PERFORMANCE OF A THREE-STAGE BANYAN-BASED ARCHITECTURE WITH INPUT AND OUTPUT BUFFERS FOR LARGE FAST PACKET SWITCHES

Fabio M. Chiussi and Fouad A. Tobagi

Technical Report No. CSL-TR-93-573

June 1993

This work has been supported in part by NASA under grant NAGW-419, by NSF under grant NCR-9016032, by Pacific Bell and by the AT&T Foundation.

Performance of a Three-Stage Banyan-Based Architecture with Input and Output Buffers for Large Fast Packet Switches

Fabio M. Chiussi and Fouad A. Tobagi

Technical Report: CSL-TR-93-573

June 1993

Computer Systems Laboratory
Departments of Electrical Engineering and Computer Science
Stanford University
Stanford, CA 943054055

Abstract

Fast packet switching, also referred to as Asynchronous Transfer Mode (ATM), has emerged as the most appropriate switching technique to handle the high data rates and the wide diversity of traffic requirements envisioned in Broadband Integrated Services Digital Networks (B-ISDN). ATM switches capable of meeting the challenges posed by a successful deployment of B-ISDN must be designed and implemented. Such switches should be nonblocking and capable of handling the highly-bursty traffic conditions that future anticipated applications will generate; they should be scalable to the large sizes expected when B-ISDN becomes widely deployed; accordingly, their complexity should be as low as possible; they should be simple to operate; namely, their architecture should facilitate the determination of whether or not a call can be accepted, and the assignment of a route to a call.

In this paper, we describe a three-stage banyan-based architecture, also referred to as the Memory/Space/Memory (MSM) switching fabric, which meets these challenges. It combines input and output shared-memory buffer components with space-division banyan networks, making it possible to build a switch with several hundred I/O ports. The MSM achieves output buffering, thus performing very well under a wide variety of traffic conditions, and is self-routing, thus adapting easily to different traffic mixes. Under bursty traffic, by implementing a backpressure mechanism to control the packet flow from input to output queues, and by properly managing the buffers, we can increase the average buffer occupancy; in this way, we can achieve important reductions in total buffer requirements with respect to output-buffer switches (e.g., up to 70% reduction with bursts of average length equal to 100 packets), use input and output buffers of equal sizes, and achieve sublinear increase of the buffer requirements with the burst length.

Key Words and Phrases: fast packet switching, broadband integrated services digital networks, asynchronous transfer mode, high-speed packet networks, banyan interconnection networks, bursty traffic, backpressure.

1. Introduction

Broadband Integrated Services Digital Networks (B-ISDN) for both local-area and wide-area communications are being designed to carry all types of information, such as voice, video, image and data, in an integrated fashion. Envisioned communication applications have widely different characteristics; specifically, the nature of their traffic is diverse, ranging from steady to highly bursty; their average data rates are quite different, ranging from few bits per second to hundreds of megabits per second; their reliability requirements differ greatly, some requiring full reliability while others tolerating various levels of data loss; finally, their delay requirements vary from a few milliseconds for real-time and interactive applications to several seconds for non-interactive applications.

The need to handle both steady and bursty traffic and to accommodate a wide range of expected data rates, delay requirements and tolerable losses has led to the choice of fast packet switching as the switching mode for B-ISDN. Fast packet switching and its protocols are currently being standardized by CCITT under the name of Asynchronous Transfer Mode (ATM). The ATM standard specifies 53-byte fixed-size packets (referred to as cells) comprising 48 bytes of payload and 5 bytes of control information, and line speeds of 155.52 Mb/s and 622.08 Mb/s [CCI90]. ATM is connection-oriented; that is, for each connection, a virtual circuit (VC) is established from source to destination, and used by all packets belonging to the connection; within a virtual circuit, packet sequence must be maintained.

In using B-ISDN, the user must specify at call set-up time the type of service requested for the connection. It does so by specifying the so-called Quality of Service (QoS) parameters which comprise bandwidth requirements (average, peak, etc.), burst characteristics (e.g., maximum duration), tolerable cell loss rate, maximum acceptable delay, etc.. For example, the request may be for a connection with constant data rate of

1.44 Mb/s, virtually no cell loss (e.g., $\leq 10^{-9}$), and fixed end-to-end delay of a few millisecond, as would be required to emulate a T1 circuit in circuit-switched networks; another example of a request may be for a 64-Kbit/s connection with constant data rate, 10^{-1} tolerable packet loss rate, and 100-ms maximum cell delay as would be needed for interactive voice communication; yet a third possible example of a request could be for a connection with variable data rate, whereby the peak bandwidth is equal to 100 Mb/s, and where the average burst size is 100 packets, the tolerable cell loss rate 10^{-4}, and the maximum end-to-end delay per packet 300 ms, as would be required for an interactive image-communication application.

Our interest in this paper is in the design of ATM switches which are capable of meeting the challenges posed by a successful deployment of B-ISDN, challenges pertaining to three major aspects: *performance*, *operation*, and *implementation*.

With the technological progress made in transmission facilities, the performance of a B-ISDN network is by and large limited by the performance of the switches. Thus, it is essential that, once the speed of transmission lines has been selected, the switch be capable of supporting all input/output connection patterns that the input and output lines are capable of carrying; *i.e.*, that the switch be *nonblocking*. An ATM switch must be capable of handling traffic of the various types in presence of one another, while meeting the QoS requirement of each connection. A particularly challenging requirement is that of supporting highly-bursty traffic (which will arise in many envisioned B-ISDN applications such as image communications) while achieving a low loss rate, as this case requires the presence of large-size buffers and intelligent buffer-management procedures. Finally, an ATM switch must maintain packets belonging to each virtual circuit in their original order.

When a call is placed, the network has to determine whether or not the call can be accepted; it does so by attempting to find a route from source to destination with sufficient bandwidth in the links and sufficient resources in the switches (switching bandwidth, buffer space, etc.) to meet the QoS requirements. In general, this problem is not a simple one, particularly if the bandwidth required is variable over time, and statistical multiplexing is used to increase network efficiency; indeed, the determination as to whether a request should be accepted or not is then dependent on the specific mix of traffic being multiplexed and on available resources. It can be divided into two steps. In the first step, most QoS requirements (specifically, packet loss and delay) are ignored, and an attempt is made to find a path from source to destination with residual bandwidth in both the links and the switches to support the ***average*** bandwidth required by the call. From an operational point of view, this step is easier to execute if the switches used in the network are ***nonblocking***, rather than blocking. The determination of the route is further simplified if, internally to the switch, packets belonging to a virtual circuit are routed independently of each other and may take different routes (***packet-by-packet*** routing), rather than being routed along the same route (*VC-by-VC* routing), since in the former case no knowledge of the specific characteristics of the virtual circuit (for example its bandwidth requirement) is required. The advantage is even more substantial in case that a multicast call has to be routed. If the first step is unsuccessful, the call is rejected; otherwise, the second step consisting of verifying that the QoS requirements can be met is performed. This step requires knowledge of the performance of the switch under the specific traffic mix being handled. Here too, the step is more or less difficult to execute depending on the difficulty involved in optimizing the allocation of internal resources in the switch to the various calls and in analyzing its performance under the various mixes of traffic. A facilitating factor is also that buffers be grouped only at the outputs of the switch (or only at the inputs and at the outputs) rather than being spread internally to the switch.

The design of the switch should be scalable to the (large) sizes expected when B-ISDN becomes widely deployed; accordingly, the complexity of the design should be as low as possible. In particular, the synchronization aspect is closely tied with the complexity of the design; thus, in a large switch consisting of multiple stages of components interconnected together, in order to facilitate synchronization, the number of components and the chip count should be kept low, the number of stages should be small, and the internal links should not be run at a speed higher than the speed of the I/O links. In general, the nonblocking property of the switch is achieved at lower complexity with packet-by-packet routing than with VC-by-VC routing, because with the former it is possible to distribute the traffic internally to the switch uniformly over all possible routes and utilize the internal resources more efficiently. It is also highly desirable that the preservation of packet sequence be a natural by-product of the architecture and its routing procedure, rather than having to be achieved by adding resequencing mechanisms at its outputs, so as to avoid additional complexity in the design.

Several basic architectural designs of ATM switches have been proposed, and many among them fully prototyped [JSA91]. They typically fall into three categories; namely *shared-memory* [Kuw89,Koz9 1,Sho9 1], *shared-medium* [Suz89], and *space-division* [Hua84,Tur88,Tob91], each having its features and shortcomings. Regardless of the particular architectural design used, however, given an implementation technology and its constraints, the size of a switch is bound to be limited. (For the prototypes so far realized, the largest-size switch has been 32 *x* 32 or 64 *x* 64 depending on the architecture used.) In order to build larger sizes, several switching modules (of the same or of different types) are combined in a multi-module configuration. As with the basic switch architectures, here too, several multi-module switch architectures have been conceived. They differ by the specific modules used (shared-memory switching modules, shared-medium switching modules, bufferless crossbars or banyans, buffered crossbars or banyans, etc.), the specific interconnection topology used to interconnect these modules (banyan arrangement, Benes arrangement, Clos arrangement, etc.), and the specific routing procedure

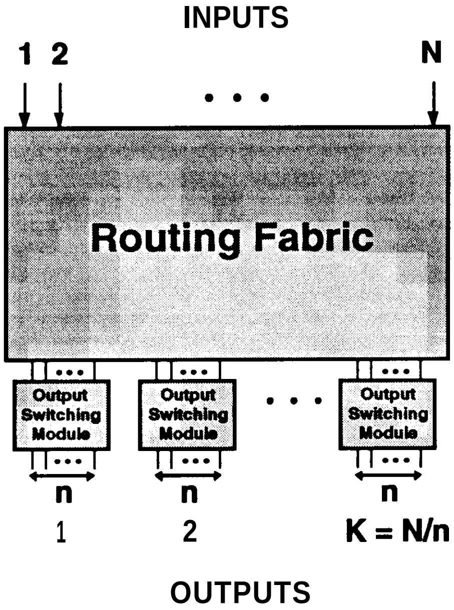


Fig. 1.1. Multi-module architecture for large fast packet switches.

used. They all share one thing in common, though, and that is, in all architectures, the modules used in the last stage are complete switching modules (with buffer capabilities). They typically are of the shared-memory or shared-medium type (but need not be limited to these two). Accordingly, a multi-module switch consists of a row of output switches, each handling a group of output lines (e.g., 32 or 64 lines), and a routing network which routes packets from the input lines to the appropriate output switching module, as shown in Fig. 1.1. The various multi-module architectures fall then into two basic classes:

i) the class of architectures in which the routing network uses *non-blocking* (and thus buffered) switching modules, typically of the same type, and thus comprises buffers at every stage of the network [Suz89,Koz9 1,Sho9 1,Ban91];

ii) the class of architectures in which the routing network uses *bufferless* modules (crossbars, banyans, Benes networks, etc.). Examples are the Growable Switch [Eng89], and the Group-Banyan Switch [Wan92].

These multi-module architectures are reviewed in Section 2, and the extent to which they meet the challenges of B-ISDN assessed; it is concluded that the routing fabric in a multimodule switch should preferably be ***bufferless***, so as to be able to route packets independently without causing them to get out-of-sequence, should be ***self-routing***, in order to simplify the routing control, and should have low ***complexity***, so as to facilitate its synchronization.

With these guidelines in mind, among the existing architectures, we consider with particular interest the Growable Switch, which consists of two stages of bufferless crossbars in a Clos topology that provides the smallest number of paths needed to make the fabric nonblocking; its primary limitation on size comes from the fact that it uses intelligent routing to assign the paths to incoming packets in each time slot. We observe, however, that this architecture can be made self-routing by using banyan routing networks instead of crossbars in the second stage of the routing network. By providing a small amount of buffering in the input components, in which to store packets that are not routed by the banyans in a given slot for further processing, the number of banyans needed to make the fabric nonblocking can be reduced to the minimum possible. The routing network consists therefore of a number of banyans placed in parallel, with input components (with buffer capabilities) that dispatch the packets to the banyans. Note that the use of banyans in parallel with input and output buffers has originally been proposed by Newman in the context of unslotted switching systems [New88b], and then described by Sarkies for a slotted switch [Sar91]; in these works, to achieve self-routing operation, packets are dispatched to the banyans in sequence.

Our work has its origin in the Tandem Banyan Switching Fabric (TBSF), and stems from our desire to increase the size that the TBSF can accommodate [Tob90b,Tob91b]. The TBSF uses a series arrangement of the banyans, thus achieving maximum utilization of each banyan by routing packets sequentially through the banyans.

Clearly, the same result can be obtained with a parallel arrangement of the banyans, by dispatching packets to the banyans in sequence. The arrangement of the banyans in parallel instead of in series leads to important advantages in terms of achievable switch size, since it results in an architecture which allows input and output components to be shared by many lines, and the number of banyans to be increased to that needed to accommodate the traffic, thus increasing the overall size of the switch beyond the achievable size of the banyan networks.

Having converged to the same architectural configuration coming from different starting points such as obviating the limitation due to routing control in the Growable Switch, and expanding the achievable size in the TBSF, in this paper we show that an architecture consisting of parallel banyans with input and output buffers, which we refer to as the Memory/Space/Memory (MSM) switching fabric, constitutes a very convenient switching structure to build fast packet switches that meet the challenges of B-ISDN. Although such configuration has been previously introduced in the aforementioned papers, it has not been characterized in terms of its performance under various traffic conditions, and under highly-bursty traffic in particular, nor has it been placed in the context of large switches by studying its scalability and implementation issues.

The MSM switching fabric meets the performance, operation, and implementation objectives outlined above. In fact, it is made nonblocking with low complexity, and is scalable to large sizes; it is self-routing (thus packets are routed on a packet-by-packet basis), and is simple to operate; despite the presence of buffers in the input controllers, it maintains packet sequence by using first-come-first-served service discipline in dispatching packets. The MSM achieves output buffering, thus it performs very well under a wide diversity of traffic patterns, including traffic patterns exhibiting correlation in the space domain (e.g., ***communities-of-interest*** traffic pattern) and in the time domain (e.g., ***bursty*** traffic pattern). The difficulty with highly-bursty traffic is that the size of the buffers in the switch required to meet a certain packet loss rate increases with the burst length and can easily exceed what can be implemented in hardware. In the MSM architecture, due to the presence of buffers in the input controllers, it is possible to reduce congestion in the output buffers by means of a backpressure mechanism, having part of the bursts wait at their respective input buffers instead of letting them "collide" at the outputs. In this way, we can evenly distribute the buffers at the inputs and at the outputs, and reduce the maximum buffer requirements in each buffer component as well as the total buffer requirements in the switch.

Finally, the MSM offers important advantages at the implementation level in terms of achievable sizes and data rates, owing to:

i) a two-phase operation of the banyans (a routing phase and a data-transfer phase), which allows to use different clock rates in each phase to meet different circuit requirements, thus making it possible to sustain high internal data rates (622.08 Mb/s instead of only 155.52 Mb/s) with currently available technologies.

ii) easier synchronization of the various components (banyans, input controllers, output buffers), which makes it possible to build a large switching fabric (> 1000 input/output ports), and operate it at 622.08 Mb/s.

This paper is organized as follows. In Section 2, we review existing architectural solutions for building large fast packet switches, identifying their distinctive features and sources of limitation. In Section 3, we describe the architecture of the Memory/Space/ Memory switching fabric, and present a complete characterization of the switch design to achieve different switch sizes and accommodate various traffic conditions, including correlated traffic patterns. In Section 4, we study the MSM with backpressure under bursty traffic, and show that by properly managing the input buffers, important savings in buffer requirements can be achieved under highly-bursty traffic conditions.

2. Architectures for Large Fast Packet Switches

Given the limitations induced by technology constraints on the size of current switching modules, regardless of their architecture, several modules have to be interconnected together to build large fast packet switches. As mentioned in the introduction, a multi-module architecture is defined by the specific selection of three elements:

a) the ***modules*** used; this refers to their function and size, and thus their resulting blocking characteristics. Modules may be ***nonblocking*** (i.e., they have sufficient switching capacity to accommodate the traffic for all sets of connection requests that can be carried by their input and output links) or ***blocking*** (***i.e.***, there exists at least one set of connection requests that cannot be accommodated). Note that nonblocking modules must provide buffering capabilities to accommodate output contention. Examples are shared-memory switches, shared-medium switches, or crossbars with buffers at the crosspoints (which are nonblocking, and thus include buffers) or bufferless crossbars, banyan, or Benes networks (which are blocking).

b) the ***multi-module configuration***; this refers to the number of modules of the various types used, the specific topology used in interconnecting the modules, and the speed of interconnection links. Typical topologies are the banyan, Benes and Clos arrangements; the banyan arrangement offers a single path from each input to each output, while the Benes and Clos arrangements offer multiple paths from each input to each output. Clearly, the arrangement of the modules is necessarily space division.

c) the ***routing procedure*** used which specifies how packets belonging to a virtual circuit are routed in the multi-module topology. two procedures are possible: (i) the *packet-by-packet* procedure in which packets belonging to a virtual circuit may be routed independently of each other and may take different routes, and (ii) the *virtual-circuit-by-virtual-circuit* (VC-by-VC) procedure in which all packets belonging to a virtual circuit are routed along the same route.

In a multi-module architecture, two aspects are of interest; namely, its performance characteristics (namely, its blocking characteristic and its ability to deliver packets in order) and the buffer requirements needed to guarantee other QOS requirements. In this section, we divide multi-module architectures into two categories (*i.e.*, the class of architectures in which the routing fabric uses *nonblocking*, and thus buffered, switching modules and the class of architectures in which the routing network uses *bufferless* modules) and discuss them first in terms of their performance characteristics. Then, we analyze their buffer requirements for various traffic types; in particular, we study bursty traffic, which is the traffic type with the most demanding buffer requirements. It should be noted that we are not trying here to define methodologies for assessing QOS capabilities, nor for guaranteeing them. Our purpose is simply to make sure that switches are designed with the right resources (number of paths, routing algorithm, buffer size) to meet bandwidth and other QOS requirements.

2.1. Architectures Based on Nonblocking Modules

A first example of a multi-module switch based on nonblocking modules is the Bus Matrix [Noj87]. Its architecture is essentially that of a crossbar with buffers at the crosspoints, and is therefore nonblocking. The modules here may be either single crosspoints with buffers, or crossbars of relatively small size (with buffers at the crosspoints), interconnected together in a square array. Such architecture requires $(N / n)^2$ components (where n is the size of the modules); the buffers are distributed over the N^2 crosspoints, and there can be no sharing among them.

