1998 International Computer Symposium
Workshop on Computer Networks, Internet, and Multimedia
December 17-19, 1998, N.C.K.U., Tainan, Taiwan, R.O.C.

Architecture Optimization of Broadband Fast Packet Switches with Clustering and Speedup Constraints

Yu-Sheng Lin[*], Christos J. Georgiou, and Chung-Sheng Li
IBM T. J. Watson Research Center, P. O. Box 704, Yorktown Heights, NY 10598
email: {georgiou,csli}@us.ibm.com

Abstract

In this paper, we study how clustering and speedup at the input and output ports of a generic nonblocking packet switch affect switch throughput and port buffer size. By determining the maximum allowable clustering and speedup, an optimal switch configuration can be established for a given VLSI technology. Our performance analysis shows that output port speedup is most effective in increasing throughput but has no effect on buffer reduction, while input speedup has a moderate effect on both increasing throughput and decreasing buffer size. Input-port grouping is useful on buffer reduction but has no effect on throughput, while output-port grouping has a moderate effect on increasing throughput and a negligible effect on buffer reduction.

1 Introduction

Broadband packet-switched networks have recently become a reality due to the rapid advances in high-speed VLSI and fiber-optic technologies. Several emerging standards for packet-switched LAN, MAN, and WAN environments, such as Asynchronous Transfer Mode (ATM), Fiber Channel Standard (FCS), and Gigabit Ethernet can support data transmission at or beyond gigabit data rates. While each of these standards may be targeted towards a different application environment, the high-speed packet switch is invariably the most essential component for these gigabit networks.

Various architectures for high-throughput packet switches that provide a wide range of cost/performance characteristics have been extensively studied [1]. In the evaluation of these architectural alternatives, tradeoffs are made based on an analysis of the switch performance, VLSI technology constraints, and cost estimates. A typical model of a packet switch consists of three elements: input ports, switching fabric, and output ports. Because of the random nature of packet flow in the network, buffers are usually provided in one or more of these elements to allow data traffic smoothing and contention resolution.

Input buffers based on FIFO queues are easy to implement, but the maximum switch throughput for fixed-size packets is limited to 0.586 due to the *head-of-line* (HOL) blocking phenomenon [2]. The HOL blocking problem can be eliminated by using non-FIFO input buffers [3], with which the throughput can be increased to 0.8, at the expense of slightly increased overhead in packet scheduling. An output buffering scheme can potentially provide higher throughput by allowing more than one packet to be transmitted simultaneously to an output port over the switching fabric. The tradeoffs of input queuing versus output queueing have been investigated in [4] in which the authors show that the mean queue length and mean waiting time are larger for input queueing than for output queueing. However, an output-buffer switch usually requires higher memory bandwidth and greater switching fabric speedup. It was shown in [5] that the switch throughput can reach or exceed 0.99 for an output speedup of four.

[*] on leave from Dept. of Electronic Engineering, National Chiao Tung Univ. Taiwan, R.O.C. while performing this work.

The average buffer size per port can be reduced if multiple input or/and output links are grouped together to share the same buffer [8][9]. Moreover, grouping output links can reduce output port contention because packets with the same destination contend for a group of output links rather than just a single output link. The shared-memory switch [10] architecture is the extreme case of link grouping and buffer sharing.

The purpose of this paper is to study the system impacts when multiple architecture optimization techniques are applied under two technological constraints: switch fabric speedup and buffer memory bandwidth, both of which arise from using VLSI technology for switch design. A generic packet switch architecture characterized by four design parameters is proposed to specify the degree of speedup and link grouping (buffer sharing). Using a three-stage queuing model, we analyze the maximum achievable switch throughput and buffer requirement for different parameter sets. The results show that a throughput of one can be achieved by simultaneous output speedup (with a speedup factor of 1.65) and input speedup (with a speedup factor of 2). The effects of each design parameter on the buffer size are also investigated. Based on the observation of these effects, principles for optimal allocation of memory bandwidth and switch fabric speed up are developed.

