters of SMCC are $R_a = 256$ and $R_b = 64$. In the same way, similar experiments on QCN, where QCN is configured with the standard parameters setting, are conducted ten times. Each experiment lasts 2 seconds. In this way, we expect to eliminate the random factor of the hardware experiments. The cumulative distribution function graphs of the queue length are summarized in Fig. 16. Obviously, SMCC can hold the queue length mostly around the

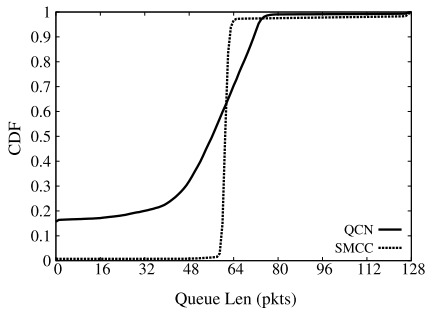


Fig. 16. Probability distributed function of the queue length.

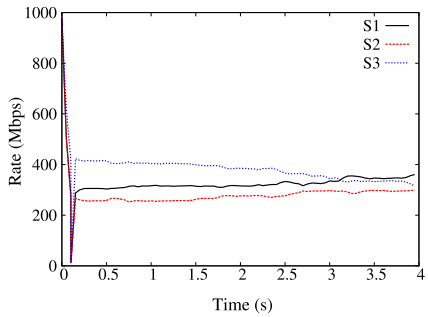


Fig. 17. Sending rates of the individual sources in SMCC.

target point 64 KB. While QCN can only hold the queue length about between 40 and 80 KB. In addition, the probability that the queue length becomes 0 increases up to 10 percent in the QCN system.

We also measure the sending rates of the individual sources when SMCC is configured with $R_a = 256$ and $R_b = 64$. As shown in Fig. 17, each source can eventually reach its fair share. Therefore, SMCC is fair. Experiments on the fairness of QCN are omitted, because QCN has been shown to be unfair in lots of literature [14], [15].

5.3.2 The Sliding Mode Motion Pattern

Fig. 15 also indicates that the performance of SMCC is insensitive to the change of parameters. With any of the four pairs of parameters, SMCC can enter into the sliding mode motion pattern, although the "deal-breaker" phenomena may occur.

Next, we will show that the performance of SMCC is also insensitive to the change of network configurations. In Scenario I, let two RPs start controlled flows at the speed of 1 Gbps at the beginning. And then, an extra background flow with fixed sending speed G enters into the SMCC system at the 2nd seconds. The parameters of SMCC are $R_a = 256$ and $R_b = 64$. The evolutions of the queue length at the bottleneck link are shown in Fig. 18. When $G = 500$ Mbps, the queue length almost keeps unchanged. When $G = 750$ Mbps, the queue length oscillates slightly. Note that R_a is already larger than the available bandwidth in this condition. When $G = 875$ Mbps, the queue length oscillates heavily. In this condition, when the sending rate is adjusted once, the variance of the sending rate is probably larger than the available bandwidth. Therefore, it is the improper granularity of rate adjustment that results in the oscillations. Next, we will try the two stages parameters setting method of SMCC.

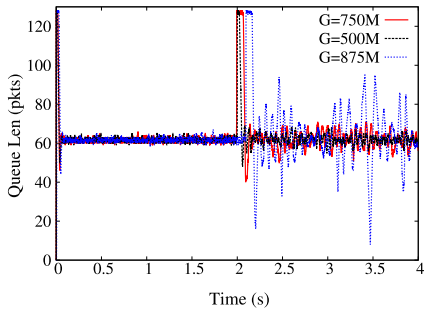


Fig. 18. Evolutions of the queue length of SMCC at the bottleneck link when network configurations change.

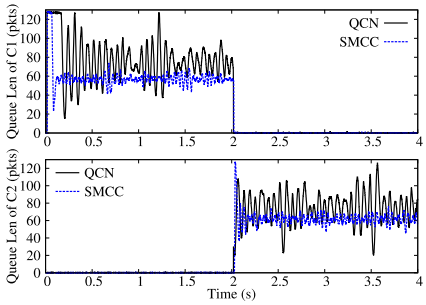


Fig. 20. Evolutions of the queue length at the bottleneck links with traffic pattern I.

