# The Throughput of a Buffered Crossbar Switch

Mingjie Lin, *Student Member, IEEE,* and Nick McKeown, *Senior Member, IEEE*

***Abstract*— The throughput of an input-queued crossbar switch – with a single FIFO queue at each input – is limited to $2-\sqrt{2} \approx 58.6\%$ for uniformly distributed, Bernoulli i.i.d. arrivals of fixed length packets. In this letter we prove that if the crossbar switch can buffer one packet at each crosspoint, then the throughput increases to $100\%$ asymptotically as $N \to \infty$, where $N$ is the number of switch ports.**

***Index Terms*— Input-queued switch, buffered crossbar switch, throughput.**

## I. Introduction

In their seminal paper in 1987, Karol *et al.* proved that the throughput of an input queued (IQ) crossbar switch – with a single FIFO queue at each input – is limited by head-of-line (HOL) blocking [1]. In particular, the authors showed that when packets are fixed length, arrivals are a Bernoulli i.i.d. process, and the destination of each packet is picked uniformly, independently and at random from among the set of switch outputs, then the throughput is limited to $2-\sqrt{2} \approx 58.6\%$. This widely cited result led to a collection of techniques to overcome head-of-line blocking and increase throughput, such as lookahead schemes, virtual output queues (VOQs), and speedup, *et al.* (see [2][3] for more references).

There has also been interest in crossbars with a single packet buffer at each crosspoint. A buffered crossbar is interesting because it can achieve 100% throughput with a much simpler scheduling algorithm compared with crossbar switch without crosspoint buffer [2], [3]. Buffered crossbars are possible because of large improvements in gate count; nowadays, crossbar capacity is limited by I/O speed, not die size. This leaves room on the die for a buffer at each crosspoint.

Early simulation studies suggested that buffered crossbar switch can achieve high throughput under various admissible traffic patterns [4][5][6]. Recent analytical studies show conditions for which a buffered crossbar with VOQs can achieve 100% throughput [7][8][9].

In this paper, we introduce what we believe to be the first analytical result for crossbar switches with a buffer at each crosspoint, but without speedup or VOQs.

## II. Buffered Crossbar Switch

Figure 1 shows an $N \times N$ buffered crossbar fabric with a one-packet buffer at each crosspoint. For the purposes of this letter, we will assume that time is slotted and that fixed-size packets arrive according to an i.i.d. Bernoulli random process. Each input port $i$ has a single input queue with infinite capacity. Each arriving packet has a destination port number $j$ selected uniformly and at random from all $N$ output ports. At the crosspoint that connects input $i$ to output $j$ there is a buffer $B_{i,j}$ that can hold at most one packet.

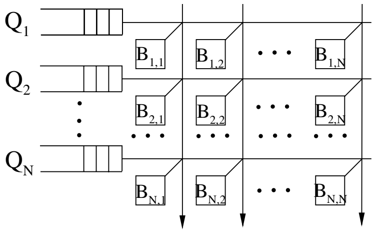


Fig. 1. Diagram of a $N \times N$ Buffered Crossbar Switch

We will divide each time slot into two phases. During the first phase, the scheduler at each input – independently and in parallel – examines the packet at the head of its FIFO and decides whether to place it into a crosspoint buffer. Specifically, if the HOL packet in $Q_i$ is destined to output port $j$, and crosspoint buffer $B_{i,j}$ is empty, then packet $i$ will be immediately removed and put into $B_{i,j}$. Otherwise, packet $i$ will either be dropped or kept depending on scheduling policy of the switch. During the second phase, the scheduler at each output port – independently and in parallel – decides whether to serve any of the crosspoint buffers destined to it. Specifically, output port $j$ will examine crosspoint buffers $B_{i,j}$, where $i = 1, 2, 3, \cdots, N$. If all the crosspoint buffers are empty, then output port $j$ will do nothing. If one or more crosspoint buffers are full, then output $j$ will pick one (from among the set of full buffers) uniformly and at random and serve it.

## III. Analytical Model

In the remainder of the letter, we will use the following notation for a $N \times N$ buffered crossbar switch.