The organization of this paper is as follows: Section 2 describes the generic switch architecture that provides the foundation for performance study of various optimization approaches. ›A three-stage queueing model which describes the the switching system is discussed in Section 3. Section 4 describes the analytical results of throughput and buffer requirement. The optimal resource allocation for various implementation constrains is presented in Section 5. A brief summary is given in Section 6.

2 Generic Switch Architecture

The switch architecture to be studied in this paper is shown in Fig. 1. This architecture consists of:

1. N input links.

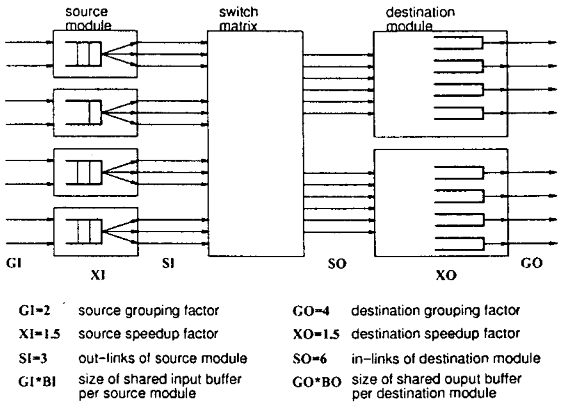

source module
switch matrix
destination module
GI
XI
SI
SO
XO
GO
GI=2 source grouping factor
XI=1.5 source speedup factor
SI=3 out-links of source module
GI*BI size of shared input buffer per source module
GO=4 destination grouping factor
XO=1.5 destination speedup factor
SO=6 in-links of destination module
GO*BO size of shared ouput buffer per destination module

Figure 1: Structure of the generic switch model.

2. N output links.
3. N/GI input modules: Each input module, IN, has a buffer of size $GI \times BI$ shared among GI input links of the module. The parameter, BI, is the storage size for each source link. An IN module is connected through SI outgoing links to the switch matrix.
4. Switch fabric: The switch matrix between IN and OUT modules has $N \times XI$ inputs and $N \times XO$ outputs. The switch matrix is bufferless and non-blocking.
5. N/GO output modules: Each output module, OUT, has a buffer of size $GO \times BO$ shared among GO output links of the module. The parameter, BO, is the storage size for each output link. An OUT module is connected to the switch fabric through SO links.

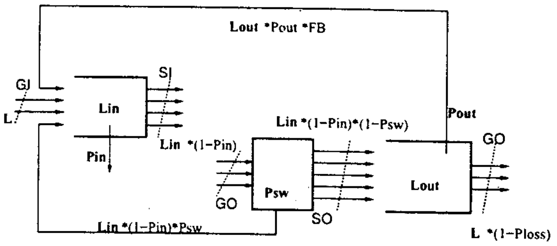
All of the links are assumed to operate at the same data rate.

The proposed switch architecture can thus be completely characterized by the tuple (GI, SI, GO, SO). The parameters GI and GO describe the grouping (clustering) factor at the input and output switch links, respectively, while SI and SO specify the speedup at the corresponding input and output of a switch fabric. The ratios $XI = SI/GI$ and $XO = SO/GO$ represent the input and output speedup factor, respectively. This architecture can be used to model those switch architectures reported in previous works by choosing appropriate (GI,SI,GO,SO). For example, a baseline switch without any speedup or grouping is represented as (1,1,1,1). Output speedup architecture discussed in [5] can be represented as (1,1,1,SO), while output speedup with grouping in [8] can be represented as (1,1,GO,SO). An N-port shared-buffer switch can be represented by $(N, N, 1, 1)$[1] We choose the (N,N,N,N) configuration, in which the input and output buffer are fully shared by all ports, as the benchmark for buffer-size comparisons.

3 Queuing Model

The switch architecture described in the previous section is approximated by a three-stage queueing model, as shown in Fig. 2. By assuming uniformly distributed Poisson arrival and exponential packet length distribution, the behavior of each stage is as follows:

[**Input modules**] Each IN module is modeled as an FCFS (First-Come-First-Serve) queue with SI servers and a buffer capable of storing $GI \times BI$ packets. The arrival process is assumed to be Poisson while the length of a packet is assumed to be exponentially distributed with uniformly distributed destinations. Thus, the input module can be modeled as an (M/M/SI/GI×BI+GI) queueing system. The probability of a packet arriving at an overflow input buffer is denoted by P_{in}.



