International Journal of Computers, Communications & Control
Vol. II (2007), No. 3, pp. 269-278

Efficient Variable Length Block Switching Mechanism

Jaidhar C.D, A.V. Reddy

Abstract: Most popular and widely used packet switch architecture is the crossbar. Its attractive characteristics are simplicity, non-blocking and support for simultaneous multiple packet transmission across the switch. The special version of crossbar switch is Combined Input Crossbar Queue (CICQ) switch. It overcomes the limitations of un-buffered crossbar by employing buffers at each crosspoint in addition to buffering at each input port. Adoption of Crosspoint Buffer (CB) simplifies the scheduling complexity and adapts the distributed nature of scheduling. As a result, matching operation is not needed. Moreover, it supports variable length packets transmission without segmentation. Native switching of variable length packet transmission results in unfairness. To overcome this unfairness, Fixed Length Block Transfer mechanism has been proposed. It has the following drawbacks: (a) Fragmented packets are reassembled at the Crosspoint Buffer (CB). Hence, minimum buffer requirement at each crosspoint is twice the maximum size of the block. When number of ports are more, existence of such a switch is infeasible, due to the restricted memory available in switch core. (b) Reassembly circuit at each crosspoint adds the cost of the switch. (c) Packet is eligible to transfer from CB to output only when the entire packet arrives at the CB, which increases the latency of the fragmented packet in the switch. To overcome these drawbacks, this paper presents Variable Length Block Transfer mechanism. It does not require internal speedup, segmentation and reassembly circuits. Using simulation it is shown that proposed mechanism is superior to Fixed Length Block Transfer mechanism in terms of delay and throughput.
Keywords: Crossbar switch, Un-buffered Crossbar switch, Buffered Crossbar switch, Combined Input Crossbar Queue switch.

1 Introduction

Packet switching technology has become the predominant technology for high speed data networks and has begun to be used for applications like voice communication which have traditionally relied on circuit switching. Recently, many large-scale fast routers have used Input Queued (IQ) switches. In an Input Queued switch, variable length packets arriving at the inputs are segmented into fixed size packets known as cells for transmission over the switch and reassembled into packets at the output before being transmitted. Cell transmission time is fixed and is called as cell time or time slot. When a packet size is not an integral multiple of cell size, padding bytes are needed for the last fragment and is called as segmentation overhead. Smaller cell size generates more number of cells per packet and leads to a large switch header overhead. In high-speed switches/routers, segmentation results in heavy load. Moreover, if an optical switching technology is introduced, it is even more difficult to segment optical domain packets.

In high speed optical networks, it is more reasonable that incoming IP packets pass through a switch fabric based on packet by packet switching scheme. Variable length packet switching considers the entire IP packet as a single switching unit, doing away with padding bytes and reassembly buffers. For high speed switches/routers, non-existence of reassembly buffers and circuits is an attractive feature. In IP networks, the speed at which a scheduler can switch cells is not really significant, as even if only one cell is remaining for switching in an input queue, it is impossible to reassemble a complete IP packet in the reassembly buffer. Hence, it is reasonable to compare the packet switching with cell switching, from the point of view of Quality of Service (QoS).

In general, latency problem in packet switching scheme is worse than in cell switching scheme because of a decrease in the statistical multiplexing effect. However, from the packet latency view point, packet switching scheme performs better than cell switching scheme. In addition, packet latency is much better than that of cell latency for QoS requirement, because each packet represents complete information.

Variable length packets dominate network traffic (e.g., IP packets in Ethernet frames). Thus, a study of high-speed switches that support variable length packet switching is needed. A high performance variable length packet switching mechanism is proposed, for efficiently switching variable length IP packets. Performance of the proposed mechanism is evaluated in terms of packet latency and throughput. Overall, performance of our mechanism is better than earlier one.

2 Previous work

Buffers in an Input Queued (IQ) switch can be a single queue or multiple queues. Simplest one is single First In First Out (FIFO) queue, in which cells are served according to their arrival order i.e., First Come First Served (FCFS) basis. Due to the Head of Line (HoL) blocking problem, FCFS service discipline limits the maximum throughput to 58.6% [1]. To overcome HOL blocking problem several techniques have been proposed. One of the techniques is Virtual Output Queue (VOQ) [2] in which each input maintains separate queue for each output. VOQ approach overcomes the HOL blocking problem. However, it creates an input contention. Output contention occurs when more than one input wishes to send a packet to the same output at the same time. To resolve both input and output contentions, schedulers are used, whose function is to find one to one conflict free match between inputs and outputs in every time slot. Recently many scheduling algorithms have been proposed for Input Queued switch [3–8].

