Real-Time Packet Switching: A Performance Analysis

ISRAEL CIDON, MEMBER, IEEE, INDER GOPAL, SENIOR MEMBER, IEEE, GEORGE GROVER, AND MOSHE SIDI, SENIOR MEMBER, IEEE

Abstract—**In this paper, we model the internal structure of a packet switching node in a real-time system and characterize the tradeoff between throughput, delay, and packet loss as a function of the buffer size, switching speed, etc. We assume a simple shared single path switch fabric, though the analysis can be generalized to a wider class of switch fabrics. We show that with a small number of buffers the node will provide a guaranteed delay bound for high-priority traffic, a low-average delay for low-priority traffic, no loss of packets at the input and low probability of packet loss at output.**

I. Introduction

REAL-TIME traffic is defined to be traffic with a delivery time bound. In other words, it is traffic which loses its value if delayed by more than a certain time. Examples of such traffic include conversational voice, interactive video, process control, etc. The approach to carrying such traffic in most existing communication systems has been to employ dedicated circuit switched switching techniques. Conventional wisdom has argued that the nodal processing overheads necessary for each packet make it unsuitable for real-time traffic. One of the first works to disprove this conventional wisdom was [1], wherein the basic idea of off-loading the packet switching function onto high-speed specialized hardware was proposed. This idea has been developed further by several groups [2]–[5], and it is now generally accepted that packet switching is an attractive approach to the integrated transport of real-time traffic.

In most real-time packet switching systems, the packet protocols executed in the intermediate nodes are considerably simplified. This simplification is what enables hardware implementation. Typically, flow control and error recovery are performed only at the end-points and the intermediate node performs only the packet routing function. A typical example is the PARIS system described in [3]. The buffering available at the input and output ports is limited and, if congestion occurs causing the buffers to be filled up, the packet is simply discarded. Conventional congestion control techniques which "slow down" input data if queues build up are not employed.

Manuscript received November 24, 1987; revised June 14, 1988. This work was performed while M. Sidi was with IBM Thomas J. Watson Research Center, Yorktown Heights, NY, on sabbatical leave from Technion—Isreal Institute of Technology, Haifa 32000, Israel.

I. Cidon, I. Gopal, and G. Grover are with IBM Thomas J. Watson Research Center, Yorktown Heights, NY 10598.

M. Sidi is with the Department of Electrical Engineering, Technion—Israel Institute of Technology, Haifa 32000, Israel.

IEEE Log Number 8824399.

The limited buffering enforces a strict upper bound on the network delay. This is what ensures real-time delivery of packets. However, the limited buffering also causes packet loss, which must be minimized in order to provide a reasonable quality of service. Since the delays are bounded, the problem in such a packet switched network is to *maximize* throughput while keeping the probability of packet loss to a reasonable value. This is quite different from conventional packet switched networks which attempt to *maximize* throughput subject to constraints on the delay.

In this paper, we model the internal structure of a packet switch node in a real-time system and characterize the tradeoff between throughput and packet loss as a function of the buffer size, switching speed, etc. Other models [4], [5] of such nodes have focused on the actual packet switch fabric, and have attempted to provide a detailed analysis of the probability of blocking, etc., within the fabric. In our model, we are mainly interested in the management and behavior of the input and output packet buffers. We assume a simple shared single path [3], [9] switch fabric (i.e., ring, star, or bus), though the analysis can be generalized to a wider class of switch fabrics. We assume a separate pool of packet buffers at the input and output of each link.

Our analysis shows that for the input and switch fabric stages, a very small buffer pool together with standard buffer management techniques, is sufficient to completely eliminate packet loss and to guarantee a small bounded delay. For the output buffers at a node it is not possible to completely eliminate packet loss, and we examine the impact of different priority schemes on the packet loss and delay.

The paper is structured in the following manner. In Section II, we present the structure of the packet switch node. In Section III, we provide the analysis and the numerical results of the input and switch section and in Section IV, we provide the analysis and the numerical results of the output section.

II. The Node Model

In this section, we describe the various components that are involved in the transport of packets. We refer to these components collectively as the switching subsystem. In addition to these components, a node will have other components (that are not addressed in this study) such as those that implement network control functions, those that interface with external attachments, etc.

The switching subsystem of a node (Fig. 1) is composed of the following three parts.

1) The *switching kernel* performs the basic switching function of the transferring packets from input to output. The input and output adaptors are attached to the switching kernel.

2) The *input adaptors* are attached to incoming links. They receive incoming packets from the link, buffer them as necessary (in memory on the adaptor) and send them through the switching kernel.

3) The *output adaptors* are attached to the outgoing links. They receive packets from the switching kernel, buffer them as necessary, and transmit them over the outgoing link.

In the following, we trace the packet path through a switching node and describe the operation of each of the switching components in some more detail. A packet arrives over the incoming link as a high-speed serial bit stream. In our analysis, we assume that packets are of variable length, typically ranging from 250 to 8000 bits. The input adaptor first has the job to recognizing the packet, performing a serial to parallel transfer, and storing it in its buffers. We assume that all buffers are composed of FIFO queues. Once the input adaptor has stored a complete packet, it signals the switching kernel that it wishes to transmit a packet. No further processing is done at the input adaptor. Specifically, operations such as error recovery, etc., are assumed to be done on an end-to-end basis (if needed).

The method by which the switching kernel transmits a packet from the input buffer is, of course, dependent on its internal structure. In this paper, we assume a single-path [3], [9] architecture composed of a shared broadcast medium such as a star, ring, or a bus. (The results can be generalized to other switching fabrics.) Upon receipt of a full packet, an input adaptor will seek access to the broadcast medium. Only one input adaptor can have access to the medium at any time. We assume that a central arbitrator is responsible for resolving contention for access among the various input adaptors. The arbitration is pipelined with the transmission of previous packets so that no separate arbitration delay is experienced. Several arbitration policies are possible (round robin, fixed priority, etc.) and in the next section, we shall demonstrate the effect of the arbitration policy upon the overall performance.

Once the input adaptor has access to the broadcast medium it transmits one or more complete packets. This transmission is done at the speed of the shared medium which is typically much faster than the incoming link. Packets are of variable size, but we do not permit packets to be fragmented into smaller pieces. This transmission of a packet as a complete entity has the basic advantage

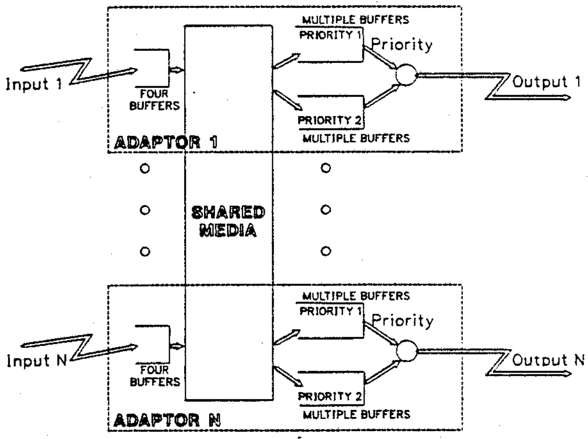

Fig. 1. Schematic of internal structure.

of eliminating contention at the output adaptor. In other words, an output will receive only one packet at a time, and does not have to consider the possibility of simultaneously receiving fragments of packets from two separate inputs. This simplifies considerably the design and analysis of the output adaptor.