A common solution that has been proposed to build a large fast packet switch is to interconnect together several nonblocking modules (which inherently must comprise buffering) of identical architectural type in a multistage configuration [Suz89,Koz91, Sho91,Ban91]. Here, the modules can be of any of the three architectural types mentioned above.

The multistage switch may or may not be blocking depending on the specific arrangement of the modules and on the routing algorithm used. A configuration which uses the smallest amount of resources is one that provides a unique path from each source to each destination, such as a banyan arrangement. A N x N switch can be built using log, N stages of N/n modules of size $n \times n$. For example, a 1024 x 1024 switch may be built by using 2 stages of 32 modules of size 32, as illustrated in Fig. 2.1.a. The banyan arrangement is a blocking switch, for the same reason that a banyan network is blocking (the difference is that here the switching elements have larger size, e.g., 32 x 32, as opposed to 2 x 2). In fact, since there is only one path from an input to an output, and each internal link is shared by several paths, blocking occurs as soon as the traffic does not use all internal links equally. For example, a simple scenario where such a problem occurs is one in which all the connection requests arriving at the n inputs of a module in the first stage are destined to ports connected to the same output module, and thus have to be routed through the same internal links; as soon as the sum of the requested bandwidths exceeds the capacity of the links, blocking occurs.

To overcome blocking and improve the performance of the switch, multiple paths must be provided from each input to each output. One solution is to use 2 log,, N $-$ 1 stages of N/n identical modules of size $n \times n$ interconnected in a Benes arrangement [Ben65]. For example, to build a 1024 x 1024 switch with a Benes configuration, we can use three stages of 32 modules of size 32, offering 32 concurrent paths for each input/ output pair (see Fig. 2.1.b). A Benes arrangement is a nonblocking switch provided that

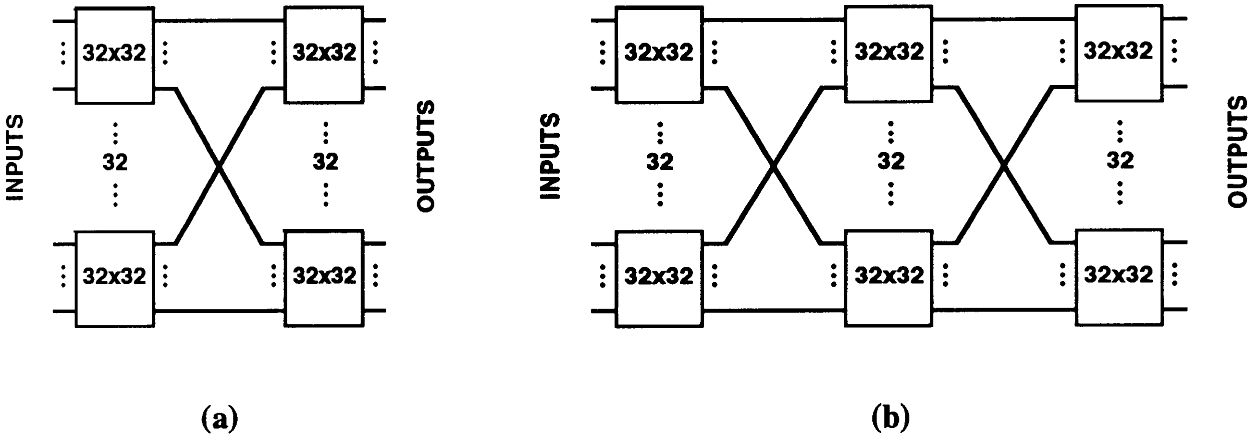


Fig. 2.1. 1024 x 1024 multistage configurations of modules of size 32 x 32; a) two-stage Banyan arrangement, single path from each input to each output; b) three-stage Benes arrangement, 32 paths from each input to each output.

packets belonging to a virtual circuit are routed independently from each other, and the first log,, N − 1 stages of the network are used as a randomizing network and the remaining log, N stages as a banyan routing network. Indeed, the nonblocking property is due to the fact that packets for a given connection are uniformly distributed over all possible paths from input port to output port (again, this is the same reason that a buffered Benes network could be made nonblocking). The Benes network is the smallest network (in terms of number of switching modules) to achieve nonblocking behavior.

A second solution is to use a three-stage Clos arrangement, which provides m paths from each input to each output **[Clo53].** It consists of a first stage of N/n modules of size $n \times m$, a middle stage of m modules of size $(N / n) \times (N / n)$, and a third stage of N/n modules of size $m \times n$; each module in the first and third stages is interconnected to each module in the second stage, as shown in Fig. 2.2 (note that if N/n is too large, the three-stage Clos arrangement can be further generalized by expanding each module in the middle stage into a three-stage Clos arrangement of size $(N / n) \times (N / n)$). It should be noted that in the particular case $N = n^2$, *the* Benes arrangement consists of three stages,

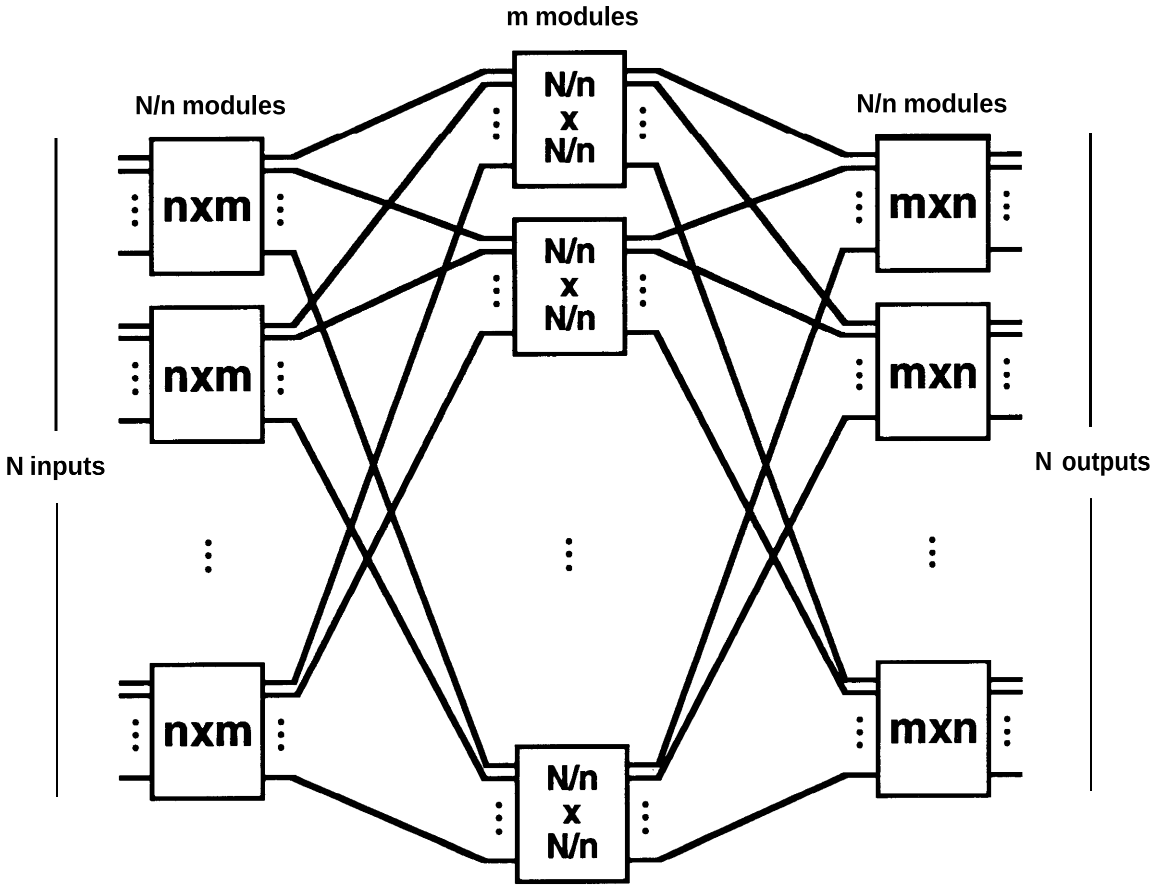


Fig. 2.2. Three-stage Clos arrangement.

and is equal to a three-stage Clos arrangement with $m = n$. The three-stage Clos arrangement is nonblocking provided that $m = n$, and packets belonging to a virtual circuit are routed independently from each other and uniformly distributed over all m paths from source to destination.

By definition, if the modules used in the multistage configurations are nonblocking, they must include buffering. With multiple paths between each input/output pair, since different paths traverse different intermediate modules and the queue length in each module is random, packets routed through different paths may experience different queueing delays. Thus, in both Benes and Clos arrangements, if packets are routed independently from each other, packets belonging to the same virtual circuit may be delivered out of sequence. A possible solution is to provide resequencing buffers at the outputs of the switch to reconstruct the proper sequence of the packets for each connection. This solution, although feasible, constitutes an additional level of complexity at the large sizes of interest, especially with the high degree of multiplexing for which ATM switches are designed, and thus it has been rarely attempted [Ban91].

In order to avoid the out-of-sequence problem, a common solution is to route packets belonging to the same virtual circuit along the same path and use FCFS queueing discipline within each module [Suz89,Koz91,Sho91]. We refer to this mode of operation as routing on a ***virtual-circuit-by-virtual-circuit*** basis. In this case, in order to accept a call, the switch must guarantee that sufficient bandwidth to carry the connection is available in all the links of the selected path.

With this mode of operation, the Benes arrangement, which is nonblocking when packets are routed independently from each other, becomes blocking. In order to make the Benes configuration nonblocking, the internal links must be run at a higher speed than the input/output links, as derived in [Mel89] through a generalization of classical circuit-switching theory to the case of multirate connections. Denoting as ***speed advantage*** α ($\alpha > 1$) the ratio of the speeds of internal and external links, it has been shown in [Mel89] that an r-stage Benes arrangement of switching modules of size $n \times n$ ($r = 2\log_n N - 1$) is nonblocking if it has a speed advantage:

$$\alpha = r - \frac{2}{n}\log_n \frac{N}{n} \qquad (1)$$

so, for example, a 1024 x 1024 3-stage Benes ($n = 32$) requires a speed advantage of 2.9375. Note that if all virtual circuits have the same data-rate requirements, it can be shown that a = 2 suffice to make the 3-stage Benes nonblocking.

Similarly, with multirate connections, in order to make the Clos arrangement nonblocking, the internal links must be operated at a higher speed than the external links, *i.e.*, $\alpha > 1$. For a given α, there is a condition on the number of internal paths needed to make the Clos arrangement nonblocking, as also derived in **[Mel89]**:

$$m > 2\frac{1}{\alpha - 1}(n-1) \tag{2}$$

The required m to satisfy the nonblocking condition (2) is plotted as a function of the speed advantage α **in** Fig. 2.3, for $n = 32$. For $\alpha = 3, m = n$. For α to be smaller, a larger m must be used. It is important to note that m diverges as the speed advantage approaches 1. From (2), we also note that switches that are nonblocking with $\alpha = 1$ when all virtual circuits have the same data-rate requirements, are blocking when their rates differ, and $\alpha > 1$ must be used to make them nonblocking. An example is a three-stage Clos arrangement with $m = 2n - 1$ internal paths between each input/output pair all running at the same rate, which is nonblocking for single-rate connections [Clo53], but is blocking for multirate connections [Mel89]; from (2), a speed advantage α equal to 2 is required for such a switch to be nonblocking.

It is possible to obtain nonblocking behavior with $\alpha = 1$ (*i.e.*, with the internal links running at the same rate as the external links; system with no speed advantage), provided that all connections have a bandwidth strictly lower than the capacity of the links. Let B denote the maximum bandwidth allowable for a connection (B < 1, normalized for convenience to the capacity of the links). The condition, shown in [Mel89], is then given by:

$$m > 2\left\lfloor \frac{n-B}{1-B} \right\rfloor \tag{3}$$

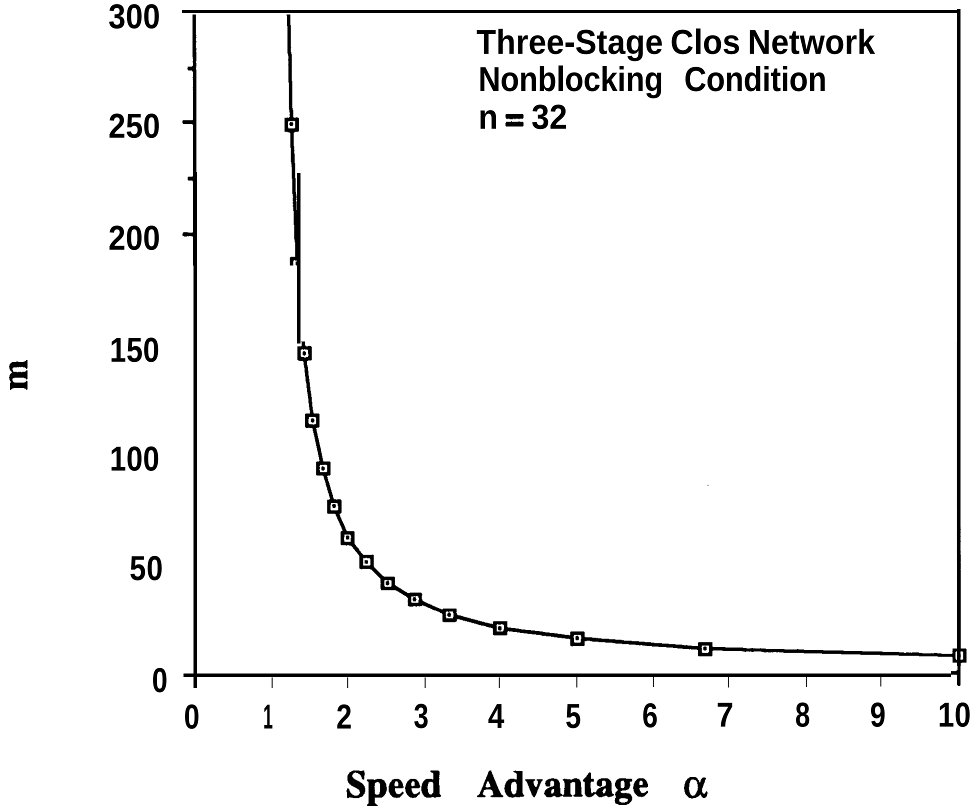


Fig. 2.3. Required *m* for a non-blocking Clos network as a function of the speed advantage of the system $\alpha, n = 32$.

The required m is shown in Fig. 2.4 as a function of B. Note that m diverges as B approaches 1. The required m is always above $2n + 1$, and increases rapidly even for relatively small values of B. For $B = 0.5$, m should be $4n - 1$.

This discussion makes evident that either a speed advantage in the system or a restriction on the bandwidth of the calls is necessary to obtain nonblocking behavior in Benes and Clos-type fast packet switches operated with virtual-circuit routing.

In switches that are blocking, a routing algorithm that can achieve high utilization of the available switching capacity under generic multirate traffic conditions is necessary in order to achieve low probability that a connection is blocked. With calls of different types (e.g., wide-bandwidth and narrow-bandwidth calls), the blocking behavior for each type is heavily dependent on how the bandwidth of the internal links is allocated to the calls. The bandwidth allocation method must balance the blocking behavior over all types of calls and contain the decrease in link utilization caused by fragmentation of available link capacity. Restrictions have to be imposed on the amount of bandwidth allocated to each type of call on the various links. In fact, if calls have different bandwidth requirements and no restrictions in link sharing are enforced, calls which require higher bandwidths will suffer higher blocking; depending on the dynamic behavior of the bandwidth distribution, fragmentation of the link capacity will reduce the available capacity to route wide-bandwidth calls. As a possible solution for these problems, since multiple links to a destination are available, narrow-bandwidth calls may be packed together on the same link in order to leave some links with as much available capacity as possible to route wide-bandwidth calls when they occur. In addition, a limitation on the maximum number of calls of a given type that may be routed through each link may be imposed, similarly to what has been proposed in the context of bandwidth allocation on a trunk [Lia89,Oda90]. Albeit these methods in general make the switch performance more robust under given traffic conditions, the achievable internal-link utilization remains well below optimal.

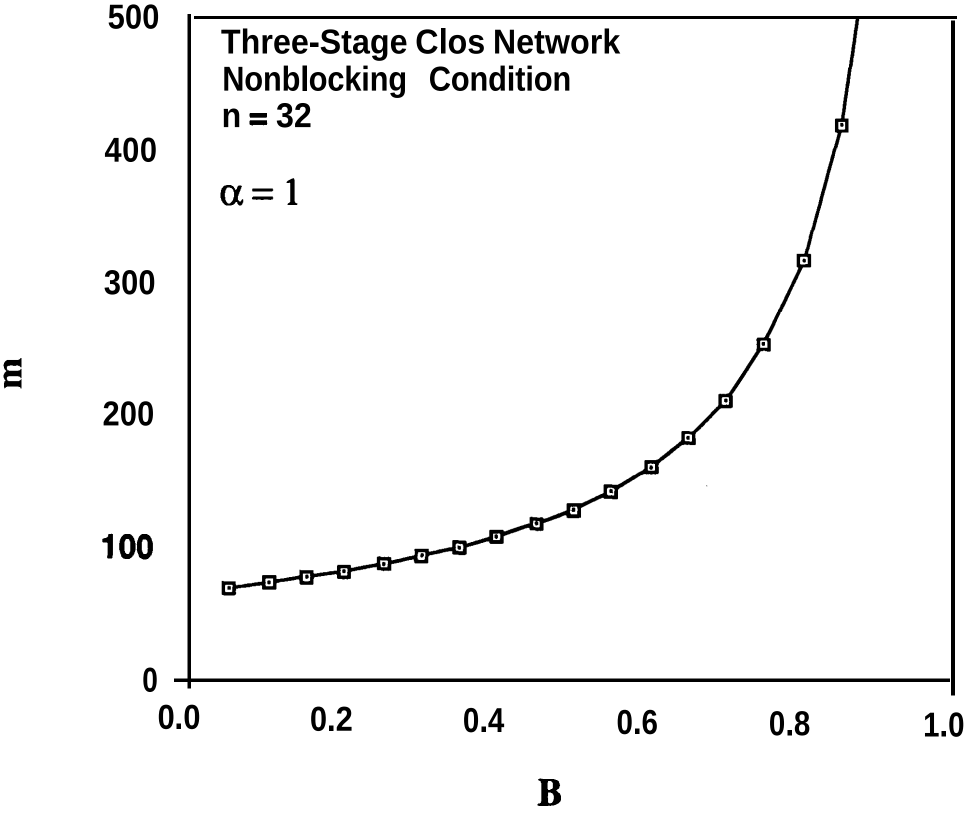


Fig. 2.4. Required m for a non-blocking Clos network with no speed advantage as a function of the maximum bandwidth of a connection B, $n = 32$.