To cope with the overheads and the scheduler inefficiencies, internal speedup is used as an alternative solution. The switch speedup S is defined as the ratio of the switch bandwidth to the bandwidth of the line rates. Internal speedup is a good solution but, it incurs significant cost. If the speedup is N for $N \times N$ switch, arriving cells at the inputs get immediately transferred to their corresponding outputs. Hence, buffers are used only at the output side and this kind of architecture is named as Output Queued (OQ) switch. An OQ switch can provide QoS guarantees to individual data flow or groups of data flow [9]. However, OQ switch is inherently less scalable when number of ports is more or link rate is higher. This makes an IQ switch an attractive candidate when line rate is higher or number of ports is more. But, IQ switch fails to provide guaranteed QoS. To satisfy both the requirements of high switching capacity and guaranteed QoS simultaneously, Buffered Crossbar switch was proposed as an alternative.

Buffered Crossbar (BC) switch overcomes the limitation of un-buffered crossbar switch by employing buffers at each crosspoint. Adoption of Crosspoint Buffer (CB) drastically improves the overall performance of the switch. The first BC switch was implemented as a large multi cabinet unit [10].

Pure Buffered Crossbar (PB) switch employs buffers only at each crosspoint and nowhere else. Incoming cells at the input, enter directly into the switch core to reside in their corresponding CB. PB switch consists of a FIFO CB, preceded by an address filter (AF) was proposed [11]. Yet another PB switch with a restricted CB of size 16 KB was introduced [12]. There is a possibility for packet loss due to the restricted CB size. To prevent packet loss completely, larger CB is needed, but CB size is inversely proportional to switch size. When port size increases, it forces larger memory requirement to switch core. The requirement of larger CB could be minimized by employing buffers at the input side, in addition to CBs. Such an architecture is known as Combined Input Crossbar Queue (CICQ) switch. It separates input and output contentions. Schedulers at each input and output port work independently and in parallel. CICQ switch to use FIFO input buffer was proposed [13, 14]. Using simulation it is shown that the throughput of the FIFO Input Buffered CICQ switch is limited to 91% due to the HoL blocking problem [13]. To overcome the HoL blocking problem, VOQ's have been used in most of the proposed

CICQ architectures. In the rest of the paper, Virtual Output Queued CICQ switch is referred as CICQ switch. The CICQ switch having VOQ was first proposed in [15]. To schedule the cells, Oldest Cell First (OCF) selection strategy is employed at all the contention points. CICQ switch in which CB size is one cell, has been proposed [16] in which the Longest Queue First (LQF) scheduling is used at the input side and Round Robin (RR) scheduling is used at the output side. Yet another CICQ switch architecture, where CB is restricted to a single cell, has been proposed [17] in which RR scheduling style used at all the contention points. All these scheduling algorithms were just simple mapping of earlier algorithms, proposed for un-buffered crossbar switch into the new CICQ switch architecture.

3 Combined Input Crossbar Queue Switch (CICQ) Architecture

Our $N \times N$ CICQ switch model is shown in Fig. 1 and has a structure as described below.

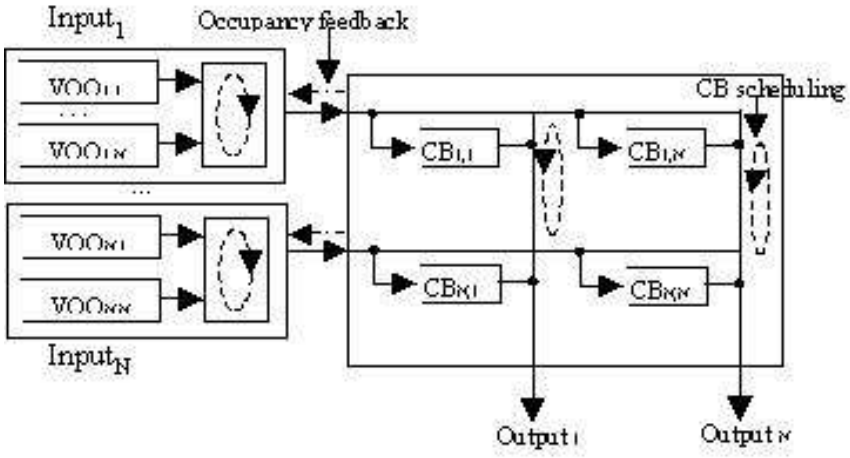


Figure 1: Combined Input Crossbar Queue switch with Round Robin Scheduling at all the contention points.

