Input- and Output-Based Shared-Memory Crosspoint-Buffered Packet Switches for Multicast Traffic Switching and Replication

Ziqian Dong and Roberto Rojas-Cessa

Abstract—**The incorporation of broadcast and multimedia-on-demand services are expected to increase multicast traffic in packet networks, and therefore in switches and routers. Combined input-crosspoint buffered (CICB) switches can provide high performance under uniform multicast traffic, however, at the expense of N^2 crosspoint buffers. In this paper, we introduce an output-based shared-memory crosspoint-buffered (O-SMCB) packet switch where the crosspoint buffers are shared by two outputs and use no speedup. The proposed switch provides high performance under admissible uniform and nonuniform multicast traffic models while using 50% of the memory used in CICB switches. Furthermore, the O-SMCB switch provides higher throughput than an SMCB switch with buffers shared by inputs, or I-SMCB, previously proposed, despite the strong similarities between the architectures of these two switches. In this paper, we study the performance of the O-SMCB switch under uniform and nonuniform multicast traffic models and compare it to the I-SMCB switch.**

Index Terms—**Multicast, buffered crosspoint, buffered crossbar, shared memory, packet switch.**

I. Introduction

The migration of broadcasting and multicasting services, such as cable TV and multimedia-on-demand to packet-oriented networks is expected to take place in a dominant role in the near future. These highly popular applications have the potential of loading up the next generation Internet. To keep up with the bandwidth demand of such applications, the next generation of packet switches and routers need to provide efficient multicast switching and packet replication.

A lot of research has focused on unicast traffic, where each packet has a single destination. It has been shown that unicast switches achieve 100% throughput under admissible conditions, $\sum_i \lambda_{i,j} < 1$ and $\sum_j \lambda_{i,j} < 1$, where i is the index of inputs $(0 \leq i \leq N-1)$, j is the index of outputs $(0 \leq i \leq N-1)$ for an $N \times N$ port switch, and $\lambda_{i,j}$ is the data rate from input i to output j, in a plethora of switch architectures and switch configuration schemes.

Although it is difficult to describe actual multicast traffic models, switches also need to provide 100% throughput under admissible multicast traffic. In multicast switches, the admissibility conditions are similar to those for unicast traffic, however, with the consideration of the fanout of multicast packets. The fanout of a multicast packet is the number of different destinations that expect copies of the packet. This implies that the average fanout of multicast traffic increases the average output load of a switch. Therefore, the average output load in a multicast switch is proportional to the product of the average input load and the average fanout for a given multicast traffic model.

Here, we consider having incoming variable-size packets being segmented into fixed-length packets, also called cells, at the ingress side of a switch and being re-assembled at the egress side, before the packets leave the switch. Therefore, the time to transmit a cell from an input to an output takes a fixed amount of time, or time slot.

Multicast switching has been largely considered for input buffered (IB) switches. In these switches, matching has to be performed between inputs and outputs to define the configuration on a time-slot basis. This matching process can be complex when considering multicast traffic.

In IB switches, the replication of multicast packets can be handled in two main different ways: 1) at the input unicast buffers (in switches with buffers that store packets, one per output, or virtual output queues, VOQs), and 2) by performing the replication at the switch fabric. To resolve which multicast packets are transmitted per output (and therefore, per input) in IB switches, matching between input and output ports is performed. In the matching process, the selection of input-output pairs is based on assigned weights or heuristically. Switches that replicate multicast packets at the inputs require large input buffers for heavy traffic loads with large fan outs. Matching schemes for IB switches with VOQs have shown to provide 100% throughput under unicast admissible traffic using a maximum weight matching (MWM) scheme with, however, prohibitively high processing complexity that causes large configuration delay under high data rates or large capacity switches. The complexity of matching schemes for multicast packets is higher than that of unicast packets as the contention degree might be higher. Furthermore, since the retention of multicast packets at the inputs can easily overload the queues, multicast matching schemes have to be highly efficient and with low complexity. Alternatively, 100% throughput under unicast traffic can be achieved with lower complexity schemes with, however, speedup (memory and switch fabric running faster than the line speed). Considering the lag of memory speed in current technologies, discussion on switches with speedup is out of the scope of this paper.

