Throughput Analysis of Shared-Memory Crosspoint Buffered Packet Switches

Ziqian Dong and Roberto Rojas-Cessa

Abstract

This paper presents a theoretical throughput analysis of two buffered-crossbar switches, called shared-memory crosspoint buffered (SMCB) switches, in which crosspoint buffers are shared by two or more inputs. In one of the switches, the shared-crosspoint buffers are dynamically partitioned and assigned to the sharing inputs, and memory is sped up. In the other switch, inputs are arbitrated to determine which of them accesses the shared-crosspoint buffers, and memory speedup is avoided. SMCB switches have been shown to achieve a throughput comparable to that of a combined input-crosspoint buffered (CICB) switch with dedicated crosspoint buffers to each input but, with less memory than a CICB switch. The two analyzed SMCB switches use random selection as the arbitration scheme. We model the states of the shared crosspoint buffers of the two switches using a Markov-modulated process and prove that the throughput of the proposed switches approaches 100% under independent and identically distributed uniform traffic. In addition, we provide numerical evaluations of the derived formulas to show how the throughput approaches asymptotically to 100%.

Index Terms

Buffered crossbar, throughput, speedup, shared memory, crosspoint buffer.

I. Introduction

Combined input-crosspoint buffered (CICB) switches, also referred to as combined input-crosspoint queued (CICQ) switches, have recently captured significant research interest because they offer higher performance than input-queued (IQ) packet switches using configuration schemes with lower complexity [1]–[22]. In addition, CICB switches allow a longer time than an IQ switch to select the packets that are to be forwarded from an input to an output port while requiring a memory speed equal to that of IQ switches to deliver high switching performance.

Ziqian Dong is with the Department of Electrical and Computer Engineering, New York Institute of Technology, New York, NY 10023. Email: ziqian.dong@nyit.edu.

Roberto Rojas-Cessa is with the Department of Electrical and Computer Engineering, New Jersey Institute of Technology, Newark NJ 07102. Email: rojas@njit.edu.

This paper follows the mainstream practice of segmenting incoming variable-size packets into fixed-length packets, called cells, at the ingress side of a switch. Packets are reassembled at the egress side, before they depart from the switch. Therefore, the time it takes to transmit a cell from an input to an output is constant, and it is called time slot. In this paper, the terms cells and time slots are used interchangeably.

This paper considers admissible traffic, defined as $\sum_{i=0}^{N-1} \rho_{i,j} \leq 1$, $\forall j$ and $\sum_{j=0}^{N-1} \rho_{i,j} \leq 1$, $\forall i$, where i is the input port number ($0 \leq i \leq N-1$), j is the output port number ($0 \leq j \leq N-1$), and $\rho_{i,j}$ is the input load from input i destined to output j.

CICB switches, as IQ switches, adopt virtual output queues (VOQs) at the inputs to avoid head-of-line (HOL) blocking [23]. A VOQ stores cells from an input port destined to an output port. The amount of memory in the buffered crossbar of a CICB switch is kLN^2 bytes, where N is the number of switch ports, L is the cell size in bytes, and k is the number of cells that can be stored in a crosspoint buffer. A CICB switch uses a dedicated crosspoint buffer for each i, j pair [2], [5].

The CICB also uses a credit-based flow control mechanism to avoid crosspoint-buffer overflow. Buffer underflow is another phenomenon that can decrease the throughput of buffered-crossbar switches. Buffer underflow occurs when the cells arrive in the buffers at a lower rate than that at which cells are served from the buffers. This may occur when the line cards of a CICB switch are placed far from the buffered crossbar. Buffer underflow is avoided by setting the size of crosspoint buffers equal to or larger than the round-trip time (RTT) measured between the line cards (inputs) and the buffered crossbar. Here, RTT is defined as the sum of the time it takes to transmit a cell from an input to the buffered crossbar, $d1$; the time it takes to transmit flow-control information from the buffered crossbar to the input that indicates crosspoint-buffer occupancy, $d2$; and the input and output arbitration delays, IA and OA, respectively ($RTT = d1 + d2 + IA + OA$). The unit of RTT is the number of time slots, however, as one cell can be transmitted in one time slot; it can be also referred to as the number of cells. In a CICB switch, the required crosspoint-buffer size to avoid underflow for flows with data rate R_c b/s (or cells/time slot), where R_c is the port speed, is such that $k \geq \frac{R_c \ RTT}{L}$. Herein, a flow is defined as the set of packets from input i destined to output j.

When the amount of the memory in buffered crossbar of a CICB is small, such that $k < RTT$, the throughput of the switch under large-rate flows may decrease because a crosspoint buffer may become empty before a new cell is received from the inputs [8], [9]. This underflow problem was demonstrated under these conditions [12] using the unbalanced traffic model [5], [6]. Buffer underflow was also observed in a buffered crossbar switch but, with internal variable-length segments [11].

To reduce the effect of buffer underflow, a method to improve the utilization of crosspoint buffers has been adopted in a switch with prioritized services [10]. In this study, a crosspoint buffer hosts logical queues, one for each service priority. The size of crosspoint buffers is set to $k \leq P + 1$, where P is the number of service priorities, so that $k > RTT$. Therefore, the

switch uses the prioritized buffers to either support long RTTs for high-priority flows or services with different priorities.

As buffered crossbars are often implemented on a single chip, the amount of memory that can be implemented on a single chip is limited by physical space [24], [25]. The interconnection technology that supports high-speed requires large on-chip real estate [26]. Increasing the port speed requires more on-chip real estate for interconnection, which limits the amount of space for on-chip memory implementation. A solution to reduce memory amount (to support long RTTs under best-effort traffic) was proposed using shared-memory crosspoint buffered (SMCB) switches using round-robin arbitration [12]. The shared-memory buffered switches were also considered to support multicast traffic [16]–[18]. Virtual crosspoint queues (VCQs) in ingress ports of a buffered crossbar were later proposed as another approach to reduce crosspoint buffer size [13]. The VCQs resemble VOQs; however, they are placed at the buffered crossbar. This work showed that the adoption of VCQs requires more memory in the crossbar to achieve similar performance to that of a shared-memory crosspoint switch [14].

Another approach focused on increasing the efficiency of the flow control mechanism, to decrease the dependency in memory [15]. Furthermore, the implementation of input arbiters placed at the buffered crossbar was then proposed [19]. Here, the latency of the information exchange between input and output arbiters is avoided as the arbiters are placed in the same chip. This approach increases the efficiency of the flow control mechanism but the minimum memory requirement remains at kLN^2.

These works showed that an SMCB switch is a buffered-crossbar switch with the smallest amount of memory in the buffered crossbar. Therefore, a question arises: What is the maximum throughput of SMCB switches under uniform traffic?