Input Queue: There are N VOQ's at each input, one for each output. Packets arriving at the input i destined for output j are stored in $\text{VOQ}_{i,j}$. Internal fabric consists of N^2 Crosspoint Buffers. The $\text{CB}_{i,j}$ stores the packets coming from input i destined to output j, where $i, j = 1, 2, \ldots, N$ and its size is set to 2250 bytes.

Scheduler: Each input and output port has its own scheduler and each of them work independently and in parallel. At all the contention points Round Robin (RR) scheduling is used. Input scheduler selects VOQ from among the active VOQs. A $\text{VOQ}_{i,j}$ is said to be active for being scheduled in the input scheduling process, if it is not empty and the corresponding $\text{CB}_{i,j}$ has enough space to accommodate the incoming block. i.e., the value of its credit counter is greater than or equal to the size of its head block. The output scheduler is responsible for: (a) selecting the next eligible flow (CB) in its column; (b) initializing the transmission of packets to the specific switch output and sending a credit back to the appropriate input credit scheduler. A flow is eligible when the corresponding CB is not empty. If there is more than one eligible flow, the output scheduler has to select one of them in a RR fashion.

Flow Control: A credit based flow control mechanism is used in order to provide lossless transmission between input port and CB [18]. Each input i maintain N credit counters, one for each VOQ. Initially, value of these counters is set to the CB size. Whenever an input scheduler forwards a block from a VOQ, it decrements its respective credit counter by the size of the forwarded block. When a block departs from $\text{CB}_{i,j}$, its corresponding output scheduler sends a credit back to the respective $\text{VOQ}_{i,j}$.

3.1 Variable Length Packet Switching

Significant advantage of the CICQ switch is their capacity to directly switch variable length packets without segmenting it. All the input and output schedulers operate independently and in parallel. Hence, there is no global "time-frame" that constrains the system to transmit fixed size packets. This does not hold good for unbuffered crossbar switch.

Native switching of variable length packets eliminates the internal speedup and reassembly circuit at all the outputs. However, native switching of variable length packets results in unfairness. Consider two VOQ's at a port i, where $VOQ_{i,1}$ is saturated with large packets and $VOQ_{i,2}$ is saturated with small packets. In this scenario, RR polling alternatively selects $VOQ_{i,1}$ and $VOQ_{i,2}$ regardless of packet size and $VOQ_{i,1}$ achieves high transfer rate than $VOQ_{i,2}$. Fig. 2 shows unfairness caused by RR selection strategy for a 2×2 switch. To overcome this unfairness, Fixed Length Block Transfer mechanism has

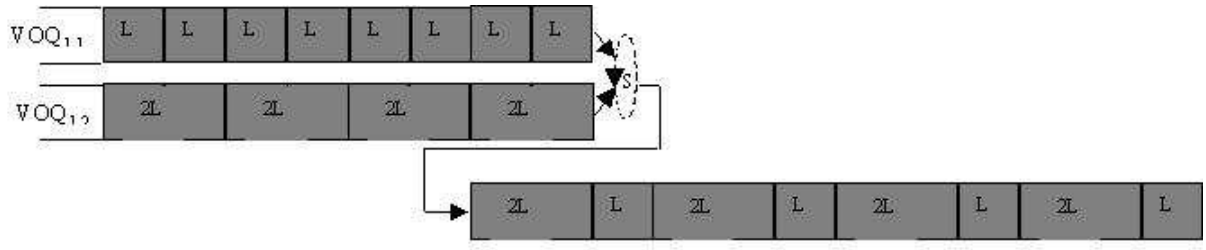


Figure 2: Native Switching of Variable Length Packets.

been proposed [19]. In this mechanism, VOQ is eligible to transfer predefined block bytes of data (i.e., 1500 B) to its CB, when it gets the chance. Each block consists of a set of entire packets and/or packet segments and packet reassembly is performed at the CB. Hence, minimum CB requirement at each crosspoint is twice the maximum size of the block. Due to restricted memory available to switch core, it does not work when port size is larger. Packet is eligible to transfer from CB to output port only when entire packet has arrived at the CB. As a result, latency of the fragmented packet is increased. Consider the worst case scenario where packet P_1 of size 40 B and P_2 of size 1500 B alternatively arrive at the input port. Each block contains a fragmented packet. Fragmented packet is delayed at the CB till the arrival of the complete packet, even when there is no output contention. Moreover, number of reassembly circuits is square of the switch size, which adds to the cost of the switch. To overcome these problems Variable Length Block Transfer mechanism is proposed in this paper.