In switches with cell replication at the switch fabric, multicast cells can be stored in a single queue at the input, separated from unicast cells. Here, cell replication is performed by using the replication capabilities of the switch fabric [1]. Although with reduced memory amount at the input ports, this approach suffers from severe head-of-line (HOL) blocking as a number of ports need to be available for receiving a cell at a time slot.

This work is supported in part by National Science Foundation under Grant Award 0435250.

The authors are with the Department of Electrical and Computer Engineering,New Jersey Institute of Technology, Newark, NJ 07102. Emails: {zd2, rojas}@njit.edu.

Following the VOQ strategy for unicast traffic, virtual multicast queues (VMQs) can be used to avoid HOL blocking, where there is a queue for each different fan out combination, or up to $2^N - 1$ VMQs in an input. This large number of VMQs makes it an impractical approach, even for small N. As a practical alternative, it has been of interest to find the smallest number of multicast queues k (where each queue can store multicast cells with a set of fan out combinations), where $2 \leq k \leq 2^N - 1$, and optimal queuing policies to reduce HOL blocking [2]. This approach, although it reduces the number of queues, has the drawback of being difficult to find a suitable value of k for different traffic patterns. It also depends on the queuing policies and on the multicast traffic model used.

The switching performance is also a function of the call splitting policy used. No call splitting, or one-shot policy, is the forwarding of multicast cells to their destination ports in a single time slot. If one of the multicast copies cannot be forwarded because of output contention, then the cell remains at the buffer and no cell replication is performed. No call splitting requires VMQs to avoid HOL blocking. On the other hand, call splitting allows forwarding of some or all of the multicast copies to different output ports in one time slot. Therefore, call splitting mitigates HOL blocking (although it doesn't completely remove it), and at the same time, it relaxes the requirement of VMQs. Call splitting is then widely considered in multicast switches. In this paper, we consider that cell replication is performed at the switch fabric by exploiting its space capabilities [1].

Combined input crosspoint-buffered (CICB) packet switches have shown higher performance than IB switches at the expense of having crosspoint buffers, which run at the same speed as that of input buffers in an IB switch, under unicast traffic. In these switches, an input has up to N buffers where each one stores cells destined to a particular output to avoid head-of-line blocking [3]. The crosspoint buffers in CICB switches can be used to provide call splitting (or fanout splitting) intrinsically. Different from IB switches, CICB switches are not required to have cell transmission after inputs and output have been matched [4]-[7]. This feature makes CICB switches attractive for implementation. In CICB switches, one input can send up to one (multicast) cell to the crossbar, and one or more cells destined to a single output port can be forwarded from multiple inputs to the crossbar at the same time slot [8], [9]. Therefore, CICB switches have natural properties favorable for multicast switching as contending copies for a single output can be sent to the crosspoint buffers from several inputs at the same time slot without blocking each other. In general, CICB switches have dedicated crosspoint buffers for each input-output pair, for a total of N^2 crosspoint buffers. Since memory used in the crosspoint buffers has to be fast, it is desirable to minimize the amount of it as fast memory is expensive.

In response to this need, we propose an output-based shared-memory crosspoint-buffered (O-SMCB) packet switch. This switch requires less memory than a CICB switch to achieve comparable performance under multicast traffic and no speedup. Furthermore, the O-SMCB switch provides higher throughput under uniform and nonuniform multicast traffic models than our previously proposed input-based SMCB (I-SMCB) switch, where two inputs share the crosspoint buffers [10].

We show the performance improvement of the O-SMCB switch over the I-SMCB switch, both using round-robin selections for input and output arbitrations, under multicast traffic. We adopt this selection scheme for its simplicity and as an example. Other selection schemes (e.g., weight-based selection) can also be used.

The remainder of this paper is organized as follows. Section II introduces the CICB switch for multicast traffic. Section III introduces the I-SMCB switch model. Section IV describes the proposed O-SMCB switch model. Section V presents the throughput evaluation of both switches under multicast traffic with uniform and nonuniform distributions. Section VI summarizes our conclusions.

II. Combined Input-Crosspoint Buffer Switch

In an $N \times N$ CICB switch, the buffered crossbar has N inputs and N outputs. There is one multicast first-in first-out (FIFO) queue at each input. A multicast cell uses a bitmap of size N to represent the destinations of multicast-cell copies. If the bitmap has a value of 1 at the j^{th} position, output j is one of the destinations of the multicast cell. Otherwise, the bit value is indicated by 0. A crosspoint (CP) element in the buffered crossbar that connects input port i to output port j is denoted as $CP(i, j)$. The crosspoint buffer of $CP(i, j)$ is denoted as $CPB(i, j)$. The size of $CPB(i, j)$ is k cells, where $k \geq 1$. Figure 1 shows a CICB switch with dedicated crosspoint buffers.