Comparing Fig. 18 with Fig. 5a, which shows the behaviors of QCN with the change of the available bandwidth, we know that SMCC has a more stable queue than QCN. Note that the network configurations in Figs. 18 and 5a are the same when $G = 500$ Mbps.

In summary, SMCC is insensitive to the changes of both the system parameters and the network configurations, and thus is superior to QCN.

5.3.3 Parameters Setting

In Scenario I, let two RPs start controlled flows at the speed of 1 Gbps at the beginning. And then, an extra background flow with fixed sending rate 875 Mbps enters into the SMCC system at the 1st second. Parameters $R_b = 64$, $T_1 = 8$ KB and R_a is set in the two stages way: when $\Delta Q > T_1$, $R_a = R_{large} = 256$; or else, $R_a = R_{small} = 128$. The evolutions of the queue length at the bottleneck are shown in Fig. 19. The queue length almost keeps unchanged when there is no background flow in the 1st second. After the 1st second, the way of two stages parameters setting can reduce the aptitude of the oscillations of the queue length in the steady state. Note that R_{small} is larger than the available bandwidth and R_{large} is comparable to two times of the available bandwidth after the 1st second. Hence, it's the application of R_{small}, which results in the better granularity of rate adjustment, that reduces the aptitude of the oscillations of the queue length. In other words, the simple two stages parameters setting can improve the performance of SMCC.

Remark. Mathematically, ΔQ is the function of the derivative of Q_{off}. Consequently, the variance of ΔQ is small comparing to that of Q_{off} in reality. In fact, ΔQ keeps a small value throughout our experiments and thus R_b is hardly reached. That's why parameter R_b is set a fixed small value in reality, rater than changed with the available bandwidth as R_a. Knowing that the variance of Q_{off} is larger than that of ΔQ, we can infer that parameter R_a dominates the granularity of the rate adjustment. Referring that parameter a also dominates the response time, we can explain why parameter a should be set in the two stages way.

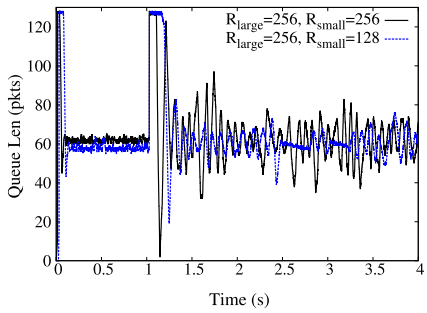


Fig. 19. Evolutions of the queue length of SMCC with two stages parameters setting.

5.4 Scenario II

Here we test the performance of QCN and SMCC on the multiple bottlenecks Scenario II. Referring to Section 3.1, which shows that the small T can improve the performance of QCN in certain condition, we set T to be the time of sending 30 KB data in the QCN system. The initial parameters setting of SMCC is that $R_a = 256$ and $R_b = 64$.

5.4.1 Traffic Pattern I

At the beginning, two RPs S1 and S2 start controlled flows at the speed of 1 Gbps and S3 starts a background flow of fixed sending speed 500 Mbps destined to R3. At the 2nd second, the background flow of S3 stops, but S4 starts a background flow of fixed sending speed 500 Mbps destined to R4, which stops at the 4th second. Note that the bottleneck is at C1 in the first 2 seconds and at C2 in the last 2 seconds. The evolutions of the queue length at the bottleneck links are shown in Fig. 20. After the bottleneck link changes at the 2nd seconds, the evolutions of the queue length almost keep unchanged in both the QCN system and the SMCC system. Moreover, the evolutions of the queue length in the QCN system and the SMCC system are the same as that in the $[4, 6\,\text{s}]$ of Figs. 5a and 18, respectively. Hence, both QCN and SMCC suit for the change of the bottleneck links. Besides, the aptitude of the oscillations of the queue length in the SMCC system is smaller than that in the QCN system.