- $C_j$ – The $j$th column of crosspoint buffers, i.e., the union $\cup_{i=1}^{N} B_{i,j}$. We also denote the number of packets residing in $C_j$ as $\left|C_j\right|$.
- $\pi_k$ – For any column of crosspoint buffers $C_j$, the invariant probability of $\left|C_j\right|$ equals $k$ at the end of a time slot.
- $p_{i,j}$ – Probability that, in a single time slot, a column of crosspoint buffers will transition from having $i$ packets to $j$ packets. This will be used to define the state transition probability of a Markov chain representing the occupancy of crosspoint buffers.
- $S_N$ – The saturation throughput of a $N \times N$ buffered crossbar switch with size 1 at each crosspoint buffer.

Manuscript received September 21, 2004. The associate editor coordinating the review of this letter and approving it for publication was Prof. David Petr. This work was supported by Cisco Systems and the Stanford Networking Research Center.

The authors are with the Computer Systems Laboratory, Stanford University (e-mail: mingjie@stanford.edu).

Digital Object Identifier 10.1109/LCOMM.2005.05012.

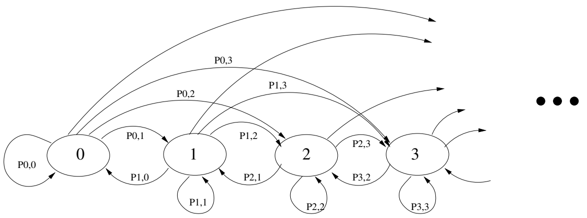


Fig. 2. Diagram of a Markov chain to model state transition of the switch.

Because of the symmetry of the traffic and the switching fabric, the maximum utilization of any output port will be identical. We define saturation throughput simply as the maximum utilization of the output port $j$.

At the beginning of a time slot $n$, if $|C_j| > 0$, the output port will output a packet; the probability of that event is $1-\pi_0$. Furthermore, even if $|C_j| = 0$ at the beginning of the time slot $n$, the column might receive an incoming packet with probability $1-(1-1/N)^N$ during the buffer-in phase of time slot $n$, and hence output port $j$ will send a packet by the end of the time slot $n$. Therefore, we have the following formula of the saturation throughput.

$$S_N = (1-\pi_0) + \pi_0 \cdot [1-(1-1/N)^N] \qquad (1)$$

In what follows, we prove our main result in two stages. First, we prove that a buffered crossbar switch that drops blocked packets (rather than buffering them at the input) has a throughput of $100\%$ as $N \to \infty$. We then consider the more interesting case in which blocked packets are buffered in the input queue. We show that the difference between the two cases is sufficiently small that throughput is still $100\%$ as $N \to \infty$.

### *A. A Buffered Crossbar Switch that Drops Blocked Packets*

We assume the following in the analysis:

- The input data traffic at each input port are Bernoulli i.i.d. uniform arrivals and the sum of all input traffic load is 1.
- At the beginning of each time slot, if a packet at the head of an input queue (Head-of-line packet) finds that its crosspoint buffer is occupied, then the packet will be immediately dropped.

Our rationale for making the above assumption is to simply ease the analysis; as we will see in the next section, this also leads us to the more interesting case when the HOL packet is not dropped. The system can then easily be modeled by a Discrete Time Markov Chain. The number of occupied crosspoint buffer cells in $C_j$ is chosen as the state of the Markov chain, and - by symmetry - represents the distribution of the occupancy of all columns. The following is a transition diagram of the Markov chain:

Let $p_{i,j}$ denote the probability of a transition from state $i$ to state $j$. According to the theorem on total probability, we have:

$$\pi_k = \sum_{i=0}^{N} \pi_i \cdot p_{i,k} \text{ where } k = 1, 2, \cdots, N \qquad (2)$$

To solve this set of linear equations, we need to find $p_{i,j}$. Note that during the time slot $n$, the state transition $i \to j$ occurs if and only if $j+1-i$ packets arrive to empty buffers in column $j$, which happens with probability $C_{N-i}^{j+1-i} \cdot \left(\frac{1}{N}\right)^{j+1-i}\left(1-\frac{1}{N}\right)^{N-(j+1-i)-i}$ by a simple combinatoric argument. If $j+1 \geq i$ and $i, j \in [0, N-1]$, then:

$$p_{i,j} = C_{N-i}^{j+1-i} \cdot \left(\frac{1}{N}\right)^{j+1-i}\left(1-\frac{1}{N}\right)^{N-j-1} + \alpha \qquad (3)$$

where, $\alpha = \delta(i,j) \cdot \left(1-\frac{1}{N}\right)^N$ [1]. Otherwise, $p_{i,j} = 0$.

