Analyzing the Impact of Buffer Capacity on Crosspoint-Queued Switch Performance

Guo Chen, Youjian Zhao, Dan Pei*, and Yongqian Sun

Abstract: **We use both theoretical analysis and simulations to study the impact of crosspoint-queued (CQ) buffer size on CQ switch throughput and delay performance under different traffic models, input loads, and scheduling algorithms. In this paper, we present the following. 1) We prove the stability of CQ switch using any work-conserving scheduling algorithm. 2) We present an exact closed-form formula for the CQ switch throughput and a non-closed-form but convergent formula for its delay using static non-work-conserving random scheduling algorithms with any given buffer size under independent Bernoulli traffic. 3) We show that the above results can serve as a conservative guide on deciding the required buffer size in pure CQ switches using work-conserving algorithms such as the random scheduling, under independent Bernoulli traffic. 4) Furthermore, our simulation results under real-trace traffic show that simple round-robin and random work-conserving algorithms can achieve quite good throughput and delay performance with a feasible crosspoint buffer size. Our work reveals the impact of buffer size on the CQ switch performance and provides a theoretical guide on designing the buffer size in pure CQ switch, which is an important step toward building ultra-high-speed switch fabrics.**

Index Terms: **Crosspoint-queued switch, performance evaluation, switch fabric.**

I. INTRODUCTION

As content-rich Internet applications such as video streaming, audio streaming, file sharing, and live video/voice call, become increasingly popular, the demands for higher backbone bandwidth have grown extremely fast. For the increasingly growing link rate, the switch fabric in core routers has only a very short time (e.g., 5.12 ns for a 64 bytes long packets to be transmitted in a 100-Gbps link) to schedule and send out a packet. Thus, how to reduce the scheduling time in a switch fabric, becomes a huge challenge. Most of the previous switch fabrics, including the input-queued switch [1], [2], combined-input-and-crosspoint-queued (CICQ) switch [3], [4] and multi-stage switch fabrics such as those in [5] and [6], allocate major buffers at line cards instead of at switch fabrics. To prevent packets from conflicting and being corrupted at the switch fabrics, every scheduling cycle in these approaches requires a round-trip communication between the line cards and the switch module, which limits the switching speed. To reduce power consumption, the line cards and switch module in modern core routers are often placed in different racks with a distance of from a few meters up to 60 m, as presented in [7]. Assuming that the length of the inter-rack cable is 2 m long and the propagation speed is 2×10^8 m/s [7], the back-of-envelope calculation shows that each scheduling cycle has at least a 20 ns delay caused by the round-trip communication, which becomes a bottleneck in a high-speed switch.

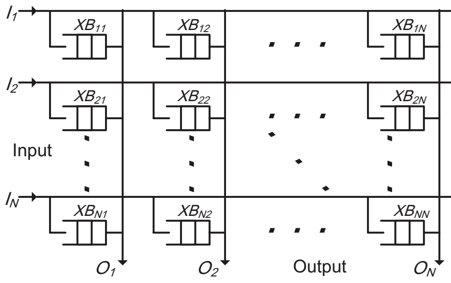

Fig. 1. The CQ switch model.

Recently, to overcome the above limitations, both the academia [8]–[12] and industry [13] have shown growing interest in CQ switches (shown in Fig. 1). Packets are buffered only at each crosspoint using on-chip memory; thus, switch decision can be independently and locally made by each output scheduler solely based on the conditions of the buffers in the same column as the output scheduler. Therefore, scheduling algorithms can be developed without communications between the line cards and switch module, which greatly reduces scheduling delay.

Despite the great promise of the CQ switch[1], a clear understanding is lacking on how to design the crosspoint buffers to meet the overall performance requirement of the switch fabric. In this paper, we take the first step toward this direction. We focus on understanding the impact of buffer size on the CQ switch performance. Previously, [8] presented an accurate analytical model for pure CQ throughput and delay, assuming a *buffer size of one* and independent and identically distributed (i.i.d.) uniform Bernoulli traffic. However, for larger buffer sizes, the authors introduced only approximate analytical models and simulation results for the *throughput* only. No theoretical or simulation analysis on the switch average delay was presented for crosspoint buffer size larger than one. Later on, several pa-

Manuscript received March 11, 2015; approved for publication by Jeonghoon Mo, Division III Editor, August 13, 2015

This work was supported in part by the National High Technology Research and Development Program of China (863 Program) under Grant No. 2013AA013302, the National Key Basic Research Program of China (973 program) under Grant No. 2013CB329105 and the State Key Program of National Natural Science of China under Grant No. 61233007.

The authors are with the Department of Computer Science and Technology, Tsinghua University, Beijing 100084, China, email: {chen-g11, sunyq12}@mails.tsinghua.edu.cn, {zhaoyoujian, peidan}@tsinghua.edu.cn.

*D. Pei is the corresponding author, Tel:+86(10)62792837

Digital object identifier 10.1109/JCN.2016.000067

[1] Although CQ switch was considered many years ago to be difficult to implement due to the scarcity of on-chip memory, modern technology has made feasible the implementation of CQ switch fabrics with large crosspoint buffers. Recently, [8] has revisited the CQ switch and proven the feasibility using semiconductor integration technology at that time by showing that a crosspoint buffer could store over 3 Mb packets for a switch with more than a hundred ports.

pers [9]–[11] have used *simulations* to study pure CQ switch performance for buffer sizes larger than one under traffic models such as the uniform Bernoulli and bursty traffic.

Compared with these related works, the present paper is the first one to provide an *exact* theoretical performance formula for a pure CQ switch in terms of *both throughput and delay performance* with buffer sizes of *one and larger* under any independent Bernoulli (both uniform or non-uniform) traffic. The contributions of this work are summarized as follows:

- We prove that the CQ switch can achieve 100% throughput (or stability) as the crosspoint buffer size approaches infinity, under any work-conserving scheduling algorithms only under the assumption that the traffic obeys the strong law of large numbers (SLLN) and without output oversubscription.
- To the best of our knowledge, this paper presents the first *exact* closed-form formula of the CQ switch throughput for any given buffer size and presents the first *exact* non-closed-form (but convergent) formula of the delay for any buffer size, both under independent Bernoulli traffic, using a static non-work-conserving random (nWCRand) scheduling algorithm.
- Using mathematical proof as well as the comparison between the theoretical value and simulation results, we show that the theoretical value can serve as a conservative guide (a loose lower bound performance) for designing buffer sizes of a CQ switch using work-conserving scheduling algorithms.
- Our real-trace simulation results show that using simple work-conserving algorithms, the CQ switch can realize a very good performance with moderate memory resource consumption, which shows its feasibility in practical use.