With many types of calls, since the temporal characteristics of the calls are in general different for each type, it is evident that the design of a routing algorithm able to maintain high resource utilization and consistent performance over time by adapting to different mixes in the bandwidth requests is a challenging task. Furthermore, besides some knowledge of the characteristics of the incoming calls, such a routing algorithm would require information of the occupancy of all links throughout the switch in order to decide whether or not there is an available route to accept a call. Given the global scope of this information, the routing algorithm has to be centralized and the link status stored in a central memory accessed by the algorithm. As the switch size increases, this elaborate centralized algorithm would necessarily become a source of limitation. Overall, it is clear that the routing algorithm brings a considerable additional complexity in the design of the switch and its operation.

In summary, multistage configurations of nonblocking switching modules are natural solutions for building large fast packet switches. However, they suffer from numerous drawbacks. In these configurations, in order to overcome blocking, multiple paths for each input/output pair must be provided. Due to intermediate buffering along the paths, packets belonging to the same connection, if routed independently from each other, may experience different queueing delays and be delivered out of sequence. Resequencing buffers may be provided to restore the sequence of the packets, bringing an additional level of complexity in the design. Alternatively, out of sequence is avoided by routing packets belonging to the same virtual circuit through the same path. Under the multirate traffic conditions for which fast packet switches are designed, nonblocking behavior is obtained by using a sufficient number of paths and by running the internal links at a higher speed. If some blocking in the switch is tolerated, a routing algorithm is needed to efficiently allocate the bandwidth of the internal links and maintain consistent low blocking probability for the different types of calls by adapting to the dynamic

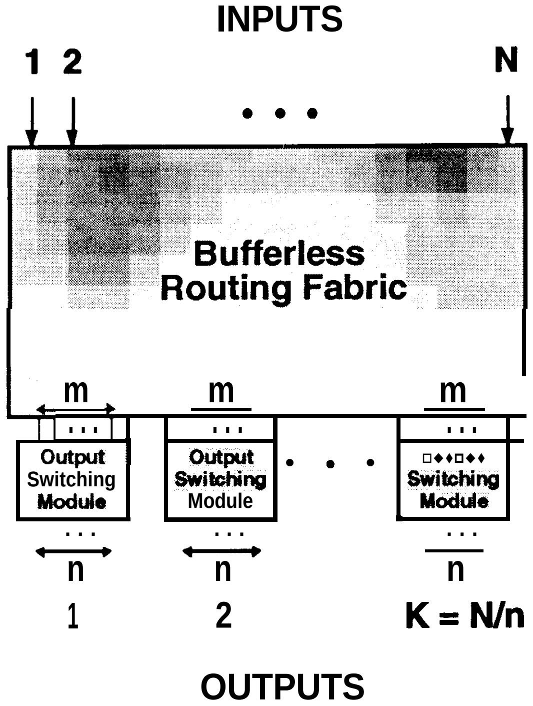


Fig. 2.5. Architectures based on bufferless routing fabrics.

variations in the traffic mix. Such a routing algorithm also brings considerable additional complexity to the design and operation of the switch.

2.2. Architectures Based on Bufferless Routing Fabrics

Given the aforementioned problems experienced in multistage configurations of buffered switching modules, alternative architectures for large fast packet switches that are nonblocking, route packets on a packet-by-packet basis, do not incur the out-of-sequence problem and achieve efficient resource utilization have emerged; namely, the Growable Switch [Eng89], and the Group-Banyan Switch [Wan92]. The approach underlying these alternative architectures, as shown in Fig. 2.5, is to provide several nonblocking output switching modules, each serving a subset of output ports, and provide a bufferless routing fabric which routes packets independently from the input ports to the

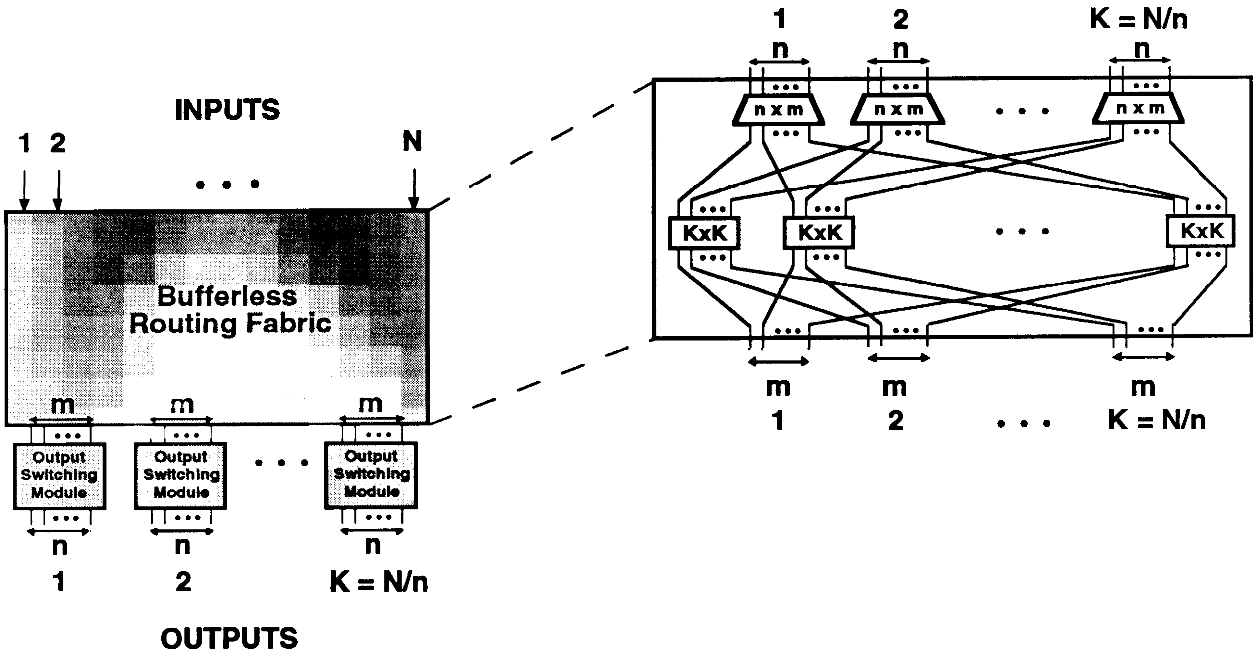


Fig. 2.6. Architecture of the Growable switch.

requested output modules. Since the routing fabric is bufferless, packets do not go out-of-sequence.

a) The Growable Switch

The Growable Switch, shown in Fig. 2.6, has its starting point in a three-stage Clos arrangement in which the output modules consist of nonblocking packet switching modules of a size that is implementable, and the modules in the first two stages are bufferless cross-connect modules (e.g., crossbars) which can simultaneously connect any disjoint pairs of inputs and outputs; the switch uses a routing procedure which routes packets from inputs to outputs on a slot-by-slot basis [Eng89]. The Clos arrangement of cross-connect modules has been originally introduced to build large circuit switches [Clo53]; in that context, it has been shown that the Clos arrangement is *rearrangeably* nonblocking (*i.e.*, the switch is nonblocking provided that existing connections can be rerouted in order to accommodate new ones) for $m \geq n$, where m is the number of modules in the intermediate stages (equal to the number of paths for each input/output pair). In order to use such a Clos arrangement as a packet switch, the output modules become packet switches; the nonblocking properties of the Clos arrangement are recognized, and the problem is to size the middle stage to make the switch nonblocking (*i.e.*, to achieve a desired low packet loss rate) despite the fact that packets are independently destined and can have output conflicts. Assuming that the destinations of packets arriving at different input lines are statistically independent, it is observed that the number of incoming packets destined to a given output group in a single slot is small compared to N. Considering a group of n outputs, there is a value m such that by receiving up to m packets destined to that group in a given slot (*i.e.*, by providing m modules in the middle stage), and discarding those in excess, a desired packet loss rate can be achieved. For example, it is shown that for $N \rightarrow \infty$ and n larger than 16, an ***expansion factor m/n*** lower than 2.5 is sufficient to guarantee a packet loss rate equal to 10^{-6}, under uniform traffic and loads as high as 0.9. The expansion factor needed to achieve a given packet loss rate decreases for increasing values of n. It should be noted that a similar observation is at the basis of the Knockout Switch (*i.e.*, that the number of packets destined to a given output in a single slot is small compared to N) [Yeh87]; however, in the Knockout Switch, since there is no output grouping, a larger expansion factor than in the Growable Switch is needed (e.g., an expansion factor equal to 8 is necessary in the Knockout switch for 10^{-6} packet loss rate, under uniform traffic at 0.9 load).

The concern then becomes one of devising a routing procedure to route packets from inputs to outputs on a slot by slot basis which is implementation-wise feasible. In fact, an optimal assignment of the paths would require a centralized routing algorithm to select up to m packets destined to each output group and to assign paths to all selected packets (the existence of such an assignment for the packets is guaranteed by the nonblocking property of the Clos arrangement for $m \geq n$). Since up to N packets would have to be processed and routed in a time slot, it is evident that this centralized routing algorithm would rapidly become a severe limiting factor on the switch size. To overcome this limitation, a suboptimal routing algorithm which can operate in a distributed fashion has been proposed.

The basis of the distributed algorithm is to assign paths to an output module sequentially from one input module to the next. The distributed routing algorithm operates in $K = N / n$ steps; accordingly, each time slot is divided into K mini-slots. Each input module in the first stage of the switch includes a routing processor that assigns routes to the packets arriving to the module. During each mini-slot, each input module is paired with an output module, and the corresponding routing processor assigns paths to all packets that are destined to that output module. In the first mini-slot, input module i $(i = 1,\ldots, K)$ assign paths to its incoming packets destined to output module i; in the second mini-slot, input module i assigns paths to all its packets destined to output module $i + 1(\text{mod } K)$; similarly for all the mini-slots. Assigning a path from an input module to an output module consists of finding a matching pair of available lines consisting of an output line of the input module and an input line of the output module connected to the same intermediate module, and reserving them for the packet. The assignment is accomplished by means of two sets of tables; namely, one set of input-module tables, one per module, which represents the status (free or used) of the output lines of the input modules, and one set of output-module tables, one per module, which represents the status of the input lines of the output modules. The input-module tables are kept in the respective input modules, while the output module tables are circulated among the input modules in a way that is synchronized with the pairing of input modules with output modules. To find a route from a input module to a given output module, the routing processor in the input module compares its input-module table with the output-module table, locating a free pair of matching links (see Fig. 2.7). Note that this sequential way of

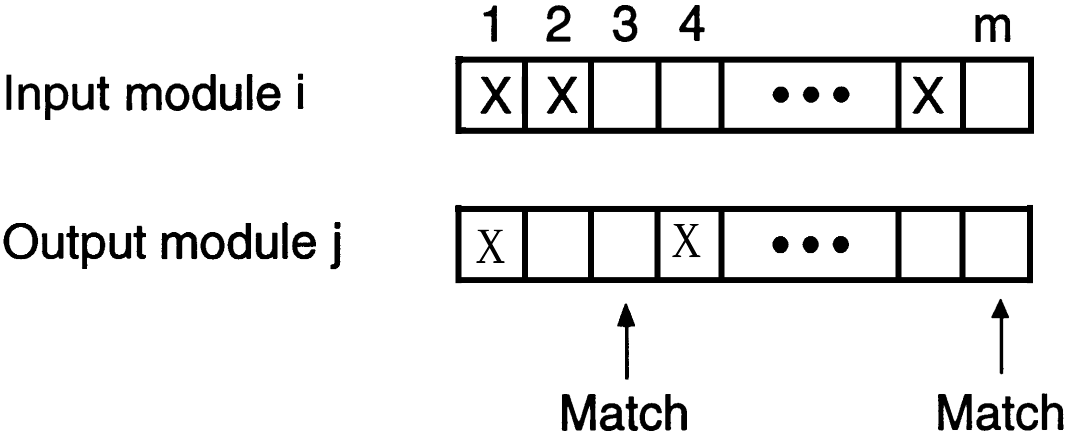


Fig. 2.7. Tables describing the availability of the output lines of input module *i* and of the input lines of output module j, and matching of available lines to assign paths from input module *i* to output module j.

assigning paths leads to unequal access of the modules if the order of assignment remains the same from slot to slot; to avoid unfairness, the pairing of input modules with output modules should be cyclically shifted from slot to slot.

Since the assignment of the paths from each input module to all output modules is performed in a predefined sequence, and since the paths, once assigned, cannot be rearranged during the assignment process, it is possible that some packets (among the up to *m* packets that can be routed to an output group in a slot) cannot be assigned a path, and have to be dropped. In other words, with suboptimal routing some loss is incurred due to the inability of the routing algorithm to assign a route to some packets, in addition to the loss due to the fact that more than *m* packets may be destined to an output group in a given slot. Consequently, with suboptimal routing, an expansion factor slightly larger than what needed with optimal routing is necessary to achieve a desired packet loss rate in the routing fabric. For example, for $N \rightarrow \infty$ and $n = 16$, $m = 35$ is necessary with suboptimal routing for a 10^{-6} packet loss rate, under uniform traffic and load = 0.8, instead of m = 34 required for the same loss rate with optimal routing (clearly, at higher loads, larger m would be required to achieve the same packet loss rate).

Due to intelligent routing, the Growable switch achieves high link utilization, equal to n/m. However, in each input module, the assignment of the paths to up to n packets destined to an output module, and the transmission of the output-module table to the next input module in the sequence has to be completed within a mini-slot (furthermore, in order to assign paths to all packets arriving at its inputs and destined to the same output module in a single mini-slot, an input module must sort the arriving packets according to their destination output module before the routing algorithm is performed). For given packet duration and group size n, the duration of a mini-slot is inversely proportional to the switch size; thus, the speed of the routing algorithm directly limits the practical size of the switch.

b) The Group-Banyan Switch

The Group-Banyan switch achieves self-routing operation by using a banyan arrangement of special modules, referred to as binary-grouping modules, which sort packets arriving at their inputs into two groups as a function of a particular bit in the destination address of the packets [Wan92]. More specifically, the bufferless routing fabric, as shown in Fig. 2.8, consists of an array of log, K stages of $K/2$ binary-grouping modules interconnected by groups of m links ($m > n$) in a banyan topology. Each binary-grouping module has two groups of m input links and 2 groups of m output links. The binary-grouping modules in stage i (i = 1,..., K) route the packets arriving on their $2m$ input lines to one of the two groups of m output lines according to the i-th bit in the binary representation of their destination addresses. A binary-grouping module may switch a packet arriving at an input line to any output line in the desired group. If the number of packets destined to one of the two groups of output lines exceeds m, only m packets are

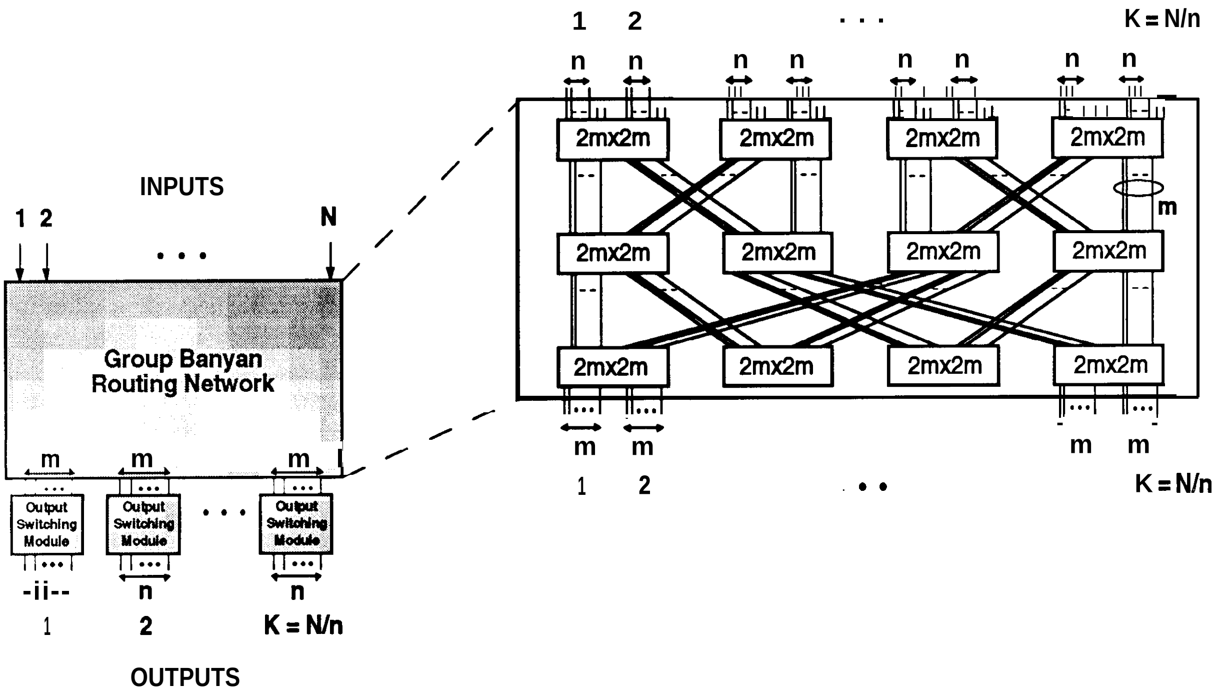


Fig. 2.8. Architecture of the Group-Banyan switch.

routed to that group, and the remaining ones are lost. In the modules in the first stage of the routing network, only ***n*** lines for each group of ***m*** input lines are used, and are connected to the inputs of the switch.

The Group-Banyan switch is based on an observation somewhat similar to the one at the basis of the **Growable** switch. If the destinations of packets arriving at different input lines are statistically independent, in each stage of the routing network the probability that, among the packets arriving at a routing module, more than ***m*** packets are destined to the same output group is small for some ***m*.** Thus, by properly selecting ***m*,** a desired packet loss rate in the routing fabric can be achieved. By the Law of Large Numbers, as ***n*** increases, the required expansion factor ***m/n*** necessary for a given packet loss rate decreases. For example, for ***n*** = 32, ***m*** = 63 is necessary for a 10^{-7} packet loss rate for any switch size of interest, under uniform traffic at full load.

A possible architecture for the $2m \times 2m$ -routing module consists of $2m$ 1×2-binary switches connected to a pair of $2m$ X m-concentrators, as illustrated in Fig. 2.9.a. Alternatively, the routing module can be realized using a self-routing multistage sorting network, as in Fig. 2.9.b. In this case, a routing module in stage i (i = 1,. . . , K) sorts the packets according to the i-th bit in their address. If there are more than m packets destined to an output group, packets in excess end up in the wrong output group after the sorting operation; one-bit filters at the outputs of the sorting network check if the i-th bits are consistent with the output group and, if not, discard the packets in excess.