Variable Length Block Transfer mechanism transfers up to a block of bytes, of a data packet from a selected VOQ to CB. Block size may vary from block to block and its maximum size is restricted to 2250 bytes. Block may contain set of entire packets or a single complete packet. Packets that share a common destination, are packed inside the block continuously one after other. When entire packet cannot be accommodated in a single block, it is packed into a new block instead of fragmenting it. Unlike cell switching, our mechanism does not use padding bytes to fill the block. Hence, speedup is eliminated. Fig. 3 shows Variable Length Block Transfer mechanism for the input port 1. The block 1 at the $VOQ_{1,1}$ consists of set of entire packets $P_{1,1}$ and $P_{1,2}$. Sum of the size of $P_{1,1}P_{1,2}$ and $P_{1,3}$ is greater than 2250 bytes. Hence, packet $P_{1,3}$ is packed into new block, block 2 without fragmenting it. Under heavy load, block size may be maximized. As a result, header overhead is reduced and crossbar operates very close to maximum efficiency. Table 1 compares the proposed mechanism and Fixed Length Block Transfer mechanism.

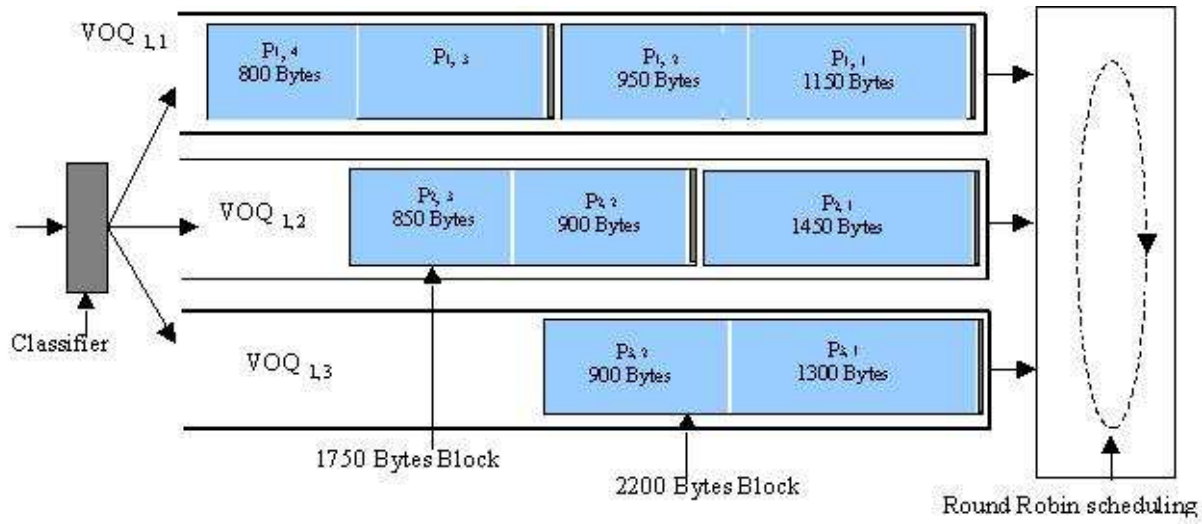


Figure 3: Variable Length Block Transfer Mechanism.

Table 1: Comparison of the proposed mechanism and Fixed Length Block Transfer mechanism.

Characteristics	Variable Length Block Transfer mechanism	Fixed Length Block Transfer mechanism
Cross-point Buffer Size	2250 Bytes	3000 Bytes
Segmentation and Reassembling Circuit	No	Yes
Scheduler	Round Robin	Round Robin
Packet Segmentation	No	Yes
Block Size	Variable Size	Fixed Size

4 Simulation Experiment

Three simulation experiments were designed in order to compare the performance of the proposed mechanism with earlier ones. For all experiments, a 32×32 switch, port speed of 10 Gbps, no internal speedup and single priority is assumed. Round Trip Time (RTT) between line cards and switch fabric has been set to 40 B times (corresponding to 32 ns at 10 Gbps line rate) which is the sum of the following delays (a) input arbitration (b) the transmission of a packet from an input port to the switch crossbar (c) the output arbitration and (d) the transmission of the flow control information back from the crossbar to the input port. Delay is measured as the time interval between the first byte of the packet arriving at the input port and its first byte departing from the output port. The reported delay is averaged over all packets.

Experiment #1: Poisson arrivals of variable length packets are assumed and each of the 32 input ports chooses an output port with a uniform distribution over the 32 output ports ($\lambda_{i,j} = \rho/N$ for all i and j). Every input port has identical offered load ranging from 80 to 98%.

Experiment #2: Each input i hosts two active flows, flow $i \rightarrow i$ and $i \rightarrow (i+1)$ mod N. The former flow consumes two thirds (2/3) of the incoming load and the latter consumes the remaining one third (1/3). Poisson arrival of variable length packets is assumed and the offered load ranges from 50 to 98% and 80 to 98% for small and large size packets respectively.

