ATM SWITCHING WITH INPUT QUEUEING

E. Del Re, R. Fantacci

Dipartimento di Ingegneria Elettronica, Università di Firenze,
Via S. Marta,3 50139 Firenze, ITALY.

ABSTRACT

This paper deals with an efficient high-speed packet switching in which each packet arrived at an input is stored in one of N possible queues, one for each possible output link. An implementation architecture which permits to share by the N separate queues the same input buffer is considered and studied. An important result shown in the paper is that the proposed multiple input queueing approach outperforms the output queueing approach without requiring a speed-up in the switching operations.

I. INTRODUCTION

The evolution in the field of communications which has been available advanced transmission systems using fiber optics, has led to advanced switching techniques able to handle multimedia traffic. The fast packet switching technique (FPS) seems to be a promising approach to be used in future high-speed networks. To highlight the main characteristics of the fast packet switching technique we note that every type of switch architecture must perform two basic function, i.e. routing and output contention resolution. The packet routing is usually based on hardware techniques by making use of the information contained in the header of each packet (usually called a cell). The solution of the output contention often represents the main source of complexity in the switch architecture [1]-[4]. It occurs whenever two or more packets arriving simultaneously at different switch inputs require to be routed to the same output. Only one of these contending packets achieves routing. In order to avoid loss, queueing is necessary for the others to wait for next routings. The two classic alternatives to queue the unrouted packets are the input queueing and the output queueing. Switching fabrics with input queueing are quite simple in architecture but unfortunately, the maximum possible throughput is bounded by 0.586 because of the head-of-line blocking problem [4]. Switches with output queueing avoid this drawback and achieve optimal delay throughput performance, in particular the maximum possible throughput approaches 1 as the mean arrival rate of packets per slot p approaches 1.

The main problem which arises with output queueing is the requirement of a faster switching fabric to route packets arriving at the switch inputs to the appropriate output buffers within a time slot therefore, by considering the worst case of N packets that simultaneously require to be routed to the same output the switch fabric has to operate N time faster than the input output links. It is evident that this requirement makes it difficult to use switching fabrics with output queueing in high speed networks. This paper deals with the input queueing approach, in particular to avoid the head of line blocking problem each input queue is splitted in N separate queues, one for each possible output link. Any arrival packet is stored in one of these queues according to its destination. Packets at the head of input queues

Work carried out under the financial support of the National Research Council (C.N.R.) in the frame of the Telecommunication Project.

for the same output link are contenders for routing. In this way it is possible to route to the outputs more than one packet for input queue, providing they have different output destinations. In this paper it is assumed that all the packets queued at each input share the same input buffer. In particular, it will be shown later that in this way it is possible to achieve the same throughput-delay performance as the output queueing approach without having to resort to a N time faster switch fabric and to reduce the buffer size requirements.

II THE MULTIPLE INPUT QUEUEING APPROACH

In the switch fabric under consideration whenever a new packet arrives at an input it is stored in the queue associated to its output destination. In particular, we are focusing here on an implementation architecture (Fig. 1) in which all the queued packets at each input share the same buffer. Any new arrived packet is store in the shared input buffer (SB). The routing requests of such packets jointly with the memory locations are broadcasted over the input bus to all the output controllers (Arbiters). By means of routing request (address) filters (AF) a routing request may arrive only at the input of the arbiter corresponding to the desidered output as a routing request can only pass through the filter whose address matches the routing requests destination address. Collisions over the same bus are impossible because at almost one packet may arrived per slot at each input port. Each arbiter handles all the requests according to the First-In-First-Out (FIFO) selection policy. Therefore, a queue, named as destination queue, is formed by each arbiter and updated by placing at its end any new request.

From above, it follows that the fast packet switching is performed here in two stages. In stage one, the routing request, associated with each packet is analyzed while in stage two the packet itself is transmitted into the output link, whenever the associated routing request reaches the head of the appropriate destination queue. It is evident that in our model the routing requests may arrive in batches of random size. Looking to the worst case analysis, we may have to store N routing requests simultaneously within a time slot. However, the problem of a speed up typical of the output queueing implementation architecture, doesn't arise here because the routing requests are formed by few bits.