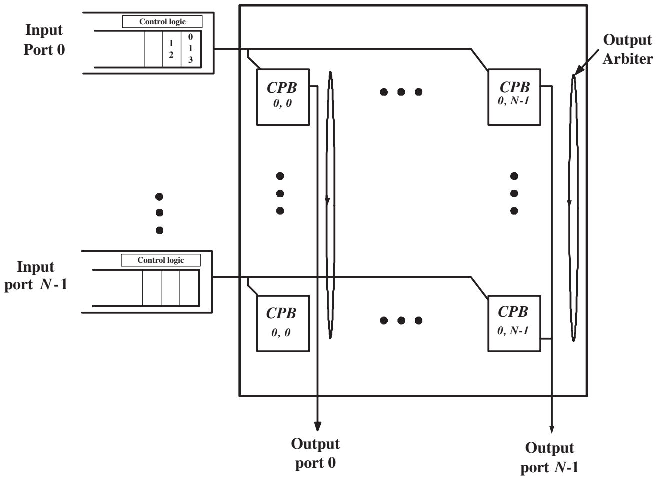

Fig. 1. CICB switch with dedicated CPBs.

To avoid cell loss within the switch, a credit-based flow control mechanism is used. This mechanism indicates input i whether $CPB(i, j)$ has room available for a cell or not. To implement this mechanism, each input has a credit counter for each $CPB(i, j)$ where the maximum count is the number of cells that a CPB can hold. When the number of cells sent by the FIFO to $CPB(i, j)$ reaches the maximum count k, the FIFO stops sending cells to that CPB to avoid buffer overflow. The count for $CPB(i, j)$ is increased by one each time a cell is sent by input i to this CPB and decreased by one each time that $CPB(i, j)$ forwards a cell to output j. A CPB that has available room for at least one cell is considered eligible to receive a cell.

This switch uses call splitting to forward copies of a multicast cell to the buffered crossbar. This is, as long as a CPB is eligible to receive a copy of the multicast cell and there is a multicast cell for this buffer at the HOL multicast cells of the input FIFO, the multicast cell is sent to the buffered crossbar. The crossbar performs multicast cell replication. Each time the HOL multicast cell is dispatched to destination j, the j^{th} bit of the multicast bitmap is reset. When the bitmap of the HOL cell is zero, the multicast cell is considered served.

III. Input-Based Shared-Memory Crosspoint Buffered (I-SMCB) Switch

The input-based shared-memory crosspoint buffered (I-SMCB) switch also has one multicast FIFO at each input, N^2 crosspoints and $\frac{N^2}{m}$ crosspoint buffers in the buffered crossbar. Each crosspoint buffer is shared by m inputs. Since previous results have shown that $m = 2$ delivers the optimum performance [10], we consider that the crosspoint buffers are shared by two inputs in the remainder of this paper. In this switch model, we denote each shared crosspoint buffer as SMB to differentiate from the notation in the CICB switch. The flow control mechanism used in the I-SMCB switch is similar as that used in the CICB switch.

A crosspoint in the buffered crossbar that connects input port i to output j is also denoted as $CP(i, j)$. The buffer for $CP(i, j)$ and $CP(i', j)$, where $0 \leq i' \leq N - 1$ and $i \neq i'$, that stores cells for output port j and is shared by these two crosspoints (or inputs i and i') is denoted as $SMB(q, j)$, where $0 \leq q \leq \frac{N}{2} - 1$. We assume an even N for the sake of clarity. However, an odd N can be used with one input port using dedicated buffers of size 0.5 to 1.0 the capacity of an SMB.

To eliminate the speedup at SMBs, only one input is allowed to access an SMB at a time. To schedule the access to the N SMBs from the two inputs, an input-access scheduler is used. Figure 2 shows the architecture of the mSMCB switch when $m = 2$. The size of an SMB, in the number of cells that can be stored, is k_s. There are $\frac{N}{m}$ input-access schedulers in the buffered crossbar. An input-access scheduler selects which non-empty inputs access the SMBs that have room for storing a cell.