As an answer to this question, we present a theoretical throughput analysis of two SMCB switches whose crosspoint buffers are shared by m inputs and that use random selection as the arbitration scheme. In these switches, one of them uses dynamic partitioning of buffer space and speedup of m, and it is called the SMCBxm switch, and the other arbitrates inputs to access the shared crosspoint buffers to avoid memory speedup, and it is called the mSMCB switch. Random selection has been used to analyze the throughput of packet switches [27], [28] and here, we adopt it for a similar reason. The analysis proves that the achievable throughput of the two shared memory CICB switches is 100% under i.i.d. uniform traffic. Furthermore, the high throughput achieved by the mSMCB switch indicates that memory speedup is not required to achieve high switching performance.

The remainder of this paper is organized as follows. Section II describes the SMCBxm and mSMCB switches. Section III demonstrates that the throughput of the SMCB switches with random selection approaches 100% throughput under i.i.d. uniform traffic. Section IV presents the conclusions.

II. Shared-Memory Crosspoint Buffered (SMCB) Switches

To reduce the required memory amount in the buffered crossbar of an SMCB switch, a crosspoint buffer is shared by m inputs, where $2 \leq m \leq N$. This section introduces two SMCB switches. Previously, the SMCB switches were considered with round-robin for both input and outputs arbitrations, and the SMCB switches presented in this paper consider random selection for the input and output arbitrations.

A. Shared-Memory Crosspoint Buffered Switch with Memory Allocation and Speedup of m (SMCBxm)

The SMCBxm switch has N VOQs at each input, N^2 crosspoints, and $\frac{N^2}{m}$ crosspoint buffers in the buffered crossbar. A crosspoint in the buffered crossbar that connects input i to output j is denoted as $CP(i,j)$, as in the CICB switch. The buffer shared by $CP(i,j)$ and $CP(i',j)$, where $0 \leq i' \leq N-1$ and $i \neq i'$, that stores cells for output j is denoted as $SMB(q,j)$, where $0 \leq q \leq \lfloor \frac{N}{2} - 1 \rfloor$. The size of $SMB(q,j)$ is k_s cells. Each VOQ has a service counter to count the number of outstanding cells, and a counter limit, $C_{i,j}^{max}$, to indicate the maximum number of cells that $VOQ(i,j)$ can send to the corresponding SMB. A sharing control unit (SCU) at each SMB decides the size of the partition of an SMB for each sharing input. This memory partitioning is in function of the occupancy of $VOQ(i,j)$, $Z_{i,j}$. Table I shows the size of the shared-buffer partition, indicated by $C_{i,j}^{max}$, in number of cells, in function of $Z_{i,j}$.

Because m inputs might need to access the shared memory at the same time, this switch requires a speedup of m for the shared memory. A credit-based control mechanism, in combination with the dynamic memory allocation, is used to avoid buffer overflow. With the flow control mechanism, an input is kept from sending more cells to the SMB than the permitted allocation. To minimize the speedup of the shared memory in a practical implementation, the number of inputs sharing a crosspoint buffer is set to two (i.e., m=2). This description considers an even N for the sake of clarity. However, an odd N can also be adopted (with one dedicated crosspoint-buffer for the non-sharing input).

Figure 1 shows the SMCBxm switch with two inputs sharing the crosspoint buffers (i.e., SMCBx2). The SMCBx2 switch uses random selection for input and output arbitrations. The switch works as follows: When a cell arrives at $VOQ(i,j)$, a request is sent to the corresponding SCU. Based on $Z_{i,j}$, the SCU at every SMB sets up the size of the memory partition available for each input. The allocated amount of memory sets C^{max}, and the flow control keeps track of the occupancy of the SMBs and notifies the inputs. A grant is sent back from the SCU to the input to enable the forwarding of a cell, and the input dispatches a cell in the next time slot. The count of the service counter is increased by one when a cell from $VOQ(i,j)$ is sent to $SMB(q,j)$ and reduced by one when a cell of $VOQ(i,j)$ at $SMB(q,j)$ is dispatched to the output. When the count of the service counter reaches $C_{i,j}^{max}$, $VOQ(i,j)$ is inhibited from sending more cells to the SMB. After cells are stored at the

SMBs, the output arbiter selects a cell to be forwarded to the output. The selected cell is sent to the output in the next time slot.

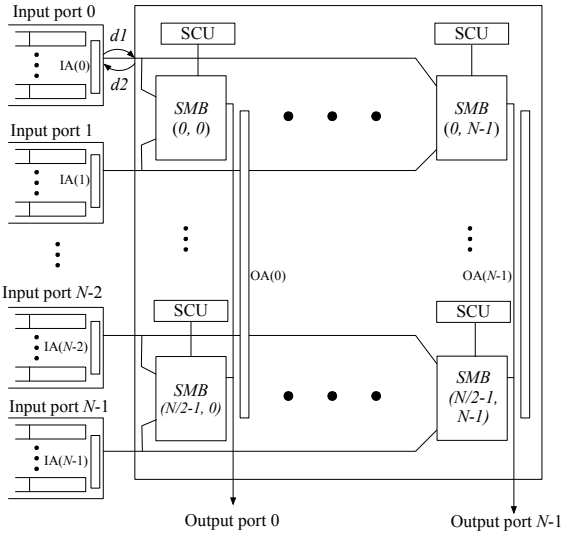


Fig. 1. An $N \times N$ SMCBx2 switch.