The performance analysis of the multiple input queueing approach (with shared input buffers) discussed above have been derived by making use of well-known results for discrete-time queueing system [11],[14]-[16]. Let us assume that the arrival processes on the N input links as N independent Bernoulli processes with the probability of an arrival per slot equal to p. Each packet has an equal probability to be addressed to any of the other N output links and successive packets require independent routing. Each destination queue, in its turn, may be modeled as a discrete M/D/1/N/N queueing system. The goodness of this assumption will be verified in that follows by comparing theoretical and simulation results.

By means of the previous considerations it follows immediately that each input queue may be modeled as a discrete M/G/1 queueing system with the service time per packet equal to the total delay spent by the corresponding routing request in the destination queue.

Fixing our attention on a particular input queue, the imbedded Markov chains approach developed for the continuous M/G/1 model is applicable here also to derive the mean total delay per packet. The probability generating function of the number of packets in the input queue, assuming equilibrium, is:

$$Q(z)=\frac{Q_0 A(z)(z-1)}{z-A(z)} \qquad (1)$$

where Q_0 is the probability of having an idle queue and A(z) is the probability generating function of the number of arrivals during the service period of a customer. We have also [14],[17]:

$$A(z)=(1-p+pz)G(1-p+pz) \qquad (2)$$

where G(z) is the probability generating function of the waiting time (normalized to the packet duration time τ) or equivalently in our case the probability generating function of the total time spent by packets waiting for reaching the head of the destination queue.

In deriving an expression for G(z) it must be taken into account that in the considered case, the routing requests may arrive to the appropriate destination queue in batches of random size [14], [16],[17]. We assume that routing requests which arrive at the destination queue at a same instant are served in a random order. However, the routing requests arriving in earlier instants, are served first on the basis of the FIFO discipline. Therefore, the total time spent in the destination queue waiting for service by any routing request is due to the sum of two contributions, e.g. w_1 and w_2. The term w_1 takes into account the time necessary to serve all the routing requests which are waiting in the queue at the arrival instant. The second term w_2, is an additional delay due to the service of the routing requests which arrived at the same instant and were randomly selected to be served first.

Focusing on a destination queue (the tagged destination queue), and assuming that k packets are already waiting for routing, the probability that a routing request (the tagged routing request) arrives in a batch of size i is given by :

$$P(i|k)=\frac{i}{(N-k)\alpha}\binom{N-k}{i}\alpha^i(1-\alpha)^{N-k-i} \qquad (3)$$

where α is the probability of having a packet from one of the input queues requesting routing to the tagged output link.

The probability generating function of the waiting time, on condition that k requests are waiting for routing in the tagged destination queue and that the tagged request arrives in a batch of size i is:

$$G(z|i,k)=\sum_{j=0}^{i-1}\frac{z^{j+k}}{i}=\frac{1-z^i}{i(1-z)}z^k \qquad (4)$$

Therefore, G(z | k) is given by:

$$G(z|k)=\sum_{i=1}^{N-k}G(z|i,k)P(i|k)= \qquad (5)$$

$$\frac{1-(1-\alpha+\alpha z)^{N-k}}{(N-k)\alpha(1-z)}z^k$$

The probability $P_R(k)$ of having k routing requests, $0 \le k \le N-1$, in the destination queue can be obtained numerically by an application of the Markov chain balance equations. Fig. 2 shows an example of the Markov chain to be considered when N=4. The final results is:

$$P_R(1)=P_R(0)\frac{(1-a_{0,0}-a_{0,1})}{a_{1,0}} \qquad (6)$$

$$P_R(k)=\frac{1-a_{k-1,1}}{a_{k,0}}P_R(k_1)-\sum_{i=2}^{k}\frac{a_{k-i,i}}{a_{k,0}}P_r(k-i) \qquad (7)$$

$$2\le k\le N-1$$

with $P_R(0)$ determined in order to verify the following equation:

$$\sum_{k=0}^{N-1}P_R(k)=1 \qquad (8)$$

and the terms $a_{i,j}$ (Fig. 2), given by:

$$a_{i,j}=\binom{N-i}{j}\alpha^j(1-\alpha)^{N-i-j} \qquad (9)$$

Therefore, the probability generating function of the waiting time G(z) can be derived

as a function of α as:

$$G(z)=\sum_{k=0}^{N-1} G(z|k)P_R(k)= \sum_{k=0}^{N-1} \frac{[1-(1-\alpha+\alpha z)^{N-k}]}{(N-k)\alpha(1-z)}P_R(k) \quad (10)$$

In deriving an expression for α we define:

- A_m as the overall number of routing requests arrived at all the destination queues at the mth time slot;
- F_m as the number of free input queues at the mth time slot.