Experiment #3: Both Variable Length and Fixed Length Block Transfer mechanisms are modeled under non-uniform traffic such as unbalanced traffic as defined in [17]. It uses a probability w, as the

fraction of the input load directed to a single predetermined output, while the rest of the input load is directed to all outputs with uniform distribution. Let us consider input port s, output port d, and the offered load for each input port $\rho_{s,d}$. The traffic load from input port s to output port d, $\rho_{s,d}$, is given by

$$\rho_{s,d} = \begin{cases} \rho\left(w + \frac{1-w}{N}\right), & \text{if } s = d \\ \rho\left(\frac{1-w}{N}\right), & \text{otherwise} \end{cases}$$

When $w = 0$, the offered traffic is uniform. On the other hand, when $w = 1$ the traffic is completely directional from input i to output j, i.e., $i = j$. Poisson arrivals of variable length packets are assumed and the throughput is measured as a fraction of the maximum possible one (320 Gbps in our simulation).

5 Experimental Results

Fig. 4(a) shows the results for experiment #1 under Bimodal packet size distribution in which packet size of 40 B and 1500 B alternatively arrived at the input ports. Average delay of the proposed mechanism is lower than Fixed Length Block Transfer mechanism. In a Fixed Length Block Transfer mechanism, each block contains a fragmented packet. When the block arrives at the CB, the fragmented packets' reassembly is delayed until the next packet arrives at the CB. Segmentation and reassembly delay increases the latency of the packet. These types of delay are non-existent in our mechanism. Hence, our mechanism exhibits shorter delay than the earlier mechanism. Block of data bytes are eligible for transfer from CB to output, when CB gets the chance to transfer. If there is no output contention in a column, block of data bytes can immediately be transferred without waiting for the next packet. Fig. 4(b) shows the results of experiment #1 in which packet size is uniform. Our mechanism shows lower average delay than Fixed Length Block Transfer mechanism due to the non existence of segmentation and reassembly delay.

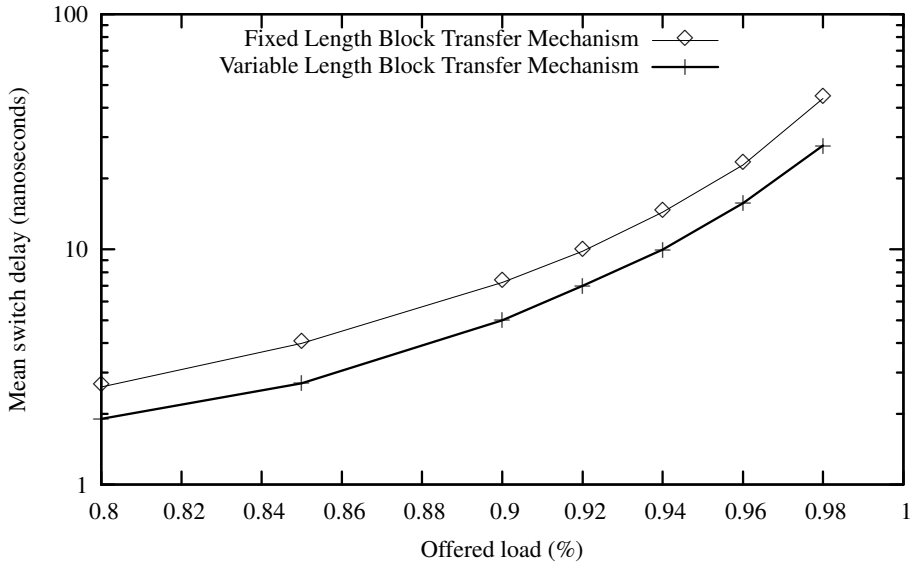
Figs. 5(a) and 5(b) show the results for the experiment #2 for small and large packets respectively. Mean delay of Block transfer mechanism is higher for larger packets than Output Queued switch. However, it shows smaller average delay for small packets.

Presence of two-stage buffering in a CICQ switch introduces priority, based on packet lengths. The transfer time required for packets from a VOQ to a CP is proportional to the size of the packet. Thus, a smaller packet requires less transfer time from a VOQ to a CP. Suppose the transfer of a small packet from port 1 to $CB_{1,1}$ and the transfer of a large packet from port 2 to $CB_{2,1}$ begin at the same time. The small packet arrives at a $CB_{1,1}$ before the large packet does at a $CB_{2,1}$. Thus, the small packet will be transmitted before the large packet, to an output link if the remaining $CB_{i,1}$ for all ports i are empty. The effect is demonstrated by the smaller mean delay than that of the OQ switch, for small packets, at a high offered load. The Block transfer mechanism further adds the packet size-based priority within each port. In the Block transfer mechanism, a multiple of small packets of a single block, are eligible for transfer from a single VOQ in the RR polling at an input port, giving higher priority to smaller packets over larger packets within the input port. This explains the lower small packet mean delay of the block transfer mechanism than the Output Queued switch for all offered loads considered.