In summary, in order to solve for the saturation throughput (equation (1)), we need to first find $\pi_0$ by solving a set of linear equations $\mathcal{L}$ that consists of $\pi_N = 0$, $\sum_{k=0}^{N} \pi_k = 1$, and equations from (2). Finding a closed form solution to $\mathcal{L}$ is hard, but numerical solutions are easy for finite $N$.

*Theorem 1:* The throughput of a $N \times N$ buffered crossbar that drops blocked packets is 100% as $N \to \infty$ (i.e. $S_\infty = 1$) for Bernoulli i.i.d. uniform arrivals.

*Proof:* We will first prove that $\pi_0 = 0$ which, from Equation (1) leads to $\lim_{N\to\infty} S_N = 1$.

From Equation (3), when $N \to \infty$, $p_{i,j} = \frac{e^{-1}}{(j+1-i)!} + \delta(i,j) \cdot e^{-1}$, and the set of linear equations $\mathcal{L}$ becomes $\pi_0 = \pi_0 \cdot 2e^{-1} + \pi_1 \cdot \frac{e^{-1}}{0!}$ and $\pi_k = \sum_{i=0}^{k+1} \pi_i \cdot \frac{e^{-1}}{(k+1-i)!}$, where $k \in [1, \infty)$. Adding all the above linear equations in $\mathcal{L}$, and merging terms:

$$\begin{aligned}
\sum_{i=0}^{\infty} \pi_i &= \pi_0 \cdot e^{-1} + \sum_{k=0}^{\infty} \sum_{i=0}^{k+1} \pi_i \cdot \frac{e^{-1}}{(k+1-i)!} \\
&= \pi_0 \cdot e^{-1} + \sum_{i=0}^{\infty} \sum_{k=i-1}^{\infty} \pi_i \cdot \frac{e^{-1}}{(k+1-i)!} \\
&= \pi_0 \cdot e^{-1} + \sum_{i=0}^{\infty} \pi_i \left(\sum_{j=0}^{\infty} \frac{e^{-1}}{j!}\right) \quad (\text{let } j \text{ be } k-i+1) \\
&= \pi_0 \cdot e^{-1} + \sum_{i=0}^{\infty} \pi_i
\end{aligned}$$

which implies $\pi_0 = 0$, and hence from equation (1), $S_\infty = \lim_{N\to\infty} S_N = 1$. ■

Consider a particular buffer column $C_j$ in a $N \times N$ buffered crossbar switch that drops blocked HOL packets, i.e. each input queue contains at most one packet, we now derive the probability $P_N(b)$ that incoming packet blocking/dropping actually happens during each time slot for a long time duration. Assume that a new packet arrives to every input at the beginning of time slot $n$ and the incoming packets have a destination picked uniformly, independently, and at random from the $N$ output ports. We denote the probability that none of the arriving packets is blocked at the column $C_j$ is $\overline{P_N(b)}$, i.e., $1 - P_N(b)$. Clearly, $\overline{P_N(b)}$ is only a function of the total number of packets $k$ in $C_j$ and switch size $N$. In order to find the total non-blocking probability, we condition on the total number of packets in $C_j$ and have $\overline{P_N(b)} = \sum_{k=0}^{N} \overline{P_N(b|k)} \cdot \pi_k$, where $\overline{P_N(b|k)}$ denotes the conditional

[1] The additional term $\alpha$ is to account for the special case when $i, j = 0$ and there is no incoming packet for any crosspoint buffer in $C_j$. $\delta(i,j)$ is 1 iff $i, j = 0$.