We assume that the output adaptor for a packet is identified by an address included in the header of the packet. All output adaptors receive the packets, but only the adaptor identified by the header address copies the packet into its buffers. Note that this output adaptor address could either have been in the arriving packet (for example, if source routing were performed) or could have been generated by the input adaptor through a table lookup.

The output adaptor performs the inverse operation of the input adaptor and transmits the packet on the outgoing serial link. If the buffer space on an adaptor is full, arriving packets are discarded. As will become apparent from later sections, the major queueing point in our system is at the output adaptor. Thus, we investigate the effect of priorities in reducing delays for delay critical traffic. We assume two priority classes with two separate pools or buffers at each output. A nonpreemptive priority is accomplished, whereby the buffer for the lower priority class (Type 2) is served only if the buffer for the higher priority class (Type 1) is empty.

There are several performance parameters of interest in the system, among them delay, throughput, packet loss, etc. In the next two sections, we shall focus on these parameters. We divide our analysis into two parts. The first part deals with the input adaptor and the switching kernel. The second part deals with the queue in the output adaptor and output link.

III. Input and Switch Analysis

This portion of the analysis deals with the packet path from the time it arrives over an incoming link till it is placed in the output buffer.

Ideally, a well-designed switching node will have only a single queueing point in the packet path. This minimizes overall delays and permits more efficient buffer management. The main result of this section is that if the shared broadcast medium is "sufficiently fast" and the arbitration among the adaptors is performed according to a "fair" policy, packet loss can be completely eliminated at the input with minimal buffering requirements. Thus, the delays experienced by a packet in this portion of its path are bounded by a small value, resulting in effectively a single queueing point at the output buffer.

The section takes the following form. We first define various properties of the arbitration policies that we are considering. We then show, for any policy within a large class, upper and lower bounds on the amount of buffering necessary at the input in order to avoid packet loss. We then focus on a specific round robin policy and prove some tighter bounds on this policy.

A. Definitions

Fig. 2 shows the system under consideration. The various input adaptors are attached to the shared broadcast medium and contend for access when they have one or more full packets to transmit. The arbitration function determines which adaptor has access.

We make the following definitions and observations regarding an arbitration policy.

1) Packet Integrality: A policy is packed integral if it considers only complete packets. Thus, an adaptor can ask for access only when it has a complete packet to transmit. Once it has access it must transmit one or more complete packets. Packet fragments cannot be transmitted. As mentioned earlier, packet integrality simplifies the output buffer management.

2) Exhaustive: A policy is exhaustive if, when an adaptor is given permission to transmit, it can always transmit *every* complete packet in its buffer. Nonexhaustive policies that, say, permit an adaptor to transmit a fixed number of packets per transmission opportunity are inherently unfair when variable size packets are considered. This is because, adaptors receiving small packets will not transmit as many bits as adaptors receiving large packets.

3) Work Conserving: A policy is work conserving if the shared medium (which we shall refer to as the server) is never idle if there is a complete packet to transmit at some input adaptor. This property is guaranteed if the arbitration is pipelined with the previous packet transmission.

The remainder of the section is devoted to finding bounds on the amount of buffering necessary at the input adaptor in order to guarantee no packet loss. This buffering bound, in turn, gives a delay bound. Let S_i be the transmission rate of input line i (in bits/s) and S_s be the rate of the shared medium (again, in bits/s). Also, let $S_L = \Sigma_{i=1}^{N} S_i$ be the total of the transmission rates of all input lines, and let $\alpha = S_s/S_L$ be the ratio between the service rate and the total input rate. Finally, we define P to be the maximal length of a packet (in bits).

We consider only systems in which $\alpha \geq 1$, i.e., where the shared medium is fast enough to support the total rate of all the incoming links.

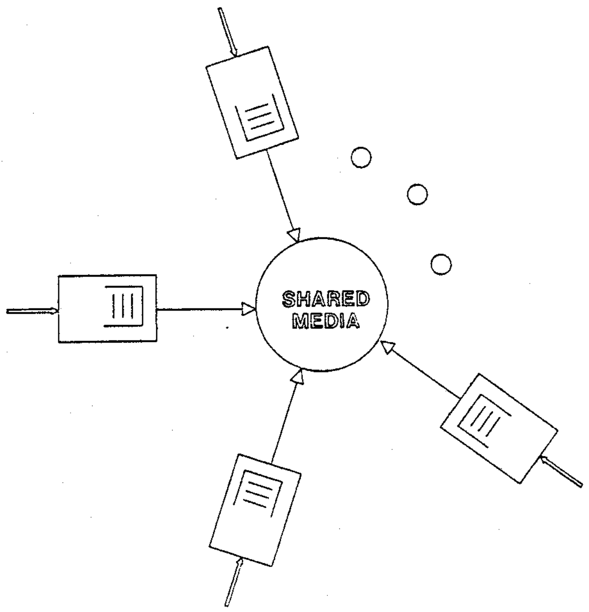

Fig. 2. The model for the input buffers.

B. General Bounds

We now derive some upper and lower bounds on the buffering necessary at the input adaptor in order to avoid packet loss for *any* policy that is work conserving and packet integral.

Theorem 1: For *any* arbitration policy that is work conserving and packet integral,

1) The total number of bits that are queued in *all* link adaptors is bounded by NP.

2) The number of bits queued in adaptor i is bounded by

$$B_i \leq \frac{P((\alpha - 1) + NS_i/S_L)}{\alpha - 1 + S_i/S_L} \quad i = 1, 2, \cdots, N.$$

Proof: Consider the point in time when the switch changes from the idle to the busy state. At that point each link input queue might contain no more than exactly one packet of the longest length P.

Consequently, the sum of the lengths of all the queues is bounded by NP. As the server is no longer idle this number can grow no further, the server will remove bits from the input buffer set at least as fast as they can be added by all links combined. Because of this speed ratio, it is only possible for the total number of bits in the buffer set to increase when the server is idle (recall that the server can only idle when no complete packet is present in *any* of the link adaptor buffers). It is therefore apparent that NP is an upper bound (and, since it is achievable, a least upper bound) for the total number of bits in the buffer set. This completes the proof of 1).

Let $T_{\max}$ be the time between the end of the last idle period until buffer i is served. We will bound the value of

$T_{\max}$. In the period $T_{\max}$, the switch will serve exactly $T_{\max} \times S_s$ bits. Consider the time when the switch moves from the idle state to the busy state. At this time, each link adaptor can have at most one packet in its input buffer. This corresponds to a total maximum of NP bits in the system. Assume that K_i bits are contained in the ith buffer at the beginning of this time interval. In the time interval $T_{\max}$ the number of bits transmitted can be no larger than the number of bits that were present at all other buffers before the period began plus the number of bits that arrive to these buffers during the period. Thus, we have

$$T_{\max} S_s \leq NP - K_i + T_{\max}(S_L - S_i) \tag{1}$$

or

$$T_{\max} \leq \frac{NP - K_i}{S_s - S_L + S_i}. \tag{2}$$

The number of bits in the ith queue is thus bounded by the sum of the number of initial bits K_i and the number of bits that arrive to that adaptor until it is being served.

$$\begin{aligned} B_i &\leq K_i + T_{\max} S_i \\ &\leq \frac{K_i(S_s - S_L + S_i) + NPS_i - K_i S_i}{S_s - S_L + S_i} \\ &= \frac{K_i(S_s - S_L) + NPS_i}{S_s - S_L + S_i}. \end{aligned} \tag{3}$$

Since $K_i \leq P$,

$$B_i \leq \frac{P((S_s - S_L) + NS_i)}{S_s - S_L + S_i} \tag{4}$$

$$B_i \leq \frac{P((\alpha - 1) + NS_i/S_L)}{\alpha - 1 + S_i/S_L}. \tag{5}$$

Equation (5) proves the second part of theorem 1. ■

Assume that, $S_i = S_L/N$, then

$$B \leq \frac{P}{1 - (N-1)/N\alpha}. \tag{6}$$

Since B is strictly increasing with N taking N to infinity makes the bound independent of N.

$$B \leq \frac{P}{1 - 1/\alpha}. \tag{7}$$

Theorem 2: For *any* arbitration policy that is work conserving and packet integral, if the input link speeds are all equal and $\alpha = 1$, then the buffering needed to avoid packet loss is at least $2P$ (for each link adaptor).