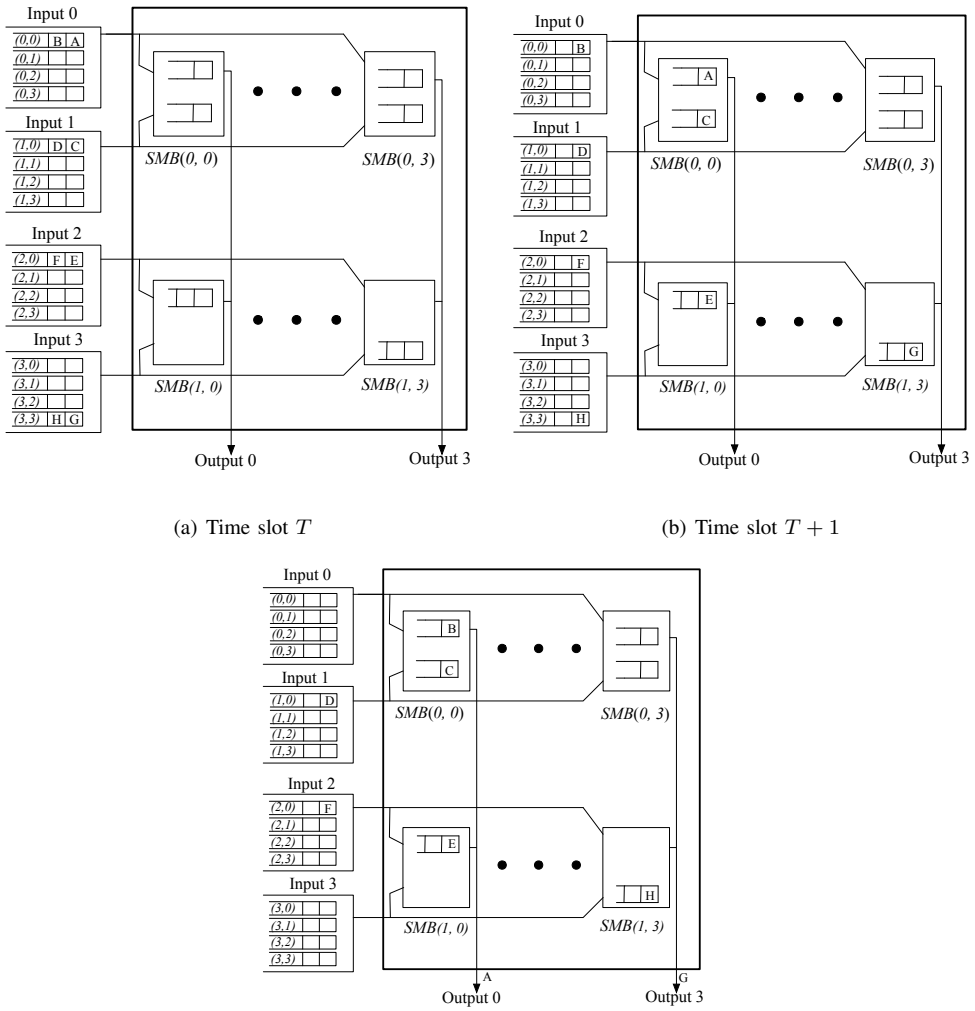
TABLE I

MEMORY ALLOCATION OF AN SMB IN THE SMCBX2 SWITCH.

$Z_{i,j}$	$Z_{i',j}$	$C_{i,j}^{max}$	$C_{i',j}^{max}$
0	0	$\lfloor k_s/2 \rfloor$	$\lfloor k_s/2 \rfloor$
$[0, RTT)$	$[0, k_s - RTT]$	$Z_{i,j}$	$k_s - Z_{i,j}$
$[RTT, \infty)$	0	k_s	0
$[0, RTT/2]$	$[0, RTT/2]$	$\lfloor k_s/2 \rfloor$	$\lfloor k_s/2 \rfloor$
$(RTT/2, \infty)$	$[0, RTT/2]$	$k_s - Z_{i',j}$	$Z_{i',j}$
$(RTT/2, \infty)$	$(RTT/2, \infty)$	$\lfloor k_s/2 \rfloor$	$\lfloor k_s/2 \rfloor$

Figure 2 shows an example of a 4×4 SMCBx2 switch with $k_s = 2$. To simply the explanation, we only show the VOQs and SMBs. Inputs 0 and 1 share $SMB(0, j), 0 \leq j \leq 3$. Inputs 2 and 3 share $SMB(1, j), 0 \leq j \leq 3$. At time slot T, $VOQ(0, 0)$ holds Cells A and B, $VOQ(1, 0)$ holds Cells C and D, $VOQ(2, 0)$ holds Cells E and F, and $VOQ(3, 3)$ holds Cells G and H, as shown in Figure 2(a). $VOQ(0, 0)$, and $VOQ(1, 0)$ share $SMB(0, 0)$. Since the occupancy of both VOQs is the same, $SMB(0, 0)$ allocates $k_s/2 = 1$ cell to each VOQ, and $C_{0,0}^{max} = C_{1,0}^{max} = 1$. Because $VOQ(2, 3)$ and $VOQ(3, 0)$ have no cells, $SMB(1, 0)$ and $SMB(1, 3)$ allocate their full capacity to $VOQ(2, 0)$ and $VOQ(3, 3)$, respectively, and $C_{1,0}^{max} = C_{1,3}^{max} = k_s = 2$ at this time slot.

At time slot $T + 1$, Cells A (from Input 0) and C (from Input 1) are forwarded to $SMB(0, 0)$ and Cells E and G are forwarded to $SMB(1, 0)$ and $SMB(1, 3)$, respectively, as shown in Figure 2(b). The output arbiters at Outputs 0 and 3 select



(a) Time slot T

(b) Time slot $T + 1$

(c) Time slot $T + 2$

Fig. 2. Example of a 4×4 SMCBx2 switch

a cell to be forwarded to the output in a random fashion. At time slot $T + 2$, Cells A and G are forwarded to Outputs 0 and 3, respectively, as shown in Figure 2(c). Flow control information is sent back to Inputs 0 and 3 to indicate the availability of the SMBs.

B. Shared-Memory Crosspoint Buffered Switch with Input-Crosspoint Matching (mSMCB)

In the mSMCB switch, only one input is allowed to access an SMB in a time slot; therefore, memory speedup is not required. To schedule the access to the SMB among m inputs, an input-access scheduler, denoted as S_q, is used to match m inputs to N SMBs. Figure 3 shows the architecture of the mSMCB switch for $m = 2$ (i.e., 2SMCB). The size of an SMB, in number of cells that can be stored, is also denoted as k_s. There are $\frac{N}{m}$ S_qs in the buffered crossbar. S_q matches non-empty inputs to SMBs that have room for storing at least one cell. The matching in S_q follows a three-phase process, as that used by some IQ switches [27], [29]. In this section, the matching scheme in S_q uses random selection [27]. A credit-based flow