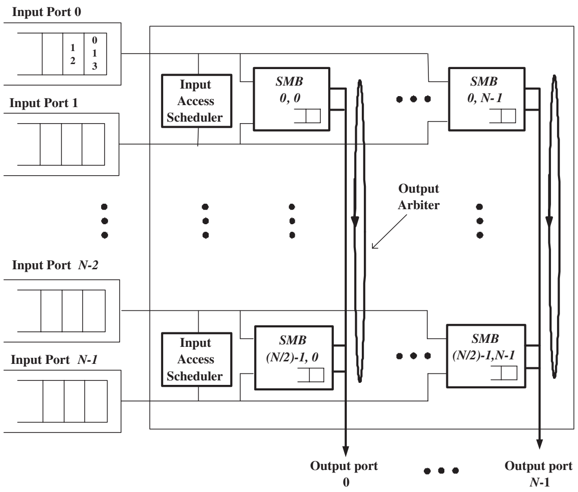

Fig. 2. $N \times N$ I-SMCB switch with shared-memory crosspoints by inputs.

Multicast cells at the inputs are handled in similar way as in the CICB switch, also using a multicast bitmap to indicate the destination of the multicast cells.

IV. Output-based Shared-Memory Crosspoint Buffered (O-SMCB) Switch

To observe the response of the proposed switch under multicast traffic only, the O-SMCB switch is also provisioned with one multicast first-in first-out (FIFO) queue at each input. This switch has N^2 crosspoints and $\frac{N^2}{2}$ crosspoint buffers in the crossbar as in the I-SMCB switch. Figure 3 shows the architecture of the O-SMCB switch. A crosspoint in the buffered crossbar that connects input port i to output j is also denoted as $CP(i, j)$. Similarly to the I-SMCB switch, the buffer shared by $CP(i, j)$ and $CP(i, j')$ that stores cells for output ports j or j', where $j \neq j'$, in the O-SMCB switch is denoted as $SMB(i, q)$, where $0 \leq q \leq \frac{N}{2} - 1$. The size of an SMB, in number of cells that can be stored, is also k_s. In this paper, we study the case of minimum amount of memory, or when $k_s = 1$ (equivalent to having 50% of the memory in the crossbar of a CICB switch). Therefore, $SMB(i, q)$ with $k_s = 1$ can store a cell that can be directed to either j or j'. The SMB has two egress lines, one per output.

Similarly to the I-SMCB switch, only one output is allowed to access an SMB at a time to avoid the need for speedup at SMBs in the O-SMCB switch. The access to one of the N SMBs by each output is decided by an output-access scheduler. A scheduler performs a match between SMBs and the outputs that share them by using round-robin selection. There are $\frac{N}{2}$ output-access schedulers in the buffered crossbar, one for each pair of outputs. Multicast cells at the inputs have an N-bit multicast bitmap to indicate the destination of the multicast cells. Each bit of the bitmap is denoted as D_j, where $D_j = 1$ if output j is one of the cell destinations, otherwise $D_j = 0$. Each time a multicast copy is forwarded to the SMB for the cell's destination, the corresponding bit in the bitmap is reset. When all bits of a multicast bitmap are zero, the multicast cell is considered completely served. Call splitting is used by this switch to allow effective replication and to alleviate a possible head-of-line blocking.

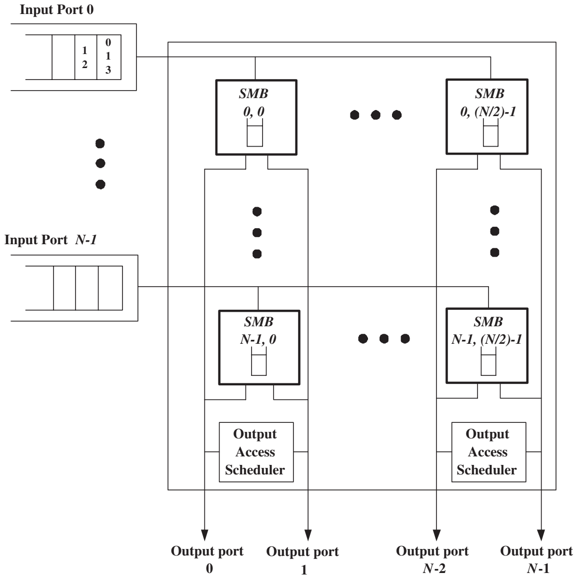

Fig. 3. $N \times N$ O-SMCB switch with shared-memory crosspoints by outputs.