According to our assumptions, an input queue is free at the mth time slot if it is idle or if the packet at its head has been selected to be routed during the (m-1)th time slot. It is evident that an arrival at a destination queue must come only from a free input queue. It follows that:

$$P\{A_m=j\}=\binom{F_m}{j}(N\alpha)^j(1-N\alpha)^{F_m-j} \quad (11)$$

Therefore, the mean number of arrivals is:

$$A_m(F_m)=F_m N\alpha \quad (12)$$

Letting Q_m denote the number of routing requests in the tagged destination queue, we can write:

$$F_m=N-Q_m \quad (13)$$

Therefore, in a steady state condition:

$$\overline{F}=N-\overline{Q} \quad (14)$$

where the mean number of routing requests in the tagged destination queue can be derived as:
By assuming equilibrium we have also:

$$\overline{Q}=\sum_{n=0}^{N-1} nP_R(n) \quad (15)$$

$$(N-\overline{Q})N\alpha=Np \quad (16)$$

Hence:

$$\alpha=\frac{p}{N-\sum_{n=0}^{N-1} nP_R(n)} \quad (17)$$

Eq. (17) defines a non-linear equation in α. Solving this equation numerically it is possible to determine α.

Starting from the previous considerations, it is straightforward to derive the mean delay per packet (normalized to the packet duration time τ) for the FIFO selection policy as:

$$T=1+\sum_{k=0}^{N-1}\frac{(N+k-1)\alpha}{2}P_R(K)+ p\{\frac{\sum_{k-0}^{N-1} k[k-1+\alpha(N-k-1)]P_R(k)}{2N-p[2+\sum_{k-0}^{N-1}(N-k-1)\alpha P_R(k))]}+ \frac{\sum_{k=o}^{N-1}(N-k-1)(N-k-2)\alpha^2 P_R(k)}{3\{2N-p[2+\sum_{k=0}^{N-1}(N+k-1)\alpha P_R(k)]\}}\} \quad (18)$$

Fig. 3 shows T as a function of p for different values of N. It is evident in this figure that the maximum possible throughput approaches 1 as p approaches 1.

Fig. 4 shows T as a function of p for the single queueing on inputs and for the proposed multiple queueing on inputs in comparison with that obtained by using the output queueing

approach [4]-[13]. It is evident in this figure that the proposed multiple queueing on inputs achieves the same performance as the output queueing approach without resorting to a more complex switching fabric [4],[18].

III FINITE BUFFER ANALYSIS

We can't ignore at this point that in any practical implementation, as that sketched in Fig.1, the buffers size is finite. It is evident that in this case packets may be loss. Unfortunately, for the switching system under consideration the analytical evaluation of the packet loss probability leads to a too complex queueing problem to be solved in a closed form. However, an upper bound on the packet loss probability can be derived. The tightness of this upper bound will be verified later by comparing analytical and simulation results.

We start our analysis by considering an input shared buffer of infinite size. Under this assumption, it is evident that the number of packets stored in each input queue is independent of the number of packets stored in the other N-1 queues. Therefore, the overall number of packets stored in the input shared buffer results in the sum of N independent identically distributed (i.i.d.) random variables n_i, denoting the number of packets stored in the queue for output i ($1 \leq i \leq N$). It follows that the probability generating function of such number results to be:

$$B_t(z)=\sum_{k=0}^{\infty} z^k q_t(k)=B^N(z)=[\sum_{k=0}^{\infty} z^k q(k)]^N \quad (19)$$

with B(z) the probability generation function of the number of packets in one of the N separate input queues, q(k) the probability of having k packet in an input queue and $q_t(k)$ the probability of having k packets stored in the shared input buffer. In deriving an expression for B(z) we resort again to the imbedded Markov chain approach. The imbedded Markov chain to be considered is shown in Fig 5 where the instants of service completion has been assumed as the imbedded points. In this figure the terms a_i are defined as:

$$a_i=\sum_{m=1}^{\infty}\binom{m}{i}(p/N)^i(1-p/N)^{m-i}p(m) \quad (20)$$

where p(m) denotes the probability of having for the packet at the head of the queue a service time equal to m (slots). These probabilities can be derived numerically from (2), (10). Fig. 6 shows as an example the probabilities of the service time for the case N=16 and p=0.8. The terms q(k) can be derived as:

$$q(i)=q(0)\frac{(1-a_0-a_1)}{a_1} \quad (21)$$

$$q(k)=\frac{1-a_1}{a_0}q(k-1)-\sum_{i=2}^{k}\frac{a_i}{a_0}q(k-i) \quad (22)$$

$$2 \leq k \leq N-1.$$

with as usual q(0) defined in order to verify the following equation:

$$\sum_{k=0}^{\infty} q(k)=i \quad (23)$$

Therefore, it is possible to derive numerically the probability of having n packet stored in the shared input buffer. It is well known that the packet loss probability P_B for a finite shared buffer size equal to L cells can be upper bounded by the probability of having a number of packets greater than L for the case of an infinite size of the shared buffer. Figs. 7-9 show the derived upper bound in comparison with the packet loss probability achieved by using the output queueing approach as a function of the buffer size (cells) for different values of p. These figures clearly point out that the proposed multiple input queueing approach with an input shared buffer outperforms the output queueing approach. In Fig 10 the derived upper bound analytically derived is compared with the simulation results we obtained for N=16. It is evident in this figure that the derived upper

bound is tight for low values of the packet loss probability, (tail of the distribution) which, typically, are the values of interest.

IV. CONCLUDING REMARKS

In this paper a fast packet switching fabric has been described and analyzed. The presented results clearly show that the multiple input queueing approach with input shared buffers outperforms the output queueing approach in terms of buffer size requirements. An important result is that the same delay-throughput performance as the output queueing can be achieved through the use of the proposed technique, without using a switch fabric which runs N times faster as the input and output links.

It is important to recall that the increasing in the switching fabric speed operations can be avoided by introducing internal parallelism as in the Knockout switch [4]. However, the implementation complexity of the Knockout switch seems to be higher with respect to the proposed multiple input queueing approach with shared buffers. This is principally due to the use by the Knockout concentrator/shifter at each output. The basic function of the Knockout concentrator is to select U packets out of N possible contenders by mean of an algorithm analogous to a knockout tournament. For a concentrator with N inputs and U outputs, there are U rounds of competition. The basic building block of the concentrator is a 2x2 switching element which randomly selects one input packet as the winner and, of course, the other as loser. The losers try to win the next rounds to became winners themself. After the last round the losers are lost. Delay lines must be included in the concentrator to keep the competition synchronous. The shifter permits to store sets of at the most U winners in U separate buffers according with the order of their arrival, therefore implementing the FIFO selection policy. The Knockout concentrator permits to reduce the number of separate buffers keeping low the packet loss probability (packets can be loss in the case of an infinite buffer size if they arrive in batches with more than U elements).

The Knockout switch philosophy permits indeed to save memory but it could be evident that it gives rise to an increase in the implementation complexity of the switch.

We would like to point out, however, that the switch fabric studied in this paper, makes use of a different switching approach. Diversely from the Knockout switch, the packet switching is performed in two stages. In stage one the routing requests associated with the packets at the heads of the input queues are analyzed while in stage two the packets can be transmitted onto the output links. The contention among the routing requests for the same output are handled by the appropriate output controller, which selects the winner slot by slot according to the FIFO selection policy which means that the routing requests are served according with their order of arrival at the output controller. It is evident that the connection requests may arrive in batches of random size. In the worst case we may have to store N routing requests simultaneously within a time slot. However, the problem of a speed up doesn't arise here because the connection requests are formed by few bits (those strictly necessary to identify without ambiguity the output destination and the input queue). Once again, this permits to avoid the use of the knockout concentrator and of the shifter and therefore, a notable reduction in the implementation complexity is achieved.

REFERENCES

[1] J. S. Turner, "Design of an Integrated Services Packet Network", **IEEE J. Select. Areas Commun.**, vol. SAC - 4, pp 1373-1380, Nov. 1986.

[2] J. S. Turner, L. F. Wyatt, "A Packet Network Architecture for Integrated Services", in **Proc. IEEE Globecom '83**, pp. 45- 50, Dec. 1983.