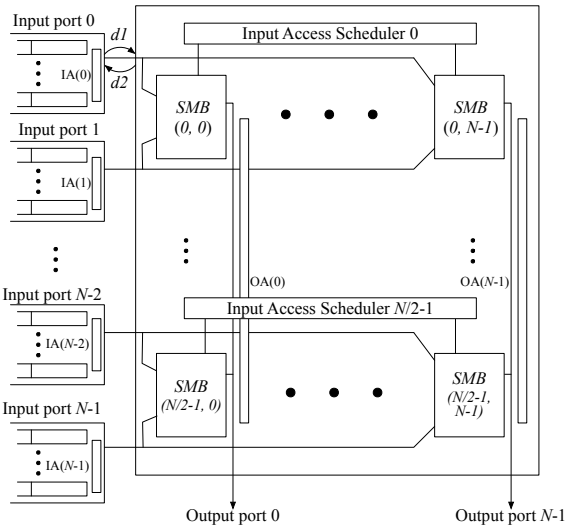


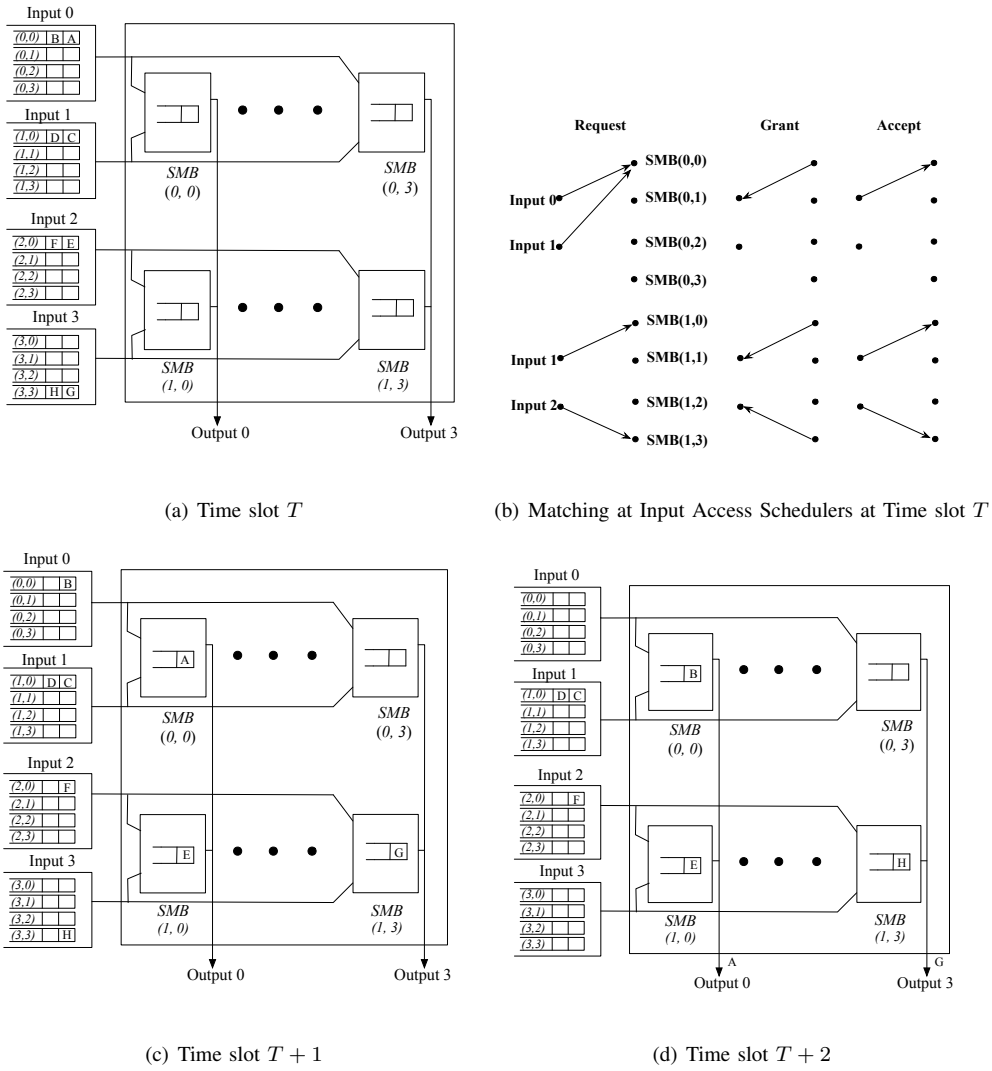
Fig. 3. An $N \times N$ 2SMCB switch.

control is used to monitor the available space in SMBs and to avoid buffer underflow. S_q determines non-empty VOQs and which corresponding SMBs have room available for at least one cell as eligible VOQs for matching. Allocation of memory is not used in this switch as the matching process regulates access to the SMBs because only one input is allowed to be matched to one SMB.

At each output in the buffered crossbar, there is an output arbiter to select a cell from non-empty SMBs. An output arbiter considers up to two cells from each SMB, where each cell belongs to a different input. The output arbiter uses random selection and is represented as a rectangle block in Figure 3.

The 2SMCB switch works as follows: Cells destined to output j arrive at $VOQ(i, j)$ and wait for dispatching. Input i notifies S_q about new cell arrivals. S_q selects the next cells to be forwarded to the crossbar by performing matching between inputs and SMBs. After a cell (or VOQ) is matched by S_q, the input is notified and sends the cell in the next time slot. A cell going from input i to output j enters the buffered crossbar and is stored in $SMB(q, j)$. Cells leave output j after being selected by the output arbiter.

Figure 4 shows an example of a 4×4 2SMCB switch with $k_s = 1$, where the occupancies of VOQs are the same as those in Figure 2. At time slot T, Input Access Schedulers perform matching between inputs and SMBs. As shown in Figure 4(b), S_0 has two requests for $SMB(0, 0)$ and grants Input 0 to access $SMB(0, 0)$. S_1 has a request for $SMB(1, 0)$ from Input 1 and one request for $SMB(1, 3)$ from Input 2. Since both SMBs are available, both inputs are granted access to the SMBs. At time slot $T + 1$, Cell A is forwarded to $SMB(0, 0)$, Cell E is forwarded to $SMB(1, 0)$, and Cell G is forwarded to $SMB(1, 3)$. The output arbiters at Outputs 0 and 3 select a cell, in a random fashion, to be forwarded to each corresponding output. Flow



(a) Time slot T

(b) Matching at Input Access Schedulers at Time slot T

(c) Time slot $T + 1$

(d) Time slot $T + 2$

Fig. 4. Example of a 4×4 2SMCB switch.

control information is sent back to Inputs 0 and 3 to indicate the availability of the SMBs. Cells A and G are forwarded to their destined outputs in time slot $T + 2$.

III. Throughput of the SMCB Switches with Random Selection

This section presents the throughput analysis of the mSMCB and SMCBxm switches that use random-based selection schemes under Bernoulli uniform traffic. The presented analysis is based on the following assumptions of the incoming traffic:

1 Arrivals at each input are i.i.d.

2 Arrival processes at each input are independent of previous arrivals and they are modeled as Bernoulli arrivals.

3 Cell destinations are uniformly distributed over all outputs.

It has been shown that the throughput of a buffered crossbar switch with dedicated crosspoint buffers increases to 100% asymptotically as $N \to \infty$ under Bernoulli i.i.d. traffic if the crossbar switch can buffer one cell at each crosspoint [28], [30].