5.4.2 Traffic Pattern II

Let two RPs S1 and S2 start controlled flows at the speed of 1 Gbps at the beginning, S3 starts a background flow of fixed sending speed 500 Mbps destined to R3 at the 1st second. At the 3rd second, the background flow from S3 stops, but S4 starts a background flow of fixed sending speed 750 Mbps destined to R4, which stops at the 5th second. Note that the bottleneck is at C1 in the first 3 seconds and at C2 in the last 2 seconds. The evolutions of the queue length

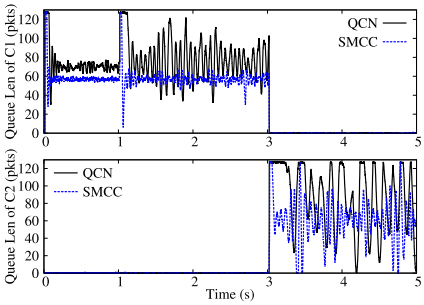


Fig. 21. Evolutions of the queue length at the bottleneck link with traffic pattern II.

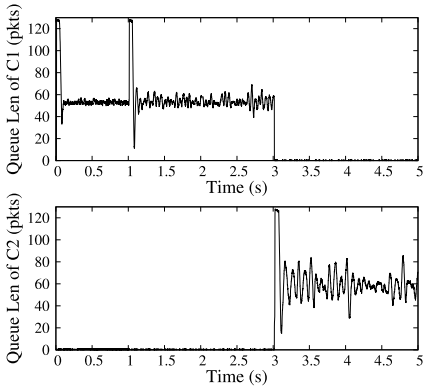


Fig. 22. Evolution of the queue length under Scenario II, $T_1 = 8$ KB, $R_{large} = 256$, $R_{small} = 128$ and $R_b = 32$.

at the bottleneck link are shown in Fig. 21. With the increase of the background flow size, the aptitude of the oscillations of the queue length becomes large in the QCN system. This is consistent with what we have shown on the single bottleneck link scenario in Section 3.1. The aptitude of the oscillations of the queue length in the SMCC system almost keeps unchanged when the background flow size increases to 500 Mbps, but greatly increases when the background flow size increases to 750 Mbps. This is because the available bandwidth becomes smaller than R_a. In sum, SMCC has a more stable queue than QCN. What's more, when the two stages parameters setting is used in the SMCC system, the aptitude of the oscillations of the queue length reduces even if the size of the background flow is 750 Mbps, as shown in Fig. 22.

According to our experiments, the recommended parameters setting for 1 Gbps links are $T_1 = 8$ KB and choosing a_{large}, a_{small}, b such that the maximum adjustment range of sending rate in a sampling interval is 256, 128, 64 Mbps, respectively.

5.5 Simulation

Because each NetFPGA card has only four Ethernet ports, we can only do experiments on small topology. Therefore, it is very hard to make the experiments more realistically on the NetFPGA platform. However, the consistence of above experimental results with both dumbbell and parking-lot topologies implies that the performance of SMCC is not sensitive to particular network topology. Moreover, to make the network configuration more realistic, we implement QCN and SMCC with NS2 and conduct simulations with bursty, short-lived flows and the tree topology here.

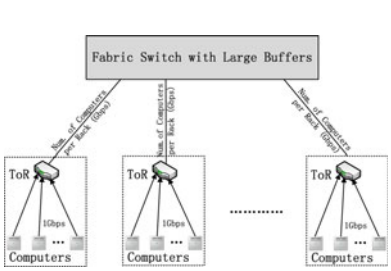
Simulations are conducted with the same fat-tree topology as [29]. As shown in Fig. 23a, the link capacity between Top of Rack (ToR) switches and computers are 1 Gbps and the rest of the fat-tree topology is replaced with one large fabric switch with large buffers because the bottleneck in datacenter networks is usually the ToR switches [30], [31]. Let N denote the number of ToR switches and M denote the number of computers connected to each ToR switches in each rack. To simulate the Many-to-One traffic pattern [6], we let $M = 4$, $N = 4$, the leftmost computer be the receiver, and all the other computers concurrently start controlled flows of initial sending rate 1 Gbps. In this scenario, flows are bursty since the initial incoming rate faced by congested switch is $4 * 4 * 1$ Gbps $= 16$ Gbps. We firstly let flows to be long-lived and subsequently let the sizes of flows evenly distribute between $[10\,\text{kb}, 500\,\text{Mb}]$ such that short-lived flows and long-lived flows are mixed together. Both QCN and SMCC use above recommended parameters configuration. The results are shown in Fig. 23.