Our work reveals the impact of buffer size on the CQ switch performance and provides a theoretical and conservative guidance on deciding the required buffer size in pure CQ switch, which is an important step toward building ultra-high-speed switch fabrics. Acquiring a better understanding of the CQ switch is also an important step toward building multi-stage and multi-plane switch fabrics with large capacity. Scaling up the CQ switch to a larger self-sufficient switch fabric is worthy of further study, which is beyond the scope of this current work.

II. CQ SWITCH

In this section, we briefly describe the CQ switch model and provide some fundamental definitions used for the rest of our paper.

A. CQ Switch Model

Let us consider an $N \times N$ CQ switch, as shown in Fig. 1. The ith input and ith output are denoted by I_i and O_i, respectively. XB_{ij} represents the crosspoint buffer between I_i and O_j, where $i, j = 1, \cdots, N$. We assume that time is slotted, and all the packets are segmented into fixed cells before being sent to the switch, and all the internal and external links of the CQ switch have the same capacity to transfer one cell per time slot. We follow this assumption for the rest of this paper. The XB_{ij} size is L_{ij} cells.

At the beginning of a time slot, one or no cell arrives at each input. If a cell arrives at input i heading to the output j at the start of a time slot, it is buffered in XB_{ij} in a first-in-first-out manner if the buffer is not full. The cell will be dropped when XB_{ij} is full. Within the same slot, the scheduler of each output independently selects one of the buffers in its column according to a certain scheduling algorithm and sends the head-of-line (HOL) cells out of the switch through the output if the selected buffer is not empty. If an empty buffer is selected, no cell is scheduled out through this output in this time slot. It is noteworthy that the departure steps at different output schedulers are performed in parallel.

B. Definitions

First, we provide some definitions that are related to the performance of a switch fabric.

Definition 1. The *throughput* of a switch fabric is the ratio of the number of cells that traversed the switch to the number of cells that arrive at the switch as time goes to infinity. We define TP as the throughput of the switch.

Definition 2. The *loss rate* of a switch fabric is the ratio of the number of cells dropped by the switch to the number of cells that arrive to the switch as time goes to infinity. We define LR as the loss rate of the switch.

Proposition 1. *For a switch fabric with finite buffers, the throughput of the switch is equal to one minus the loss rate of the switch if the average cell-arrival rate to the switch is greater than zero.*

Proof. Let us assume that the total buffers of the switch can contain L cells. We let λ denote the average arrival rate in all inputs of the switch as time goes to infinity and $L^*(n)$ denotes the total number of cells in the buffers of the switch at time slot n; hence $L^*(n) \le L$. We define C_a, C_l, and C_t as the total number of cells that arrive, are lost, and traverse the switch as time goes to infinity, respectively. Obviously, $C_a = \lim_{n\to\infty} \lambda \cdot n$, and $C_l = \lim_{n\to\infty} \lambda \cdot n \cdot LR$. Then, we have $C_t = \lim_{n\to\infty}(\lambda \cdot n - \lambda \cdot n \cdot LR - L^*(n))$. Thus,

$$TP = \frac{C_t}{C_a} = \lim_{n\to\infty}(1 - LR - \frac{L^*(n)}{\lambda \cdot n}) = 1 - LR.$$

□

Definition 3. The *delay* of a switch fabric is the average delay of all the cells that have traversed the switch as time goes to infinity. We define DL as the delay of the switch.

Then, we provide some definitions related to the scheduling algorithms.

Definition 4. A scheduling algorithm is called *work conserving* if, by using this scheduling algorithm, any output of the switch will always be busy if at least one buffer destined to this output is not empty. Otherwise, the scheduling algorithm is called *non-work conserving*.

Definition 5. A scheduling algorithm is called *static* if the rule of scheduling remains the same regardless of the system state. Otherwise, it is called *dynamic*.

Definition 6. A static random scheduling algorithm is called *fair* if at each time slot a column output scheduler randomly (and with the same probability) selects one of the crosspoint to send out its HOL cell.

Finally, we present a definition related to the arrival traffic.

Definition 7. The traffic in an input is considered to be *uniform* if each cell that arrives at the input has equal probability of heading to any output of the switch.

III. STABILITY PROOF

In this section, we first prove that the CQ switch can achieve 100% throughput (or stability) using any work-conserving scheduling algorithms as the buffer size approaches infinity. We draw this conclusion only under the assumption that the traffic sum of all inputs heading to the same output obeys the SLLN and the output ports are non-oversubscribed.

Let us consider the CQ switch model shown in Fig. 1. We use the fluid model introduced in [14] to prove the stability of the CQ switch and follow the same notations for ease in understanding. Before the proof, some definitions are given.

We let $A_{ij}(n), n = 1, 2, \cdots$ denotes the cells arriving at the ith input heading to the jth output up to the nth time slot. $A_j(n)$ denotes the total cells that arrive at all inputs heading to output j up to the nth time slot. Hence, $A_j(n) = \sum_{i=1}^{n} A_{ij}(n)$. $A(n)$ denotes the total number of cells that arrive to the switch up to time slot n.

Definition 8. The traffic heading to the same output is called as obeying the *SLLN*: with probability one,

$$\lim_{n\to\infty} \frac{A_j(n)}{n} = \lambda_j, \quad j = 1, \cdots, N. \tag{1}$$

We denote λ_j as the arrival rate of the traffic heading to O_j.