In the Group-Banyan switch, because of the banyan arrangement of the modules, there is a single route from each input to each output; the links along that route have capacity m. The function of the binary-grouping modules is to allocate the available capacity of the links to the packets. Compared with the Growable Switch, the Group-Banyan avoids the need of intelligent routing at the cost of providing log, K stages of modules. Consequently, the hardware requirements are fairly demanding, and certainly constitute a limitation on the switch size. In particular, the synchronization of several stages of fairly large and complex routing modules is a difficult task, especially considering that the interconnections between modules consists of large groups of links interconnecting modules physically located well apart from each other. In addition, the Group-Banyan switch is based on the assumption that the traffic is uniformly distributed across all output destinations. Due to its banyan structure, the performance of the switch is sensitive to the input traffic pattern, and the switch is prone to congestion under unbalanced and correlated traffic (such as the communities-of-interest scenario). As a solution, a randomization network in front of the routing network to break the spatial correlation in the input traffic pattern and uniformly distribute the traffic in the routing fabric is necessary. A randomization network over all N inputs substantially adds to the complexity of the switch.

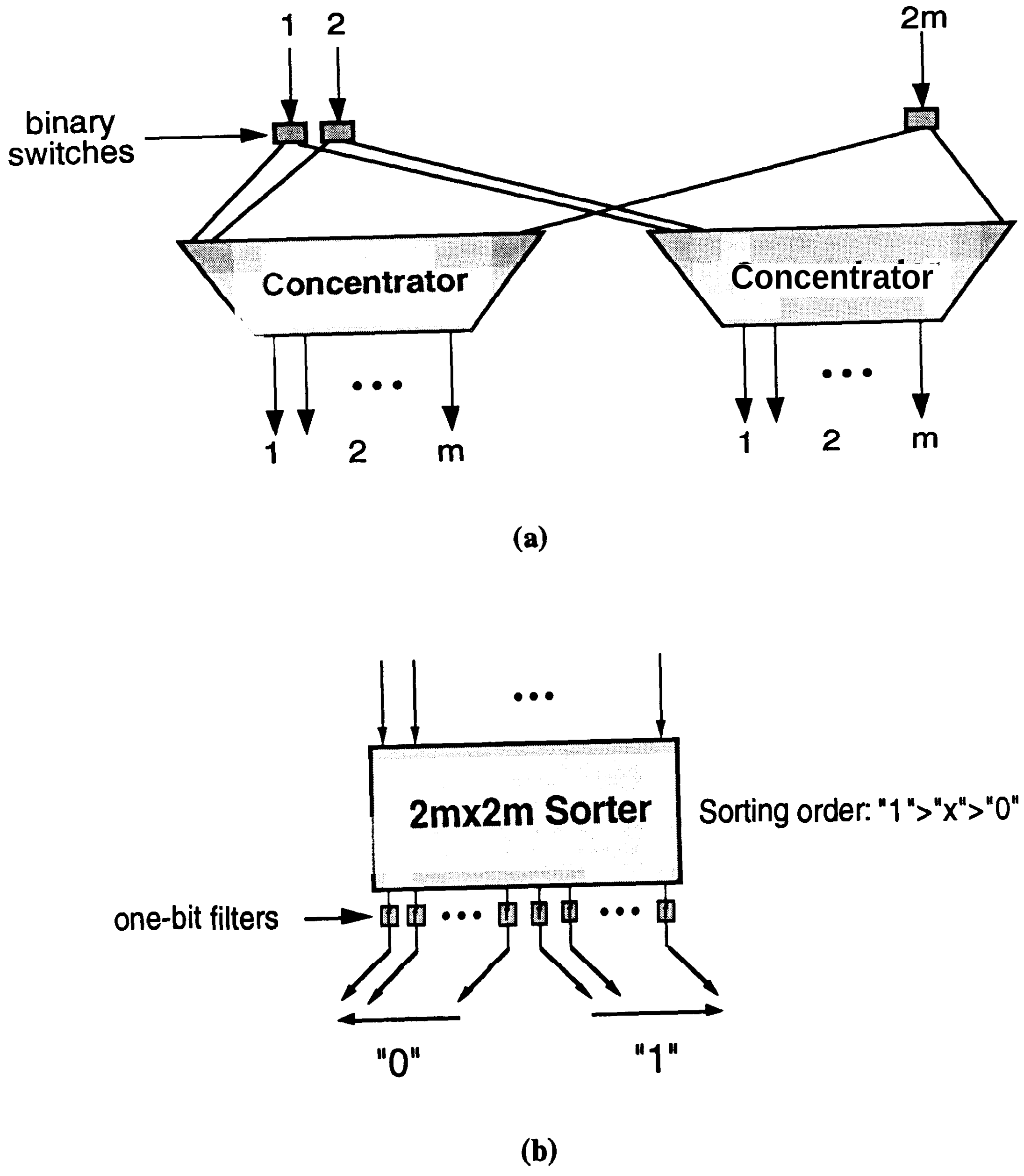


Fig. 2.9. Possible architectures for the routing modules in the Group Banyan switch; a) binary switches and concentrators; b) sorting network and one-bit filters.

2.3. Buffer Requirements

Besides the blocking characteristics of multi-module switches, we are interested in their buffer requirements to guarantee a desired packet loss rate under given traffic conditions. Here, we study the buffer requirements for a specific multistage switch, namely, a 1024 *x* 1024 Benes arrangement built from nonblocking switching modules of size 32 *x* 32 (see Fig. 2.1.b), with internal links running at the same rate as the external links (no speed advantage), using both packet-by-packet routing and virtual-circuit-by-virtual-circuit routing; we consider the switching modules to have a shared-buffer architecture. (Similar considerations, with the proper adjustments, would apply to multistage configurations of output-buffer switching modules with completely partitioned buffers.) We compare the buffer requirements of such a switch with those of multi-module output-buffer switches such as the Growable Switch and the Group-Banyan Switch.

We consider both uniform and bursty traffic conditions, at different loads (where the load is the rate with which packets arrive at the switch). In the uniform traffic pattern, the process describing the arrival of packets at each input line is a Bernoulli process with parameter p (which represents the load offered to the switch), independent from all other input lines, and the output port for a packet is uniformly chosen among all output ports, independently for each packet. We consider a commonly used bursty traffic model in which each input alternates between active and idle periods of geometrically distributed durations [Lie90,End90]; during an active period, packets destined to the same output arrive continuously in consecutive time slot. Let L the average length of the bursts (active periods), and L' the average length of the gaps between bursts (idle periods); then the load offered to the switch is equal to $L / (L + L')$. We assume that the output requests of the bursts are uniformly distributed over all destinations.

Single shared-buffer switching modules have been extensively studied both under uniform traffic [Hlu87,Hlu88,Eck88] and under bursty traffic conditions [End90,Lie90]. To serve as reference, we consider a single 32 *x* 32 shared-buffer switch, and show in Fig. 2.10.a. the packet loss rate under uniform traffic at different loads as a function of the buffer size per port (these results, obtained by simulation, are very close to those obtained by approximate analysis in [Hlu87,Hlu88]); in Fig. 2.10.b and Fig. 2.10.c, we show the packet loss rate (also obtained by simulation) under bursty traffic at different loads, for average burst length $L = 10$ and $L = 100$, respectively. As expected, the buffer size per port needed to achieve a desired packet loss rate increases with the load, as well as with the burstiness of the traffic (under bursty traffic, the increase is approximately linear).

The packet loss rate in each stage, and the cumulative packet loss rate as a function of the buffer size per port in each stage of a 1024 *x* 1024 Benes arrangement of *identical* shared-buffer switching modules of size 32 *x* 32 (thus, modules in all stages have equal buffer size), with internal links running at the same speed as the external links, using packet-by-packet routing, under uniform traffic at different loads is plotted in Fig. 2.11 (in this study, we do not consider the use of backpressure from output to input modules). As expected, each module in the first stage has the same buffer requirements as a single shared-buffer switch under uniform traffic. Also the modules in the other two stages have buffer requirements similar to the single switch. In fact, the packet arrival processes in these stages are independent and the packet destinations are uniformly distributed; the arrival processes, however, are not strictly Bernoulli processes, due to queueing at the previous stage; hence, the modules in the second and third stages actually require marginally larger buffer sizes than what is necessary in a single switch under uniform traffic to achieve a given packet loss rate. Of course the difference decreases for small buffer sizes (large loss rates), as the reduction in traffic in the second and third stage due to loss in the upstream stage(s) becomes noticeable. In general, as expected, the

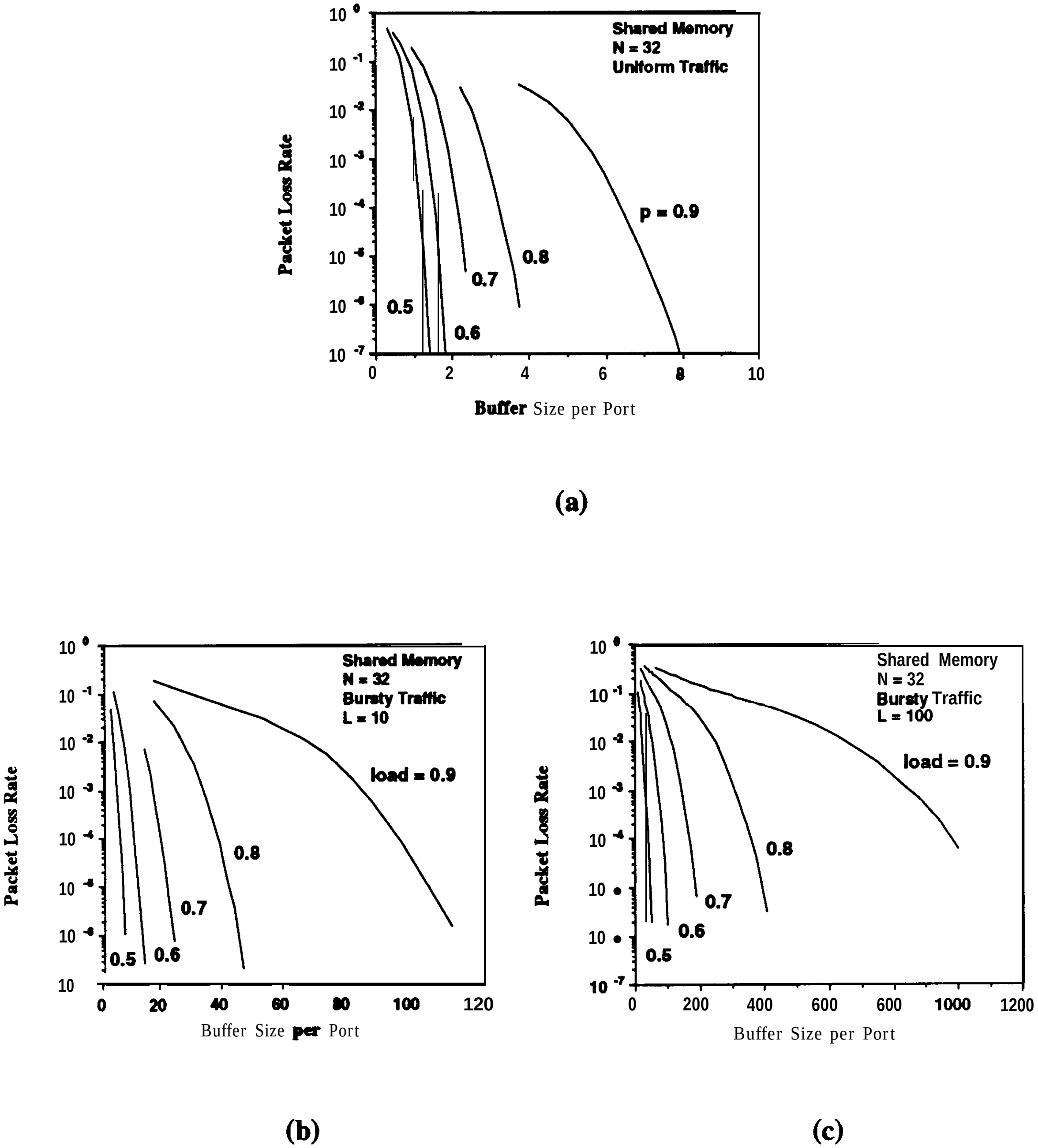


Fig. 2.10. Packet loss rate in a shared-buffer switch of size 32 *x* 32 as a function of the buffer size per port; a) uniform traffic conditions, for different loads *p* (obtained by simulation); b) bursty traffic conditions, burst lengths geometrically distributed with average *L* = 10, for different loads (obtained by simulation); c) bursty traffic conditions, burst lengths geometrically distributed with average *L* = 100, for different loads (obtained by simulation)

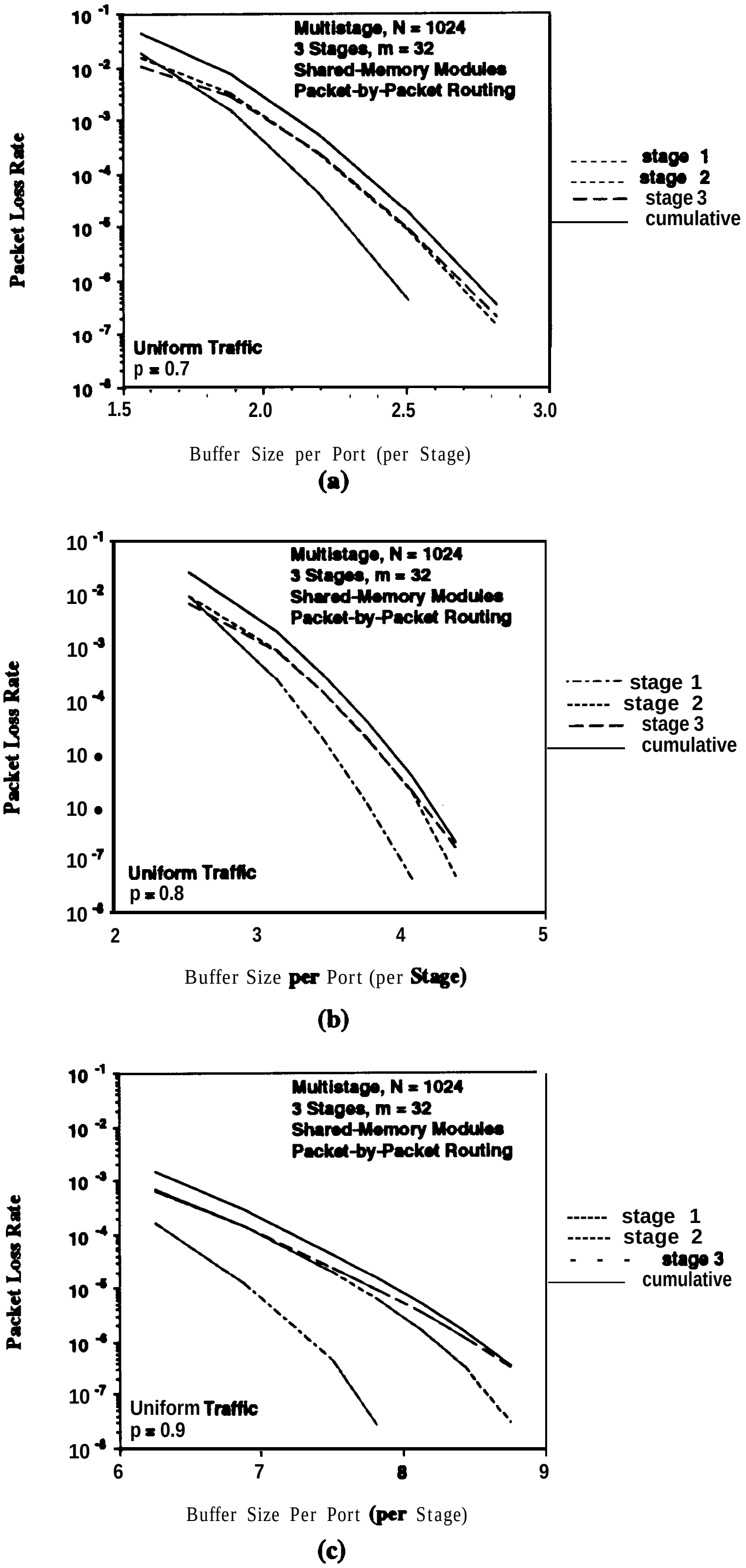
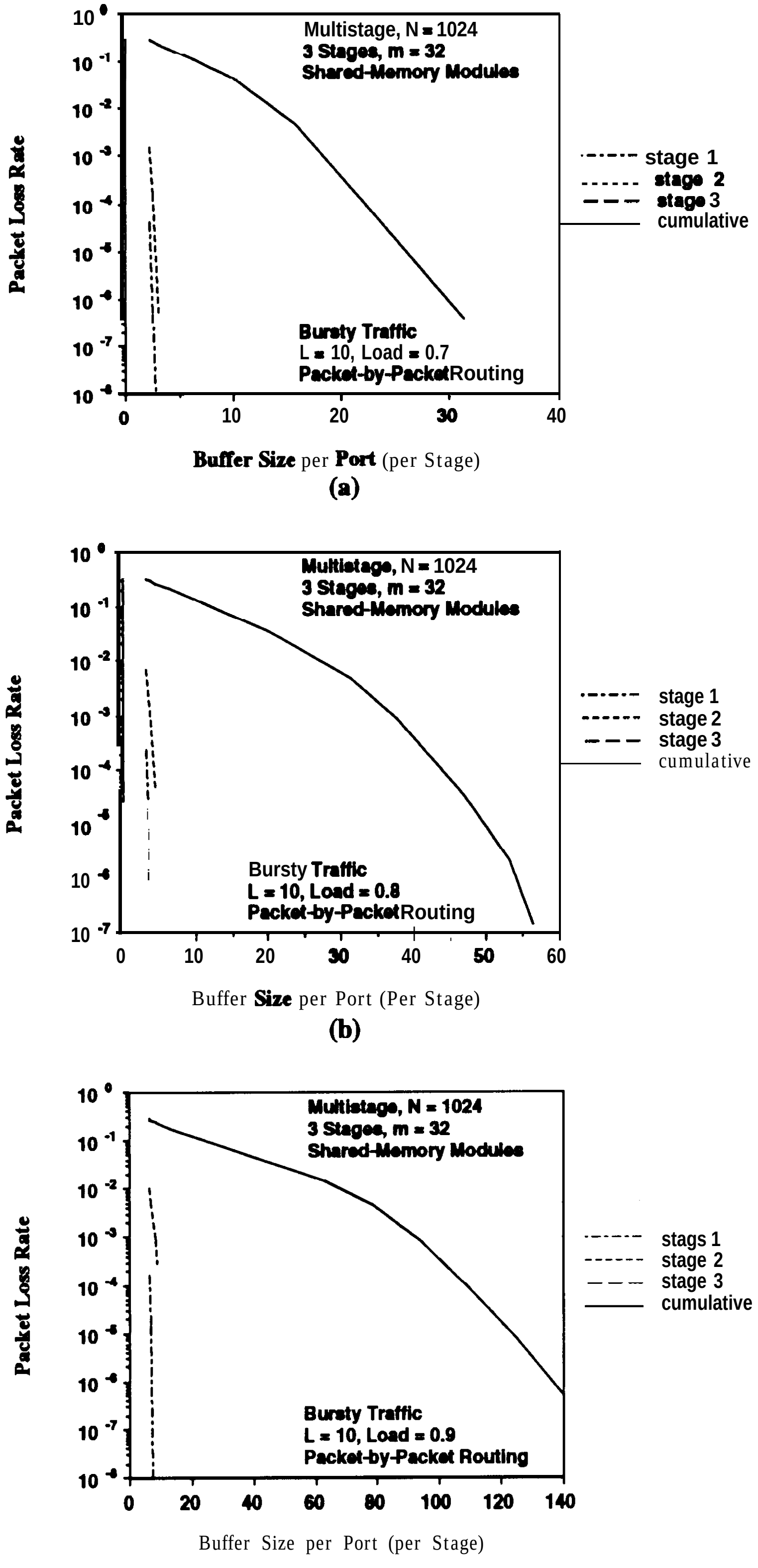


Fig. 2.11. Packet loss rate in each stage and cumulative packet loss rate in a 1024 *x* 1024 three-stage configuration of shared-buffer switching modules of size 32 *x* 32, with packets independently routed, under uniform traffic; a) p = 0.7; b) p = 0.8; c) p = 0.9 (obtained by simulation).

buffer requirements in each stage and the total buffer requirements under uniform traffic are rather small.