[3] P. Newman, "A Fast Packet Switch for

Integrated Service Backbone Network", **J. Select Areas Commun.**, Vol. Sac - 6, pp. 1468-1479, Dec. 1988.

[4] H. Ahmadi, W. E. Denzel, "A Survey of Nodern High - Performance Switching Techniques", **J. Select. Areas Commun.**, vol. 7, pp. 1091 - 1103, Sept. 1989.

[5] L. Kleinrock, "**Queueing System**", vol. 1, New York, Wiley, 1975.

[6] J. F. Hayes, "**Modeling and Analysis of Computer Communications Networks**", New York, Plenum Press, 1984.

[7] M. Schwartz, "**Telecomunication Network: Protocols, Modeling and Analysis**", Reading, Massachusetts, U.S.A., Addison - Wesley Publishing Company, 1987.

[8] D. Bertsekas, R. Gallager, "**Data Network**", Engleewood Cliffs, New Jersey, Prentice Hall, 1987.

[9] J. L. Hammond, P. J. P. O'Reilly, "**Performance Analysis of Local Computer Networks**", Reading, Massachusetts, U.S.A., Addison - Wesley Publishing Company, 1986.

[10] A. S. Tanenbaum, "**Computer Networks**", Englewood Cliffs, NJ. Prentice - Hall, 1989.

[11] M.J. Karol, M.G. Hluchyj, S.P. Morgan, " Input Versus Output Queueing on a Space-Division Packet Switch", **IEEE Trans. on Commun.**, Vol. COM-35, NO. 12, pp. 1347-1356, Dec. 1987.

[12] J.Y. Hui, E. Arthurs, " A Broadband Packet Switch for Integrated Transport", **IEEE J. Select. Areas Commun.**, Vol.SAC-5, NO. 8, pp.264-1273, Oct. 1987.

[13] M.G. Hluchyj, M.J. Karol, " Queueing in High-Performance Packet Switching", **IEEE J. Select. Areas Commun.**, Vol. SAC-6, NO. 9, pp. 1587-1597, Dec. 1988.

[14] T. Meisling, " Discrete-Time Queueing Theory", **Oper. Res.**, Vol. 6, pp.99-105, Jan-Feb. 1958.

[15] H. Kobayashi, A.G. Konheim, " Queueing Models for Computer Communications System Analysis", **IEEE Trans. on Commun.**,Vol. COM-25, NO. 1, pp. 2-28, Jan. 1977.

[16] P.J. Burke, "Delays in Single-Server Queues with Batch Input", **Oper. Res.**, Vol. 23, pp. 830-833, July-Aug. 1975.

[17] E. Del Re, R. Fantacci, "A Fast Packet Switching Satellite Communication Network", IEEE INFOCOM'91, Miami, Florida, U.S.A., April, 7-8 1991.

[18] J.Y. Hui, "**Switching and Traffic Theory for Integrated Broadband Networks**", Kluwer Accademic Publishers, Norwell, Massachutes, U.S.A., 1990.