Definition 9. Output j is said to be *non-oversubscribed* if the traffic satisfies

$$\lambda_j \leq 1, \quad j = 1, \cdots, N. \tag{2}$$

We present the following theorem:

Theorem 1. *A CQ switch can achieve 100% throughput (i.e., stability) using any work-conserving scheduling algorithm when the traffic heading to any output of the switch obeys the SLLN and all the outputs are non-oversubscribed.*

Proof. We let $D_j(n)$ be the number of cells scheduled out from the switch through output j, i.e., the total departure of all the crosspoint buffers belonging to set $\{XB_{ij} | i = 0, 1, \cdots, N\}$ up to time slot n. We let $Z_j(n)$ be the total number of cells in all the crosspoint buffers belonging to set $\{XB_{ij} | i = 0, 1, \cdots, N\}$ at the beginning of time slot n.

Then, for the CQ switch that uses any work-conserving scheduling scheme, we can obtain the equation that for any $n \geq 0$ and $j = 0, 1, \cdots, N$,

$$Z_j(n) = Z_j(0) + A_j(n) - D_j(n) \geq 0. \tag{3}$$

$$D_j(n) = \sum_{l=1}^{n} 1_{\{Z_j(l)>0\}}. \tag{4}$$

Equation (3) shows the evolution of Z_j where the gross number of cells in all crosspoint buffers in column j at time slot n is equal to the initial number of cells in XB_j plus the total arrival $A_j(n)$ minus the total departure $D_j(n)$ both until time slot n. For a work-conserving algorithm, if $Z_j(n) > 0$, a cell is definitely scheduled out through output j. Thus, Equation (4) shows that total departure $D_j(n)$ of output j until time slot n is equal to the cumulative number of time slots when at least one crosspoint buffer in column j is not empty.

Following the convention in [14], we denote $\dot{f}_j(t)$ as the derivative of function f with respect to t. Now, the fluid model of the switch can be constructed as follows:

$$\begin{aligned} &Z_j(t) = Z_j(0) + \lambda_j t - D_j(t) \geq 0 \\ &\dot{D}_j(t) = 1, \text{ if } Z_j(t) > 0. \end{aligned} \tag{5}$$

According to Lemma 1 and Definition 3 in [14], the above fluid model of the CQ switch is defined as *weakly stable* if, for almost every t such that $Z_j(t) > 0$, we can prove that $\dot{Z}_j(t) \leq 0$.

Next, we prove that the above fluid model is weakly stable. We construct the following expression:

$$Z_j(t)\dot{Z}_j(t) = Z_j(t)(\lambda_j - \dot{D}_j(t)). \tag{6}$$

From Equation (5), when $Z_j(t) > 0$,

$$Z_j(t)\dot{Z}_j(t) = Z_j(t)(\lambda_j - 1). \tag{7}$$

According to the previous assumption, all outputs are non-oversubscribed. Therefore, $\lambda_j \leq 1$. Thus, we obtain $Z_j(t) \cdot \dot{Z}_j(t) \leq 0$ when $Z_j(t) > 0$, which means that $\dot{Z}_j(t) \leq 0$ for every t such that $Z_j(t) > 0$. Thus far, we have proven the weak stability of Equation (5).

Finally, from Theorem 3 in [14], a switch is *stable* if the corresponding fluid model is *weakly stable*. Therefore, the stability of the CQ switch has been proven. A stable switch has

$$\lim_{n\to\infty} \frac{D_j(n)}{n} = \lambda_j, \quad j = 1, \cdots, N \tag{8}$$

which represents the fraction of time when output j is busy as the time approaches infinity. Here, we can see that as arrival rate λ_j of the traffic heading to output j increases up to 100%, the switch can achieve 100% throughput. □

IV. PERFORMANCE ANALYSIS USING DIFFERENT BUFFER SIZES

Next, we focus on presenting a theoretical throughput and a delay calculation expression according to the buffer size in this section. We derive an exact closed-form formula for the throughput and an exact non-closed-form formula for the delay of the CQ switch assuming a static nWCRand scheduling algorithm and independent Bernoulli input traffic. Deriving such exact formula for the work-conserving scheduling algorithms is difficult [8]. Considering the feasibility and simplicity of the derivation, we use a non-work-conserving algorithm here. Moreover, we prove that the theoretical analysis result of the nWCRand scheduling algorithm can serve as a conservative guide (a loose lower bound performance) for designing the buffer sizes of a CQ switch using work-conserving scheduling algorithms.

We consider the CQ switch model shown in Fig. 1. We assume that the cell arrivals at each input are governed by an independent Bernoulli process and with fixed probability heading to each output. Each output scheduler uses a static nWCRand scheduling algorithm. We use the following notations:

$\rho_i \triangleq$ The Bernoulli parameter of the cell-arrival process in input I_i.

$a_{ij}^k \triangleq$ The probability of k cells arrived at XB_{ij} in a given time slot. $k = 0, 1$.

$d_{ij} \triangleq$ The probability of any cell arrived at I_i heading to output O_j. $\sum_{j=1}^{N} d_{ij} = 1$ and $0 \leq d_{ij} \leq 1$ for $i = 1, \cdots, N$.

$s_{ij} \triangleq$ The probability of crosspoint buffer XB_{ij} being selected by output O_j. $\sum_{i=1}^{N} s_{ij} = 1$ and $0 < s_{ij} < 1$ for $j = 1, \cdots, N$.

For a better understanding of these notations, we present a specific example. We consider the condition that the CQ switch uses fair random scheduling with uniform i.i.d Bernoulli traffic. Under this assumption, we can have $\rho_1 = \rho_2 = \cdots = \rho_N$ because the input arrivals are governed by the i.i.d Bernoulli process, and all the Bernoulli parameters are the same. Additionally, because the input traffic is uniform and the CQ switch uses fair random scheduling, we have $d_{ij} = s_{ij} = 1/N (i, j = 1, \cdots, N)$.

Let us assume that the cells that arrive at all the inputs are governed by an independent Bernoulli process and the CQ switch uses a static nWCRand scheduling algorithm. We present a formal description of the scheduling cycle in a time slot as follows:

- *Arrival Step:* At the beginning of a time slot, for input i, a probability of ρ_i exists that one cell will arrive, and a probability of $1 - \rho_i$ exists that no cell will arrive. The cell that arrives at input I_i has the probability d_{ij} to head to output O_j. The succeeding cells and cell arrivals at different inputs are independent.
- *Departure Step:* Within the same slot after the arrival step, each output scheduler picks a crosspoint buffer out of all the buffers in its column with a static nWCRand scheduling algorithm. For output O_j, it selects crosspoint buffer XB_{ij} with the probability s_{ij} and schedules the HOL cell out of the switch if the selected buffer is not empty. Otherwise, no cells are transmitted through O_j in this time slot. Each output independently schedules cells in parallel.