This throughput is also referred to as the saturation throughput as $N \to \infty$.

In this paper, the presented analysis focuses on SMCB switches with the minimum number of inputs sharing an SMB, i.e., $m = 2$, and it shows that the 2SMCB and SMCBx2 switches achieve 100% throughput under uniform i.i.d. traffic with Bernoulli arrivals as $N \to \infty$. The results show that the memory speedup, as required by the SMCBxm, is not a strict requirement for a switch of moderate and large sizes (i.e., N=32 or larger). Furthermore, the analysis considers the case where m=N for the mSMCB switch to show the relationship between switching performance and the amount of memory required. The performance analysis is based on the probability that a VOQ receives service to identify the maximum throughput of the proposed switches. Regarding to the mSMCB switch, the analysis focuses on the effect that the matching process has on the switching performance.

In an SMCBx2 switch, SMBs are partitioned before the cells are forwarded from the input to the crosspoints, and therefore, a partition can be considered as a dedicated queue for input i if the ratio of the occupancies of the two VOQs sharing the SMB remains unchanged. To simplify the description, the remainder of this section refers to the partition of an SMB as a queue. Considering the characteristics of the two SMCB switches, k_s is one or more cells for the mSMCB switch, and two or more cells for the SMCBxm switch.

The probability that a queue is full is denoted as P_f, and the probability that a VOQ is blocked (from sending a cell to the corresponding crosspoint buffer) is denoted as P_b. The probability that a VOQ sends the HOL cell to the queue is denoted as P, where $P = 1 - P_b$.

A. mSMCB Switch with $k_s = 1$

The blocking probability of a VOQ in the mSMCB switch with $k_s = 1$, denoted as P_b, is defined by two possible cases: I) when the SMB is full with probability P_f. In this case, P_b is in function of the probability that a cell is forwarded to the corresponding output. The probability that there is a cell destined to this specific output is $\frac{1}{N}$. II) When a given input contends with t inputs (where $0 \leq t \leq m-1$) for access to an available SMB, and the input is not granted because another input is matched. The probability that an input receives no grant is $\frac{t}{t+1}$. Considering these two cases, the blocking probability for the mSMCB switch is stated as

$$P_b = \frac{1}{N} P_f + \sum_{t=0}^{m-1} \binom{m-1}{t} \left(\frac{1}{N}\right)^t \left(1 - \frac{1}{N}\right)^{m-1-t} \frac{t}{t+1}(1 - P_f). \tag{1}$$

Figure 5 shows a Markov chain describing the occupancy of $SMB(q, j)$ in an mSMCB switch. P_{Sy} represents the state probability, where $0 \leq y \leq k_s$. P_{uv} is the transition probability from state u to state v. The probability that the SMB is full

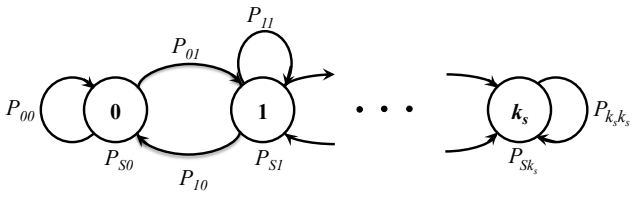


Fig. 5. Diagram of the Markov chain of a shared crosspoint buffer of size k_s.

P_f is equivalent to the state probability P_{Sk_s}. For any k_s, P_{01} is defined by the product of the probability of input arrival $\rho_{i,j}$ and the matching probability between the inputs and the SMBs,

$$P_{01} = \rho_{i,j} \sum_{t=0}^{m-1} \binom{m-1}{t} \left(\frac{1}{N}\right)^t \left(1 - \frac{1}{N}\right)^{m-1-t} \frac{1}{t+1} \tag{2}$$

The service probability $P_{service}$ is the probability that the output arbiter selects a non-empty $SMB(q,j)$ to forward a cell to the output. For $m = 2$, $P_{service} = \frac{2}{N}$ for any state. P_{10}, which occurs when input i has no requests and $SMB(q,j)$ is selected by the output arbiter, is defined by

$$P_{10} = (1 - \rho_{i,j}) P_{service}. \tag{3}$$

The service probability $P_{service}$ is the probability that the output arbiter selects $SMB(q,j)$ to forward a cell, or $\frac{1}{N}$. The following balance equations are obtained when $k_s = 1$:

$$\begin{cases} P_{01} P_{S0} = P_{10} P_{S1}; \\ P_{S0} + P_{S1} = 1; \end{cases} \tag{4}$$

The probability that the SMB is full is represented as

$$P_f = P_{S1} = \frac{P_{01}}{P_{01} + P_{10}}. \tag{5}$$

B. mSMCB Switch with $k_s = 2$

To compare the performance of the two SMCB switches with the same amount of memory, the size of SMBs is set to $k_s = 2$. The probability that an SMB is full follows (1). The following balance equations are obtained when $k_s = 2$:

$$\begin{cases} P_{01}P_{S0} = P_{10}P_{S1}; \\ P_{12}P_{S1} = P_{21}P_{S2}; \\ P_{S0} + P_{S1} + P_{S2} = 1; \end{cases}$$

and from these equations:

$$P_f = P_{S2} = \frac{P_{01}P_{12}}{P_{01}P_{12} + P_{01}P_{21} + P_{10}P_{21}}. \tag{6}$$

Here, the transition probabilities are defined as:

$$\begin{cases} P_{01} = P_{12} = \rho_{i,j} \sum_{t=0}^{m-1} \binom{m-1}{t} \left(\frac{1}{N}\right)^t \left(1 - \frac{1}{N}\right)^{m-1-t} \frac{1}{t+1}; \\ P_{10} = (1 - \rho_{i,j}) \frac{1}{N-1}; \\ P_{21} = (1 - \rho_{i,j}) \frac{2}{N}; \end{cases}$$

*C. SMCB*xm *Switch with* $k_s = 2$

Without losing generality, let us assume that all VOQs in the SMCBx2 switch have a backlog longer than RTT such that SMBs are partitioned into two equally sized parts, or $k_s/2$ each, one part for each VOQ that shares the SMB. In this case, the blocking probability of a VOQ is in function of the occupancy of the allocated portion of the memory, where the portion is either full or available. The blocking probability of a VOQ is obtained by considering whether the allocated portion of the SMB is full. The blocking probability of a VOQ to forward a cell to the corresponding SMB, p_b is represented as

$$P_b = \rho_{i,j} P_f. \tag{7}$$

The allocated SMB partition of one-cell size in the SMCBx2 is modeled as a queuing system, as shown in Figure 5, used for the mSMCB switch. The state transition probability p_{01} for the SMCBx2 switch is the product of the probability of input arrival $\rho_{i,j}$ and the probability that the input arbiter selects $VOQ(i,j)$ is $\frac{1}{N}$, as in (8).