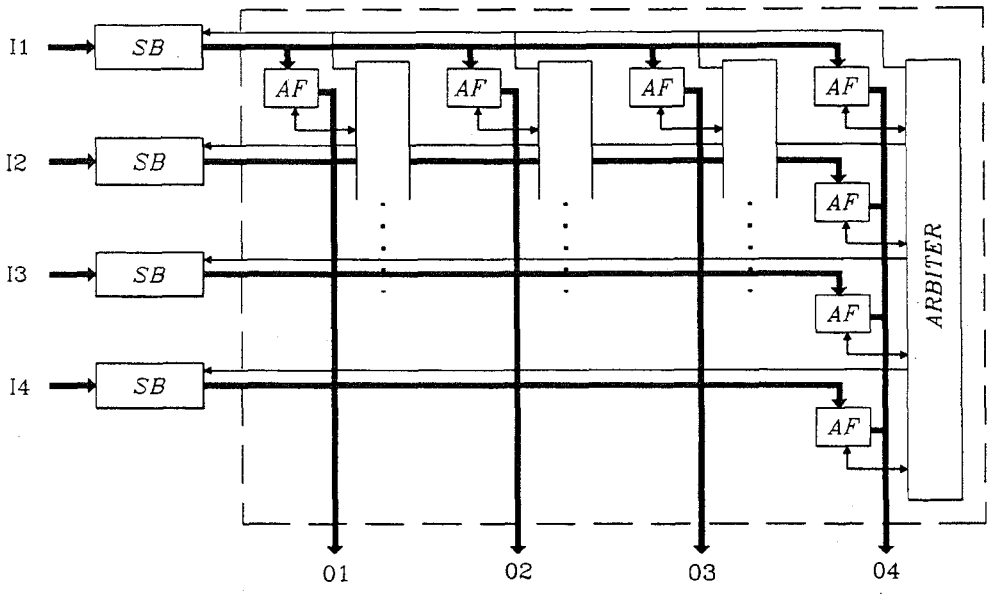

Fig. 1 - The proposed switch fabric architecture (N=4).

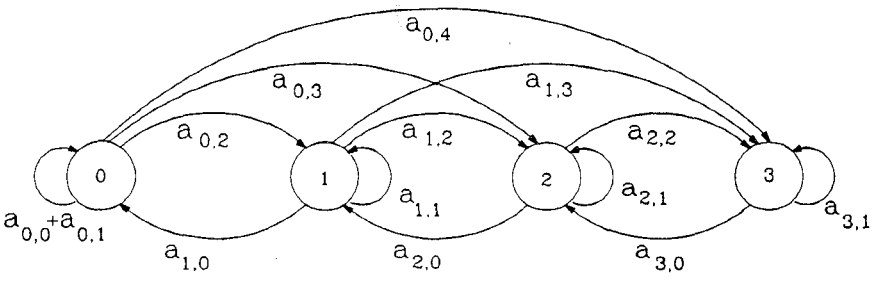

Fig. 2 - The discrete Markov chain state transition diagram for the routing requests queue size (N=4).

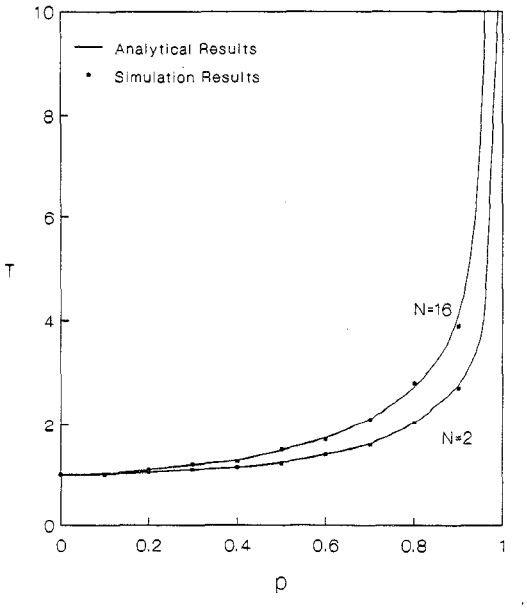

Fig. 3 - The mean normalized total delay.

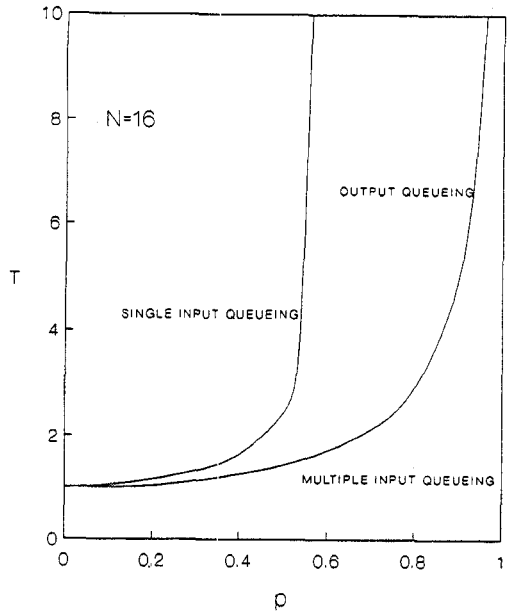

Fig. 4 - Mean Total delay comparison (N=16).