The packet loss rate under bursty traffic at different loads is shown in Fig. 2.12 and Fig. 2.13 for $L = 10$ and $L = 100$, respectively (again, all stages have equal buffer size, and no backpressure is considered). The first stage, since it acts as a randomizer, is still offered uniform traffic conditions; consequently, the buffer requirements of each module in the first stage are the same as those of a single shared-buffer switch under uniform traffic. At the inputs of the second stage, the burstiness in the traffic has been diluted by the randomization, so that the incoming traffic closely resembles uniform traffic; thus, the buffer requirements of the modules of the second stage are also rather similar to those of a single switch under uniform traffic; as the burst length increases with respect to the size of a module (i.e., with respect to the number of lines over which packets belonging to a burst are uniformly spread by the first stage), however, more and more residual burstiness is noticeable, and some increase in the required buffer size for a given loss rate is necessary. The modules in the last stage receive bursty traffic. The burst characteristics are marginally affected by queueing in the first two stages, and the buffer size needed in the modules is slightly larger than the size required for the same loss rate in a single shared-buffer switch under the same bursty traffic conditions.

Clearly, with packet-by-packet routing, the switch is nonblocking and the multistage arrangement achieves output buffering. The buffers in the first two stages are needed only to distribute the traffic inside the switch and balance the utilization of the internal switch capacity. Thus, under bursty traffic, rather than using identical modules, it proves convenient to use modules with buffers of different size at each stage; for example, assuming that buffers in each stage are sized so as to equally distribute the packet loss among the stages, a relatively small buffer size is needed in each module in the first two stages (similar to the buffer size required in a single 32 x 32 switch under uniform traffic to achieve the



(a)

(b)

(c)

Fig. 2.12. Packet loss rate in each stage and cumulative packet loss rate in a 1024 *x* 1024 three-stage configuration of shared-buffer switching modules of size 32 *x* 32, with packets independently routed, under **bursty** traffic conditions, burst lengths geometrically distributed with average $L =$ 10; a) load = 0.7; b) load = 0.8; c) load = 0.9 (obtained by simulation).

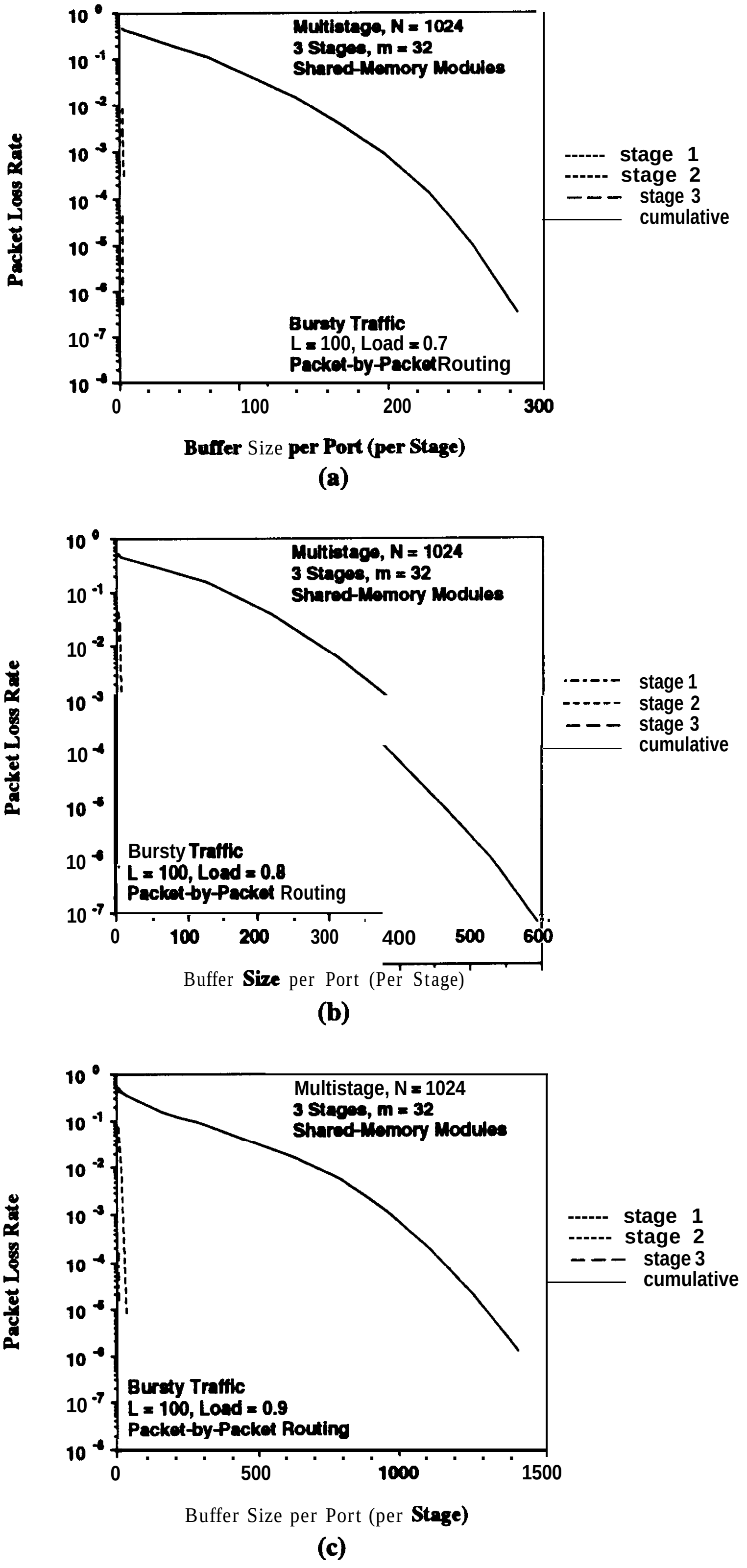
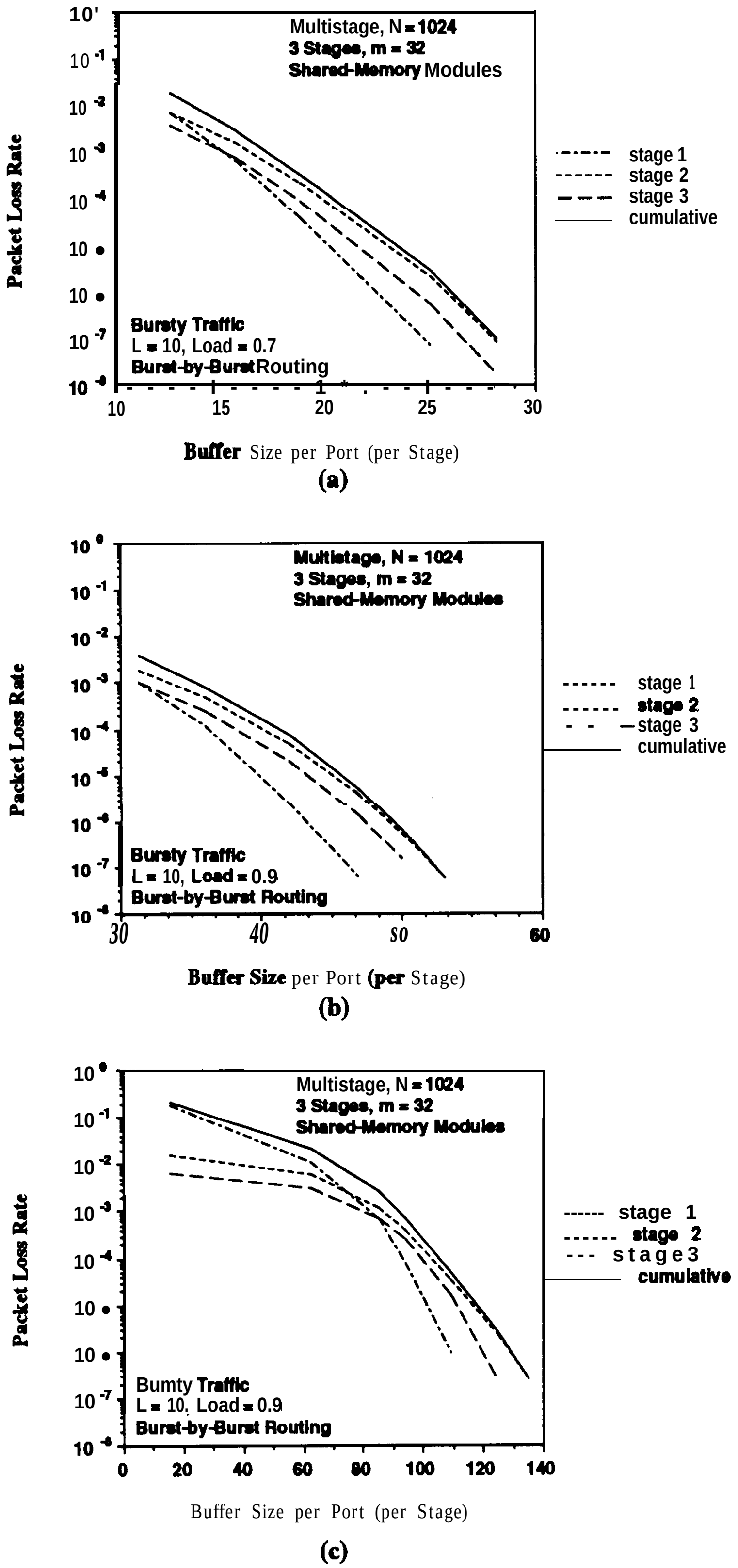


Fig. 2.13. Packet loss rate in each stage and cumulative packet loss rate in a 1024 *x* 1024 three-stage configuration of shared-buffer switching modules of size 32 *x* 32, with packets independently routed, under bursty traffic conditions, burst lengths geometrically distributed with average $L = 100$; a) load = 0.7; b) load = 0.8; c) load = 0.9 (obtained by simulation).

desired loss rate), and a large buffer size is needed in each module in the last stage only (similar to the buffer size required in a single 32 x 32 switch under bursty traffic to achieve the desired loss rate). In this way, the total buffer requirements in the switch under bursty traffic conditions are not much larger than the sum of the buffer requirements of N/n individual switches each serving n ports. As in any output buffer switch, the buffer size to meet a desired loss rate in the last stage increases roughly linearly with the burst length.

Operating the multistage configuration by routing packets on a virtual-circuit-by-virtual-circuit basis, rather than on a packet-by-packet basis, has also an important impact on the buffer requirements necessary to guarantee a desired packet loss rate. Evidence of this effect can be obtained by again considering the 1024 × 1024 three-stage Benes configuration built from shared-buffer modules of size 32, with the internal links running at the same speed as the external links. Since with routing on a virtual-circuit-by-virtual-circuit basis the switch is blocking, the reduction in offered traffic due to the fact that some virtual circuits are actually not accepted by the switch because they are blocked should be factored in the evaluation; this makes the assessment of the buffer requirements difficult; (clearly, a comprehensive simulation study at the packet level of the switch operated with routing on a virtual-circuit-by-virtual-circuit basis would be computation-wise very demanding). For our purposes, we need therefore to identify which mix of bandwidth requests for the virtual circuits is likely to be the most representative in terms of buffer requirements. We distinguish three cases. A first scenario is one in which all virtual circuits have high bandwidths (consequently, on each incoming line, the degree of multiplexing is low); under these conditions, the blocking behavior of the switch must be close to the circuit switching case, and the buffer requirements should be rather small. A second scenario is one in which all virtual circuits have low bandwidths (and the degree of multiplexing is high); in this case, blocking is likely to be low, but buffer requirements are large, due to the large amount of statistical multiplexing that is taking place in the switch. A third and final scenario is one in which virtual circuits have a generic mix of bandwidths. In these conditions, blocking is high, as discussed in Section 2.1 above, and the buffering requirements arc difficult to evaluate because they have to be coupled with blocking. From these considerations, however, we conclude that, as far as buffer requirements are concerned, perhaps the most interesting scenario is the one in which a relatively large number of virtual circuits of relatively low bandwidth are multiplexed on each input link; of interest is the case where each connection generates bursty traffic. In this case, it can be assumed that the bursts of a given connection would arrive so far apart from each other that they can be considered independent. Thus, an indication of the buffer requirements in the switch with virtual-circuit-by-virtual-circuit routing can be obtained by simulating the switch with ***burst-by-burst*** routing; namely, by selecting the routes on a burst-by-burst basis (specifically, upon arrival of a burst, the switching modules in the first stage randomly select one of the possible 32 routes to its destination; then, all packets in that burst follow the same route).

The packet loss rate in each stage and the cumulative packet loss rate in the switch as function of the buffer size per stage, with routing on a burst-by-burst basis, under bursty traffic at different loads with average burst length $L = 10$ and $L = 100$, is shown in Fig. 2.14 and Fig. 2.15, respectively (also in this case, all stages have equal buffer size, and no backpressure is considered). Since all packets in a burst are routed through the same route, modules in the first stage are offered bursty traffic; thus, their buffer requirements are the same as those of a single shared-buffer switch under bursty traffic (compare with Fig. 2.10.a and Fig. 2.10.b). Each module in the second stage receives the entire bursts, destined to the single output link connected with the output destination modules. The buffer requirements of the modules in the second stage are even slightly larger than those in the first stage, due to the effect of queueing in the upstream stage. Of course, for small buffer sizes (large loss rates), the reduction in traffic due to loss in the



(a)

(b)

(c)

Fig. 2.14. Packet loss rate in each stage and cumulative packet loss rate in a 1024 X 1024 three-stage configuration of shared-buffer switching modules of size 32 x 32, with packets routed on a burst basis, under bursty traffic conditions, burst lengths geometrically distributed with average $L =$ 10; a) load = 0.7; b) load = 0.8; c) load = 0.9 (obtained by simulation).

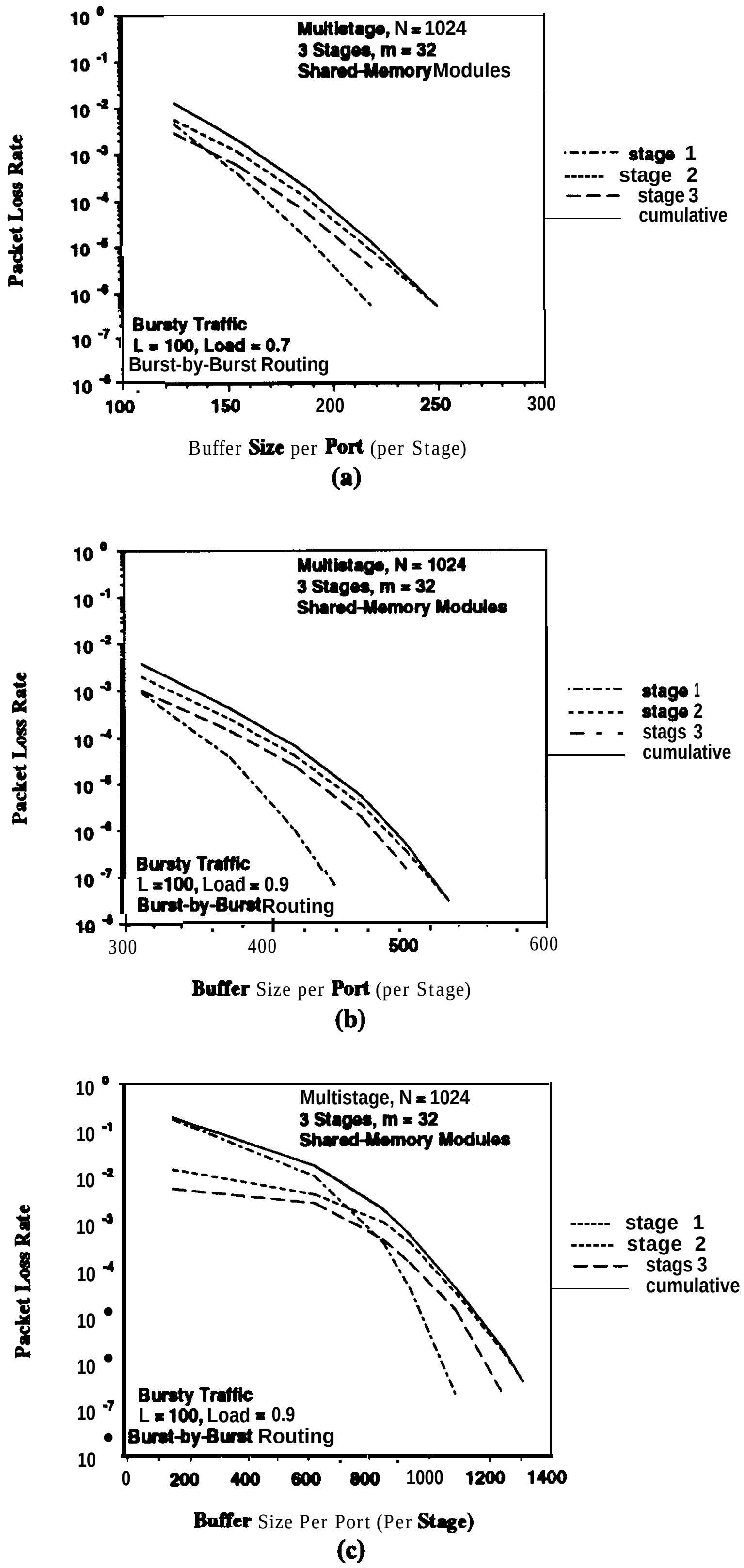


Fig. 2.15. Packet loss rate in each stage and cumulative packet loss rate in a 1024 × 1024 three-stage configuration of shared-buffer switching modules of size 32 x 32, with packets routed on a burst basis, under bursty traffic conditions, burst lengths geometrically distributed with average $L =$ 100; a) load = 0.7; b) load = 0.8; c) load = 0.9 (obtained by simulation).

first stage results in lower losses in the second stage. Clearly, in the last stage, the offered traffic is also bursty. Hence, the buffer requirements of the modules in all stages are similar to those of a single shared-buffer switch under bursty traffic, and increase approximately linearly with the burst size. The buffer requirements of the modules in the first two stages are dramatically worse than those observed in the switch with packet-by-packet routing, where uniform distribution of the packets in the first two stages is achieved by random routing of the packets. Here, the bursts are conserved inside the switch, thus heavily affecting the characteristics of the traffic offered to the switching modules.