Proof: We will examine the worst case environment where the speed of the switching node server is just equal to the sum of the speeds of all the incoming links. This is as slow as the server can operate and still have a buffer requirement which is bounded. Next, consider the situation when the switch changes from the idle to the busy state at a point when each link input queue contains exactly one packet of the longest length P. Before the last of these N packets is served, all of the others (at least) comprising a total of $(N-1)P$ bits, have been served. While this service is in process, the link adaptor associated with the last packet can obtain additional input, whose magnitude, since the adaptor operates at a speed exactly one Nth of that of the server, may be as large as

$$\frac{(N-1)P}{N}.$$

Since one packet of length P existed initially, the required buffer size B is therefore such that

$$B \geq \left(1 + \frac{N-1}{N}\right)P. \tag{8}$$

And since

$$\lim_{N \to \infty} \left(1 + \frac{N-1}{N}\right) = 2$$

it follows that

$$B \geq 2P. \tag{9}$$

C. The Round Robin Exhaustive Policy

In this section, we examine a specific policy (the RR policy) in some detail. The policy is work conserving, packet integral and exhaustive. It is based on round robin scheduling, i.e., the links are arranged in some prearranged cyclic order and all complete packets in a link adaptor's input buffer are transmitted before proceeding to the next link adaptor in the cyclic order. If the last packet in the adaptor buffer is still incomplete when all earlier packets in the buffer have been served, the partial packet is left for the next turn. If a link adaptor does not have any packets to transmit when its turn arrives, it is bypassed.

As the RR policy is a work conserving policy, it is apparent from Theorem 1 that NP is an upper bound on the total number of bits in *all* link adaptor buffers. When a particular link queue is served, the only bits remaining after service is completed can be those of an incomplete packet, and the length of that queue is therefore less than P. This bound remains valid as long as the server idle. The size of this adaptor queue, just before it is served in the next turn, will thus be less than P plus the maximum length to which a queue can grow between two successive service turns. When the link is being served the length of its queue decreases and all full packets are served. At the end of the service there will be again less than P bits in its buffer. If the amount by which a queue may grow between two successive service turns can be bounded, then an upper bound on the length of the queue is obtained.

Assume that a specific link adaptor (without loss of generality, we shall say link number N) has just completed service or that the system has just transited from idle to busy. We will call the time it takes until the system serves link N again to be *an iteration*. We will bound the length of a single iteration, i.e., the time it takes to serve links 1 through $N - 1$ before link N can be served. The results are proven for a system with all link speeds equal. The results for different link speeds are similar but the bookkeeping involved in their derivation is more complex. These results are summarized at the end of this section.

Assume that at the beginning of the iteration, there are K_i, $i = 1, 2, \cdots, N$ bits buffered at the ith link. Define X as the maximal amount by which the Nth queue may grow during this iteration.

Lemma 1: If the RR policy is employed and the number of bits queued at adaptor i, $i = 1, 2, \cdots, N$ at the beginning of the iteration is K_i, then

$$X = K_N + \frac{1}{N\alpha} \sum_{n=1}^{N-1} K_n \left(1 + \frac{1}{N\alpha - 1}\right)^{N-n}.$$

Proof: Define the maximum time it takes for the system to serve all links up to (including) link $n < N$ as T_n. It is clear that the number of bits that might be buffered at the Nth queue are the sum of the initial number of bits K_N plus the number of bits that can arrive to that link over a time period of T_{N-1}.

We define S_l to be the link speed in the system ($S_l = S_L/N$). Define $\gamma = S_l/S_s = 1/\alpha N$, and $\beta = 1/(1 - \gamma)$. We start by computing T_1. The maximum time it takes to completely serve queue 1 is the time it takes to transmit the initial K_1 bits plus the time it takes to transmit the bits that have arrived during the initial period of transmission plus those bits that have arrived during the second transmission, etc. For the worst case, we assume that the link is always receiving at the full speed.

$$T_1 = \frac{K_1}{S_s} + \frac{K_1}{S_s}\left(\frac{S_l}{S_s}\right) + \frac{K_1}{S_s}\left(\frac{S_l}{S_s}\right)^2 + \frac{K_1}{S_s}\left(\frac{S_l}{S_s}\right)^3 + \cdots$$
$$= \frac{K_1}{S_s} \sum_{n=0}^{\infty} \left(\frac{S_l}{S_s}\right)^n = \frac{K_1}{S_s}\left(\frac{1}{1-\gamma}\right) = \frac{K_1}{S_s}\beta. \tag{10}$$

To calculate T_2 have to add to T_1 the time it takes to serve the second queue. In order to do so, we have to add the initial number of bits K_2 to the maximal number of bits that may join this queue during the time interval T_1.

$$T_2 = \left(T_1 + \frac{K_2}{S_s}\right) \sum_{n=0}^{\infty} \left(\frac{S_l}{S_s}\right)^n = \left(T_1 + \frac{K_2}{S_s}\right)\beta$$
$$= \frac{K_1}{S_s}\beta^2 + \frac{K_2}{S_s}\beta. \tag{11}$$

Similarly,

$$T_3 = \left(T_2 + \frac{K_3}{S_s}\right) \sum_{n=0}^{\infty} \gamma^n$$
$$= \frac{K_1}{S_s}\beta^3 + \frac{K_2}{S_s}\beta^2 + \frac{K_3}{S_s}\beta. \tag{12}$$

And finally,

$$T_{N-1} = \left(T_{N-1} + \frac{K_{N-1}}{S_s}\right) \sum_{n=0}^{\infty} \gamma^n$$
$$= \frac{K_1}{S_s}\beta^{N-1} + \frac{K_2}{S_s}\beta^{N-2} + \cdots + \frac{K_{N-1}}{S_s}\beta$$
$$= \frac{1}{S_s} \sum_{n=0}^{N-1} K_n \beta^{N-n}. \tag{13}$$

The total number of bits in the Nth queue can be expressed as $K_N + S_l T_{n-1}$, which is the initial number of bits in the queue plus those bits that can join the queue during the first iteration. This serves as an upper bound to the queue length at the end of the iteration when the distribution of bits at all adaptors is given by $K_1, K_2, \cdots, K_N$.

$$X = K_N + T_{N-1}S_l = K_N + \frac{S_l}{S_s} \sum_{n=1}^{N-1} K_n \beta^{N-n}$$
$$= K_N + \gamma \sum_{n=1}^{N-1} K_n \left(\frac{1}{1-\gamma}\right)^{N-n}. \tag{14a}$$

Using $\alpha = 1/N\gamma$

$$X = K_N + \frac{1}{N\alpha} \sum_{n=1}^{N-1} K_n \left(1 + \frac{1}{N\alpha - 1}\right)^{N-n}. \tag{14b}$$

■

This completes the proof of lemma 1.