	IN queue	Switching point	OUT queue
	(M/M/C/K) =(M/M/SI/GI*BI+GI)	(M/M/C/K) =(M/M/SO/SO)	(M/M/1/K) =(M/M/1/GO*BO+GO)
arrival:	GI * Lin	GO * Lin*(1-Pin)	GO * Lout
server:	SI	SO	GO

FB =1: Packets are not switched if output buffer is full

FB =0: Packets are lost if output buffer is full

L	original rate / port	Psw	prob. of failure in arbitration
Lin	effective input rate / port	Pin	prob. of input buffer overflow
Lout	effective output rate / port	Pout	prob. of output buffer overflow
GI	source grouping factor	GO	destination grouping factor
XI	source speedup factor	XO	destination speedup factor
SI	out-links of source module	SO	in-links of destination module
BI	input buffer size per link	BO	output buffer size per link

Figure 2: Three-stage queueing model for performance analysis

[**Switch fabric**] Each switch point in a switch matrix is modeled as a queue with SO servers and no buffer. The output port contention is thus equivalent to contending for the SO servers. A packet lost in the contention is considered as arriving at an all-server-busy condition, and is put back to the input queue for rescheduling to avoid HOL blocking. We assume the waiting time for rescheduling is exponentially distributed and independent of the arrival process. The input process, which is a combination of the arrival process and the feedback process, to an IN module is thus still Poisson with a rate equal to the sum of the arrival rate and the feedback rate. Thus, the switch fabric can be modeled as an (M/M/SO/SO) queueing system. We denote the probability that a packet is rejected at the switching point due to output port contention as P_{sw}.

[**Output modules**] Each OUT module is modeled as GO single-server queues. Since the output buffer is shared by all of the queues in the same OUT module, overflow happens only when the total queue size becomes larger than the buffer size. Thus, the output module can be modeled as an (M/M/1/GO×BO+GO) queueing system. The probability of a packet arriving at an overflow output buffer is denoted by P_{out}.

The value of the parameter, FB, is used to indicate the existence of feedback of packets to the input queue. In case buffer overflow occurs in an OUT module, the packet can be either discarded ($FB = 0$) or fedback to the input queue ($FB = 1$) where it waits for rescheduling. The feedback case ($FB = 1$), which is used for all analytical results, models the backpressure mechanism used to prevent output buffer overflow. If the waiting time for rescheduling of the over-

[1] In fact, the switch fabric and output ports do not exist.

flow packets in output modules is exponentially distributed and independent of the arrival process and feedback process from the switch points, the combined input process to the IN modules is still Poisson with a rate equal to the sum of all three components.

Due to the assumptions for the input process and packet length distributions, the input queue can be modeled as an M/M/SI/GI×BI+GI queue with an input rate of GI×L_{in}, where L_{in} is the effective input rate to the IN module.

The switching point can be modeled as an M/M/SO/SO queue with an input rate of $GO \times L_{sw}$, where L_{sw} is the effective input rate to the switch point. according to the uniform traffic assumption. The output queue is an M/M/1/GO×BO+GO queue with an effective input rate of L_{out}.

Assuming the system is in the steady state with an input rate L to each input link, the effective input rate to an IN module is

$$L_{in} = L \times GI + L_{out}P_{out}FB + L_{in}(1 - P_{in})P_{sw} \quad (1)$$

The effective input rate to a switch point is

$$L_{sw} = L_{in}(1 - P_{in}) \quad (2)$$

and the effective input rate to an OUT module is

$$L_{out} = L_{in}(1 - P_{in})(1 - P_{sw}) \quad (3)$$

The overflow probabilities for above queueing systems with finite buffer size can be calculated, for example, from [12]. For an M/M/C/K queue (for the input module and switch fabric), the loss probability is:

$$\begin{aligned} P_{loss} &= \frac{P_0 \times \rho^K}{C! \times C^{(K-C)}} \\ P_0 &= [\sum_{n=0}^{C} \frac{\rho^n}{n!} + \frac{\rho^C}{C!} \times \sum_{n=1}^{K-C} (\frac{\rho}{C})^n]^{-1} \end{aligned} \quad (4)$$

and for an M/M/1/K queue (the output module) the loss probability is:

$$P_{loss} = \frac{(1-\rho) \times \rho^K}{1 - \rho^{(K+1)}} \quad (5)$$

By substituting Eq.(4) to P_{in} in Eq. (1) and and P_{sw} in Eq.(2), while substituting Eq. (5) to P_{out} in Eq. (3), the overflow probability of the queueing system can be obtained for a given buffer size and system configuration, or vice versa.

Due to the difficulty in finding a closed-form solution as a result of the exponential relationship between the buffer size and the loss probability, a numerical approach is adopted to obtain the solution. In this approach, the equations are solved iteratively by imposing an initial condition $P_{in} = P_{out} = P_{sw} = 0$. In most cases, numeric results converge within 0.1% of its asymptotic values in just a few iterations.

In a practical switch design, a separate queue is usually implemented for each destination if the buffer is shared among several output links. Using this implementation, rescheduling of the packets dropped from the switching points or output queues due to buffer overflow can be handled in a more intelligent way (instead of just waiting for exponentially distributed random interval assumed in the model). Therefore, our analytical model can only provide a *pessimistic* performance estimation, resulting in a conservative design.

4 Performance Analysis

The throughput of a switch is defined as the utilization of the output links. The maximum throughput of a switch architecture can be derived by assuming infinite buffer size so that there is no packet loss due to buffer overflow. Assuming $P_{in} = P_{out} = 0$ and all of the queues are in the steady state, the input rate to an IN module equals

$$L_{in} = min\{L + P_{sw}L_{in}, XI\} \quad (6)$$

The combined input rate is limited by the speedup factor XI since the queue length inside the IN module could become infinite if the input rate exceeds XI. Similarly, the input rate to an OUT module cannot exceed 1 and equals

$$L_{out} = min\{L_{in}(1 - P_{sw}), 1\} \qquad (7)$$

Combining the above two equations, the maximum achievable throughput thus equals

$$L = min\{XI - P_{sw}min\{XI, \frac{1}{1 - P_{sw}}\}, 1\} \qquad (8)$$

The actual throughput can then be obtained by substituting Eq. (4) and (5) into Eq. (8) for a given buffer size (BI,BO) and configuration (SI,SO,GI,GO), The throughput as a function of output grouping and speedup for various parameter sets is shown in Fig. 3. If only output speedup is applied, e.g., XO=2, the throughput is improved from 0.5 to 0.8. For XO larger than four, the throughput is higher than 0.99 since the event that more than four packets have the same destination is rare. This is consistent with the previous results [5]. If only input speedup is applied, e.g., XI=2, the throughput is improved from 0.5 to 0.67. For XI=8, the throughput becomes 0.89. The improvement is less significant than the output speedup because the input speedup only increases the number of candidate packets available for routing at the switching points. There is no saturation behavior as in the output speedup case. Output grouping increases the number of servers at the switching points but also increases the load, resulting in a modest throughput improvement. The throughput only increases from 0.6 to 0.76 as GO increases from 2 to 8. Due to the assumption that destinations are uniformly distributed, input grouping does not improve the throughput because it has no effect on the traffic at the switching points.

If both input and output speedup are applied, throughput improvement is even more significant since the number of both candidate packets and servers are increased. With output speedup, e.g., XO=2, 100% throughput can be achieved with an input speedup XI=1.35. Comparing to the cases in which only output speedup is applied, the required speedup of the switch matrix in order to achieve the same throughput is reduced from 4 (XI=1,XO=4) to 2.7 (XI=1.35,XO=2).