$$P_{01} = \rho_{i,j} \frac{1}{N}. \tag{8}$$

The transition probability P_{10} is the probability that an SMB is selected by the output arbiter while there is no request from the input, or

$$P_{10} = (1 - \rho_{i,j}) \frac{1}{N}. \tag{9}$$

P_f is calculated as in (5):

$$P_f = P_{S1} = \frac{P_{01}}{P_{01} + P_{10}}. \tag{10}$$

D. Maximum Throughput of the 2SMCB and SMCBx2 with Random Selection

The throughput of the proposed switches is determined by considering the blocking probability of the VOQs. This section demonstrates that these two switches can achieve 100% throughput for large switch sizes. The analysis is performed through numerical evaluations of the blocking probability of VOQs for the SMCB switches. The following equations state the limits of P_b for the 2SMCB switch for $k_s = 1$ and $k_s = 2$ cells, respectively, and for the SMCBx2 switch when $k_s = 2$.

In the case of the 2SMCB switch with $k_s = 1$:

$$\lim_{N\to\infty} P_f = \lim_{N\to\infty} \frac{1 - \dfrac{1}{2N}}{3 - \dfrac{5}{2N}} = \frac{1}{3} \tag{11}$$

and

$$\lim_{N\to\infty} P_b = \lim_{N\to\infty} \left\{ \frac{1}{N} P_f + \frac{1}{2N}(1 - P_f) \right\} = 0. \tag{12}$$

For the case of the 2SMCB switch with $k_s = 2$:

$$\lim_{N\to\infty} P_f = \lim_{N\to\infty} \frac{1 - \dfrac{1}{N} + \dfrac{1}{4N^2}}{3 - \dfrac{4}{N} + \dfrac{5}{4N^2} + \dfrac{2N}{N-1} - \dfrac{4}{N-1} + \dfrac{2}{N^2 - N}} = \frac{1}{3} \tag{13}$$

then, the limit follows:

$$\lim_{N\to\infty} P_b = \lim_{N\to\infty} \left\{ \frac{1}{N} P_f + \frac{1}{2N}(1 - P_f) \right\} = 0. \tag{14}$$

For the SMCBx2 switch with $k_s = 2$, using (8), (9), and (10):

$$\lim_{N\to\infty} P_b = \lim_{N\to\infty} \frac{1}{N^2} = 0. \tag{15}$$

Because the minimum possible values of k_s are addressed, the nonblocking probability for these two switches for any k_s value approaches 1.0 as N grows. Therefore, both switches achieve 100% throughput for a large N. This high throughput with random-based selection schemes is achieved because of the expansion gain provided by the matching size, 2-to-N, in the 2SMCB switch, and because a speedup of two is used for the SMCBx2 switch.

Considering that the limit of the non-matching probability is

$$\lim_{N\to\infty}\sum_{t=0}^{N-1}\binom{N-1}{t}\left(\frac{1}{N}\right)^{t}\left(1-\frac{1}{N}\right)^{N-1-t}\frac{t}{t+1}=\lim_{N\to\infty}\left(1-\frac{1}{N}\right)^{N}=0.368 \tag{16}$$

and that the probability that an SMB is full converges to

$$\lim_{N\to\infty}P_f=\lim_{N\to\infty}\frac{\left(1-\frac{1}{N}\right)^{N}}{\left(1-\frac{1}{N}\right)^{N}+N-1}=0, \tag{17}$$

the blocking probability when all N inputs share the SMBs converges to

$$\lim_{N\to\infty}P_b=\lim_{N\to\infty}\left\{\frac{1}{N}P_f+\left(1-\frac{1}{N}\right)^{N}(1-P_f)\right\}=0.368 \tag{18}$$

In other words, the nonblocking probability converges to 0.632. Therefore, the throughput of this switch when $m = N$ is 63.2% (as in an $N \times N$ IQ switch with random selection [27], [31]). This shows that the performance of the mSMCB switch is also determined by the matching process. Therefore, the performance of the mSMCB switch increases as the number of inputs sharing the crosspoint buffers decreases, i.e., $m \leq N$, as a product of the m-to-N matching. While a larger m would save more memory at the buffered crossbar, the throughput would decrease.

E. Nonblocking Probability of Small Switch Sizes

We evaluated the nonblocking probabilities of switches, shown in Figure 6, using (5) and (10). We further compare the nonblocking probabilities of the 2SMCB and the SMCBx2 switches with a CICB switch with random selection for both input and output arbitration and an input-queued (IQ) switch with parallel iterative matching (PIM) [27]. The nonblocking probability of the CICB switch follows that of the SMCBx2 switch without memory speedup. We use k to represent the crosspoint buffer size of the CICB switch. When $k = k_s = 1$, the amount of memory in the SMCB switches is half of that in the CICB switch. Figure 6 shows the nonblocking probability of a VOQ in the 2SMCB switch for $k_s = 1$, in the SMCBx2 switch for $k_s = 2$, in the CICB switch for $k = 1$, and the IQ switch with one iteration, all under uniform traffic with i.i.d. Bernoulli arrivals. The results show that the SMCBx2 and CICB switches achieve slightly higher throughput than the 2SMCB switch when the switch size is small (e.g., $N < 16$). As the switch size increases, the throughput of both switches approaches 100%, and the throughput of the IQ switch with PIM approaches 63%.

The throughput of the switches with the same amount of memory in the SMCB switches and the CICB switch (i.e., $k_s = 2$ and $k = 1$) are compared. Figure 7 was generated using (5) for the SMCBx2 and CICB switches and (6) for the 2SMCB

switch. This figure shows that as the switch size increases, the throughput of both switches approaches 100%. The SMCBx2 switch has the same performance as that of the CICB switch, but at the cost of a memory speedup of 2.

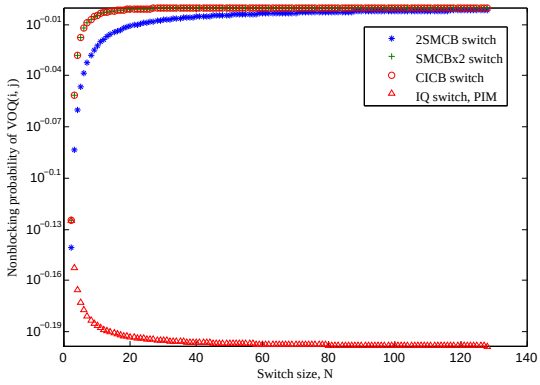


Fig. 6. Nonblocking probability under Bernoulli uniform traffic with $\rho = 1$, for 2SMCB ($k_s = 1$), SMCBx2 ($k_s = 2$), CICB ($k = 1$), and IQ (PIM).