Theorem 3: Under the RR policy for any N the maximal number of bits that may be queued in any adaptor is bounded by

$$B_0 = P\left(\frac{e^{1/\alpha}}{\alpha} + 1\right).$$

Proof: The bound B_0 is derived by dividing the total NP bits into $K_1, \cdots, K_N$ such that X is maximized.

$$B_0 = \max_{K_N \le P, \sum_{n=1}^{N} K_n \le NP} \{X\}. \tag{15a}$$

From (14b), since $\beta = 1 + 1/N\alpha - 1 > 1$, it is clear that X is increased by moving one or more bits from K_j to K_i as long as $i < j < N$. This is since K_i is multiplied by $\beta^{N-i} > \beta^{N-j}$. Consequently, we have

$$B_0 = \max_{K_N \le P, K_1 + K_N \le NP} \cdot \left\{K_N + \frac{K_1}{\alpha N}\left(1 + \frac{1}{N\alpha - 1}\right)^{N-1}\right\}. \tag{15b}$$

In order to complete the bound we just have to divide the NP bits between the first and the Nth queue such that $K_N \le P$ and $K_N + K_1 \le NP$. $K_N = P$ as long as the coefficient of K_1 in (15b) is less then 1. It is easy to see that since $\alpha \ge 1$

$$\frac{1}{\alpha N}\left(1 + \frac{1}{N\alpha - 1}\right)^{N-1} \le \frac{1}{N}\left(1 + \frac{1}{N-1}\right)^{N-1} \le \frac{e}{N}. \tag{16}$$

Clearly, since $e < 3$, for any $N \geq 3$ the coefficient of K_1 in (15b) is less then 1. From (15b), equality holds for $N = 2$. Consequently, in the worst case there are P bits in the last queue and $(N - 1)P$ is the first for all N. Therefore,

$$B_0 = P + \frac{(N-1)P}{\alpha N}\left(1 + \frac{1}{N\alpha - 1}\right)^{N-1}. \quad (17)$$

B_0 is always increasing with N, therefore, for all N's

$$B_0 \leq \lim_{n\to\infty} P + \frac{(N-1)P}{\alpha N}\left(1 + \frac{1}{N\alpha - 1}\right)^{N-1}$$
$$= P\left(1 + \lim_{n\to\infty}\left(1 + \frac{1}{N\alpha - 1}\right)^{N-1}\right). \quad (18)$$

The evaluation of this limit is simplified by a change of variables. Let us set $\alpha N - 1 = \alpha M$ so that $N = M + 1/\alpha$. Then

$$\lim_{N\to\infty}\left(1 + \frac{1}{\alpha N - 1}\right)^{N-1} = e^{1/\alpha}. \quad (19)$$

To conclude, substituting the result of (19) back into (18), the upper bound is

$$B_0 = P\left(\frac{e^{1/\alpha}}{\alpha} + 1\right). \quad (20)$$

This completes the proof of Theorem 3. ■

We now return to improving still further the upper bound B_0 of Theorem 3. To facilitate the presentation, we consider only the case where $\alpha = 1$, i.e., the switch speed is equal to the sum of the link speeds. However, the same approach can be carried also for any $\alpha > 1$.

The calculations up to now have estimated a first-order upper bound on the maximal number of bits that can be stored at any queue. To calculate this bound, we have used a worst case scenario where $(N - 1)P$ bits are stored in the first queue. However, once a new upper bound B_j is found, we can use this upper bound to tighten the possible worst case scenario.

$$B_{j+1} = \max_{K_N \leq P, \sum_{n=1}^{N} K_n \leq NP, K_n \leq B_j, n = 1,2,\cdots,N} \{X\}.$$

This leads to a sequence of upper bounds each of which improves the previous one. We focus on systems with $N \geq 4$ since for $N = 1, 2, 3$ from (17) $B_0 \geq (N - 1)P$ and thus the bound of Theorem 3 cannot be further improved. In addition, we try to get a simpler view of the bounds by exploring their asymptotic behavior as $N \to \infty$.

Theorem 4: For the RR discipline under the condition of theorem 3 and for $\alpha = 1$, the maximal number of bits that may be queued in any adaptor is bounded by 3.3502P.

Proof: We turn to the basic equation (14b). By letting $\alpha = 1$

$$X = K_N + \frac{1}{N}\sum_{n=1}^{N-1} K_n\left(1 + \frac{1}{N-1}\right)^{N-n}. \quad (21)$$

Using the same rationale by which a maximal number of bits was placed in the first queue to derive B_0, we now place B_0 bits in the first queue, the second queue, etc., until the smallest number k_0 is reached such that $k_0 B_0 \leq (N - 1)p$. Queues $k_0 + 1$ through $N - 1$ contain no bits, and queue N contains P bits.

These conditions lead to the establishment of a new upper bound $B_1 \leq B_0$. B_1 can in turn be used to generate a succeeding upper bound $B_2 \leq B_1$, and so on. A monotonically decreasing sequence of upper bounds is thus generated, which, being bounded from below (for example, by 0), must converge to a final bound B, which is a least upper bound with respect to this sequence of upper bounds for the RR policy as formulated under Theorem 4.

Let B_j be the jth upper bound in the sequence, expressed in bits.

Let l_j be the smallest integer such that $B_j l_j \leq (N - 1)P$.

Queues 1 through $l_j - 1$ will thus have B_j bits each, but queue l_j may have some number of bits less than B_j. We now add an additional number of bits δ_j to queue l_j, an amount just sufficient so that queue l_j too has B_j bits. Then

$$l_j = \frac{(N+1)P}{B_j} + \frac{\delta_j}{B_j}. \quad (22)$$

It might be conjectured that the fact that these additional bit distributions may differ from a total of $(N - 1)P$ because of the δ_j factors could cause a greater number of bits to be served, casting doubt on whether the sequence $\{B_j\}$ is indeed monotonically decreasing. However, it will be seen in the calculation that the term δ_j is not significant for large values N. From (21) by setting $K_n = B_j n \leq l_j$, we get

$$B_{j+1} = P + \frac{B_j}{N}\sum_{n=1}^{l_j}\left(1 + \frac{1}{N-1}\right)^{N-n}. \quad (23)$$

By using $\beta = (1 + 1/N - 1)$, we get

$$B_{j+1} = P + \frac{B_j \beta^{N-1}}{N}\,\frac{1 - \beta^{-l_j}}{1 - \beta^{-1}}. \quad (24)$$

Since $1 - \beta^{-1} = 1/N$, we get

$$B_{j+1} = P + B_j \beta^{N-1}(1 - \beta^{-l_j}). \quad (25)$$

Using (22)

$$B_{j+1} = P + B_j \beta^{N-1}(1 - \beta^{(N-1)P/B_j}\beta^{-\delta_j/B_j}). \quad (26)$$

By taking N to infinity, we get

$$B_{j+1} = P + B_j e(1 - e^{P/B_j}). \quad (27)$$

Equation (27) thus provides a simple recursive relation for B_{j+1} in terms of B_j and P. To eliminate P from this expression, it is only necessary to define a sequence of coefficients

$$C_j = \frac{B_j}{P}. \quad (28)$$

Equation (27) is then transformed to

$$C_{j+1} = 1 + C_j e(1 - e^{-1/C_j}). \qquad (29)$$

Using the recursion relation of (21) and by letting both sides of the equation to go to infinity we get

$$C = 1 + Ce(1 - e^{-1/C}) \qquad (30)$$

or

$$C(1 - e + e^{(C-1/C)}) - 1 = 0. \qquad (31)$$

It is possible to compute the convergence value C from (31) by numerically solving the nonlinear equation.

$$C = \lim_{j \to \infty} \{C_j\} = 3.3501. \qquad (32)$$

This completes the proof of Theorem 4. ■

The last Theorem of this section bounds the number of buffers for the case of different link speeds. The method of proving this bound is similar to the way we derived the first bound for the symmetrical case (for $j = 0$). One can improve the bound by employing similar techniques used for the symmetrical case.