A flow control mechanism is used to notify the inputs about which output replicates a multicast copy and to avoid buffer overflow. The flow control allows the inputs to send a copy of the multicast cell to the crossbar if there is at least one outstanding copy and an available SMB for the destined output. After all copies of the head-of-line multicast cell have been replicated, the input considers that cell served and starts the process with the cell behind.

V. Performance Evaluation

We compare the performance of the proposed O-SMCB switch to those of the CICB and I-SMCB switches. Models of 16×16 O-SMCB, I-SMCB, and CICB switches were implemented in discrete-event simulation programs. Similarly to the O-SMCB, the SMBs are shared in the I-SMCB switch, however, by (two) inputs. For a fair comparison, the I-SMCB also uses round-robin selections for SMB-access by inputs and for output arbitration. The CICB switch uses round-robin for input and output arbitrations. We study the maximum achievable throughput for each switch. The switches are simulated for 500,000 time slots for throughput measurement.

We consider multicast traffic models with uniform and nonuniform distributions and Bernoulli arrivals: multicast uniform, multicast diagonal with fanouts of 2 and 4 (called diagonal2 and diagonal4, respectively), and broadcast. In the uniform multicast traffic model, multicast cells are generated with a uniformly distributed fanout among N outputs. For this traffic model, the average fanout is $\frac{1+N}{2} = \frac{17}{2}$ and a maximum admissible input load of $\frac{1}{fanout}$=$\frac{1}{8.5}$. This traffic model includes a fanout=1 or unicast traffic. The multicast diagonal2 traffic model has a destination distribution to $j = i$ and $j = (i + 1)\%N$ for each multicast cell, and a maximum admissible input load of 0.5. The multicast diagonal4 traffic model has the copies of multicast cells destined to $j = \{i, (i+1)\%N$, $(i + 2)\%N$, and $(i + 3)\%N\}$ for each multicast cell, and its admissible input load is 0.25 (i.e., output load=1.0). In general, multicast diagonal$m$ traffic has copies of multicast cells destined to $j = (i + \tau)\%N$, where $\tau = 0, 1, ..., m$. Here m is the fanout value for multicast diagonal traffic. The admissible input load is $\frac{1}{m}$. A broadcast multicast has copies for all N different outputs and a maximum admissible load of $1/16 = 0.0625$.

Under admissible multicast uniform traffic, all switches deliver 100% throughput. These results are observed under both Bernoulli and Bursty arrival. Under admissible multicast diagonal2 traffic, the throughputs observed are 100% for the O-SMCB and CICB switches, and 96% for the I-SMCB switch. Under admissible multicast diagonal4 traffic, the performance of the I-SMCB switch decreases to 67%, while the throughputs of the O-SMCB and CICB switches remain close to 100%. Under broadcast traffic (fanout equal to N), the throughput of the O-SMCB switch is 99% and the throughput of the CICB switch is close to 100%, while the throughput of the I-SMCB switch is 95%. The simulation results under diagonal multicast traffic are shown in Figure 4.

Figures 5 and 6 show the throughput of 16× 16 and 32 × 32 CICB, I-SMCB and O-SMCB switches, respectively, under diagonal multicast traffic with different fanouts with maximum admissible input load. Here, the I-SMCB and O-SMCB switches have half the amount of memory in the buffered crossbar of that in a CICB switch. We can see that the throughput of O-SMCB and CICB switches remains close to 100% with different fanout values. The reason why the I-SMCB switch delivers worse performance than the others when fanout is 2 is because the diagonal multicast-traffic model generates multicast copies from the sharing inputs to the same outputs. In another words, the sharing inputs always compete for the same SMBs. Therefore, the throughput is up to 50% when the fanout is 2. On the other hand, the throughputs of the O-SMCB and CICB switches are close to 100% under all fanout values. However, the O-SMCB switch achieves this high throughput with half the amount of memory in the buffered crossbar of that in a CICB switch. This trend is observed under both switch sizes.