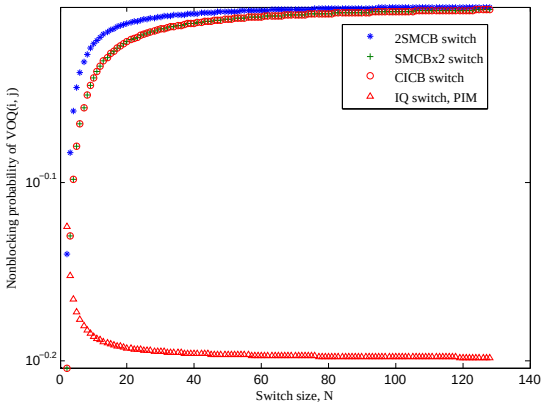


Fig. 7. Nonblocking probability under Bernoulli uniform traffic with $\rho = 1$, for 2SMCB ($k_s = 2$), SMCBx2 ($k_s = 2$), CICB ($k = 1$), and IQ (PIM).

F. Discussion of Nonblocking Probability under Hot Spot and Bursty Traffic

Internet traffic may not show uniformity. We discusse the nonblocking probability under hot spot and bursty traffic in this section. In this paper, the assumption of having cell destinations uniformly distributed over all outputs can be relaxed to allow hot spot traffic, where the traffic arriving in the inputs is directed to a single output. This is still governed by the M/M/1 model with a different arrival probability. Assuming the probability of arrival as $\frac{2}{N}$ for flow (i, j), $\rho_{i,j} = \frac{2}{N}$, the blocking probability of the $VOQ(i, j)$ for 2SMCB and SMCBx2 switches is calculated by replacing $\rho_{i,j}$ to $\frac{2}{N}$ in (2)-(10), in Sections III.A - III.C, where $\lim_{N\to\infty} P_f = 0$ still holds. The performance of the SMCB switches under unbalanced traffic depends on the number of inputs that are sharing the crosspoint buffers, m, and on the arbitration scheme used. It has been shown that the SMCB switches achieve a comparable performance to that of a CICB switch with weighted arbitration schemes and minimum

number of inputs sharing the crosspoint buffers, e.g., m=2 [12].

Bursty traffic is also of interest when analyzing switch performance. As observed by the CAIDA traffic analysis project, the probability that the average Internet packet size is smaller than 100 bytes is more than 50% [32], [33]. Packet segmentation may not be necessary when the size of a packet is small. Cell length can be determined by the expected packet length [32], [33]. An appropriate cell length reduces the level of packet segmentation and improves switching performance [11], [34]–[40]. The analysis of bursty traffic can be relaxed from fixed-size cells in the proposed analysis to variable-sized cells since cell arrival can still be modeled as Bernoulli arrivals for variable-length cells.

IV. Conclusions

This paper presents a theoretical throughput analysis of two shared-memory CICB switches, the SMCBxm and mSMCB switches, which use crosspoint buffers shared by m input ports and random-based arbitrations, under independent and identically distributed traffic with uniform distributions and Bernoulli arrivals. The SMCBxm switch uses memory speedup and dynamically partitions the shared memory among the sharing inputs. The mSMCB switch arbitrates the access to crosspoint buffers using an input access scheduler and avoids memory speedup.

In this paper, we have proved that these switches achieve 100% throughput under i.i.d. traffic with uniform distribution and Bernoulli arrivals. This result also indicates that speedup is not necessary by the mSMCB switch to achieve this high performance. Both of the analyzed SMCB switches relax the amount of memory to $\frac{1}{m}$ of that in the buffered crossbar of a CICB switch with dedicated crosspoint buffers.

References

[1] Nojima, S., Tsutsui, E., Fukuda, H., Hashimmoto, M.: 'Integrated Packet Network Using Bus Matrix,' *IEEE J. Select. Areas Commun.*, October 1987, 5, (8), pp. 1284-1291

[2] Doi, Y., Yamanaka, N.: 'A High-Speed ATM Switch with Input and Cross-Point Buffers,' *IEICE Trans. Commun.*, March 1993, 76, (3), pp. 310-314

[3] Nabeshima, M.: 'Performance Evaluation of a Combined Input- and Crosspoint-Queued Switch,' *IEICE Trans. Commun.*, E83-B, (3), March 2000, pp. 737-741

[4] Yoshigoe, K., Christensen, K.J.: 'A parallel-polled Virtual Output Queue with a Buffered Crossbar,' Proc. *IEEE HPSR 2001*, May 2001, pp. 271-275

[5] Rojas-Cessa, R., Oki, E., Jing, Z., Chao, H.J.: 'CIXB-1: Combined Input-One-Cell-Crosspoint Buffered Switch,' Proc. *IEEE HPSR 2001*, May 2001, pp. 324-329

[6] Rojas-Cessa, R., Oki, E., Chao, H.J.: 'CIXOB-1: Combined Input-crosspoint-output Buffered Packet Switch,' Proc. *IEEE GLOBECOM 2001*, 4, November 2001, pp. 2654-2660

[7] Javadi, T., Magill, R., Hrabik, T.: 'A High-Throughput Algorithm for Buffered Crossbar Switch Fabric,' *Proc. of IEEE ICC 2001*, June 2001, pp.1581-1591

[8] Abel, F., Minkenberg, C., Luijten, R.P., Gusat, M., Iliadis, I.: 'A Four-Terabit Single-Stage Packet Switch with Large Round-Trip Time Support,' *Proc. of High Performance Switching and Routing*, August 2002, pp. 5-14

[9] Abel, F., Minkenberg, C., Luijten, R.P., Gusat, M., Iliadis, I.: 'A Four-Terabit Packet Switch supporting Long Round-trip Times,' *IEEE Micro*, 23, (1), January-February 2003, pp. 10-24

[10] Luijten, R., Minkenberg, C., Gusat, M.: 'Reducing Memory Size in Buffered Crossbars with Large Internal Flow Control Latency,' *Proc. of IEEE Globecom 2003*, 7, December 2003, pp. 3683-3687

[11] Katevenis, M., Passas, G., Simos, D., Chrysos, N.: 'Variable Packet Size Buffered Crossbar (CICQ) Switches,' Proc. *IEEE ICC 2004*, 2, June 2004, pp. 1090-1096

[12] Dong, Z., Rojas-Cessa, R,: 'Long Round-Trip Time Support with Shared-Memory Crosspoint Buffered Packet Switch,' Proc. *IEEE 13th Annual Symposium on High Performance Interconnects*, August 2005, pp. 138-143