Theorem 5: Under the RR policy and with N links of speeds $S_1, S_2, \cdots, S_N$, the maximal number of bits that may be queued by the Nth adaptor is bounded by B^N where

$$B^N \le P\left(1 + \frac{N\gamma_N}{\prod_{i=1}^{N-1} (1 - \gamma_i)}\right).$$

Where $\gamma_i = S_i/S_s$.

Proof: We define $\beta_i = 1/(1 - \gamma_i)$. Following (10), it is clear that

$$T_1 = \frac{K_1}{S_S}\beta_1. \qquad (33)$$

Following (11)–(13), we get

$$T_{N-1} = \sum_{j=1}^{N-1} \frac{K_j}{S_S} \prod_{i=j}^{N-1} \beta_i. \qquad (34)$$

The number of bits that can be accumulated at the Nth adaptor is thus

$$X_N = K_N + S_N T_{N-1} = K_N + S_N \sum_{j=1}^{N-1} \frac{K_j}{S_S} \prod_{i=j}^{N-1} \beta_i. \qquad (35)$$

Since, for all k, $\beta_k > 1$, it is clear that X_N always increases by moving one or more bits from K_j to K_i as long as $i < j < N$. Consequently, we have

$$B^N = \max_{\{K_N \le P, K_1 + K_N \le NPK_N + S_N\}} \frac{K_1}{S_S} \prod_{i=1}^{N-1} \beta_i. \qquad (36)$$

Since $K_N \le P$ and $K_1 \le NP$ we have

$$B^N \le P\left(1 + N\gamma_N \prod_{i=1}^{N-1} \beta_i\right). \qquad (37)$$

Since $\beta_i = 1/(1 - \gamma_i)$, this proves Theorem 5. ■

The above results provide bounds on buffering for the RR and an arbitrary work conserving policy. We note that a simple application of Little's Theorem [6] can give the corresponding bounds on delay. For input adaptor i, if the buffer bound is B and the line speed is S_i, the corresponding delay bound is B/S_i.

In the following, we provide some numerical results on the delay and buffering bounds for the symmetrical case. We also compare the bounds on the RR policy to the bounds on an arbitrary work conserving policy.

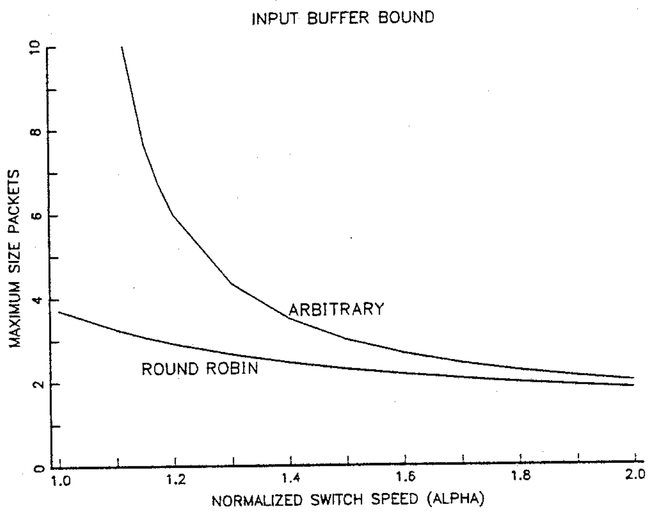

Fig. 3. Input buffering bounds.

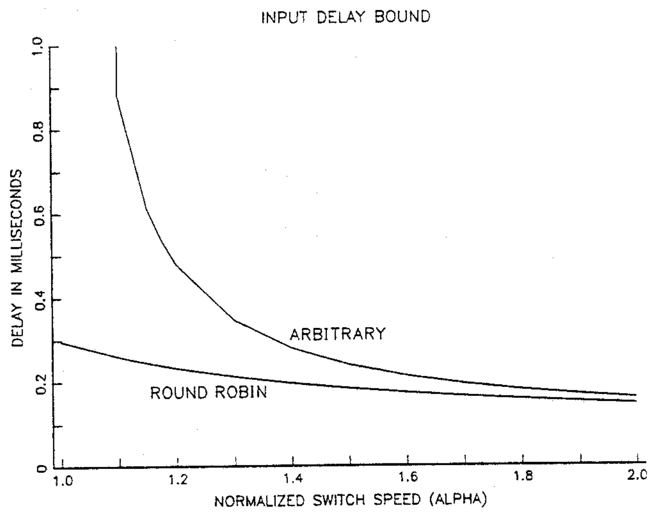

Fig. 4. Input delay bounds.

D. Numerical Results

Fig. 3 plots the bound on the input buffer size (in maximum size packets) against α (as defined in Section III). We compare the RR policy and an arbitrary work conserving, packet integral policy. We assume that N is large and use the asymptotic results of Section III. As can be seen, the differences are substantial at low values of α. For a 100 Mbit/s link, and for 8 Kbits maximum packet size, the corresponding input delay bound is plotted in Fig. 4. Again, for low α the differences are large. Notice that for the RR policy, even with $\alpha = 1$ the delays are of the order of a few microseconds.

IV. Output Analysis

At the output it is not possible to provide a bound on the buffering and guarantee no packet loss. This is because the rate of the outgoing link is typically not adequate to serve the maximum rate of packets arriving to it from the switching kernel. We therefore rely on probabilistic arguments.

To facilitate the analysis of the system, we employ a queueing model that has worse performance than the actual system. However, it is exact enough for most purposes and approximates quite well the behavior of the real system. We base our model on a $M/M/1$ queue with two types of traffic with a nonpreemptive priority given to type 1 over type 2. These two types may correspond to, for example, real-time traffic and delayable traffic.

The exponential service assumption is reasonable as we permit the transmission of variable size packets through the node. However, if voice traffic using a constant packet size dominated, then the exponential model is more of an approximation. It is a pessimistic approximation since it is well known that the $M/D/1$ system outperforms the $M/M/1$ system [6]. The Poisson arrival assumption used by our model is also a pessimistic assumption since real time traffic is typically periodic. Moreover, part of the traffic is already ordered, in sequence, over the incoming links. It is demonstrated in [7]. that the performance of a deterministic server with periodic arrivals is better than with Poisson arrivals. However, the $M/D/1$ serves as a good approximation when the number of calls is large.

Thus, the underlying queueing model for the output buffers is that of a single server system (the output link) with two arrival streams (Type 1 and Type 2) where one stream (the Type 1) has nonpreemptive priority over the other stream (the Type 2). Let the arrivals of packets of both the Type 2 and Type 1 traffic be Poisson with rates λ_1 and λ_2, respectively. Let the transmission time of a packet be exponentially distributed with mean $1/\mu$. Let $\rho_1 \triangleq \lambda_1/\mu$, $\rho_2 \triangleq \lambda_2/\mu$ and $\rho \triangleq \rho_1 + \rho_2$.