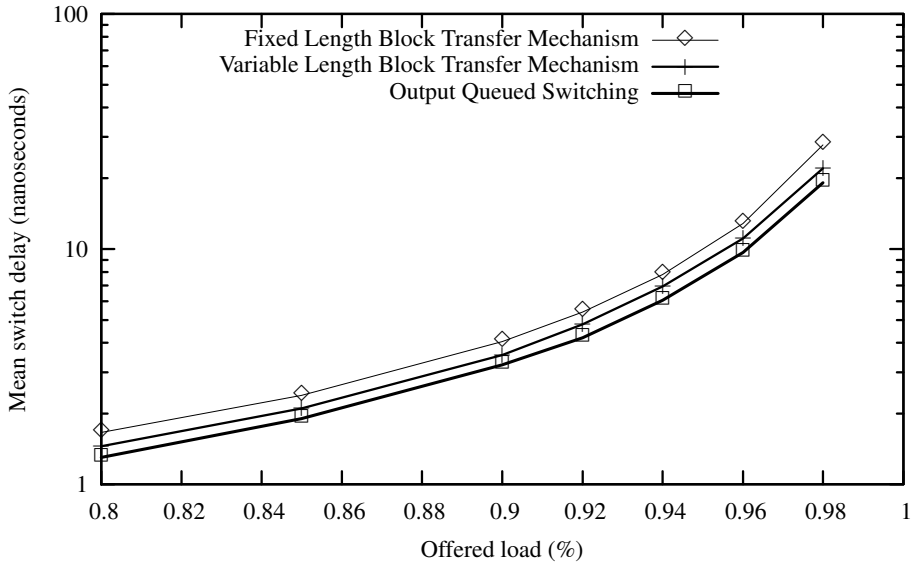
Fig. 6 shows the results for the experiment #3. It is observed that our mechanism shows higher throughput than earlier ones. Under heavy load, block size is maximized and as a result header overhead is reduced.

6 Conclusions and future work

In this paper, Variable Length Block Transfer mechanism is proposed to overcome the limitations of Fixed Length Block Transfer mechanism. Arriving packets at the inputs are not segmented as a result padding bytes and internal speedup is not required. In addition it eliminates reassembly circuit and