[13] Yoshigoe, K.: 'The CICQ Switch with Virtual Crosspoint Queues for Large RTT,' Proc. *IEEE ICC 2006*, June 2006, pp. 299-303

[14] Yoshigoe, K.: 'Threshold-based Exhaustive Round-Robin for the CICQ Switch with Virtual Crosspoint Queues,' Proc. *IEEE ICC 2007*, June 2007, pp. 6325-6329

[15] Gramsamer, F., Gusat, M., Luijten, R.: 'Optimizing Flow Control for Buffered Switches,' Proc. *IEEE ICCCN 2002*, October 2002, pp. 438-443

[16] Wang, W.F., Lee, F.C., Lu, G.L.: 'A Shared-Memory Design for Crosspoint Buffered Switches under Mixed Uni- and Multicast Traffic,' *IEEE 24th International Conference on Advanced Information Networking and Applications Workshops (WAINA)*, 2010, April 2010, pp.133-138

[17] Dong, Z., Rojas-Cessa, R.,: 'Input- and Output-Based Shared-Memory Crosspoint-Buffered Packet Switches for Multicast Traffic Switching and Replication,' Proc. *IEEE ICC*, Beijing, China, May 2008, pp. 1-6

[18] Yi, P., Li, H., Yu, J., Wang, B.,: 'Scheduling multicast and unicast traffic in buffered crossbar switches,' *IET Conf. Pub. CP525*, November 2006, pp. 1-4.

[19] Chrysos, N., Katevenis, M.: 'Crossbar with Minimally-Sized Crosspoint Buffers,' Proc. *IEEE HPSR 2007*, May 2007, pp. 1-7.

[20] Mhamdi, L., Hamdi, M.: 'MCBF: a high-performance scheduling algorithm for buffered crossbar switches,' *IEEE Commun. Letters*, September 2003, 7, (9), pp. 451-453

[21] Mhamdi, L., Hamdi, M.: 'Practical Scheduling Algorithms for High-Performance Packet Switches,' *Proc. of IEEE ICC*, May 2003, pp. 1659-1663

[22] Rojas-Cessa, R., Oki, E., Chao, H. J.: 'On the Combined Input-Crosspoint Buffered Packet Switch with Round-Robin Arbitration,' *IEEE Trans. on Commun.*, November 2005, 53, (11), pp. 1945-1951

[23] Hluchyj, M., Karol, M.: 'Queuing in High-performance Packet-switching,' *IEEE J. Select. Areas Commun.*, December 1998, 6, pp. 1587-1597

[24] Goldrian, G.A., Leppla, B., Schumacher, N.: 'Buffered crossbar switch,' U.S. Patent 7826434, November 2, 2010.

[25] Kornaros, G,: 'BCB: A Buffered CrossBar Switch Fabric Utilizing Shared Memory,' *9th EUROMICRO Conference on Digital System Design: Architectures, Methods and Tools, 2006. DSD 2006*, pp. 180-188.

[26] Xilinx Virtex-7 datasheet, Available at http://www.xilinx.com/products/silicon-devices/fpga/virtex-7/index.htm, accessed December 2011.

[27] Anderson, T.E., Owicki, S.S., Saxe, J.B., Tacker, C.P.: 'High-speed Switch Scheduling for Local Area Networks,' *ACM Trans. on Computer Systems*, November 1993, 11, (4), pp. 319-352

[28] Lin, M., McKeown, N.: 'The throughput of a buffered crossbar switch,' *IEEE Commun. Letters*, May 2005, 9, (5), pp. 465-467

[29] McKeown, N.: 'The *i*SLIP Scheduling Algorithm for Input-Queue Switches,' *IEEE/ACM Trans. Networking*, April 1999, 7, (2), pp. 188-200

[30] Sun, S., He, S., Gao, W.: 'Throughput analysis of a buffered crossbar switch with multiple input queues under burst traffic,' *IEEE Communications Letters*, April 2006, 10, (4), pp. 305-307

[31] Nong, G., Muppala, J.K., Hamdi, M.: 'Analysis of Nonblocking ATM Switches with Multiple Input Queues,' *IEEE/ACM Trans. on Networking*, February 1999, 7, (1), pp. 60-74

[32] CAIDA,: 'Packet size distribution comparison between Internet links in 1998-2008,' Available at {http//www.caida.org/research/traffic-analysis/pkt_size_distribution/graphs.xml, accessed December 2011

[33] Sinha, R., Papadopoulos, C. and Heidemann, J.: 'Internet Packet Size Distributions: Some Observations.' Technical Report ISI-TR-2007-643, USC/Information Sciences Institute, May, 2007. Available at http://www.isi.edu/~johnh/PAPERS/Sinha07a/index.html, accessed December 2011

[34] Al-saber, N., Oberoi, S., Rojas-Cessa, R and Ziavras, S.G.: 'Concatenating Packets for Variable-Length Input-Queue Packet Switches with Cell-Based and Packet-Based Scheduling,' *Proc. IEEE Sarnoff Symposium*, Princeton, NJ, April 2008, pp. 1-5

[35] Cai, L., Rojas-Cessa, R.; Kijkanjanarat, T.: 'Avoiding Speedup from Bandwidth Overhead in a Practical Output-Queued Packet Switch,' *Communications (ICC), 2011 IEEE International Conference on*, Japan 2011, pp. 1-5.

[36] Naghshineh, M., and Cuerin, R.,: 'Fixed versus Variable Packet Sizes in Fast Packet-Switched Networks,' *Proc. IEEE Infocom 1993*, (1), 1993, pp. 217-226.

[37] Marsan, M.A., Bianco, F., Giaccone, P., Leonardi, E., Neri, F.,: 'Packet Scheduling in Input-Queued Cell-Based Switches,' *IEEE Infocom 2001*, vol. 2, April 2001, pp. 1085-1094.

[38] Hu, C., Chen, X., Li, W., Liu, B.,: 'Fixed-Length Switching vs. Variable- Length Switching in Input-Queued IP Switches,' *Proc. IEEE Workshop on IP Operations and Management*, October 2004, pp. 117-122

[39] Bianco, A., Franceschinis, M., Ghisolfi, S., Hill, A.M., Leonardi, E., Neri, F., Webb, R.: 'Frame-based Matching Algorithms for Input-queued Switches,' Proc. *IEEE HPSR 2002*, November 2002, pp. 69-76

[40] Rojas-Cessa, R., Oki, E.: 'Round-Robin Selection with Adaptable-Size Frame in a Combined Input-Crosspoint Buffered Switch,' *IEEE Commun. Letters*, November 2003, 7, (11), pp. 555-557