We let L_{ij} denote the capacity of crosspoint buffer XB_{ij} in the cells. We assume that $L_{ij} = L (i, j = 1, 2, \cdots, N)$ for ease in presentation, which means that all crosspoint buffers have the same capacity of L cells. We perform our analysis on particular crosspoint buffer XB_{ij} without losing generality.

We assume random variables A_{ij}, A_i, and A to be the number of cells arriving at XB_{ij}, input I_i and the whole switch during a given time slot, respectively. According to the conditions given earlier, the value of A_{ij} can only be zero or one. We recall that a_{ij}^k denotes the probability that k cells arrive at XB_{ij} in a time slot; then,

$$\begin{aligned} a_{ij}^0 &= P\{A_{ij} = 0\} = 1 - \rho_i d_{ij} \\ a_{ij}^1 &= P\{A_{ij} = 1\} = \rho_i d_{ij} \\ a_{ij}^k &= P\{A_{ij} = k\} = 0, \quad k \neq 0, 1. \end{aligned} \tag{9}$$

We define random variable $Q_{ij}(m)$ as the cells in XB_{ij} at the end of time slot m. According to the conditions stated before,

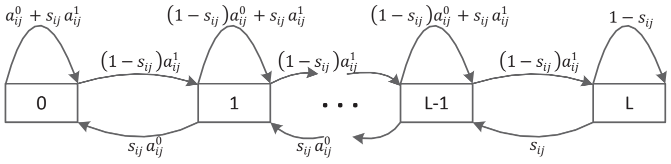

Fig. 2. Quasi-birth-death state transition diagram for XB_{ij}.

we can determine that $Q_{ij}(m)$ can be modeled as a discrete-time quasi-birth-death process, as shown in Fig. 2. The transition diagram can be interpreted as follows:

- The transitions from state l to $l + 1$ mean the probability of an arrival at the buffer, and the buffer is not selected by the output scheduler.
- The transitions from state l to itself are calculated under three different conditions. 1) When $l = 0$, it is equal to the probability of one arrival and one departure plus the probability of no arrival. 2) When $l = 1, \cdots, L - 1$, it is equal to the probability of one arrival and one departure plus the probability of no arrival and no departure. 3) When $l = L$, it is equal to the probability of one arrival and no departure (the cell will be dropped in the arrival step when the buffer is full) plus the probability of no arrival and no departure.
- The transitions from state l to state $l - 1$ are calculated under two different conditions. 1) When $l = 1, \cdots, L - 1$, it is equal to the probability of no arrival and one departure. 2) When $l = L$, it is equal to the probability of the buffer being selected (the buffer length will still be L before the departure step begins because the cell will be dropped in the arrival step when the buffer is full).

We let Q_{ij} denote the steady-state queue length of XB_{ij} according to the formula of the steady-state probabilities of the discrete-time quasi-birth-death process [15]. We can obtain the steady-state queue length distribution as follows:

$$\begin{aligned} \eta_{ij}^0 &= \frac{1}{1 + \sum_{l=1}^{L-1} \left(\frac{(1-s_{ij})a_{ij}^1}{s_{ij} a_{ij}^0} \right)^l + a_{ij}^0 \left(\frac{(1-s_{ij})a_{ij}^1}{s_{ij} a_{ij}^0} \right)^L} \\ \eta_{ij}^l &= \eta_{ij}^0 \left(\frac{(1 - s_{ij})a_{ij}^1}{s_{ij} a_{ij}^0} \right)^l, \quad l = 1, \cdots, L - 1 \\ \eta_{ij}^L &= \eta_{ij}^0 a_{ij}^0 \left(\frac{(1 - s_{ij})a_{ij}^1}{s_{ij} a_{ij}^0} \right)^L \end{aligned} \tag{10}$$

where η_{ij}^l defines the steady-state probability of the length of XB_{ij} to be equal to l, i.e., $Q_{ij} = l$.

So far, we have derived the steady-state probability distribution of the length of XB_{ij}. Next, we will use these results to analyze the throughput and delay of the CQ switch.

A. Throughput Analysis

Now, we start to analyze the throughput of the CQ switch using the results obtained earlier. Obviously, the probability of a cell arriving at XB_{ij} being dropped is equal to that of XB_{ij} being full, i.e., η_{ij}^L for steady-state.

We define random variables D_{ij}, D_i, and D as the number of cells dropped at XB_{ij}, input I_i, and the whole switch during a given time slot at steady-state, respectively. Obviously, D_{ij} and

D_i can only be zero or one. Then, we can obtain the probability of a cell arriving at input I_i being dropped in a time slot as

$$P\{D_i = 1|A_i = 1\} = \sum_{j=1}^{N}(d_{ij}\eta_{ij}^{L}). \tag{11}$$

The above equation is derived from the fact that the probability of a cell arriving at input i being dropped in a given time slot is equal to the sum of the probabilities that a cell arriving at I_i will be dropped at any crosspoint buffer of this line.

Further, we have the expectation of the dropped cells at I_i during a time slot as follows:

$$E(D_i) = \rho_i \sum_{j=1}^{N}(d_{ij}\eta_{ij}^{L}). \tag{12}$$

Thus, we derive the expectation of the dropped cells at the whole switch in a given time slot as

$$E(D) = E(\sum_{i=1}^{N} D_i) = \sum_{i=1}^{N}\left[\rho_i \sum_{j=1}^{N}(d_{ij}\eta_{ij}^{L})\right]. \tag{13}$$

Then, we obtain the loss rate of the CQ switch as follows

$$LR = \frac{E(D)}{E(A)} = \frac{\sum_{i=1}^{N}\sum_{j=1}^{N}\rho_i d_{ij}\eta_{ij}^{L}}{\sum_{i=1}^{N}\rho_i} \tag{14}$$

where random variable A denotes the number of cells arriving at the switch during a time slot and $E(A)$ denotes the expectation of A.