A. Throughput Degradation under Overload Conditions

Multicast traffic is difficult to police for admissibility. Furthermore, the performance of switches under inadmissible traffic (produced by larger fanouts than the expected average) may change. In cases of unicast traffic, the maximum throughput of a switch can remain high with a fair scheduler. However, this might not be the case under multicast traffic. In this experiment, we increased the input load beyond the maximum admissible values in the considered traffic models to observe throughput changes of the O-SMCB, I-SMCB, and CICB switches under these overload conditions. Here, we measured the throughput of the switches as a ratio between the maximum measured throughput and the maximum throughput that a switch is able to provide when all outputs are able to forward a cell.

Under uniform multicast traffic, the throughputs of O-SMCB and I-SMCB switches degrade to 93% when the input load is larger than 0.117 (i.e., output load is larger than 1.0), while the throughput of the CICB switch is 100%. This throughput degradation occurs in the SMCB switches because of the increased number of contentions for SMB access as the load increases. Under multicast diagonal2 traffic, the throughputs of the I-SMCB and CICB switch drop to 96% and 93%, respectively, while the throughput of the O-SMCB switch remains close to 100%. Under multicast diagonal4 traffic, the throughput of the I-SMCB switch drops to 68%, while the throughputs of the O-SMCB and CICB switches remain close to 100%. Under broadcast traffic, the throughput of the I-SMCB switch decreases to 79%. However, the throughputs of the O-SMCB and CICB switches remain close to 100%.

TABLE I
Throughput under multicast traffic.

Traffic type	$Ta(I)$	$Ta(O)$	$Ta(C)$	$Ti(I)$	$Ti(O)$	$Ti(C)$
Uniform	100%	100%	100%	93%	93%	100%
Diagonal2	96%	100%	100%	96%	100%	93%
Diagonal4	67%	100%	100%	68%	100%	100%
Broadcast	95%	99%	100%	79%	100%	100%

Table I summarizes the obtained throughput for all tested traffic models. In this table, Ta stands for the measured throughput under admissible traffic and Ti for the measured throughput under inadmissible traffic. The letters I, O, and C in parenthesis indicate that a result is related to the I-SMCB, O-SMCB, and CICB switches, respectively. As seen in this table, the performance of the O-SMCB switch is comparable to that of the CICB switch and higher than that of an I-SMCB switch. Therefore, the O-SMCB switch provides comparable

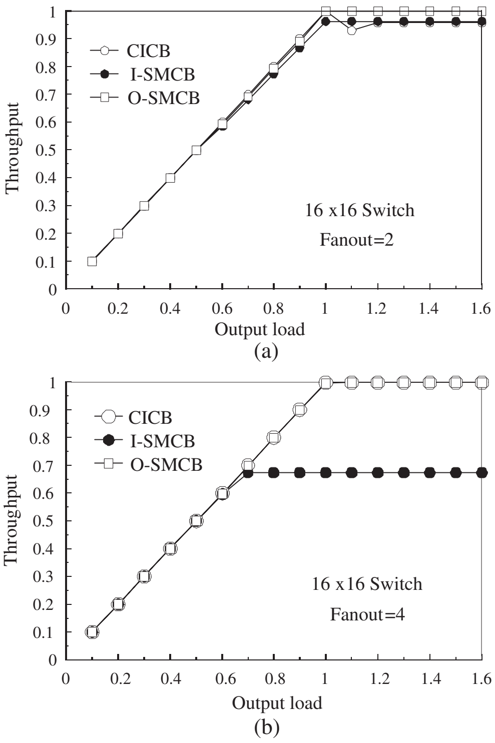

Fig. 4. Throughput performance of 16×16 I-SMCB and O-SMCB switches under a) diagonal2 traffic and b) diagonal4 traffic.

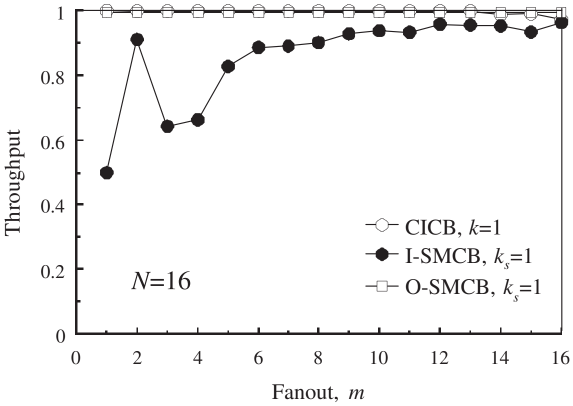

Fig. 5. Throughput performance of 16×16 I-SMCB and O-SMCB switches under diagonal multicast traffic with different fanout values.

performance but with 50% the memory amount of a CICB switch.