In this section, we compute the fraction of lost packets of each type, due to the finite number of output buffers available for the packets at the node. The finite number of buffers then provides a bound on the delay at the output link. The "brute-force" approach for solving this problem is presented in the Appendix. This approach requires the solution of a finite, but rather large system of linear equations ($2(M + 1)(N + 1) - 1$ equations where M and N are the maximum number of packets of Type 1 and Type 2, respectively, that a node can accommodate). If M and N are too large, the brute-force approach becomes impractical. We therefore introduce a simple computational method that yields both upper bounds on and approximations for the loss probabilities. The method is based on computing the occupancy probabilities of a system that has an infinite number of buffers.

We start with the following.

Let N be a random variable that represents the number of packets in a queueing system in steady state. Let $g_n = \text{Prob}(N = n)$, $n = 0, 1, 2 \cdots$ and let $G(Z)$ be the generating function of the number of packets

$$G(Z) = \sum_{n=0}^{\infty} g_n Z^n \qquad |Z| < 1. \tag{38}$$

Assume that $G(Z)$ is given by the quotient of two functions $A(Z)$ and $B(Z)$

$$G(Z) = \frac{A(Z)}{B(Z)} \tag{39}$$

and assume that the coefficients $a_i, b_i, i = 1, 2, \cdots, n$ of the Taylor expansion are known for both $A(Z)$ and $B(Z)$

$$A(Z) = \sum_{i=0}^{\infty} q_i Z^i; \quad B(Z) = \sum_{i=0}^{\infty} b_i Z^i. \tag{40}$$

Then, if $b_0 > 0$ it is easy to see that g_n, $n = 0, 1, 2, \cdots$ are computed via the following recursion:

$$g_0 = a_0/b_0; \; g_n = \left(a_n - \sum_{i=0}^{n-1} b_{n-i} g_i\right)\Big/ b_0$$
$$n = 0, 1, 2, \cdots. \tag{41}$$

Therefore, in order to compute g_n for some n, we only need to be able to compute $a_i, b_i, i = 0, 1, \cdots, n$, namely, to determine the first $n + 1$ coefficients of the Taylor series of $A(Z)$ and $B(Z)$ in (40). In the sequel, we show how to compute these coefficients (again in a recursive manner) for a nonpriority queueing system that has an infinite number of buffers.

A. High-Priority Packets

From [8] we have that the generating function of the occupancy probabilities of high-priority packets in a nonpreemptive queueing system with infinite buffers is

$$G(Z) = \frac{p_0[\mu + \lambda_1(1 - Z)] + \lambda_2}{(1 - \rho_1 Z)[\mu + \lambda_1(1 - Z)]} \tag{42}$$

where $p_0 = 1 - \rho_1 - \rho_2$. So in this case,

$$a_0 = p_0(\mu + \lambda_1) + \lambda_2; \quad a_1 = -\lambda_1 p_0; \quad a_i = 0 \quad i \geq 2$$
$$b_0 = \mu + \lambda_1; \quad b_1 = -\rho_1(\mu + \lambda_1) - \lambda_1;$$
$$b_2 = \lambda_1 \rho_1; \quad b_i = 0 \quad i \geq 3 \tag{43}$$

and now we apply recursion (41) to obtain g_n, the steady-state probability of having n high-priority customers.

B. Low-Priority Packets

From [8] we have that the generating function of the occupancy probabilities of low-priority packets in a nonpreemptive queueing system with infinite buffers is

$$G(Z) = \frac{p_0[\lambda_1 + \lambda_2(1 - Z) + \mu - \mu\alpha(Z)]}{[\lambda_2(1 - Z) + \mu][1 - \alpha(Z) - \rho_2 Z]} \tag{44a}$$

where

$$\alpha(Z) = \frac{1}{2\mu}\Big(\mu + \lambda_1 + \lambda_2 - \lambda_2 Z - \sqrt{(\mu + \lambda_1 + \lambda_2 - \lambda_2 Z)^2 - 4\lambda_1\mu}\Big). \quad (44b)$$

If we express $G(Z)$ as a quotient $A(Z)/B(Z)$, then we have

$$A(Z) = p_0; \quad B(Z) = 1 - \alpha(Z) - \rho_2 Z \quad (45)$$

and in order to use the recursive method of (41) we have to compute $a_i, b_i, i = 0, 1, 2, \cdots, n$. To that end, we first expand $\beta(Z) = \sqrt{(\mu + \lambda_1 + \lambda_2 - \lambda_2 Z)^2 - 4\lambda_1\mu}$ as a Taylor series $\beta(Z) = \Sigma_{i=0}^{\infty} \beta_i Z^i$ and determine $\beta_i, i = 0, 1, 2, \cdots, n$. We do this again in a recursive manner by using the relation

$$\left(\sum_{i=0}^{\infty} \beta_i Z^i\right)^2 = (\mu + \lambda_1 + \lambda_2 - \lambda_2 Z)^2 - 4\lambda_1\mu. \quad (46)$$

Consequently, we find

$$\beta_0 = \sqrt{(\mu + \lambda_1 + \lambda_2)^2 - 4\lambda_1\mu};$$

$$\beta_1 = -\frac{\lambda_2(\mu + \lambda_1 + \lambda_2)}{\beta_0}$$

$$\beta_2 = \frac{\lambda_2^2 - \beta_1^2}{2\beta_0}; \quad \beta_i = -\frac{\sum_{j=1}^{i-1} \beta_j \beta_{i-j}}{2\beta_0} \quad i = 3, 4, \cdots. \quad (47)$$

Therefore, the expansion of $\alpha(Z) = \Sigma_{i=1}^{\infty} \alpha_i Z^i$ is given by

$$\alpha_0 = \frac{1}{2\mu}(\mu + \lambda_1 + \lambda_2 - \beta_0); \quad \alpha_1 = -\frac{1}{2\mu}(\lambda_2 + \beta_1);$$

$$\alpha_i = -\frac{\beta_i}{2\mu} \quad i = 2, 3, \cdots. \quad (48)$$

Therefore, the a_i's are given by

$$a_0 = p_0(\lambda_1 + \lambda_2 + \mu - \mu\alpha_0); \quad a_1 = -p_0(\lambda_2 + \mu\alpha_1);$$

$$a_i = p_0\mu\alpha_i \quad i \geq 2 \quad (49)$$

and the b_i's are given by $b_i = \Sigma_{l=0}^{i} b_l^1 b_{i-l}^2, i \geq 0$ where

$$b_0^1 = \lambda_2 + \mu; \quad b_1^1 = -\lambda_2; \quad b_i^1 = 0 \quad i \geq 2 \quad (50a)$$

$$b_0^2 = 1 - \alpha_0; \quad b_1^2 = -\alpha_1 - \rho_2; \quad b_i^2 = -\alpha_i \quad i \geq 2 \quad (50b)$$

and now we apply recursion (41) to obtain g_n, the steady-state probabilities of having n low-priority packets in the system.

C. *Approximation*

From the probability distribution for the infinite buffer case, we can come up with an approximation for the finite buffer case. One simple approximation for the loss probabilities of the two types of traffic in a system that can contain at most M high-priority and N low-priority packets is given by

$$P_{\text{loss}}^{\text{high}} \simeq \frac{g_M^{\text{high}}}{\sum_{m=0}^{M} g_m^{\text{high}}}$$

and

$$P_{\text{loss}}^{\text{low}} \simeq \frac{g_N^{\text{low}}}{\sum_{n=0}^{N} g_n^{\text{low}}}$$

where g_m^{high} (g_n^{low}) are the occupancy probabilities of the high (low) priority packets.