probability that none of the newly arriving packets are blocked given that there are $k$ packets in $C_j$ and can be shown to be $\left(\frac{N-1}{N}\right)^k$ by a simple combinatoric argument. Substitute (2) into this equation [2], we obtain the following, $\overline{P_N(b)} = \sum_{k=0}^{N} \overline{P_N(b|k)} \sum_{i=0}^{k+1} \pi_i \cdot p_{i,k} = \sum_{i=0}^{N+1} \sum_{k=i}^{N} \overline{P_N(b|k)} \cdot \pi_i \cdot p_{i,k} = \sum_{i=0}^{N+1} \pi_i \left( \sum_{k=i}^{N} \overline{P_N(b|k)} \cdot p_{i,k} \right)$. We know $\overline{P_N(b|k)} = \left(\frac{N-1}{N}\right)^k$ and $p_{i,k}$ from equation (3). Therefore, for any finite switch size $N$, we can numerically solve both $P_N(b)$ and $\overline{P_N(b)}$. Once again, closed form solution is hard to obtain.

Now we consider the asymptotic case when $N \rightarrow \infty$. $\lim_{N\rightarrow\infty} \left( \overline{P_N(b|k)} \cdot p_{i,k} \right)$ can be reduced to $\frac{e^{-1}}{(k+1-i)!}$. Therefore, $\overline{P_\infty(b)} = \sum_{i=0}^{\infty} \pi_i \cdot \sum_{k=i}^{\infty} \frac{e^{-1}}{(k+1-i)!} = 1$, which leads to $P_\infty(b) = 1 - \overline{P_\infty(b)} = 0$. Therefore the following lemma is true.

*Lemma 1:* The blocking/dropping probability approaches 0 as $N \rightarrow \infty$ for a buffered crossbar that drops blocked packets with Bernoulli i.i.d. uniform arrivals.

### *B. A Buffered Crossbar Switch that Buffers Blocked Packets*

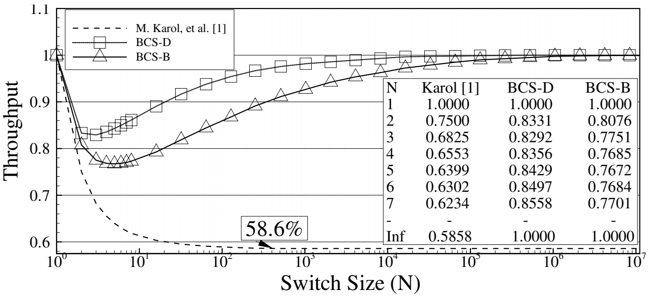
We will now remove the assumption that blocked packets are dropped. In other words, if its crosspoint buffer is occupied, the HOL packet will wait at the HOL position until the crosspoint buffer is available. Intuitively, this behavior creates HOL blocking and reduces saturation throughput for any finite size buffered crossbar switch (simulation results in Fig. 3). Unfortunately, because of HOL packet buffering, assumptions about the independence among all output ports and Bernoulli i.i.d. uniform arrivals are not longer valid. We will not be able to use Markov chain to model the switching behavior and analyze saturation throughput. Surprisingly, in the asymptotic case when $N \rightarrow \infty$, it can be shown that the saturation throughput will still converge to 1 (Simulation results in Fig.3 also supports this claim).

In previous section, lemma 1 shows that for $\infty \times \infty$ buffered crossbar switch, the dropping or blocking probability for each incoming packet actually converges to 0 for Bernoulli i.i.d. uniform arrivals, which implies that, *statistically and asymptotically*, the existence of packet dropping policy has no effect on the throughput and the switching behavior. This means that there is asymptotically no difference in throughput between the system that drops blocked HOL packets, and the system that doesn't. Therefore, the following theorem is true.

*Theorem 2:* The saturation throughput of a buffered crossbar that buffers blocked packets is still 100% as $N \rightarrow \infty$ for Bernoulli i.i.d. uniform arrivals.