VI. Conclusions

Here, we proposed a novel switch architecture to support multicast traffic using a shared-memory switch that shares crosspoint buffers among outputs to use 50% of the memory amount in the crossbar fabric that CICB switches require. Our proposed switch, the O-SMCB switch, delivers high performance under multicast traffic while using no speedup. Furthermore, the proposed switch shows an improved performance over our previously proposed I-SMCB switch with shared memory among inputs. The improved switch has the buffers shared by the outputs, instead. This has the effect of facilitating call splitting by allowing inputs directly access the crosspoint buffers. This simple improvement has a significant impact on switching performance. As a result, the O-SMCB provides 100% throughput under both admissible uniform and diagonal multicast traffic with fanouts of 2 and 4. Furthermore, our proposed switch keeps the throughput high under nonuniform traffic with overloading conditions. The disadvantage of SMCB switches is that time relaxation of CICB switches is minimized because of the matching process used in buffer access. However, the matching is performed in chip and among a moderate number of outputs. Furthermore, the matching process in the SMCB switches is simpler than those used in IB switches for multicast traffic. The O-SMCB switch, with buffer space for $\frac{N^2}{2}$ cells, provides comparable performance to that of a CICB switch, with buffer space for N^2 cells, therefore, saving 50% of the amount of memory.

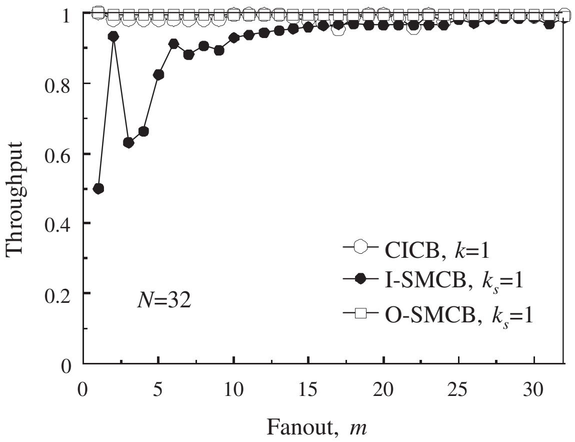

Fig. 6. Throughput performance of 32×32 I-SMCB and O-SMCB switches under diagonal multicast traffic with different fanout values.

References

[1] T.T. Lee, “Non-blocking copy networks for multicast packet switching,” *IEEE J. Select. Areas Commun.*, vol. 6, pp. 1455-1467, December 1988.

[2] A. Bianco, P. Giaccone, C. Piglione, S. Sessa, “Practical Algorithms for Multicast Support in Input Queued Switches,” Proc. *IEEE HPSR 2006*, 6 pages, May 2006.

[3] M. Karol, M. Hluchyj, “Queuing in High-performance Packet-switching,” *IEEE J. Select. Areas Commun.*, Vol. 6, pp. 1587-1597, December 1988.

[4] Y. Doi and N. Yamanaka, “A High-Speed ATM Switch with Input and Cross-Point Buffers,” *IEICE Trans. Commun.*, Vol. E76, No.3, pp. 310-314, March 1993.

[5] M. Nabeshima, “Performance Evaluation of a Combined Input- and Crosspoint-Queued Switch,” *IEICE Trans. Commun.*, Vol. E83-B, No. 3, pp. 737-741, March 2000.

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

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

[8] M. Hamdi and M. Lhamdi, “Scheduling multicast traffic in internally buffered crossbar switches,” *Proc. IEEE ICC 2004*, Vol. 2, pp. 1103-1107, June 2004.

[9] S. Sun, S. He, Y. Zheng, and W. Gao, “Multicast scheduling in buffered crossbar switches with multiple input queues,” *Proc. IEEE HPSR 2005*, 12-14, pp. 73-77, May 2005.

[10] Z. Dong and R. Rojas-Cessa, “Long Round-Trip Time Support with Shared-Memory Crosspoint Buffered Packet Switch,” *Proc. IEEE Hot Interconnects 2005*, pp. 138-143, August 2005.

[11] A. Bianco, P. Giaccone, M. Giraudo, F. Neri, E. Schiattarella, “Multicast Support for a Storage Network Switch,” *Proc. IEEE Globecom 2006*, November 2006.