- When flows are long-lived, both SMCC and QCN can hold the queue length at the target point, but SMCC responses to the burst traffic faster than QCN.
- When the sizes of flows distribute between $[10\,\text{kb}, 500\,\text{Mb}]$, QCN responses to the finishing of flows slowly and accordingly the buffer is emptied (the bottleneck link is underutilized) for longer period of time than that of SMCC.

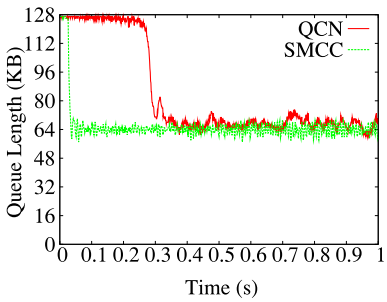
Therefore, SMCC works better than QCN in face of short-lived burst traffic.

6 Conclusion

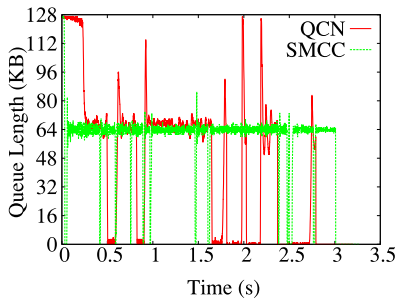
As the Internet owns its scalability and stability to TCP, the congestion management scheme would also play a central role in DCE networks. Though QCN has been ratified to be



(a) The fat-tree topology of [29]



(b) Long-lived Flows



(c) Size of flows evenly distribute between 10kb and 500Mb

Fig. 23. Simulation results.

the standard of the congestion management schemes for DCE, experiments and theoretical analysis show that QCN is sensitive to the changes of parameters and network configurations, and may become unstable in certain conditions. Inspired by the insights that QCN reaches the stable state mainly via the sliding mode motion pattern, we design the SMCC scheme, which is able to enter into the sliding mode motion pattern under any conditions. Theoretically, SMCC benefits from the advantage that the performance of congestion management system is insensitive to the changes of parameters and network configurations in the sliding mode motion pattern. Therefore, SMCC is stable and robust. Moreover, SMCC is fair, simple, and can achieve shorter response time than QCN. Experiments on the NetFPGA platform verify these results. In term of the deployment, SMCC follows the framework of the congestion management schemes developed by the IEEE 802.1Qau work group, and thus QCN can be replaced by SMCC easily through modifying only the firmware.

Acknowledgments

The authors gratefully acknowledge the anonymous reviewers for their constructive comments. This work is supported in part by National Basic Research Program of China (973 Program) under Grant No. 2012CB315803 and 2014CB347800, and National Natural Science Foundation of China (NSFC) under Grant No. 61225011.