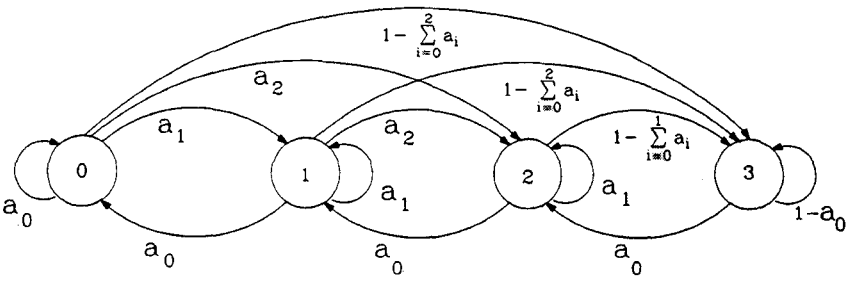

Fig. 5 - The discrete Markov chain transition diagram for the input queue.

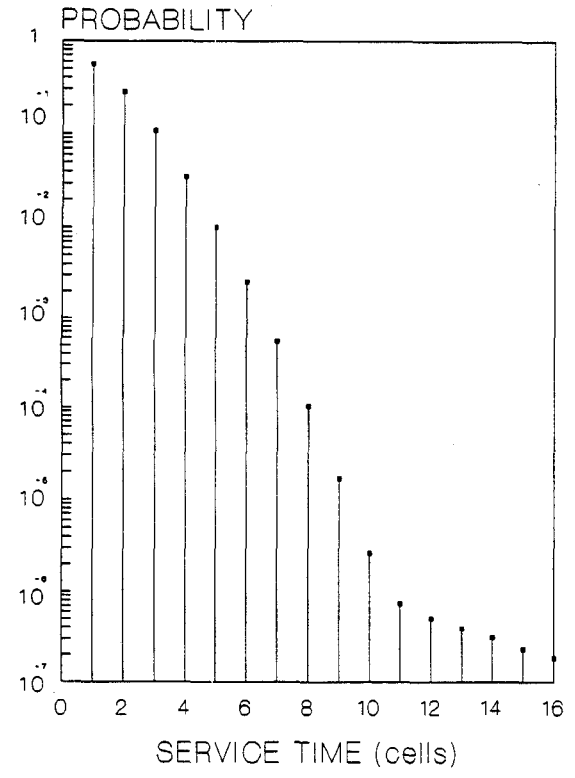

Fig. 6 - Service time probability (N=16;p=0.8).

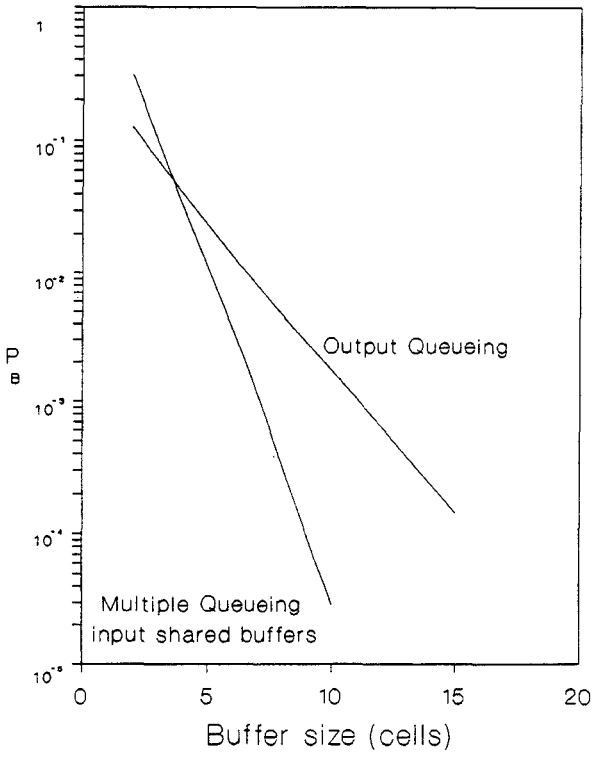

Fig. 7 - Packet loss probability comparison (N=8).