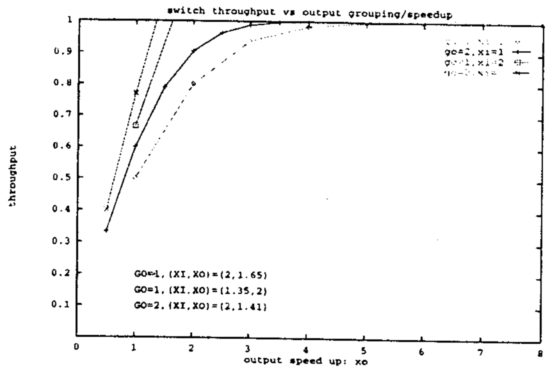


Figure 3: Throughput vs output speedup (XO) for different output grouping (GO) and input speedup (XI).

For a given load and buffer size, the steady state of the three-stage queueing model in Fig. 2 can be solved iteratively. The effects of each design parameters on buffer reduction at load=0.4 and packet loss probability 10^{-9} is shown in Fig 4. Because of the backpressure mechanism, most of the buffers are provided at the input end. Input speedup is most effective because it reduces P_{sw}, increases the service rate of the input queue, and contributes to buffer sharing. The effect of output grouping is a reduction in both P_{sw} and BO. But this effect is not as significant compare to input grouping on BI reduction. The output speedup is only effective in reducing P_{sw}. However, the effect saturates if the speedup is greater than four, as in the case of throughput analysis.

As the offered load increases, more buffer is needed in order to maintain the same packet loss rate. When the load approaches the maximum throughput of the switch, the required buffer size increases to infinity. The buffer reduction as a function of the traffic load for each design parameter is shown in Fig 5. Output speedup is most effective to push the throughput limit to the right(higher load), but has no effect on lowering buffer size. Input speedup has a moderate effect on both throughput increase and buffer reduction. Input grouping is useful on buffer reduction but

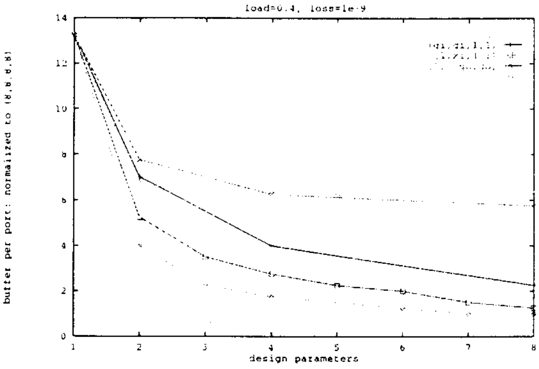


Figure 4: Effect of design parameters on buffer reduction. (8,8,8,8) represents the fully shared configuration. The input speedup case (1,xi,1,1) is most effective on buffer reduction. Follows the input grouping, output grouping, and the output speedup is the least effective.

has no effect on throughput. Output grouping has a moderate effect on throughput increase and negligible effect on buffer reduction. For switches target on 10^{-9} loss probability at heavy load (0.8 to 0.9), it is more important to push the throughput limit by speedup at both input and output. With backpressure, input grouping is more effective on buffer reduction than output grouping/speedup.

5 Configuration Optimization

The buffers in a VLSI switch chip are usually implemented with memory arrays (such as SRAM). For a given technology, the access rate of a memory array is limited. Memory bandwidth can be increased by increasing the memory word length or through memory interleaving. However, extra hardware for either performing memory interleaving or converting between the raw input packets and the wide memory words is necessary. For a given hardware complexity and chip area constraint, the maximum achievable memory bandwidth and switch fabric speedup are thus limited.

For the generic architecture in Fig. 1, there could be at most GI and SI links performing simultaneous write and read access, respectively, on the shared buffer of an IN module. Thus the shared input buffer must support data access at a link rate of $GI + SI$. Similarly, the shared output buffer of an OUT module must support memory access at a link rate of $GO + SO$. We choose the maximum memory bandwidth of IN and OUT modules C_m = max{ (GI+SI) , (GO+SO)} as the memory bandwidth cost measure of the IN/OUT modules. This is because as long as the higher access rate can be supported by a technology, so can the lower rate one. The cost to implement a switch matrix is proportional to the total number of crosspoints. For an $N \times N$ switch, the total number of crosspoints is N^2. For the generic architecture, the total number of crosspoints is $XI \times XO \times N^2$. We choose the scale factor $C_{sw} = XI \times XO$ as the cost measure of the switch matrix. The cost measures C_m and C_{sw} will be used in the discussion of optimal resource allocation.