Therefore, from Proposition 1, we can obtain the closed-form formula of the throughput of the switch as

$$TP = 1 - LR = 1 - \frac{\sum_{i=1}^{N}\rho_i\left(\sum_{j=1}^{N} d_{ij}\eta_{ij}^{L}\right)}{\sum_{i=1}^{N}\rho_i}. \tag{15}$$

B. *Delay Analysis*

Subsequently, we analyze the average delay of the CQ switch. Similarly, we begin by focusing on certain crosspoint buffer XB_{ij}.

We let random variable W_{ij} and W_i denote the time slots that a cell spent at steady-state (i.e., *delay*) in XB_{ij} and input I_i respectively. At the time that a cell arrives at XB_{ij}, we assume that buffer length $Q_{ij} = l$. We recall that the delay of a switch fabric is defined as the average delay of all the cells that have traversed (not dropped) the switch (subsection II-B). Therefore, we only consider those arriving cells that are not dropped at the buffer, which means $l < L$, when they arrive. We let τ_{ij-l}^{n} denote the conditional probability that a cell is going to spend n time slots in XB_{ij} before being scheduled out under the following conditions: The cell comes to crosspoint XB_{ij}, the buffer length is l, and the cell is not dropped (i.e., $l < L$). Thus,

$$\begin{aligned}\tau_{ij-l}^{n} =& P\{W_{ij} = n|A_{ij} = 1, Q_{ij} = l\},\ \ l < L\\ =& C_n^{n-l}\left(1 - s_{ij}\right)^{n-l}\left(s_{ij}\right)^{l+1}\end{aligned} \tag{16}$$

where $n = l, l+1, \cdots, \infty$. This equation denotes that for a cell coming and buffered in XB_{ij}, the probability of cell delay $W_{ij} = n$ is equal to that of the buffer having been selected l times during n slots to move the cell to the HOL and the buffer being selected again immediately after n slots to schedule out the cell.

We note that a cell delay is doomed to be not less than l time slots because the switch has to spend at least l time slots to drain the l packets already queued in the buffer. We also recall that η_{ij}^{l} is defined as the probability of $Q_{ij} = l$. Thus, according to the law of total probability, for a cell coming to XB_{ij} and not being dropped, the steady-state probability of this cell delay in the switch being n time slots is equal to

$$\begin{aligned}&P\{W_{ij} = n|A_{ij} = 1, Q_{ij} < L\}\\ &= \begin{cases}\sum_{l=0}^{n} P\{Q_{ij} = l\}\cdot P\{W_{ij} = n|A_{ij} = 1, Q_{ij} = l\} \\ \qquad ,\ \ 0 < n \le L-1\\ \sum_{l=0}^{L-1} P\{Q_{ij} = l\}\cdot P\{W_{ij} = n|A_{ij} = 1, Q_{ij} = l\} \\ \qquad ,\ \ n > L-1\end{cases}\\ &= \begin{cases}\sum_{l=0}^{n}\left(\eta_{ij}^{l}\cdot\tau_{ij-l}^{n}\right),\ \ 0 < n \le L-1\\ \sum_{l=0}^{L-1}\left(\eta_{ij}^{l}\cdot\tau_{ij-l}^{n}\right),\ \ n > L-1\end{cases}.\end{aligned} \tag{17}$$

Then, using (9), (10), and (17), we can derive the following formula of the mean delay of a cell buffered in XB_{ij} as follows:

$$\begin{aligned}&E\{W_{ij}|A_{ij} = 1, Q_{ij} < L\}\\ &= \sum_{n=0}^{\infty} nP\{W_{ij} = n|A_{ij} = 1, Q_{ij} < L\}\\ &= \sum_{n=1}^{L-1} n\eta_{ij}^{0}s_{ij}\left(\frac{1 - s_{ij}}{a_{ij}^{0}}\right)^{n}\\ &+ \sum_{n=L}^{\infty} n\eta_{ij}^{0}s_{ij}(1 - s_{ij})^{n}\sum_{l=0}^{L-1} C_n^{n-l}\left(\frac{a_{ij}^{1}}{a_{ij}^{0}}\right)^{l}.\end{aligned} \tag{18}$$

Next, we draw the probability of a cell delay being n time slots if the cell comes from input I_i and traverses the switch, as follows[2]:

$$\begin{aligned}&P\{W_i = n|A_i = 1, Q_i < L\}\\ &= \sum_{j=1}^{N} d_{ij}P\{W_{ij} = n|A_{ij} = 1, Q_{ij} < L\}.\end{aligned} \tag{19}$$

Therefore, the mean delay of a cell coming into I_i and traversing the switch is equal to

$$\begin{aligned}&E\{W_i|A_i = 1, Q_i < L\}\\ &= \sum_{n=0}^{\infty} nP\{W_i = n|A_i = 1, Q_i < L\}\\ &= \sum_{j=1}^{N} d_{ij}E\{W_{ij} = n|A_{ij} = 1, Q_{ij} < L\}.\end{aligned} \tag{20}$$

Next, we acquire the delay of the switch (i.e., the average delay of all the cells that have traversed the switch). Assuming that

[2] In the following equation, Q_i denotes the length of the corresponding crosspoint buffer in which the cell coming from I_i is going to be buffered in.

a total of T time slots have passed from the beginning, then according to the Bernoulli traffic model and the packet loss rate obtained earlier, $T \cdot \rho_i \cdot (1 - E(D_i))$ cells come from I_i and traverse the switch (i.e., not being dropped), on average. Because each of these cells has a mean delay equal to $E\{W_i | A_i = 1, Q_i < L\}$, the mean sum delay of all cells coming from I_i and traversing the switch is equal to $T \cdot \rho_i \cdot (1 - E(D_i)) \cdot E\{W_i | A_i = 1, Q_i < L\}$. Then, the mean sum delay of all cells coming from all input ports into the switch is equal to

$$\sum_{i=1}^{N} T \cdot \rho_i \cdot (1 - E(D_i)) \cdot E\{W_i | A_i = 1, Q_i < L\}. \quad (21)$$