To summarize this discussion, in the following table we compare the 1024 x 1024 Benes arrangement (using packet-by-packet routing and burst-by-burst routing) with switches based on bufferless routing networks (which achieve output buffering) in terms of total buffer size per port required to achieve a packet loss rate lower than 10” at 0.9 load. In this comparison, we assume that in the Benes arrangement the buffers in the modules in each stage are sized so as to equally distribute the packet loss among the stages (thus, under bursty traffic, as mentioned above, with packet by packet routing relatively small buffers are used in the first two stages, and large buffers are used in the last stage only). All the switching modules are assumed to be 32 x 32 shared-buffer switches.

TOTAL PACKET BUFFERS PER PORT	**Benes Arrangement Packet-by-Packet Routing**	**Benes Arrangement Burst-by-Burst Routing**	**Multi-Module Switches Based on Bufferless Routing Fabric**
Uniform	25	—	8
Bursty, *L* = 10	160	388	138
Bursty, *L* = 100	1560	4010	1406

As mentioned above, the Benes arrangement with packet-by-packet routing essentially achieves output buffering, and thus the buffers under bursty traffic are mostly concentrated in the last stage. Small buffers are needed in the modules of the first two stages to uniformly distribute the traffic inside the switch. In contrast, in the Benes arrangement with burst-by-burst routing, the buffer requirements are large in every stage (and comparable to those in the last stage of the switch with packet-by-packet routing).

2.4. summary

In multistage configurations of nonblocking switching modules multiple paths for each input/output pair are provided in order to overcome blocking. Due to intermediate buffering along the paths, packets belonging to the same connection, if routed independently from each other, may experience different queueing delays and be delivered out of sequence. If routing on a packet-by-packet basis is used, efficient utilization of the resources is used, but the out-of-sequence problem is incurred. Resequencing buffers may be provided to restore the sequence of the packets, bringing an additional level of complexity in the design. Alternatively, out of sequence is avoided by routing packets belonging to the same virtual circuit through the same path. Then, either the switch is more prone to be blocking, in which case a relatively complex routing algorithm needs to be implemented in order to efficiently allocate the bandwidth of the internal links, or the internal links must be run at a higher speed than the external links and more paths have to be provided to render the switch nonblocking and simplify routing.

Given these drawbacks of multistage configurations of buffered switching modules, alternative architectures for large fast packet switches that are nonblocking, route packets on a packet-by-packet basis, do not incur the out-of-sequence problem and achieve efficient resource utilization have emerged. They are the **Growable** Switch **[Eng89],** and the **Group**-Banyan Switch **[Wan92].** Such architectures consist of several nonblocking output switching modules, each serving a subset of output ports, and a bufferless routing fabric which routes packets independently from the input ports to the requested output modules, thus achieving output buffering. The Growable Switch achieves high utilization of the resources, but intelligent routing is needed to assign the paths to packets in each time slot. The Group-Banyan Switch is self-routing, but requires a larger number of stages and modules.

3. Architecture and Performance of the Memory/Space/ Memory Switching Fabric

3.1. Architecture of the Memory/Space/Memory (MSM) Switching Fabric

In the previous section, we concluded that the routing fabric in a large multimodule switch should preferably be ***bufferless***, so as to be able to route packets independently without causing them to get out-of-sequence, should be ***self-routing***, in order not to be limited by the routing control, and should have ***low complexity*** (by using a small number of stages and modules, keeping the chip count low, and avoiding to run the internal links at a higher speed than the I/O links), so as to facilitate its synchronization.

As mentioned in the Introduction, among the architectures for large switches described above, we have considered with particular interest the Growable Switch [Eng89], which features a Clos topology providing the smallest number of paths to make it nonblocking, and uses intelligent routing to assign the paths to packets in each time slot; we have observed that it can be made self-routing by using banyan routing networks instead of crossbars in the second stage of the routing fabric, and by having the components in the first stage properly dispatching the packets to the banyans. (Note that banyan networks are very desirable building blocks to use in routing networks; in fact, not only they are easy to control, due to their self-routing operation, but also can be implemented in larger sizes than crossbars on a single chip, due to their regular structure and minimum number of crosspoints.) By providing a small amount of buffering in the input components, in which to store packets that are not routed by the banyans in a given slot for further processing, the number of banyans needed to make the fabric nonblocking is reduced to the minimum possible. The architecture then consists of a number of banyans placed in parallel, with input components (with buffer capabilities) that dispatch the packets to the banyans, and output switching modules. We have converged to the same architectural configuration starting from the Tandem Banyan Switching Fabric (TBSF), motivated by our desire to increase the size that the TBSF can accommodate [Tob90b,Tob91]. The TBSF uses a series arrangement of the banyans, and its high performance results from the fact that the load offered to each consecutive banyan in the series is the highest possible (in fact, all the incoming traffic is offered to the first network in the series; then, all the traffic that has not been routed by the first banyan is offered to the second banyan, and so on for all the banyans). Since the throughput of a banyan improves as its load increases, the series arrangement is therefore a simple way to ensure maximum utilization of each banyan. Clearly, the banyans can be used in a similar way with a parallel arrangement, by dispatching the packets to the banyans in sequence. The arrangement of the banyans in parallel instead of in series results in an architecture which allows input and output components to be shared by many lines, and the number of banyans to be increased to that needed to accommodate the traffic, thus increasing the overall size of the switch beyond the achievable size of the banyan networks.

Note that a configuration using banyans in parallel with input and output buffers has originally been proposed by Newman in the context of unslotted switching systems [New88], and then described by Sarkies for a slotted switch [Sar91]. Such configuration, however, has not been characterized in terms of its performance under various traffic conditions, and under highly-bursty traffic in particular, nor it has been placed in the context of large switches by studying its scalability and its implementation issues at the large sizes of interest.

In the following, we show that an architecture consisting of parallel banyans with input and output buffers, which we refer to as the Memory/Space/Memory (MSM) switching fabric, constitutes a very convenient switching structure to build fast packet switches that meet the performance, operation, and implementation objectives of B-ISDN. In fact, it is made nonblocking with low complexity, and is scalable to large sizes;

despite the presence of buffers in the input controllers, packet sequence is maintained by using first-come-first-served service discipline in dispatching packets. The MSM achieves output buffering, thus it performs very well under a wide diversity of traffic patterns, including traffic patterns exhibiting correlation in the space domain (*e.g.*, *communities-of-interest* traffic pattern) and in the time domain (e.g., ***bursty*** traffic pattern). Since the fabric is nonblocking and self-routing, it adapts easily to different traffic mixes and is simple to operate.

Under bursty traffic conditions, in switches achieving output buffering, the size of the buffers in the switch required to meet a certain packet loss rate increases with the average burst length and, in case of long bursts, can easily exceed what can be implemented in hardware. In the MSM architecture, due to the presence of buffers in the input components, it is possible to reduce congestion in the output buffers by means of a backpressure mechanism, having part of the bursts wait at their respective input buffers instead of letting them "collide" at the outputs. In this way, we can evenly distribute the buffers at the inputs and at the outputs, and reduce dramatically the maximum buffer requirements in each buffer component as well as the total buffer requirements in the switch.

The MSM also offers important advantages at the implementation level in terms of achievable sizes and data rates. In particular, a ***two-phase operation*** of the banyans (a routing phase and a data-transfer phase), allows to use different clock rates in each phase to meet different circuit requirements, thus making it possible to sustain high internal data rates (622.08 Mb/s instead of only 155.52 Mb/s) with currently available technologies. Furthermore, given the configuration of the MSM, a system synchronization scheme which does not require strict synchronization of the various components (banyans, input controllers, output buffers) can be used, thus making it possible to build a large switching fabric (> 1000 input/output ports), and operate it at 622.08 Mb/s. We also show that, by using a general configuration of the MSM consisting of several groups of banyans in parallel, with each group serving a subset of output ports, the design parameters of the MSM can be adjusted in order to take into account different technology constraints and best accommodate requirements such as required memory bandwidth in the buffers, clock rates in each phase of operation, and silicon area.

Finally, we note that the presence of input and output buffers shared by several lines provides a convenient “interface” to combine input and output lines running at different speeds into a single switching structure; similarly, the banyans in the parallel arrangement can be easily operated at a rate different from the rates of input and output lines.

In the sequel, we first describe the basic design and operation of the routing fabric in the MSM. We then discuss how this basic structure is expanded to accommodate large switch sizes, and present a complete characterization of the switch design to achieve different switch sizes and accommodate various traffic conditions, including correlated traffic patterns. Finally, in Section 4, we show how, by implementing backpressure between input and output queues, this same architecture can support highly-bursty traffic conditions very efficiently. (Implementation issues pertaining to the MSM are studied in detail in [Chi93].)

3.1.1. Basic Configuration

a) Architecture

In its basic configuration, shown in Fig. 3.1, the Memory/Space/Memory (MSM) switching fabric consists of K banyans placed in parallel, along with input controllers, one for each of the N inputs, which dispatch the incoming packets to the banyans, and output buffers connected to the corresponding outputs of each banyan. We first describe

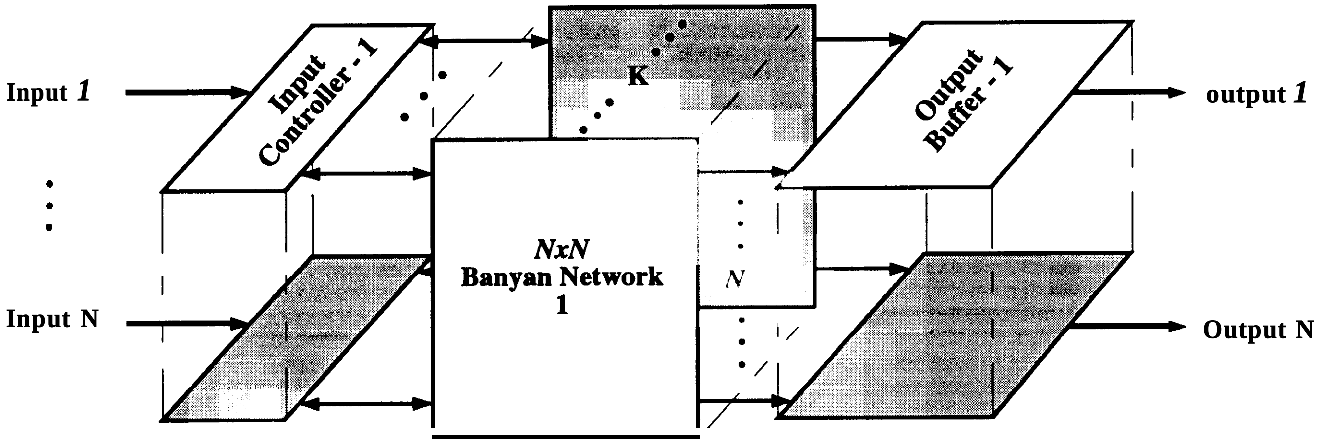


Fig. 3.1. The MSM switching fabric: basic configuration.

the switch architecture and its operation in the case in which each input and output component handles a single input and output line, respectively; we then explain how the architecture can be scaled to large sizes by sharing input and output components among several lines and by increasing the number of banyans to that needed to accommodate the traffic.

In order to minimize the number of banyans necessary to achieve a desired packet loss rate, the input controllers must dispatch the packets to the banyans so as to achieve high utilization of the banyan networks. This can be achieved by dispatching the packets to the banyans in sequence (referred to as ***sequential dispatching*** in the following). The operation of the switch is assumed to be slotted. In case no buffers are provided in the input controllers, the operation works as follows. In every time slot, each input controller that receives a packet dispatches it to the first banyan in the sequence; packets are then routed by the banyan; successful packets proceed to the output buffers, and unsuccessful packets are dispatched to the second banyan. The operation proceeds similarly for all the banyans; packets that are still unsuccessful after the last banyan has been attempted are dropped. By providing buffering in the input controllers in which to store packets that are unsuccessful in a given time slot for further processing, we increase the load on the banyans (and consequently their utilization), and achieve the same packet loss rate with fewer banyans. By organizing the input buffers as FIFO's, the sequence of the packets is maintained.

As mentioned above, the arrangement of the banyans in parallel with input and output buffers has been originally introduced by Newman in the context of unslotted switching systems [New88]. In that work, the use of sequential dispatching was indicated as a viable solution for the switch operation. The same configuration has been described by Sarkies with slotted operation of the switch and sequential dispatching [Sar91]. In this latter work, in case a packet is not successful in a banyan, and has therefore to be dispatched to the following one in the sequence, an other packet from the corresponding input queue is dispatched, in the same slot, to the banyan that the previous packet has not used. Sarkies then studies the maximum throughput of a *single* banyan network under uniform traffic as a function of the number of dispatching attempts of different packets to the banyan that are allowed in a slot (in this study, in order to find the maximum throughput, it is assumed that the inputs are saturated, *i.e.*, the inputs have always packets to dispatch to the banyan, regardless of the number of attempts). It is shown that the maximum throughput improves with the number of attempts; for example, for $N = 64$, the throughput improves from 0.36 for a single attempt to 0.66 for 16 attempts; the improvement in throughput is rapid for small values of the number of attempts (e.g., for $N = 64$, the throughput is equal to 0.58 for 6 attempts), and basically saturates at about 12 attempts. In the same paper, the impact of the overhead introduced by the dispatching attempts on the throughput of the banyan network is also considered; in this case, the *actual* throughput of the banyan is defined as the throughput of the banyan, assuming no overhead, multiplied by $1 / (1 + qh)$, where h is the fraction of time slot that is taken up by one dispatching attempt, and q is the number of attempts (in other words, the actual throughput is the throughput that the banyan would actually offer assuming that it is run at the same speed of input and output lines). In fact, as the number of attempts increases, the throughput of the banyan increases, but also the overhead increases; thus, there is a value of q for which a maximum *actual* throughput is achieved. The author shows the maximum value of the actual throughput as a function of h; of course, the maximum actual throughput decreases for increasing values of h; for example, for N = 64, and $h = 4\%$, the maximum actual throughput is 0.45, for $h = 8\%$, the maximum actual throughput is 0.39. The values of q for which the maximum of the actual throughput is achieved are not explicitly shown, but can be easily derived from the paper's result; in general, for increasing values of h, the maximum is achieved for smaller q; for example, for N = 64, with $h = 2\%$, the maximum throughput (equal to 0.53) is achieved for $q = 10$, for h = 4% the maximum throughput (equal to 0.45) is achieved for $q = 7$, and for $h = 8\%$ the maximum throughput (equal to 0.39) is achieved for $q = 4$. Based on these results on the maximum throughput of a single banyan network, the author then estimates the number of banyan networks in parallel that is required to offer sufficient routing capacity in the switch. For example, assuming $h = 2\%$, since the maximum actual throughput of a 64 x 64 banyan network is 0.53, 2 banyan networks in parallel are sufficient to build a 64 x 64 switch.

The operation of the MSM with buffers at the inputs is a special case of the switch operation described in [Sar91]. Here, we concentrate on the case in which only one packet per input controller can be dispatched to each banyan in a slot (in other words, only a single dispatching attempt per banyan is allowed in a slot). This is motivated by several reasons:

i) as amply evidenced in the remaining of this discussion, already using a single attempt per banyan, the number of banyans necessary to achieve a desired packet loss rate under a wide variety of traffic conditions is rather small; this is also valid when input and output controllers are shared by several lines.

Furthermore, we have studied the performance of the switch using multiple attempts per slot, for different switch sizes under uniform traffic, and concluded that, contrary to what expected from the substantial increase observed by Sarkies in maximum throughput in a single banyan network, by using multiple attempts, the improvement in number of banyans in the switch, although noticeable, is not significant. This is due to the fact that the traffic offered to different banyans is correlated, and the correlation increases with the number of attempts, so that banyans after the first one in the sequence do not actually ever reach a throughput equal to the maximum throughput.

ii) From an implementation point of view, the additional complexity required to support multiple attempts per slot is very substantial, (heavily affecting the attractiveness of the MSM configuration at the implementation level,) and certainly not justified by the rather small decrease in the required number of banyan networks. We also note that, since in practice in a slotted system it is convenient that the time-slot duration be equal throughout the system, the fact that multiple dispatching attempts per slot increase the overhead due to dispatching translates into a higher required clock rate in the banyan networks to accommodate a given data rate. Furthermore, we show below that, in those cases in which the required number of banyan networks is so large that it becomes difficult to handle (***i.e.*****,** in switches where input and output components are shared by many lines), an effective way to manage the banyans (which does not add complexity to the switch) is simply to use more than one group of banyans, each serving a subset of output ports.

iii) with a single dispatching attempt, as mentioned above, the sequence of the packets is easily maintained by using FCFS service discipline in the input buffers; in contrast, with multiple dispatching attempts per slot in each banyan, in order to maintain the sequence of the packets, additional means have to be provided, either in the form of resequencing buffers at the outputs, or by guaranteeing that only packets belonging to different virtual circuits are dispatched to the same banyan in a slot, as proposed in [Sar91].