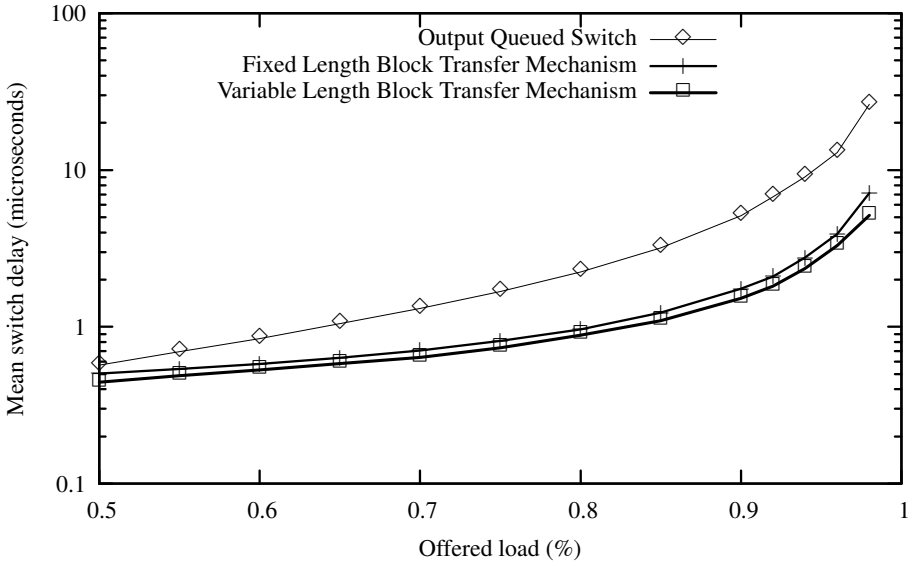


(a) Bimodel packet size

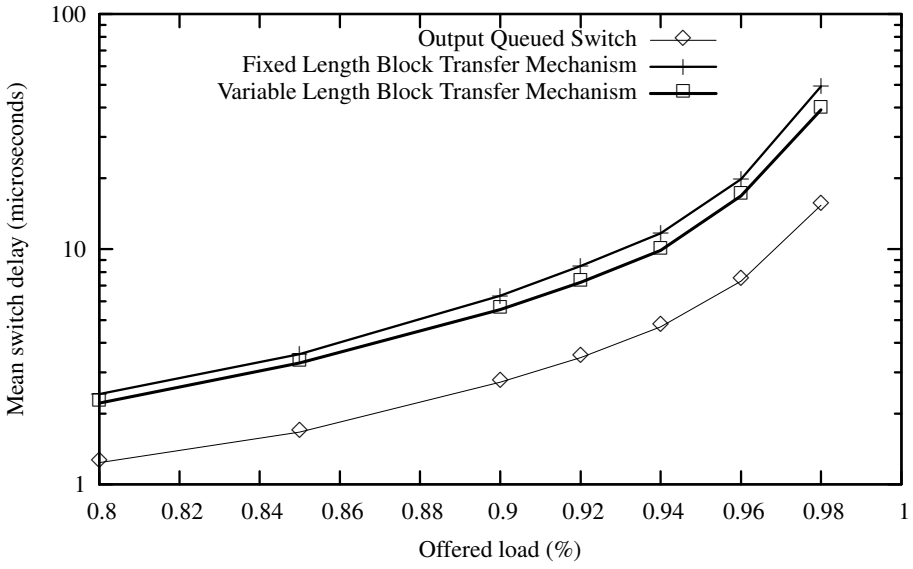


(b) Uniform packet

Figure 4: Delay performance of the proposed mechanism, Fixed Length Block Transfer under uniform traffic.



(a) Small packets



(b) Larger packets

Figure 5: Results for diagonal experiment.

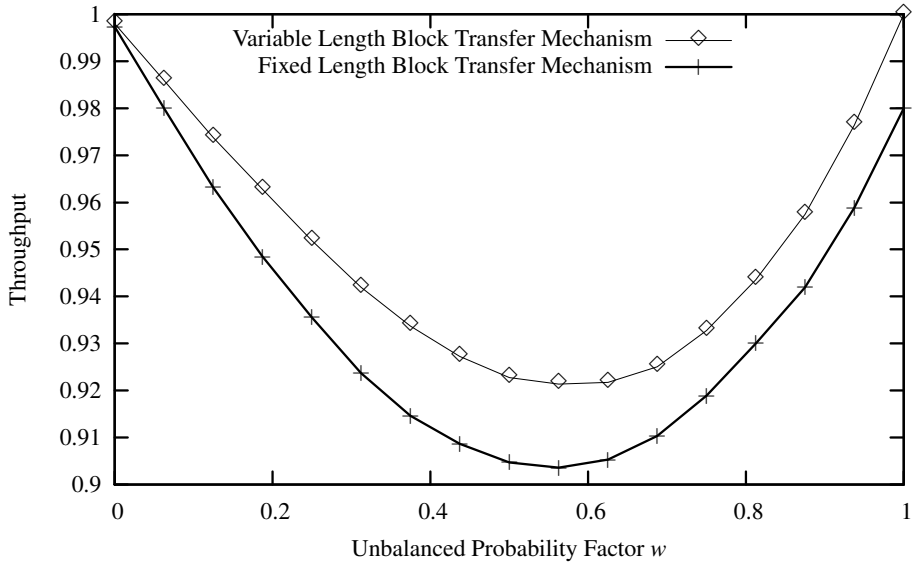


Figure 6: Throughput experiment under unbalanced traffic with load 100%.

buffers which reduces the cost of the switch. Block size is maximized under heavy load resulting in reduction of header overhead, scheduler rate and power consumption. Mean switch delay of the proposed mechanism is lower than earlier because of the not existence of the segmentation and reassembly. Memory requirement at the each crosspoint is 25% lesser than the earlier one and it is feasible to implement. Through simulation, proposed mechanism is compared with earlier one and found to be superior in terms of switch throughput and packet mean delay.

Our mechanism may produces the unfairness in terms of service rate when different VOQ's have different arrival rate which we are trying to rectify in our further work. Unfairness problem can be overcome by maintaining a counter called service counter for each VOQ and CB ($VSC_{i,j}$ and $CBSC_{i,j}$). Initial value of these counters is set to zero and the value may vary at the time of scheduling. During the time of scheduling, input port scheduler examines the content of the service counter $VSC_{i,j}$. If the value of $VSC_{i,j}$ is greater than or equal to threshold value, $VOQ_{i,j}$ is not eligible in the current round and its value is updated as $VCS_{i,j} = VCS_{i,j}$—Threshold value. Otherwise $VOQ_{i,j}$ is eligible in the current round and its service counter value is updated as $VCS_{i,j} = VCS_{i,j} + VOQ_{i,j}$ [Block size] − minimum block size.