Thus the delay of the switch can be derived as follows:

$$DL = \frac{\sum_{i=1}^{N} \rho_i (1 - E(D_i)) E\{W_i | A_i = 1, Q_i < L\}}{\sum_{i=1}^{N} \rho_i (1 - E(D_i))}. \quad (22)$$

Although the above formula of the switch delay is not closed-form, we present a proof of its convergency as follows:

Proof. Obviously, from (20) and (22), we simply need to prove the convergency of $E\{W_{ij} | A_{ij} = 1, Q_{ij} < L\}$ to prove the convergency of Equation (22). We transform (18) and obtain

$$\begin{aligned} E\{W_{ij} | A_{ij} = 1, Q_{ij} < L\} &= \sum_{n=1}^{L-1} n \eta_{ij}^0 s_{ij} \left(\frac{1 - s_{ij}}{a_{ij}^0} \right)^n \\ &+ \sum_{l=0}^{L-1} \sum_{n=L}^{\infty} n \eta_{ij}^0 s_{ij} (1 - s_{ij})^n C_n^{n-l} \left(\frac{a_{ij}^1}{a_{ij}^0} \right)^l . \end{aligned} \quad (23)$$

We define function $f(L)$, which is the sum of infinite series $\{f_n\}$, as

$$f(L) = \sum_{n=L}^{\infty} f_n \quad (24)$$

where f_n is equal to

$$f_n = n \eta_{ij}^0 s_{ij} (1 - s_{ij})^n C_n^{n-l} \left(\frac{a_{ij}^1}{a_{ij}^0} \right)^l . \quad (25)$$

Then, all we need is to prove the convergency of $f(L)$. We now use the *ratio test* [16] to prove its convergency. We have

$$\begin{aligned} \frac{f_{n+1}}{f_n} &= \frac{(n+1) \eta_{ij}^0 s_{ij} (1 - s_{ij})^{n+1} C_{n+1}^{n+1-l} \left(\frac{a_{ij}^1}{a_{ij}^0} \right)^l}{n \eta_{ij}^0 s_{ij} (1 - s_{ij})^n C_n^{n-l} \left(\frac{a_{ij}^1}{a_{ij}^0} \right)^l} \\ &= \frac{n+1}{n} \cdot (1 - s_{ij}) \cdot \frac{(n+1)!}{(n+1-l)! l!} \cdot \frac{(n-l)! l!}{n!} \\ &= \frac{(n+1)^2}{n(n+1-l)} \cdot (1 - s_{ij}). \end{aligned} \quad (26)$$

Thus,

$$\begin{aligned} \lim_{n \to \infty} \frac{f_{n+1}}{f_n} &= \lim_{n \to \infty} \frac{(n+1)^2}{n(n+1-l)} \cdot (1 - s_{ij}) \\ &= 1 - s_{ij} < 1. \end{aligned} \quad (27)$$

Therefore, according to the *ratio test* [16] method, $f(L)$ is proven to be convergent. Thus, we can conclude that Equation (22) is convergent. □

So far, we have derived the precise expression of the CQ switch throughput and delay using static nWCRand scheduling algorithms. Naturally, an appropriate work-conserving scheduling algorithm will lead to a better performance compared with the nWCRand scheduling algorithm that we used to perform theoretical analysis. Next, we briefly prove that under independent Bernoulli traffic, the WCRand scheduling algorithm (randomly selecting a crosspoint-buffer from all the non-empty ones) performs better than the static nWCRand scheduling algorithm both in terms of the throughput and average delay.

Theorem 2. *Under the same independent Bernoulli traffic, a CQ switch using a WCRand scheduling algorithm has a higher throughput and lower average delay than that using an nWCRand scheduling algorithm.*

Proof. Similarly, we could also build a discrete-time quasi-birth-death diagram for WCRand, as shown in Fig. 2. As stated earlier, for a fair nWCRand, we have $s_{ij} = 1/N$ in each steady state of the queue length. Unlike the nWCRand, s_{ij} of WCRand between different states shown in Fig. 2 are not the same. We let $s'_{ij}(m)$ denote the probability of crosspoint buffer XB_{ij} being selected by output O_j using WCRand in state m and $s^*_{ij} = \max\{s'_{ij}(m), 0 \leq m \leq L\}$. η'^{l}_{ij} defines the steady-state probability of the length of XB_{ij} to be equal to l using WCRand. Because WCRand randomly selects a crosspoint-buffer from all the non-empty ones in each time slot, we can obtain

$$s^*_{ij} \geq \frac{1}{N} = s_{ij}. \quad (28)$$

Thus, according to the formula of the steady-state probabilities of the discrete-time quasi-birth-death process, we can obtain $\eta'^{L}_{ij} \leq \eta^{L}_{ij}$. Therefore, we can conclude that WCRand has a higher throughput than nWCRand using (15). In addition, from (18)–(22), we can easily determine that WCRand has a lower average delay than nWCRand. □

Similarly, if we use *the frequency of a crosspoint queue being selected by the work-conserving round-robin (WCRR) algorithm* to approximate *the probability of a crosspoint queue being selected by the WCRand algorithm,* we can prove that WCRR also shows better performance than nWCRand. Furthermore, it is intuitive that the longest-queue-first (LQF) scheduling has the highest throughput. The strict proof of these two work-conserving algorithms is beyond the scope of this paper. Later, we will show by simulations that the above theoretical analysis provides an appropriate lower-bound for a CQ switch performance using the work-conserving algorithms.

V. VERIFICATION OF THE ANALYSIS AND REAL TRACE SIMULATIONS

In this section, we first present the simulation results under both uniform and non-uniform Bernoulli traffic to verify our previous theoretical analysis in subsection V-A. We consider four scheduling algorithms in our simulations: nWCRand, WCRand,

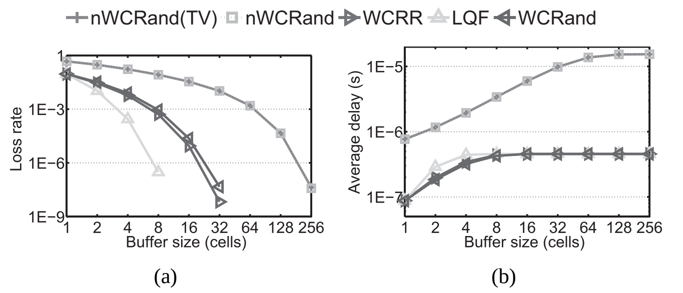

Fig. 3. Loss rate and average delay of a 16 × 16 CQ switch under uniform Bernoulli traffic with $\rho = 0.95$: (a) Loss rate and (b) average delay.