References

[1] IEEE 802.1: Data Center Bridging Task Group [Online]. Available: http://www.ieee802.org/1/pages/dcbridges.html
[2] Unified Fabric: Cisco's Innovation for Data Center Networks. (2008). white paper [Online]. Available: http://www.cisco.com/en/US/solutions/collateral/ns340/ns517/ns224/ns783/white_paper_c11-462422.pdf
[3] 100 Gigabit Ethernet [Online]. Available: http://en.wikipedia.org/wiki/100_Gigabit_Ethernet
[4] M. AI-Fares, A. Loukissas, and A. Vahdat, "A scalable, commodity data center network architecture," in *Proc. ACM SIGCOMM Conf.*, 2008, pp. 63–74.
[5] S. Kandula, S. Sengupta, A. Greenberg, and P. Patel, "The nature of datacenter traffic: Measurements and analysis," in *Proc. 9th ACM SIGCOMM Conf. Internet Meas. Conf.*, 2009, pp. 202–208.
[6] T. Benson, A. Akella, and D. A. Maltz, "Network traffic characteristics of data centers in the wild," in *Proc. 10th ACM SIGCOMM Conf. Internet Meas.*, Nov. 2010, pp. 267–280.
[7] IEEE 802.1Qbb: Priority-based flow control, working draft [Online]. Available: http://www.ieee802.org/1/pages/802.1bb.html
[8] G. F. Pfister and V. A. Norton, "Hotspot contention and combining in multistage interconnection networks," *IEEE Trans. Comput.*, vol. 34, no. 10, pp. 933–938, Oct. 1985.
[9] C. DeSanti and J. Jiang, "FCoE in perspective," in *Proc. Int. Conf. Adv. Infocomm Technol.*, 2008, pp. 138:1–138:8.
[10] D. Cohen, T. Talpey, A. Kanevsky, U. Cummings, M. Krause, R. Recio, D. Crupnicoff, L. Dickman, and P. Grun, "Remote direct memory access over the converged enhanced Ethernet fabric: Evaluating the options," in *Proc. IEEE Symp. High Perform. Interconnects*, 2009, pp. 123–130.
[11] B. Chia, DC Technology Update. (2010) [Online]. Available: http://www.cisco.com/web/SG/learning/dc_partner/files/Nexus_Update.pdf
[12] Juniper, QFX3500 SWITCH, Datasheet (2014) [Online]. Available: http://www.juniper.net/us/en/local/pdf/datasheets/1000361-en.pdf
[13] Fulcrum Announces 1 Billion Packet Per Second 10G/40G Ethernet Switch Chips for Efficient Scaling of Virtualized Data Center Networks. (2010, Nov.) [Online]. Available: http://www.granitevc.com/files/news/2010/node-447/Anncmt_10-1102.pdf
[14] A. Kabbani, M. Alizadeh, M. Yasuda, R. Pan, and B. Prabhakar, "AF-QCN: Approximate fairness with quantized congestion notification for multi-tenanted data centers," in *Proc. IEEE 18th Symp. High Perform. Interconnects*, 2010, pp. 58–65.
[15] Y. Hayashi, H. Itsumi, and M. Yamamoto, "Improving fairness of quantized congestion notification for data center Ethernet networks," in *Proc. 31st Int. Conf. Distrib. Comput. Syst. Workshops*, 2011, pp. 20–25.
[16] U. Itkis, *Control Systems of Variable Structure*. Jerusalem, Israel: Keter Publishing House, 1976.
[17] L. Xu, K. Harfoush, and I. Rhee, "Binary increase congestion control (BIC) for fast long-distance networks," in *Proc. IEEE Conf. Comput. Commun.*, 2004, pp. 2514–2524.
[18] The NetFPGA Projects [Online]. Available: http://netfpga.org/foswiki/bin/view/NetFPGA/OneGig/ProjectTable
[19] V. Misra, W. B. Gong, and D. Towsley, "Fluid-based analysis of a network of AQM routers supporting TCP flows with an application to RED," in *Proc. Appl., Technol., Archit., Protocols Comput. Commun.*, Aug. 2000, pp. 151–160.
[20] V. Jacobson, "Congestion avoidance and control," in *Proc. ACM Symp. Commun. Archit. Protocols*, 1988, pp. 314–329.
[21] T. Kelly, "Scalable TCP: Improving performance in highspeed wide area networks," *ACM SIGCOMM Comput. Commun. Rev.*, vol. 33, pp. 83–91, 2003.
[22] IEEE 802.1Qau: End-to-end congestion management, Working Draft [Online]. Available: http://www.ieee802.org/1/pages/802.1au.html
[23] D. Bergamasco and R. Pan, "Backward congestion notification version 2.0," in *Proc. IEEE 802.1 Meeting*, Sep. 2005, pp. 1–40.
[24] A. Kabbani and M. Yasuda, "Data center quantized congestion notification (QCN): Implementation and evaluation on NetFPGA," in *Proc. 1st Asia NetFPGA Develop. Workshop*, Jun. 2010, pp. 17–22.
[25] E. A. Coddington and N. Levionson, *Theroy of Ordinary Differential Equations*. New York, NY, USA: McGraw-Hill, 1975.
[26] D. P. Atherton, *Nonlinear Control Engineering*. New York, NY, USA: Van Nostrand, 1982.
[27] M. Alizadeh, A. Kabbani, B. Atikoglu, and B. Prabhakar, "Stability analysis of QCN: The averaging principle," in *Proc. ACM SIGMETRICS Joint Int. Conf. Meas. Model. Comput. Syst.*, 2011, pp. 49–60.
[28] D. Chiu and R. Jain, "Analysis of the increase/decrease algorithms for congestion avoidance in computer networks," *J. Comput. Netw. ISDN Syst.*, vol. 17, pp. 1–14, Jun. 1989.
[29] B. Vamanan, J. Hasan, and T. N. Vijaykumar, "Deadline-aware datacenter TCP (D2TCP)," in *Proc. ACM SIGCOMM Conf. Appl., Technol., Archit., Protocols Comput. Commun.*, 2012, pp. 115–126.
[30] M. Alizadeh, A. Greenberg, D. A. Maltz, J. Padhye, P. Patel, B. Prabhakar, S. Sengupta, and M. Sridharan, "Data center TCP (DCTCP)," in *Proc. ACM SIGCOMM Conf.*, 2010, pp. 63–74.
[31] C. Wilson, H. Ballani, T. Karagiannis, and A. Rowstron, "Better never than late: Meeting deadlines in datacenter networks," in *Proc. ACM SIGCOMM Conf.*, 2011, pp. 50–61.
[32] Y. Cai, Y. Yan, Z. Zhang, and Y. Yang, "Survey on converged data center networks with DCB and FCoE: standards and protocols," *IEEE Netw.*, vol. 27, no. 4, pp. 27–32, Jul./Aug. 2013.