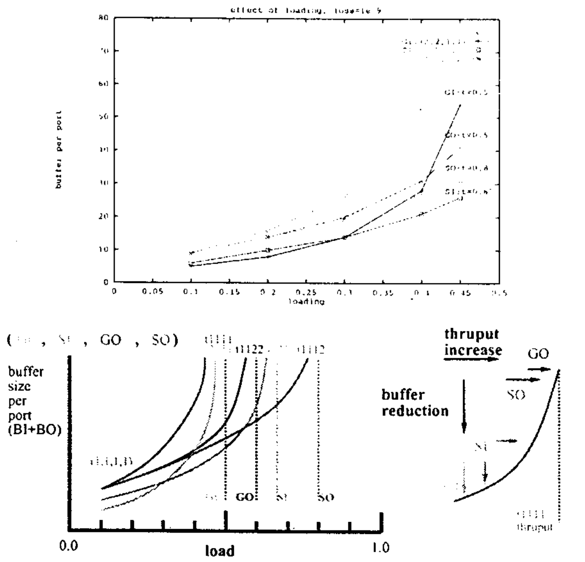


Figure 5: Loading effect on buffer size for each design parameters: upper: analysis result; lower: illustration of the effects on the curve of exponential growing buffer size.

For various configurations, the necessary buffer size to provide certain loss probability under given load can be calculated from Eq. (4) and (5). Here we com-

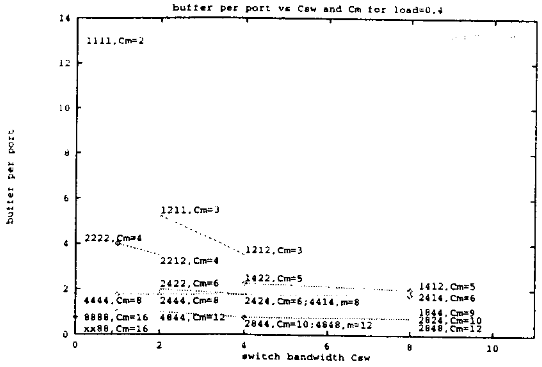


Figure 6: Minimal buffer requirement under memory bandwidth constraint C_m for different switch fabric speed constraint C_{sw} at load=0.4.

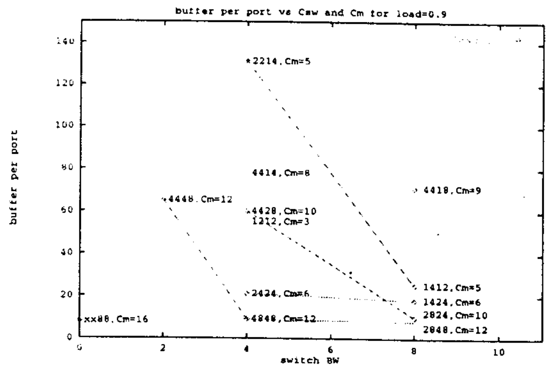


Figure 7: Minimal buffer requirement at load=0.9.

pare the necessary buffer sizes to achieve a loss probability of 10^{-9} among various configurations. The best ones are chosen for certain C_m and C_{sw}. At load=0.4, shown in Fig. 6, the buffer size required by configuration (2,2,2,2) with C_m=4 and C_{sw}=1 is only 29.1% of the baseline switch (1,1,1,1) with C_m=2 and C_{sw}=1. The small slope between the points along the constant-C_m contour indicates that internal speedup is less effective on buffer reduction at load=0.4. This is because output port contention is not so serious at light traffic load. However, at load=0.9, as shown in Fig. 7, internal speedup results in significant buffer reduction. If the speedup is allocated for both input and output, the effect is even more dramatic. As an example, the buffer size required by configuration (1,4,1,2) is only about 20% of the (2,2,1,4) for the same C_m=5.