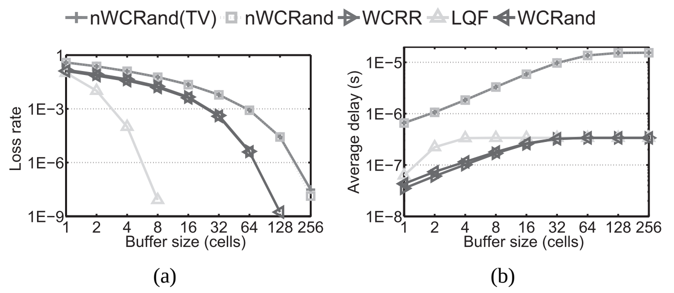

Fig. 4. Loss rate and average delay of a 16 × 16 CQ switch under non-uniform Bernoulli traffic with $\rho = 0.95$ and $\omega = 0.5$: (a) Loss rate and (b) average delay.

WCRR, and LQF. We calculate the theoretical value (TV) of the loss rate and the delay of the nWCRand scheduling algorithm according to the previous results we obtained under both uniform and non-uniform Bernoulli traffic. Various simulations have been done under different loads and using CQ switches with different port numbers. We present the results of a 16 × 16 10 Gbps CQ under a heavy load of 0.95. The results from other scenarios all verify our former theoretical analysis and are omitted here. All cells have a fixed length of 64 bytes (typical in commodity routers), and the time slot of the switch is set to 51.2 ns according to the transmission time of a cell on a 10-Gbps link. The displayed delay results are converted from time slots to seconds by multiplying 51.2 ns by the number of time slots. Each simulation run was conducted for 10^9 time slots.

Second, in subsection V-B, we present the simulation results of a 16 × 16 switch fabric under real-trace traffic using the four work-conserving scheduling algorithms mentioned above. We have shown that using the work-conserving algorithms, the CQ switch is able to realize good performance with moderate memory resource consumption. Our data consist of two parts from CAIDA [17]: two 1 min traces from 10 Gbps links, i.e., one at San Jose and another at Chicago, respectivly. All the packets are fragmented into 64-byte-long cells before being sent into the switch fabric, and the time slot is set to 51.2 ns in the same manner as presented earlier. We divide a 60 s trace into 16 equal size segments for 16 inputs. The destination port of each packet is set as the hash value of the destination IP address. The traffic distribution under this situation is not uniform but highly skewed and bursty. We believe this is similar to the real condition in the Internet. Approximately 1.7×10^7 packets with a total length of 1.15×10^{10} bytes are sent into the switch fabric during each experiment. We only present the simulation result of the San Jose trace; the results of the Chicago trace are similar.

A. Verification of the Performance Analysis

Figs. 3 and 4 show that the results of nWCRand scheduling algorithm are almost identical as the theoretical results we derived earlier under both uniform and non-uniform traffic. Investigation into the slight difference at the right end of the curves shows that the difference is due to the total experimental length of 10^9 time slots, thus unavoidably leading to an approximately 10^{-9} difference between the theoretical and simulation results. Under uniform Bernoulli traffic with heavy input load of 95%, as shown in Fig. 3(a), with crosspoint buffer size of 256 (such buffer size is easy to implement with modern semiconductor

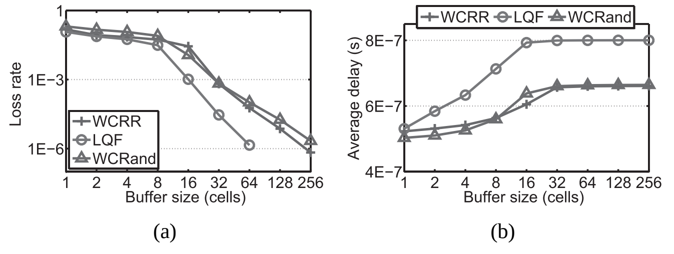

Fig. 5. Loss rate and average delay of a 16 × 16 CQ switch under real-trace traffic: (a) Loss rate and (b) average delay.

technology [8]), the loss rate of nWCRand can be as low as 10^{-7} using the theoretical results we obtained earlier. Such a loss rate is good enough for many switch fabric designs and provides a loose performance lower bound. Our results also show that with only a buffer size of 32 cells, simple work-conserving algorithms such as WCRR and WCRand can realize the same performance in which nWCRand realizes with buffer size of 256 cells. Further, the experiment shows that the theoretical results could serve as a loose performance lower bound for these algorithms. Using a more elaborate scheduling algorithm such as LQF, no packets are lost during the 10^{-9} time slot simulation with only a buffer size of 16. With regard to the average delay, our theoretical analysis shows that with a buffer size of 64, a CQ switch can have a stable average delay of approximately 10^{-5} s using nWCRand, which is shown in Fig. 3(b). Meanwhile, the average delay is much lower at less than 10^{-6} s using the work-conserving algorithms.

Similarly, under non-uniform traffic, as shown in Fig. 4, our analytic results are also verified and effectively provide loose performance lower bound to work-conserving algorithms. ω in the figure defines the unbalanced probability (refer to [3]) and $\omega = 0.5$ means the traffic is extremely non-uniform. In this situation, WCRand and WCRR have a much higher loss rate than the uniform traffic due to the blindness to the traffic distribution of their scheduling manner. In contrast, LQF can adjust to the imbalance. We have pre-adjusted the random weight of nWCRand to be equal to the unbalance degree of the input traffic; thus, its results are very similar to the uniform traffic.