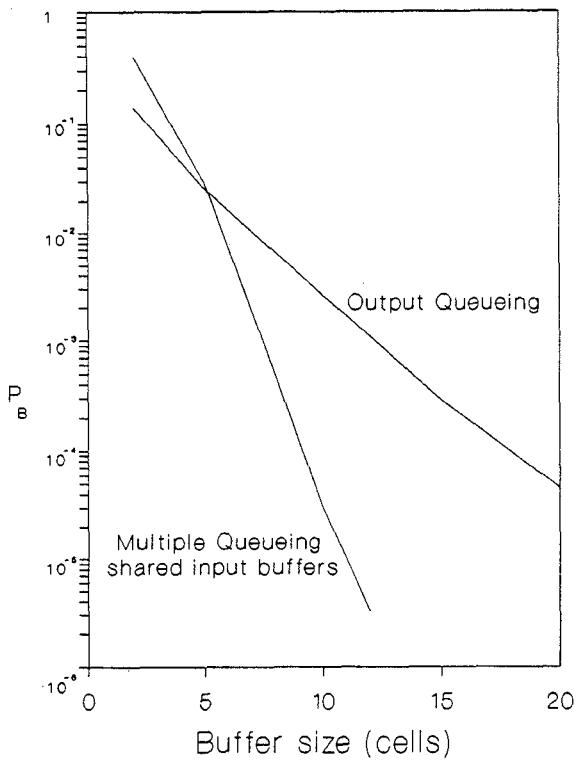

Fig. 8 - **Packet loss probability comparison (N=16).**

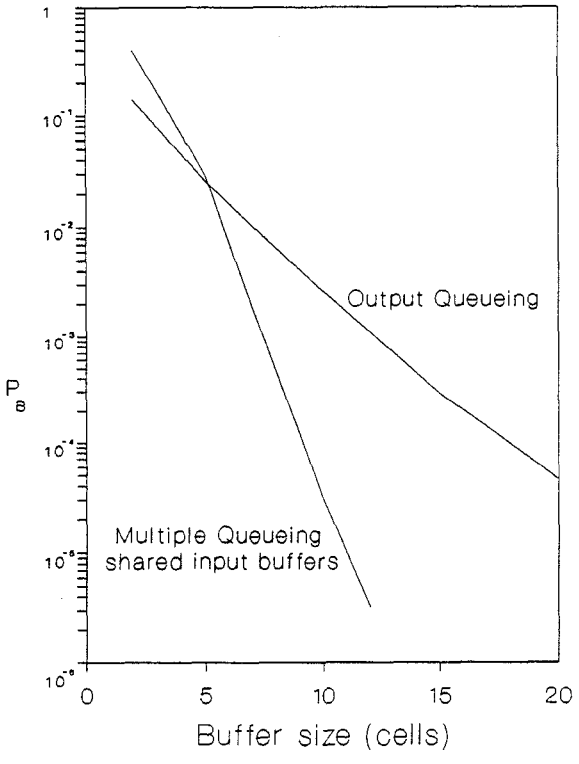

Fig. 9 - **Packet loss probability comparison (N=32).**

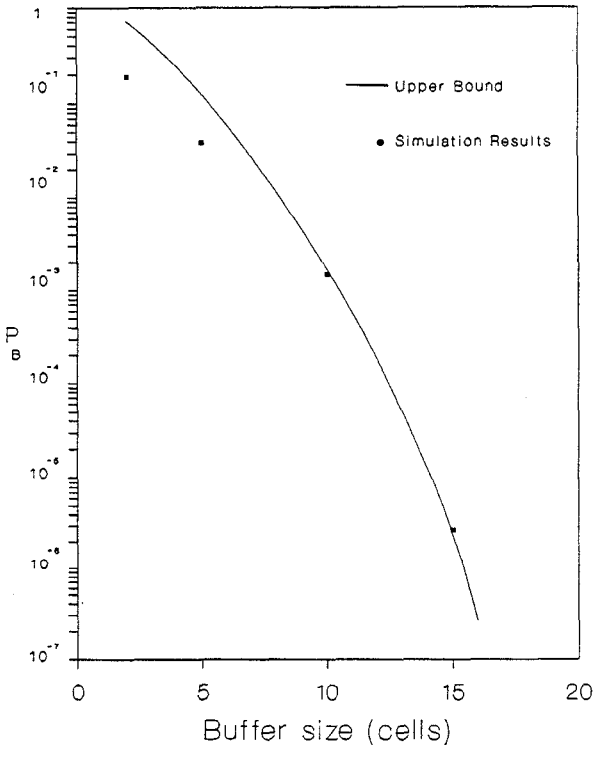

Fig. 10 - **Packet loss probability comparison (N=16; p=0.9).**