This approximation was compared to the exact result given in the Appendix for small systems and the correspondence was very good.

D. *Numerical Results*

The results are computed using the approximation of this section.

Fig. 5 shows the probability of packet loss for priority 1 as a function of the amount of buffering (for type 1 packets) in the output adaptor. This is plotted for different values of ρ. The buffering is measured in terms of the number of *average* size of packets that the output adaptor can hold. As can be seen, loss probabilities of less than 10^{-4} can be easily achieved. For the same parameters, Fig. 6 shows the probability of packet loss for priority 2.

Fig. 7 shows the corresponding delay bounds (for priority 1 traffic) for different output link speeds. Here, the probability of a packet loss is fixed at 10^{-4}. The results are for 1000 bit *average* sized packets. As can be seen, sub millisecond bounds are easily achievable at T3 (45 mbits/s) speeds or higher. For the same parameters, Fig. 8 shows the *average* delay for priority 2.

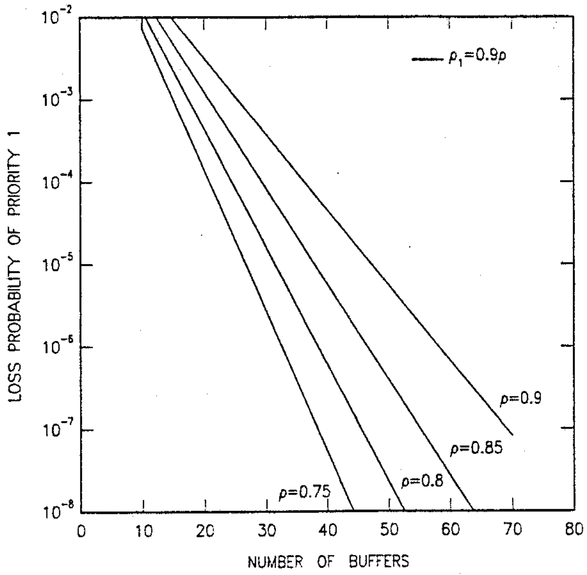

Fig. 5. Loss probability of priority 1.

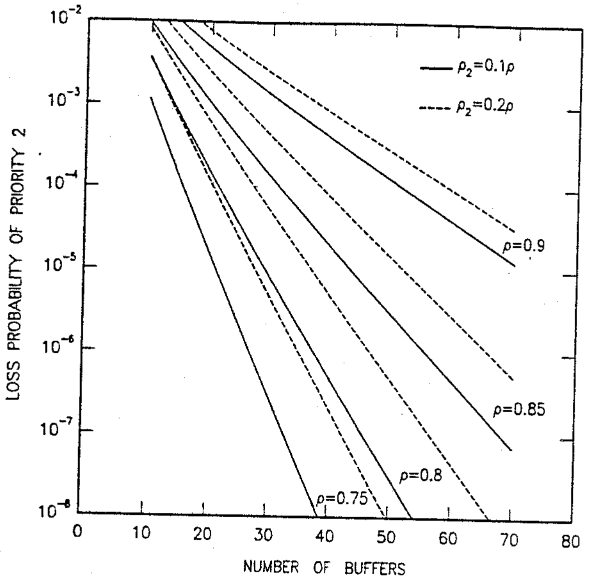

Fig. 6. Loss probability of priority 2.

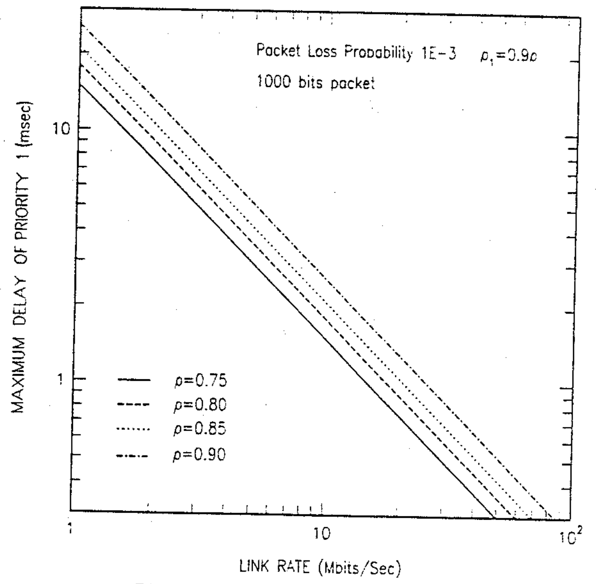

Fig. 7. Delay bound for priority 1.

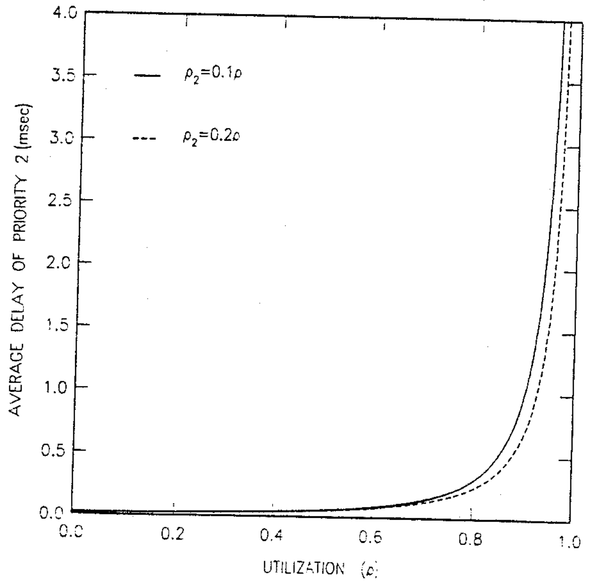

Fig. 8. Average delay for priority 2.

V. Summary

We have modeled the internal structure of a high-speed packet switching node and have identified the main queueing components in such a system. For the input and the switch fabric stages we have showed that, for a single-path switching system with a round robin exhaustive policy, if the switch is at least as fast as the sum of the input rates there is *no loss* at the input. We have derived tight bounds on the size of the input buffers required for such loss–less operation.

For the output buffers it is not possible to eliminate packet loss completely. However, for the type of systems that we consider under realistic speeds and link utilizations, one can achieve very low loss probabilities (less than 10^{-6}) with a very small delay bound (less than 1 ms). These results demonstrate the feasibility of operating a real-time system with a pure packet switching transmission.

Appendix

Let $\alpha = \lambda_1/(\lambda_1 + \lambda_2)$ and:

P_1 = probability that no packets are in the system.

$P_{1,m,n}$ = probability that a priority 1 packet is in service and that m priority 1 and n priority 2 packets are in the system (including service); $P_{1,0,0} \triangleq P_0$.

$P_{2,m,n}$ = probability that a priority 2 packet is in service and that m priority 1 and n priority 2 packets are in the system (including service); $P_{2,0,0} \triangleq 0$.