For these reasons, we are only interested in operating the MSM with each input controller dispatching a single packet to each banyan per slot; our objective is to discuss the scalability of the switch, and present a complete characterization of its performance and implementation issues.

It is interesting to note the similarities between the basic configuration of the MSM and the TBSF. With no buffers in the input controllers, as noted above, the operation of the MSM with sequential dispatching is very similar to that of the TBSF. Accordingly, its performance is also comparable; for example, under uniform traffic at full load, 9 banyans in parallel suffice to achieve a packet loss rate in the routing fabric lower than 10”. (More detailed performance results are reported in Section 3.2 below.) Providing buffering in the input controllers of the MSM in which to store unsuccessful packets, and increasing the load on the banyans is the same principle that motivated recirculation in the TBSF. In contrast with the TBSF, the “recirculation buffers” in the MSM are more naturally placed at the input of the banyans; consequently, it is possible to practically increase the load on all the banyans, instead of only on the last one as in the TBSF. Even more importantly, in the MSM with sequential dispatching, we maintain the sequence of the packets, thus solving one of the major problems of the TBSF with recirculation.

In the MSM with input buffers and sequential dispatching, a minimum of 3 banyans in parallel is necessary to achieve low packet loss rates $(<10^{-6})$ with small input buffer sizes, under uniform traffic at full load. This minimum of 3 networks in parallel is easily anticipated, since the throughput of a single banyan network of size N = 32 at full load is 0.4, making the total capacity with three networks about 1.2. The 20% extra capacity available explains the relatively small buffer sizes needed. Clearly, the MSM achieves similar performance as the TBSF with recirculation. We note, however, that in the MSM packets are dispatched to the same input in every banyan, thus increasing the likelihood of recurrent contention maintained over different banyans. We discuss the implications of this effect and propose a simple solution for it in Section 3.2.2 below.

It is interesting to contrast the performance of the MSM switch operated with sequential dispatching with that of the switch operated with ***random dispatching***, whereby packets are instead randomly dispatched, uniformly and independently, to the banyans for routing. Considering first the case of the MSM with no buffers in the input controllers, we note that, also with random dispatching, the use of the banyans in parallel improves the performance of the switch, and the larger the number of banyans, the better the performance, since as the load on the banyans decreases, the packet loss rate in each banyan decreases. As noted in **[Tob91],** however, the packet loss rate in the switch decreases very slowly with the number of banyans. For example, in an MSM with 10 banyans under uniform traffic at full load, each banyan receives a 0.1 load (since with random dispatching the traffic is equally distributed to the banyans), and its packet loss rate is 1.1×10^{-1}, yielding the same packet loss rate for the entire switch. The packet loss rate remains well above 10^{-2} for K as large as 15. Thus, with no buffers at the input, sequential dispatching in the MSM dramatically outperforms random dispatching.

With buffers at the input, however, the input controllers can actually perform any dispatching algorithm which guarantees that no packet waits in the input buffer if a corresponding input of a banyan is idle. With input buffers, therefore, sequential dispatching does not clearly outperform random dispatching in terms of packet loss rate as in the bufferless configuration; in fact, with input buffers, as soon as the queues grow, the load on the banyan networks increases regardless of the dispatching algorithm used, and the

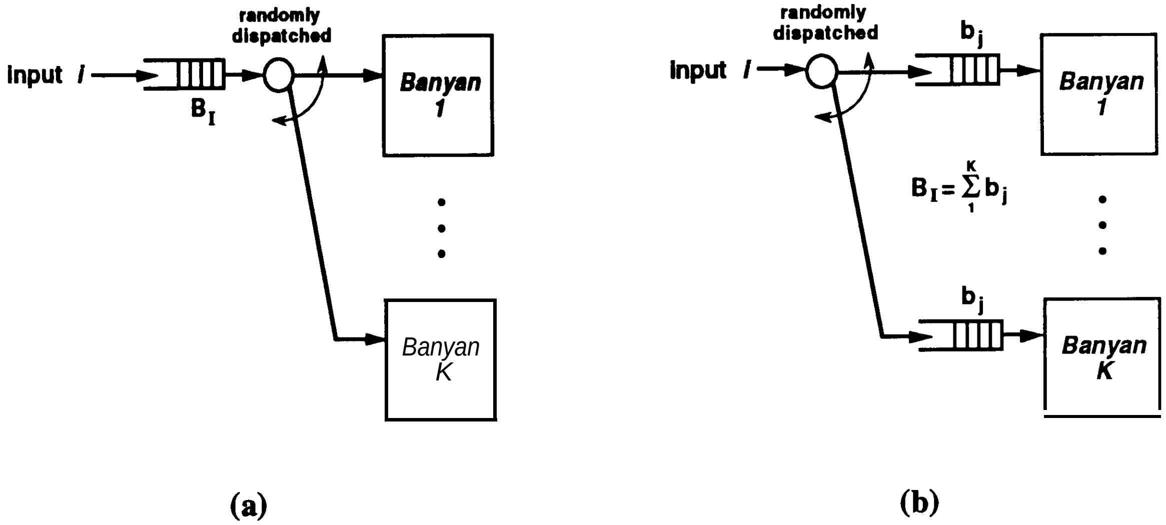


Fig. 3.2. Random dispatching in the MSM switching fabric with input buffers; a) single queue; b) multiple queues.

throughput improves. With input buffers, two random dispatching algorithms, depicted in Fig. 3.2, are conceivable. They are:

i) ***random dispatching with single queue***: packets are stored in a single input queue (see Fig. 3.2.a). In each slot, depending on the number of packets available, as many as K packets per input are randomly assigned to the banyans, independently for each slot. Packets are then routed by the banyans; similarly to sequential dispatching, successful packets are transmitted to the output buffers and unsuccessful packets remain in the input buffers to be processed in the next slot;

ii) ***random dispatching with multiple queues***: a packet, upon arrival at the input controller, is randomly assigned (independently for each packet) to an input queue corresponding to a banyan which will be thereafter in charge of its routing (see Fig. 3.2.b). Given the random assignment, the separate queues are equally and uniformly utilized.

We have studied the performance of the MSM switch operated with both these random dispatching algorithms. As expected, the performance in terms of packet loss rate and queueing delay of sequential and random dispatching are only marginally different. The essential distinction between the algorithms, is that only with sequential dispatching and FIFO operation of the input buffers packet sequence is maintained; with random dispatching, packets may get out-of-sequence, and additional means would be required to restore the sequence of packets. Given that the implementation complexity of the input controllers for the algorithms is comparable, this is certainly sufficient motivation to prefer sequential dispatching over random dispatching in the design of the MSM.

b) Switch Operation

We now describe in more detail the operation of the banyans and identify the important benefits that such an operation scheme brings at the implementation level. In our discussion, we consider the switch operation to be synchronous. All incoming lines have the same transmission capacity, packets have fixed size L (namely, we consider 53-byte ATM packets), and the time axis is slotted with slot size T equal to the transmission time of a packet on a line. We note that, in general, incoming packets from different input lines are not synchronized with each other, nor with the internal system clock. We begin here by assuming that packets on different input lines have been synchronized and aligned with the system slot boundaries. An implementation of how this synchronization may be achieved within the controllers and the banyan networks, thus obviating the need for external aligners, is reported in [Chi93].

As in any ATM switch [JSA91], for each packet, prior to entering the MSM switching fabric, header-conversion circuits generate the local switching header used by the switching fabric, based on the information in the ATM header. As shown in Fig. 3.3, the local header in the MSM comprises three fields:

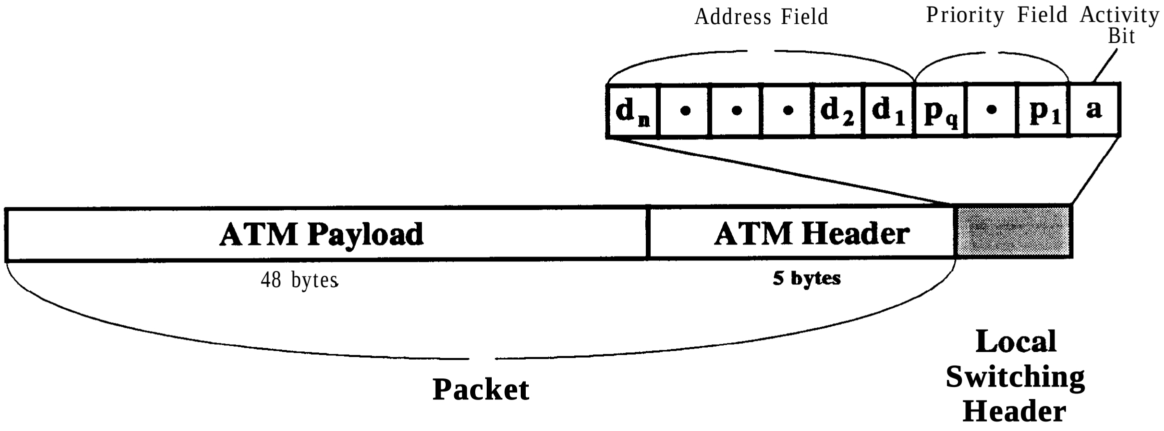


Fig. 3.3. Local switching header in the MSM switching fabric, basic configuration.

- Activity Bit a : it indicates whether the current slot contains a packet ($a = 1$), or not ($a = 0$).

- Priority Field P : this field is optional and is used only if multiple priority classes exist. It comprises $q =$ log, Q bits, where Q is the number of external priority levels.

- Address Field D: This field contains the address of the output port $d_1, d_2, \cdots, d_n$ (where n = log, N, and d_1 is the most significant bit); it is inferred from the virtual circuit information in the ATM header.

Once the local header is generated, packets are sent to the switching fabric, stored in the corresponding input buffers, and then dispatched to the banyans. To perform the dispatching algorithm, in every slot the operation of a banyan is divided in two phases:

1) ***Route Set-up Phase.*** During this phase, the local switching header portions of the packets (referred in the following simply as ***headers***) are routed through the banyan, establishing the paths in the interconnection network. At any stage s in the banyan, the proper setting of a switching element is a function of two bits in each header present at its inputs, namely a and d_s (if external-priority service is in effect, then the setting is also a function of the priority fields in the headers). An active header (identified by $a = 1$) arriving at a switching element requests to be routed to the upper output if $d_s = 0$, and to the

lower output if $d_s = 1$. Anytime there is a conflict between two active headers at a switching element (*i.e.*, d_s-bits are equal for the two headers), one of the two headers is routed properly, while the other is forced in the "wrong" direction and has its activity bit reset (clearly, this conflict resolution follows a similar idea to that used in the TBSF). The selection of the "winner" can be random, but more practical deterministic selection algorithms can be adopted with no effect on performance, as discussed below. A non-active header (identified by $a = 0$), *i.e.*, either a misrouted header or an empty slot, is routed accordingly to the current state of the switching element (determined by the other header, if active, or by the state of the switching element in the previous slot).

Once the headers reach the outputs of the banyan network, the outcome of the route set-up phase (*i.e.*, which headers have been properly routed by the network) is described by the activity bit of the headers: headers with $a = 1$ at the output of the network have been successfully routed; headers with $a = 0$ have been misrouted (or represent empty slots). The outcome of the route set-up phase is fed back to each controller at the input by having the switching elements in the last stage of the banyan network transmit backward the activity bit of each header, all the way to the input controllers, through the paths that have just been set-up.

2) *Packet Transmission Phase.* Each input controller examines the feedback signal and determines whether the header has been routed properly or not. If the header has been successful (feedback signal = 1), the controller transmits the packet (*i.e.*, the ATM header and payload) to the output buffers.

The operation of the banyans with sequential dispatching is depicted in Fig. 3.4. At the end of the route set-up phase, if an input controller determines that the header currently in process has been properly routed by a banyan, it transmits the corresponding packet to that banyan, and dispatches the next header in the input queue to the remaining banyans in the sequence. If a header has been unsuccessful in a banyan, it is simply

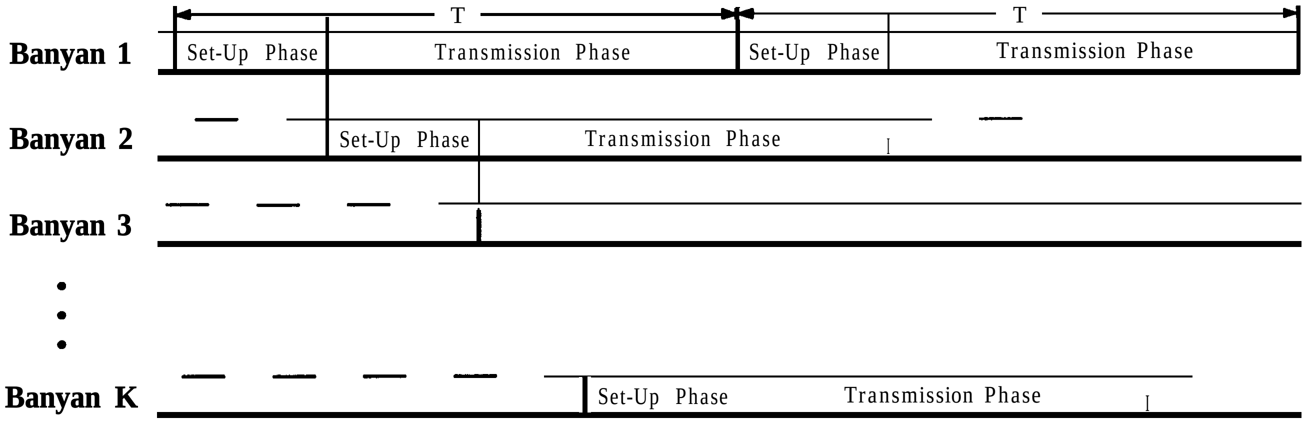


Fig. 3.4. Operation of the banyan networks with sequential dispatching.

dispatched to the following one. Packets which are unsuccessful after the last banyan has been attempted, remain in the input buffers and are processed in the following time slot. Up to ***K*** packets can be routed from each input to the outputs in a time slot.

The two-phase operation brings important benefits at the implementation level. In particular, it implies that different clock rates may be used in each phase. In fact, during the route set-up phase, the logic in charge of the switching algorithm imposes the critical constraints on the clock rate. On the contrary, during the packet transmission phase, no logic is involved, and bit streams can flow "freely" to the outputs. This means that hardware constraints and achievable clock speeds during different phases may differ substantially, and the maximum achievable clock rate in the route set-up phase may be lower than the maximum achievable clock rate in the transmission phase. The use of two clock rates makes it possible to support the 622.08 Mb/s data rate with conventional VLSI technology; at lower data rates, the use of two clock rates simplifies the design and reduces the area of the banyan network. Furthermore, with the two-phase operation, the state of the switching elements is set prior to the transmission of the packets. Thus, within each switching element, the path followed by the packets does not need to include the storage elements occupied by the control bits needed to perform the switching algorithm; this reduces considerably dynamic power dissipation internally to the chip. The two-phase operation brings important benefits also at the system-synchronization level. In fact, strict synchronization and alignment in the banyan networks is required only among the headers during the route set-up phase. At the beginning of the transmission phase, the paths in the banyans between inputs and requested outputs are fully established, and the data streams can flow independently from each other, provided that synchronization is maintained between pairs of connected inputs and outputs in the banyan. These implementation issues are discussed in more detail in [Chi93].

3.1.2. Large Size MSM Switching Modules

In order to increase the size of the MSM switch beyond the achievable size of the banyan network, we observe that, given their relatively simple functionality and small buffer requirements, the input components actually have the capacity to handle more traffic than that coming from one input line and to control a number of banyans larger than that needed to sustain the traffic from a single line; similarly, the output components have the capacity to handle more traffic than that destined to a single output line. Therefore input and output components can be shared by several (say M) inputs and outputs, respectively, making the total size of the switching fabric equal to $M \cdot N$. The resulting configuration is shown in Fig. 3.5. The switch operates as follows: in each input controller, packets arriving at its M input lines are first multiplexed into a shared buffer; packets are then sequentially dispatched to the banyans and routed to their requested output shared buffers; finally, in each output component, packets are switched to their requested output links selected among the M output ports served by the buffer. In this case, the first log, N bits of the address field D in the local switching header are used for routing the packets in the banyan networks, and the remaining log, M bits in the address field specify the desired output port among the M ports served by the corresponding output buffer “switch”.

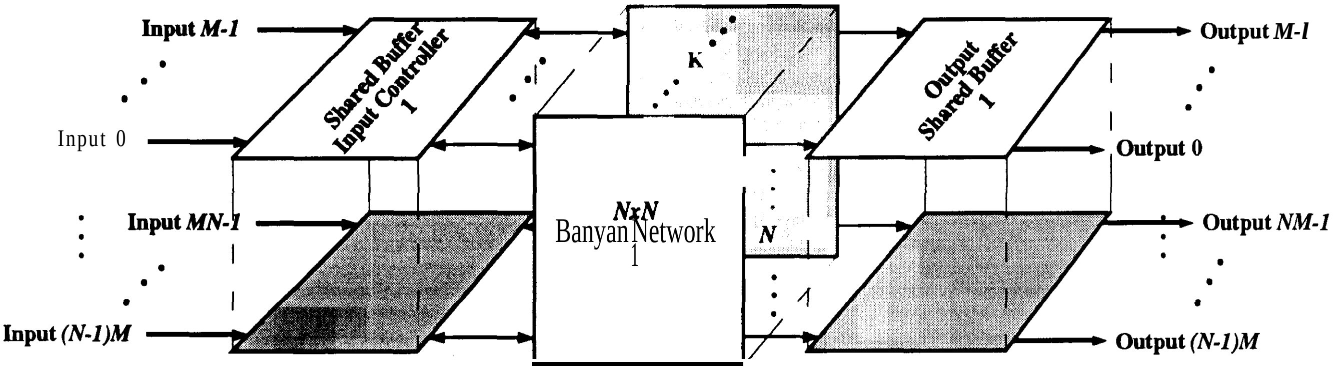


Fig. 3.5. The MSM switching fabric for large fast packet switches.

Since the routing capacity offered by the banyans in parallel must be adequate to sustain the aggregate traffic from the ***M*** links entering each input controller, the required ***K*** to maintain a given packet loss rate in the input buffers increases with ***M***. As a rule of thumb, for N = 32 and N = 64, 3 banyans for each of the input ports sharing an input controller are needed to achieve very low packet loss rates in the input buffers with relatively **small** sizes of the input shared buffers B_I, under uniform traffic at full load (again, this rule of thumb is easily anticipated, since the throughput of a banyan network with N = 32 and N = 64 at full load is 0.40 and 0.36, respectively, making the total capacity with three networks larger than 1, and the total capacity with 3*M* networks larger than *M*). For example, in a 256 x 256 switch with ***M*** = 8, N = 32, and ***K*** = 24 banyans, B_I = 30 buffers are needed to achieve a packet loss probability equal to 10^{-7}, under uniform traffic at full load. If more banyans are used, smaller buffer sizes suffice for the same packet loss rate. Similar results are achieved under other traffic patterns. A complete discussion of the design of MSM switches of different sizes for various traffic conditions is presented in Section 3.2 below.