B. Simulations under Real Trace

Fig. 5 shows the result of a 16×16 CQ switch under real-trace traffic. The loss rate shown in Fig. 5(a) refers to the packet loss rate. Once a cell of a packet is dropped, the packet is counted as lost in the switch. We can see that a pure CQ switch can achieve good performance with simple work-conserving scheduling algorithms. Fig. 5(a) shows that the switch has a loss rate down to 10^{-6} with a buffer size of 64 using LQF. A simple round-robin or random scheduling is able to reach the same performance with a buffer size of 256, which is totally within the capability of modern chip technology. In addition, the delay performance shown in Fig. 5(b) is very good. The delay here refers to the average packet delay. We can see that the WCRR and WCRand have better delay performance than LQF because starvation, which greatly increases the delay, may occur using LQF algorithm. This result under real-trace traffic demonstrates that a CQ switch with such scale can realize a very good performance with a feasible crosspoint buffer size. Thus, a self-sufficient CQ switch is suitable for ultra-high-speed link in practical use.

VI. CONCLUSION

This paper reveals the impact of buffer size on CQ switch performance and provides a theoretical guidance on designing the buffer size in a pure CQ switch. In addition, we show that CQ is a promising building block for high line-rate switch fabrics. As a next step, we plan to actually design ultra-high-speed and large-port-number switch fabrics with multi-plane or mutli-stage structure using CQ as building blocks to scale up. We also plan to design scheduling algorithms with performance better than the round-robin, random, or LQF algorithms analyzed and evaluated in this paper.

ACKNOWLEDGMENT

The authors would like to thank Kuan Cheng for his valuable comments on the mathematical demonstration, Zhiyan Zheng for the packet trace of real operational network that he provided us, and Juexing Liao for her proofreading of this paper. This paper was previously presented in part [18] at NPC'14 Conference, Ilan Taiwan, in September 2014.

REFERENCES

[1] N. McKeown, "The iSLIP scheduling algorithm for input-queued switches," *IEEE/ACM Trans. Netw.*, vol. 7, no. 2, pp. 188–201, Apr. 1999.

[2] Y. Li, S. Panwar, and H. Chao, "On the performance of a dual round-robin switch," in *Proc. IEEE INFOCOM*, 2001, pp. 1688–1697.

[3] R. Rojas-Cessa, E. Oki, Z. Jing, and H. Chao, "CIXB-1: Combined input-one-cell-crosspoint buffered switch," in *Proc. HPSR*, 2001, pp. 324–329.

[4] S. He *et al.*, "On guaranteed smooth switching for buffered crossbar switches," *IEEE/ACM Trans. Netw.*, vol. 16, no. 3, pp. 718–731, June 2008.

[5] E. Oki, Z. Jing, R. Rojas-Cessa, and H. Chao, "Concurrent round-robin-based dispatching schemes for Clos-network switches," *IEEE/ACM Trans. Netw.*, vol. 10, no. 6, pp. 830–844, 2002.

[6] S. Iyer, A. Awadallah, and N. McKeown, "Analysis of a packet switch with memories running slower than the line-rate," in *Proc. IEEE INFOCOM*, 2000, pp. 529–537.

[7] F. Abel *et al.*, "Design issues in next-generation merchant switch fabrics," *IEEE/ACM Trans. Netw.*, vol. 15, no. 6, pp. 1603–1615, Dec. 2007.

[8] Y. Kanizo, D. Hay, and I. Keslassy, "The crosspoint-queued switch," in *Proc. IEEE INFOCOM*, Apr. 2009, pp. 729–737.

[9] M. Radonjic and I. Radusinovic, "Impact of scheduling algorithms on performance of crosspoint-queued switch," *Annals of Telecommunications - Annales des Tlcommunications*, vol. 66, pp. 363–376, 2011. [Online]. Available: http://dx.doi.org/10.1007/s12243-010-0214-y

[10] I. Radusinovic, M. Radonjic, A. Simurina, I. Maljevic, and Z. Veljovic, "A new analytical model for the CQ switch throughput calculation under the bursty traffic," *AEU - International Journal of Electronics and Communications*, vol. 66, no. 12, pp. 1038–1041, 2012. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S143484111200129X

[11] Z. Cao and S. Panwar, "Efficient buffering and scheduling for a single-chip crosspoint-queued switch," *IEEE Tran. Commun.*, vol. 62, no. 6, pp. 2034–2050, 2014.

[12] G. Chen, D. Pei, and Y. Zhao, "CQRD: A switch-based approach to flow interference in data center networks," in *Proc. IEEE LCN*, 2014, pp. 107–115.

[13] "Cisco CRS carrier routing system 16-slot line card chassis system description," *Cisco Systems, Inc.*, 2012.

[14] J. Dai and B. Prabhakar, "The throughput of data switches with and without speedup," in *Proc. IEEE INFOCOM*, 2000, pp. 556–564.

[15] K. Trivedi, *Probability and Statistics with Reliability, Queuing, and Computer Science Applications.* Wiley New York, 2002.

[16] T. Bromwich, *An Introduction to the Theory of Infinite Series*. Macmillan, 1908.

[17] "Anonymized 2013 internet traces," 2013. [Online]. Available: https://data.caida.org/datasets/passive-2013/

[18] G. Chen, D. Pei, Y. Zhao, and Y. Sun, "Designing buffer capacity of crosspoint-queued switch," in *Proc. NPC*, 2014, pp. 35–48.

Guo Chen received the B.S. degree from Wuhan University (China) in 2011. He is currently a Ph.D. student in the Department of Computer Science at Tsinghua University, Beijing, China. His current research interests include data center networking and high performance switching.

Youjian Zhao received the B.S. degree from Tsinghua University, Beijing, China, in 1991, the M.S. degree from Shenyang Institute of Computing Technology, Chinese Academy of Sciences, in 1995, and the Ph.D. degree from Northeastern University, China, in 1999. He is currently a Professor with the CS Department, Tsinghua University. His research mainly focuses on high-speed switching and routing.

Dan Pei received the B.S. and M.S. degrees from Tsinghua University, Beijing, China in 1997 and 2000, respectively, and the Ph.D. degree from the University of California, Los Angeles, in 2005. He is currently an Associate Professor with Tsinghua University. Currently he is focusing on improving the Mobile Internet performance over Wi-Fi networks and data center networks.

Yongqian Sun received the B.S. degree from Northwestern Polytechnical University (China). He is currently a Ph.D. student in the Department of Computer Science at Tsinghua University, Beijing, China. His current research focuses on anomaly detection and network security.