Based on the effects of design parameters on buffer size reduction and throughput in previous section, and the minimal buffer requirement configurations for different memory bandwidth/speedup constraints, we can conclude the following heuristic resource allocation algorithm which can be used to approach minimal buffer size configuration for a given set of C_m and C_{sw}:

1. allocate C_m for output grouping and moderate output speedup XO,
2. use C_{sw} and the assigned XO to decide the maximum feasible XI, and
3. use C_m and XI to determine the GI.

The reason that the first step allocate C_m to both output grouping and speedup is to avoid over-using the C_{sw} resource since C_{sw} should be used more effectively - for input speedup. After deciding GO, XO, and XI, the input grouping factor can be derived from C_m.

6 Summary

In this paper, a generic packet switch architecture is proposed to model the effects of input grouping, output grouping, input speedup and output speedup on throughput improvement and buffer reduction subject to various technology constraints. Using this model, we conclude that, with back-pressure mechanism, output speedup is most effective in increasing the throughput limit, but has no effect on buffer reduction. Input speedup has a moderate effect on both throughput increase and buffer size reduction. Input grouping is useful for buffer reduction but has no effect on throughput. Output grouping has a moderate effect on throughput increase and negligible effect on buffer reduction. An algorithm to allocate limited memory and fabric bandwidth for minimal buffer requirement switch configuration is proposed based on these results.

The analysis methodology used in this paper can be generalized to other environments. The assumption on the input process can be modified for fixed packet length switches and/or other traffic types. The queueing model can also be applied for asymmetric packet switches with different numbers of input and output links as in [11], or even multi-stage interconnection networks.

References

[1] Hamid Ahmadi and Wolfgang E. Denzel, "A Survey of Modern High-Performance Switching Techniques," IEEE Journal on Selected Areas in Communications, vol. 7, no. 7, 1989.

[2] Yuji Oie, Masayuke Murata, Loji Kubota, and Hideo Miyahara, "Effect of Speedup in Nonblocking Packet Switch," Proc. IEEE ICC'89, pp.410-414, June 1989.

[3] Yuval Tamir, and Hsin-Chou Chi, "Symmetric Crossbar Arbiters for VLSI Communication Switches," IEEE Trans. Parallel and Distributed Systems, vol. 4, no. 1, 1993.

[4] Mark J. Karol, Michael G. Hluchyj, and Samuel P. Morgan, "Input Versus Output Queueing on a Space-Division Packet Switch," IEEE Trans. on Communications, vol. COM-35, no. 12, pp. 1347-1356, Dec. 1987.

[5] Jeane S. -C. Chen and Thomas E. Stern, "Throughput Analysis, Optimal Buffer Allocation, and Traffic Imbalance Study of a Generic Nonblocking Packet Switch," Jour. Selected Areas of Comm., vol. 9, no. 3, pp.439-449, April 1991.

[6] Ilias Iliadis and Wolfgang E. Denzel, "Performance of Packet Switches with Input and Output Queueing," Proc. of ICC'90, pp. 747-753.

[7] Anil K. Gupta and N. D. Georganas, "Analysis of a Packet Switch with Input and Output Buffers and Speed Constraints," Proc. of INFOCOM'90, v.2, pp.694-700.

[8] Arthur Y. M. Lin and John A. Silvester, "The Effect of Switch Speed and Buffer Limitations on the Performance of a Multichannel ATM Switch with Output Queueing," Proc. 1990 Int'l Telecommunication Symposium, pp.483-487.

[9] San-qi Li, "Performance of Trunk Grouping in Packet Switch Design," Proc. of INFOCOM'90, v.2, pp.688-693.

[10] H. Kuwahara, et al., "A Shared Buffer Memory Switch for an ATM Exchange," Proc. ICC, pp. 118-122, 1989.

[11] Soung C. Liew and Kevin W. Lu, "Comparison of Buffering Strategies for Asymmetric Packet Switch Modules," IEEE J. Selected Areas Commun., vol. 9, no. 3, pp. 428-438, April 1991.

[12] E. Gelenbe and G. Pujolle, "Introduction to Queueing Networks," John Wiley & Sons, New York, 1987.