Then the equations that describe the steady-state behavior of a nonpreemptive queueing system that can contain at most M priority 1 and N priority 2 packets are given by

$$P_{1,1,0} + P_{2,0,1} = \rho P_0$$

$$\rho_1 P_0 + P_{1,2,0} + P_{2,1,1} = (1 + \rho) P_{1,1,0}$$

$$\rho_2 P_0 + P_{1,1,1} + P_{2,0,2} = (1 + \rho) P_{2,0,1}$$

$$\rho_1 P_{1,m-1,0} + P_{1,m+1,0} + P_{2,m,1} = (1 + \rho) P_{1,m,0}$$
$$2 \le m \le M - 1$$

$$\rho_1 P_{1,M-1,0} + P_{2,M,1} = (1 + \rho_2) P_{1,M,0}$$

$$\rho_2 P_{2,0,n-1} + P_{1,1,n} + P_{2,0,n+1} = (1 + \rho) P_{2,0,n}$$
$$2 \le n \le N - 1$$

$$\rho_2 P_{2,0,N-1} + P_{1,1,N} = (1 + \rho_1) P_{2,0,N}$$

$$\rho_2 P_{1,1,n-1} + P_{1,2,n} + P_{2,1,n+1} = (1 + \rho) P_{1,1,n}$$
$$1 \le n \le N - 1$$

$$\rho_2 P_{1,1,N-1} + P_{1,2,N} = (1 + \rho_1) P_{1,1,N}$$

$$\rho_1 P_{2,m-1,1} = (1 + \rho) P_{2,m,1} \quad 1 \le m \le M - 1$$

$$\rho_1 P_{2,M-1,1} = (1 + \rho_2) P_{2,M,1} \quad 1 \le m \le M - 1$$

$$\rho_1 P_{1,m-1,n} + \rho_2 P_{1,m,n-1} + P_{1,m+1,n} + P_{2,m,n,+1}$$
$$= (1 + \rho) P_{1,m,n} \quad 1 \le n \le N - 1;\ 2 \le m \le M - 1$$

$$\rho_1 P_{1,M-1,n} + \rho_2 P_{1,M,n-1} + P_{2,M,n+1} = (1+\rho_2) P_{1,M,n}$$
$$1 \le n \le N-1$$
$$\rho_1 P_{1,m-1,N} + \rho_2 P_{1,m,N-1} + P_{1,m+1,N} = (1+\rho_1) P_{1,m,N}$$
$$2 \le m \le M-1$$
$$\rho_1 P_{2,m-1,n} + \rho_2 P_{2,m,n-1} = (1+\rho) P_{2,m,n}$$
$$2 \le n \le N-1;\ 1 \le m \le M-1$$
$$\rho_1 P_{2,M-1,n} + \rho_2 P_{2,M,n-1} = (1+\rho_2) P_{2,M,n}$$
$$2 \le n \le N-1$$
$$\rho_1 P_{2,m-1,N} + \rho_2 P_{2,m,N-1} = (1+\rho_1) P_{2,m,N}$$
$$1 \le m \le M-1$$
$$\sum_{m=0}^{M} \sum_{n=0}^{N} (P_{1,m,n} + P_{2,m,n}) = 1.$$

Once the above $2(M+1)(N+1)-1$ equations are solved, the loss probability of priority 1 packets is given by

$$P_{\text{loss}}^{\text{high}} = \sum_{n=0}^{N} (P_{1,M,n} + P_{2,M,n})$$

and the loss probability of priority 2 packets is given by

$$P_{\text{loss}}^{\text{low}} = \sum_{m=0}^{M} (P_{1,m,N} + P_{2,m,N}).$$

References

[1] J. S. Turner and L. F. Wyatt, "A packet network architecture for integrated services," in *Proc. IEEE GLOBECOM'83*, Nov. 1983, pp. 45-50.

[2] J. S. Turner, "New directions in communications (or which way to the information age)," *IEEE Commun. Mag.*, vol. 24, Oct. 1986.

[3] I. Cidon and I. Gopal, "PARIS: An approach to integrated high speed private networks," Int. J. Digit. Analog Cabled Syst., to be published. lished.

[4] J. Y. Hui and E. Arthurs, "A boradband packet switch for integrated transport," *IEEE J. Select. Areas Commun.*, vol. SAC-5, pp. 1264–1273, Oct. 1987.

[5] L. T. Wu, "Mixing traffic in a buffered banyan network," presented at 9th Data Commun. Symp., Whistler Mountain, B.C., Canada, Sept, 1985.

[6] L. Kleinrock, *Queueing Systems Vol. I.* New York: Wiley, 1976.

[7] A. E. Eckberg, "The single server queue with periodic arrival process and deterministic service time," *IEEE Trans. Commun.*, vol. COM-27, pp. 556-562, Mar. 1979.

[8] R. G. Miller, "Priority queues," *Ann. Math. Stat.*, no. 31, pp. 86-103, 1960.

[9] J. J. Kultzer and W. A. Motogomery, "Statistical switching architectures for future services," presented at Proc. ISS '84, Florence, Italy.

Israel Cidon (M'85) received the B.Sc. (summa cum laude) and the D.Sc. degrees from the Technion—Israel Institute of Technology, Haifa, Israel, in 1980 and 1984, respectively, both in electrical engineering.

From 1980 to 1984, he was a Teaching Assistant and a Teaching Instructor at the Technion. From 1984 to 1985 he was on the faculty with the Department of Electrical Engineering at the Technion. From 1985 he is with IBM T. J. Watson Research Center. His current research interests are in communication networks and distributed algorithms.

Inder Gopal (S'80–M'81–M'82–SM'88) received the B.A. degree in engineering science from Oxford University, Oxford, England, in 1977, and the M.S. and Ph.D. degrees in electrical engineering from Columbia University, New York, NY, in 1978 and 1982, respectively.

Since 1982, he has been a Research Staff Member in the Computer Sciences Department at the IBM T. J. Watson Research Center, Yorktown Heights. Currently, he is Manager of the Integrated Networks group. His research interests are in communication networks, high-speed packet switching, and in distributed algorithms.

Dr. Gopal is currently an Editor for the IEEE TRANSACTIONS ON COMMUNICATIONS, Guest Editor for Algorithmica, and Guest Editor for IEEE JOURNAL ON SELECTED AREAS IN COMMUNICATIONS. He was formerly a Technical Editor for IEEE COMMUNICATIONS MAGAZINE.

George Grover is engaged in Communications Research at the IBM Thomas J. Watson Research Center. He was involved in the design of deadlock free flow control and of critical synchronization processes for the IBM System 36 APPN, a peer to peer data network switch. Since 1979 he has worked extensively in the design of SNA (IBM's System Network Architecture) networking functions. Previously, he participated in assembler, compiler and operating system design and development activities in conjunction with IBM's System 360, the Stretch computer, and the 7950—a special purpose extension of the Stretch computer; and in technical planning activities relating to advanced technology, and security and privacy. He joined IBM in 1954.

Moshe Sidi (S'77–M'82–SM'87) received the B.Sc., M.Sc., and the D.Sc., degrees from the Technion—Israel Institute of Technology, Haifa, Israel, in 1975, 1979, and 1982, respectively, all in electrical engineering.

From 1975 to 1981, he was a Teaching Assistant and a Teaching Instructor at the Technion in communication and data networks courses. In 1982, he joined the faculty of Electrical Engineering Department at the Technion. During the academic year 1983–1984, he was a Postdoctoral Associate at the Electrical Engineering and Computer Science Department at the Massachusetts Institute of Technology, Cambridge, MA. During 1986–1987, he was a Visiting Scientist at IBM, Thomas J. Watson Research Center, Yorktown Heights, NY. His current research interests are in queueing systems and in the area of computer communication network