Bibliography

[1] M. J. Karol, M. G. Hluchyj and S. P. Morgan, Input Versus Output Queuing on a Space-Division Packet Switch, *IEEE Transactions on Communications*, Vol. 35, pp. 1347–1356, 1987.

[2] T. Anderson, S. Owicki, J. Saxe and C. Thacker, *High Speed Switch Scheduling for Local Area Networks*, *ACM Transactions on Computer Systems*, Vol. 11, pp. 319–352, 1993.

[3] N. McKeown, The iSLIP Scheduling Algorithm for Input-Queued Switches, *IEEE/ACM Transactions on Networking*, Vol. 7, pp. 188–201, 1999.

[4] D. N. Serpanos and P. I. Antoniadis, FIRM: A Class of Distributed Scheduling Algorithms for High-Speed ATM Switches with Multiple Input Queues, *Proceedings of the IEEE INFOCOM*, pp. 548–555, 2000.

[5] H. J. Chao and J. S. Park, *Centralized Contention Resolution Schemes for A Large-Capacity Optical ATM Switch*, Proceedings of the IEEE ATM Workshop, 1998.

[6] Y. L. S. Panwar and H. J. Chao, On the Performance of a Dual Round-Robin Switch, *Proceedings of the IEEE INFOCOM*, pp. 1688–1697, 2001.

[7] N. McKeown, V. Anantharam and J. Walrand, Achieving 100% Throughput in an Input-Queued Switch, *Proceedings of the IEEE INFOCOM*, pp. 296–302, 1996.

[8] A. Mekkittikul and N. McKeown, A Starvation-free Algorithm for Achieving 100% Throughput in an Input-Queued Switch, *Proceedings of the IEEE ICCCN*, pp. 226–231, 1996.

[9] G. Keridis and N. McKeown, Output-buffer ATM Packet Switching for Integrated Services Communication Networks, *Proceedings of the IEEE International Conference on Communications*, Montreal, Canada, 1997.

[10] E. Rathgeb, T. Theimer and M.Huber, Buffering Concepts for ATM switching networks, *Proceedings of the IEEE GLOBECOM*, pp. 1277–1281, 1988.

[11] A. K. Gupta, L. O. Barbosa and N. D. Georganas, 16×16 Limited Intermediate Buffer Switch Module for ATM Networks, *Proceedings of the EEE GLOBECOM*, pp. 939-943, 1991.

[12] A. L. Gupta and N. D. Georganas, Analysis of a packet switch with input and output buffers and speed constraints, *Proceedings of the IEEE INFOCOM*, pp. 694–700, 1999.

[13] A. K. Gupta, L. O. Barbosa and N. D. Georganas, Limited Intermediate Buffer Switch Modules and their Interconnection Networks for B-ISDN, *Proceedings of the IEEE International Conference on Communications*, pp. 1646–1650, 1992.

[14] M. Lin and N. McKeown, The Throughput of a Buffered Crossbar Switch, *IEEE communications Letters*, Vol. 9, pp. 465–467, 2005.

[15] M. Nabeshima, Performance Evaluation of a Combined Input and Crosspoint Queued Switch, *IEICE Transactions on Communications*, Vol. E83-B, pp. 737–741, 2000.

[16] T. Javid, R. Magil and T. Hrabik, A High Throughput Scheduling Algorithm for a Buffered Crossbar Switch Fabric, *Proceedings of the IEEE International Conference on Communications*, pp. 1586–1591, 2001.

[17] R. R. Cessa, E. Oki, Z. Jing and H. J. Chao, CIXB-1 Combined Input-One Cell-Crosspoint-Buffered Switch, *Proceedings of the IEEE Workshop on High Performance Switching and Routing*, pp. 324–329, 2001.

[18] H. T. Kung and R. Morris, Credit-Based Flow Control for ATM Networks, *IEEE Network Magazine*, Vol. 9, pp. 40–48, 1995.

[19] K. Yoshigoe and K.J. Christensen, An Evaluation to Crossbar Switches with Virtual Output Queuing and Buffered Cross points, *IEEE Network*, Vol. 17, pp. 48–56, 2003.

Jaidhar C.D.
Department of Computer Science and Engineering
National Institute of Technology
Tiruchirappalli – 620 015, India
E-mail: csk0305@nitt.edu

A.V Reddy
Department of Computer Applications
National Institute of Technology
Tiruchirappalli – 620 015, India
E-mail: reddy@nitt.edu
Received: March 30, 2007
Revised: May 19, 2007