Wanchun Jiang received the PHD degree and the BE degree in computer science and technology from Tsinghua University, Beijing, China in 2014 and 2009, respectively. He is currently an assistant professor in the School of information science and engineering, Central South University. His research interests include congestion control, data center networks and the application of control theory in computer networks.

Fengyuan Ren received the BA and MSc degrees in automatic control from Northwestern Polytechnic University, China, in 1993 and 1996, respectively, and the PhD degree in computer science from Northwestern Polytechnic University. He is currently a professor at the Department of Computer Science and Technology, Tsinghua University, Beijing, China From 2000 to 2001, he was at Electronic Engineering Department, Tsinghua University as a post doctoral researcher. In January 2002, he moved to the Computer Science and Technology Department of Tsinghua University. His research interests include network traffic management, control in/over computer networks, wireless networks and wireless sensor networks. He (co)-authored more than 80 international journal and conference papers. He is a member of the IEEE, and was a technical program committee member and local arrangement chair for various IEEE and ACM international conferences.

Ran Shu received the BE degree in computer science and technology from Tsinghua University, Beijing, China in 2011 and is working toward the PHD's degree at the Department of Computer Science and Technology, Tsinghua University, Beijing, China. He is currently supervised by professor Fengyuan Ren and professor Chuang Lin. His research interests include congestion control and data center networks.

Yongwei Wu received the PhD degree in applied mathematics from the Chinese Academy of Sciences in 2002. He is currently a professor of computer science and technology, Tsinghua University, Beijing, China. His research interests include data center, cloud computing, distributed processing, and parallel computing.

Chuang Lin received the PhD degree in computer science from Tsinghua University, China in 1994. He is currently a professor at the Department of Computer Science and Technology, Tsinghua University, Beijing, China. He is an honorary visiting professor, University of Bradford, United Kingdom. His current research interests include computer networks, performance evaluation, network security analysis, and Petri net theory and its applications. He has published more than 300 papers in research journals and IEEE conference proceedings in these areas and has published four books. He was the Technical Program vice chair, the 10th IEEE Workshop on Future Trends of Distributed Computing Systems (FTDCS 2004); the General Chair, ACM SIGCOMM Asia workshop 2005 and the 2010 IEEE International Workshop on Quality of Service (IWQoS 2010). He is an associate editor of the *IEEE Transactions on Vehicular Technology* and an area editor of *Computer Networks*, and the *Journal of Parallel and Distributed Computing*. He is a senior member of the IEEE and the Chinese Delegate in TC6 of IFIP.