### *C. Results Summary*

Figure 3 compares theoretical and simulation results [3] and shows that for finite switch size, the saturation throughput of a buffered crossbar is larger than the corresponding crossbar switch. Not surprisingly, for finite $N$, a buffered crossbar that drops blocked packets has a higher throughput than one that doesn't. It's interesting to note that for the buffered crossbar, the throughput has a minimum value. Overall, the throughput of a buffered crossbar increases with $N$, which is in sharp contrast with an unbuffered crossbar in which the throughput *decreases* with $N$.



| N | Karol [1] | BCS-D | BCS-B |
|---|---|---|---|
| 1 | 1.0000 | 1.0000 | 1.0000 |
| 2 | 0.7500 | 0.8331 | 0.8076 |
| 3 | 0.6825 | 0.8292 | 0.7751 |
| 4 | 0.6553 | 0.8356 | 0.7685 |
| 5 | 0.6399 | 0.8429 | 0.7672 |
| 6 | 0.6302 | 0.8497 | 0.7684 |
| 7 | 0.6234 | 0.8558 | 0.7701 |
| - | - | - | - |
| Inf | 0.5858 | 1.0000 | 1.0000 |

Fig. 3. Comparison of throughput for all 3 cases (BCS denotes Buffered Crossbar Switch, and B and NB denotes with blocking and without blocking).

## IV. Conclusion

Until recently, buffered crossbars were known only through simulations. But now there is a growing body of analytical results for buffered crossbars, to which this paper contributes one more addition. While our results probably don't have direct practical application (after all, real network traffic is neither Bernoulli i.i.d. or uniform), the results give us more understanding and intuition about how buffered crossbars behave and perform. In particular, this paper indicates that in a buffered crossbar with a large number of ports, the crosspoint buffers have a significant bearing on the system performance, and essentially dominate the buffering encountered by a typical packet. With a large enough number of ports, the crosspoint buffers provide enough diversity for the input and output schedulers, to effectively eliminate head of line blocking.

## References

[1] M. Karol, M. Hluchyj, and S. Morgan, "Input versus output queueing on a space-division switch," *IEEE Trans. Commun.*, vol. 35, pp. 1347–1356, Dec. 1987.

[2] N. McKeown, A. Mekkittikul, V. Anantharam, and J. Walrand, "Achieving 100% throughput in an input-queued switch (extended version)," *IEEE Trans. Commun.*, vol. 47, pp. 1260–1267, Aug. 1999.

[3] J. G. Dai and B. Prabhakar, "The throughput of data switches with and without speedup," in *Proc. of IEEE INFOCOM'00*, pp. 556–564, Mar. 2000.

[4] M. Nabeshima, "Performance evaluation of a combined input- and crosspoint-queued switch," *IEICE Trans. Commun.*, vol. E83-B, pp. 737–741, Mar. 2000.

[5] R. Rojas-Cessa, E. Oki, Z. Jing, and H. J. Chao, "CIXB-1: Combined input-once-cell-crosspoint buffered switch," in *IEEE Workshop on High Performance Switching and Routing*, (Dallas, TX), July 2001.

[6] D. Stephens and H. Zhang, "Implementing distributed packet fair queueing in a scalable switch architecture," in *Proc. of IEEE INFOCOM'98*, pp. 282–290, Apr. 1998.

[7] T. Javidi, R. Magill, and T. Hrabik, "A high throughput scheduling algorithm for a buffered crossbar switch fabric," in *Proc. of IEEE International Conference on Communications'01*, vol. 5, pp. 1586–1591, June 2001.

[8] R. B. Magill, C. E. Rohrs, and R. L. Stevenson, "Output-queued switch emulation by fabrics with limited memory," *IEEE J. Sel. Areas Commun.*, vol. 21, pp. 606 –615, May 2003.

[9] S. Iyer, S. T. Chuang, and N. McKeown, "Practical algorithms for performance guarantees in buffered crossbars," to be presented at IEEE INFOCOM'05, (Miami, FL), Mar. 2005.

[2] Note when $i > k + 1$, $p_{i,k}$ equals 0.

[3] The simulation program source code can be downloaded from: http://www.stanford.edu/∼mingjie/bufx-sim.c