In the MSM switching fabric, the number of input lines that can be fed to an input controller and the number of output lines that can be served by an output shared buffer are limited by four factors:

i) the bandwidth requirements of input and output memories, respectively;

ii) the functionality to be performed in the input controller and in the output buffer, respectively;

iii) the buffer requirements B_I and B_O, respectively, necessary to guarantee a desired packet loss rate in the fabric; and

iv) the number of dispatching operations to different banyans that can be completed within a slot duration.

Here, we briefly discuss the nature and extent of these limitations in the input and output components.

The memory bandwidth in the input components must be adequate to sustain a flow of M packets into the memory and K packets from the memory per slot; the memory bandwidth is therefore equal to $(M + K)V$, where V is the line speed. However, we show in Section 3.2.3 below that it can be reduced to $(2M + 1)V$ with no noticeable degradation in performance. The memory bandwidth in the output components is equal to $(M + K)V$.

In the input components, with sequential dispatching and related FIFO operation, the shared memory is a relatively simple sequentially-accessed memory. The memory control must be capable of synchronizing and multiplexing M incoming packets into the memory, of retrieving up to K packets from the memory per slot (again, the maximum number of retrieved packets per slot can be limited to $M+ 1$ with no observable performance degradation), and of delivering them to the corresponding banyans in which paths have been set up. The functionality of the input controllers has to include the implementation of the sequential dispatching algorithm to orchestrate the operation of the banyan networks. Given the simplicity of the algorithm, however, this limiting factor is not as important as synchronization, bandwidth and memory access requirements.

In contrast, the output buffer is a shared-memory switch serving M ports, and therefore is a randomly-accessed memory in which M linked lists have to be managed. The memory control logic has to synchronize and enqueue in the requested queues up to K incoming packets, and retrieve from the memory and transmit to the proper output ports M outgoing packets per slot.

The buffer requirements are fairly modest in the input controllers under any traffic condition of interest, as shown in more detail in the following section. On the contrary, since the MSM achieves output buffering, if the switch is designed to sustain bursty traffic, the local buffer requirements in the output components are large, thus becoming an important concern.

Finally, the objective of maintaining packet sequence poses an additional limitation on the number of lines served by input and output components, since the sequential scanning of all the K banyans must be completed within a slot duration T in order to maintain FIFO operation of the input buffers. Therefore, the maximum number of banyans that can be placed in parallel is:

$$K_{\max} = T / T_s = \frac{T_s + T_t}{T_s} \tag{1}$$

where T_s is the duration of the set-up phase and T_t is the duration of the transmission phase in a banyan. Thus, in practice, the limitation on $K_{\max}$ comes from the achievable clock rate during the set-up phase. It is interesting to relate this limitation on $K_{\max}$ with the limitation on the number of mini-slots required to perform the routing algorithm in the Growable Switch [Eng89], described in Section 2.2 above. The important difference is that here the limitation is on the number of banyans in parallel that can be provided, while in the Growable Switch the limitation is on the number of output modules to which an input controller can route packets. This is due to the fact that in the MSM, during a route set-up phase, each input controller routes packets destined to ***any*** output module through a ***specific*** banyan, while in the Growable Switch, during a mini-slot, each input routing module routes packets destined to a ***specific*** output module through ***any*** crossbar in the intermediate stage. From a practical point of view, we expect that, with current technology, the minimum achievable duration of the route set-up phase in a banyan be shorter than the minimum achievable duration of a mini-slot.

This discussion suggests that, as far as the limitations on bandwidth, functionality and buffer requirements are concerned, the achievable number of input lines M in a controller may be larger than the achievable number of output lines R in an output buffer. In the configuration of Fig. 3.5, however, for a given N, the more stringent limitations on R would determine the achievable switch size; furthermore, in this configuration, we note that, since K is a function of M, the limitation on K imposed by the requirement to complete the dispatching to all the banyans within a time slot directly translates into a limitation on M (and on R).

To further increase the maximum achievable switch size, we therefore propose the general configuration for the MSM, shown in Fig. 3.6. In this configuration, each input controller receives M input lines and is connected to G groups of K_g banyans each (the total number of banyans is still defined as $K = GK_g$). In each group, each of the corresponding output buffers has R output ports, where $R = M/G$. With G > 1, the input controller first selects the desired output group to which each of the packet is destined (specified by the first log, G bits in the address field of the local switching header) and enqueues the packet in the corresponding input queue. Then the operation proceeds as described above, by applying the dispatching algorithm to each of the G groups of K_g

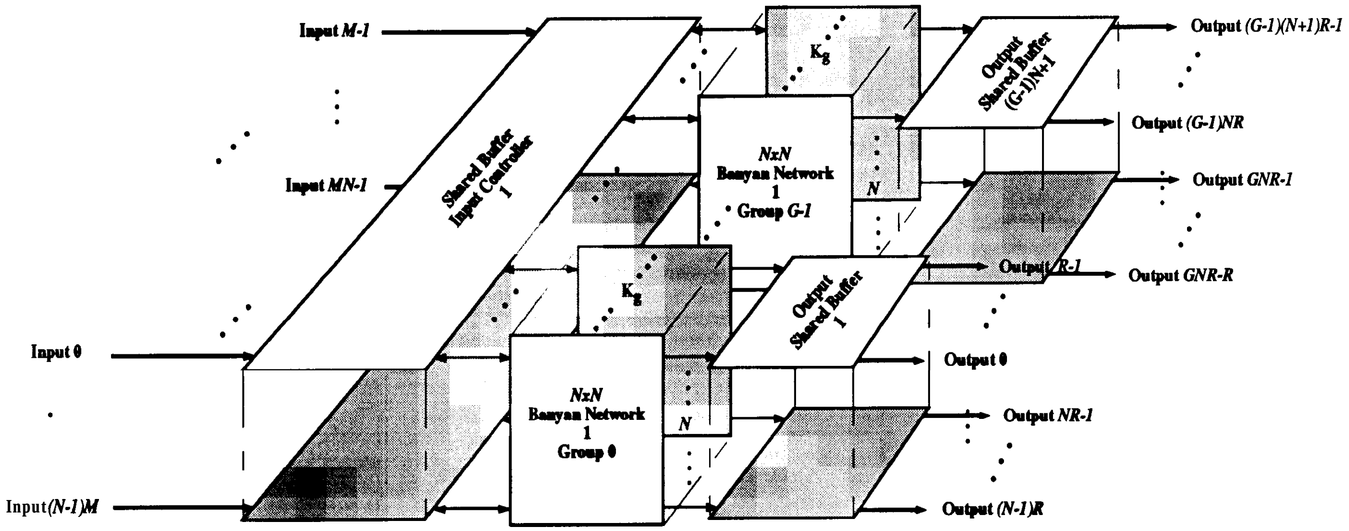


Fig. 3.6. The MSM switching fabric: general configuration, $M/R = G > 1$.

banyans. Packets are routed to their requested output shared buffer, and then switched to the requested output port selected among the R ports served by the buffer. In this general case, as a rule of thumb, in each group, 3 banyans for each of the R output ports connected to an output shared buffer are necessary to achieve low packet loss rates in the input buffers with small buffer sizes. In general, for increasing values of G, with constant total number of banyans K, the required input buffer size to achieve a desired packet loss rate increases slightly; alternatively, more banyans have to be used.

In the general configuration, the bandwidth of the output shared memories is reduced to $(K_g + R)V$. The output components have only R output ports, thus reducing the complexity of the memory management and the local buffer requirements in each component. In the input controllers, the shared memory accommodating the G input queues has still bandwidth $(M + K)V$ (which can be reduced to $(2M + G)V$ with no observable degradation in performance). If the bandwidth is not manageable, completely partitioned memories for each group can be used, each with bandwidth $(M + K_g)V$ (which can be limited to $(M + R + 1)V$, with only a modest increase in the total input buffer size, as shown in detail in the next section.

The limitation on the number of banyans in parallel coming from the requirement to complete the scanning of all the banyans within a time slot is significantly ameliorated since, given that the dispatching algorithm is performed on each group in parallel, only K_g set-up phases (as opposed to K set-up phases as in the previous configuration) have to be completed in a time slot. With this configuration, therefore, we can individually adjust the design parameters M, R, N, G, K_g, B_I and B_O to accommodate the technology constraints and achieve the desired size and performance.

It is interesting to contrast this solution of using more than one group of banyan networks in the MSM in order to overcome the limitation on K_{max} given by (1), with what can be done in the Growable Switch [Eng89] in order to overcome the limitation on the number of mini-slots required to perform the routing algorithm. In the Growable Switch, since the limitation is on the number of output modules to which an input controller can route packets, decreasing the size of the output modules by G, and separating the outputs into G groups in a similar way as in the MSM, would leave the limitation unaffected, since the number of output modules in each group is the same as the number of modules in the switch with G = 1. However, we observe that, in order to overcome the limitation on the number of output modules in the Growable Switch, the outputs could be partitioned into G groups in a different way, by keeping the size of the modules the same as in the switch with G = 1, dividing the modules into G groups, and providing separate Clos networks (which share the input modules) for each group. Therefore, as the switch with G = 1, described in Section 2.2 above, consists of K input modules of size $n \times m$, m crossbars of size $K \times K$, and K output modules of size $m \times n$, *the* switch with G > 1 consists of: (i) K input modules with n inputs and G groups of m'outputs, and (ii) G separate networks, each consisting of m'cross-connects of size $K \times K / G$ and K / G

output modules of size $m' \times n$. The number of paths m' per input/output pair in each group (equal to the number of cross-connects per group) must be chosen to guarantee a desired packet loss rate, as explained in Section 2.2; m' is equal to the number of paths required in a switch of size G times smaller the size of the switch with no partitioning of the modules. It should be noted that, since the expansion factor m' / n in the Growable switch increases for decreasing values of the switch size, m' is larger than m / G. Thus, for large G, the total number of cross-connects may be substantially larger than that needed in the switch with G = 1. In contrast, as shown in the following of this discussion, the total number of banyans in the MSM is only weakly dependent on the number of groups. We also note that, using G > 1 in the Growable Switch, only handles the limitation on the number of output modules, but does not change any limitation caused by local buffer requirements in each module (since the module size remains the same); also, the required memory bandwidth in the output modules is scarcely affected using G > 1, decreasing from $(m+n)V$ in case G=lto $(m'+n)V$ for $G>1$.

3.2. Performance of the Memory/Space/Memory Switching Fabric

In this section, we discuss how to design an MSM switching fabric of a certain size to achieve a desired packet loss rate under given input traffic characteristics. For a given switch size, different combinations of the parameters M, N, and G can be used. Packet loss in the MSM occurs both at the input and at the output buffers. For given switch size and traffic characteristics, the number of packets lost in the input buffers is a function of the number of banyan networks K, the size[1] of the input buffers B_I, and the dispatching algorithm used; the number of packets lost in the output buffers is a function of the output buffer size B_O and the rate of arrival of successfully routed packets in the output buffers. Given that packet loss occurs first in the input buffers, it is necessary to ensure that K and B_I are adequate to meet the desired packet loss rate. Thus, we begin

[1] Throughout this paper, unless otherwise stated, the buffer sizes are given in packets.

first by assuming infinite output buffers, and study how to choose the number of banyans and the size of the input buffers to achieve a desired packet loss under given traffic conditions. We first consider uniform traffic, and then analyze traffic patterns exhibiting correlation in the space domain (*e.g.*, ***communities-of-interest*** traffic pattern) and in the time domain (e.g., ***bursty*** traffic pattern). We also investigate MSM switches in which the maximum number of packets that can be retrieved from the input buffers in a slot is lower than the number of banyans in parallel (referred to as configurations with ***input-memory bandwidth limitation***), and MSM switches in which the banyan networks are replaced by crossbars. Finally, we study how to size the output buffers to achieve a desired packet loss in the various traffic conditions. The simulation results presented below have a 95% confidence interval lower than 10^{-7}. For packet loss rates lower than 10^{-6}, the 95% confidence intervals are at least an order of magnitude smaller than the packet loss rate values.

3.2.1 Uniform Traffic

a) M = 1

The packet loss rate in the routing fabric of the MSM with no input buffers and sequential dispatching, N = 32, under uniform traffic, at various loads, is shown in Fig. 3.7. As expected, the MSM with sequential dispatching achieves similar performance as the TB SF [Tob90b,Tob9 1].

The packet loss rate in the MSM with buffers at the input, sequential dispatching, and $M = 1$ is shown in Fig. 3.8 under uniform traffic at full load[2], for N = 32. A minimum

[2] Throughout this paper, we assume that packet(s) currently under process are actually stored in the input queues (e.g., the MSM with no input buffers corresponds to $B_I = 1$ in Fig. 3.8). **We also assume that the dispatching of a packet is not initiated until the packet has been received and stored in the input buffer in its entirety (consequently, a packet arriving at an empty input queue incurs a minimum queueing delay at the input of 1 time-slot). Similarly, transmission from the output queues is not started until the packet has been fully stored in the output buffer, making the minimum switching delay equal to 2 time slots.**

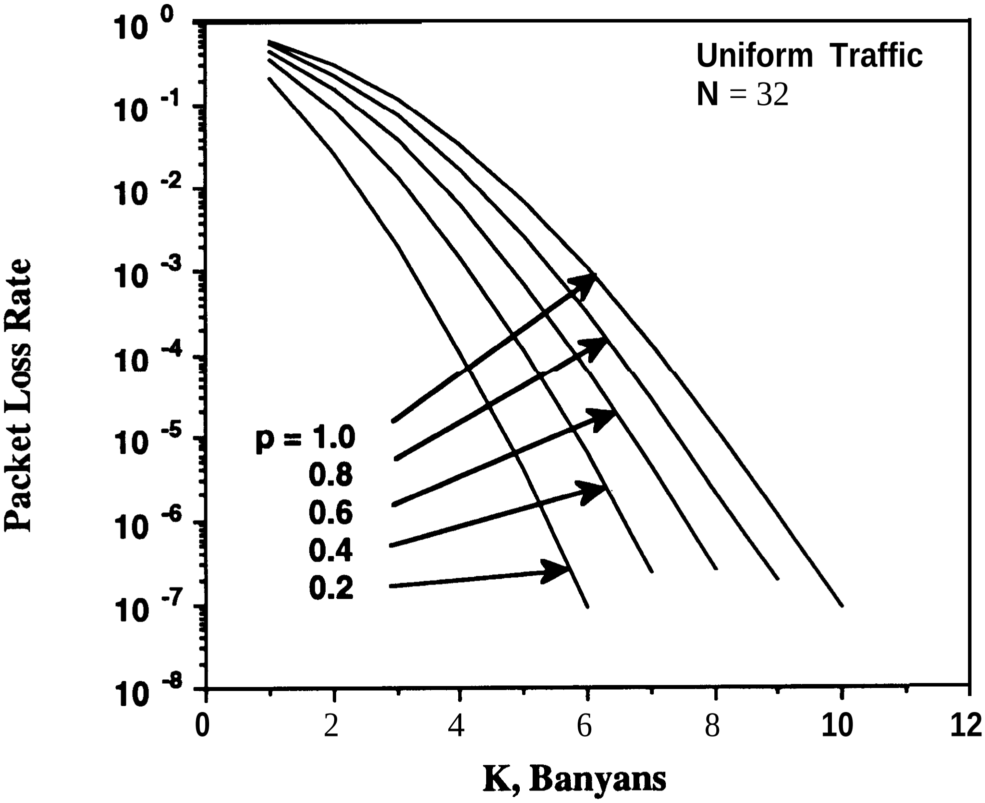


Fig. 3.7. Packet loss probability in the MSM routing fabric with no input buffers and sequential dispatching, under uniform traffic, M = 1, N = 32 (obtained by simulation).

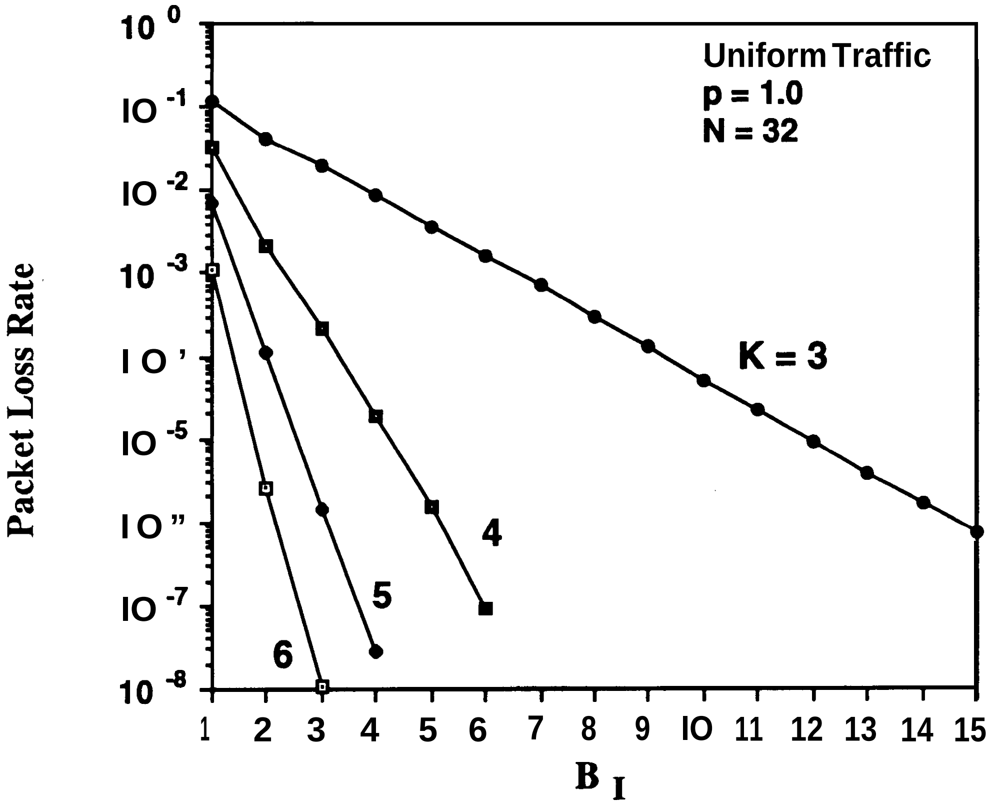


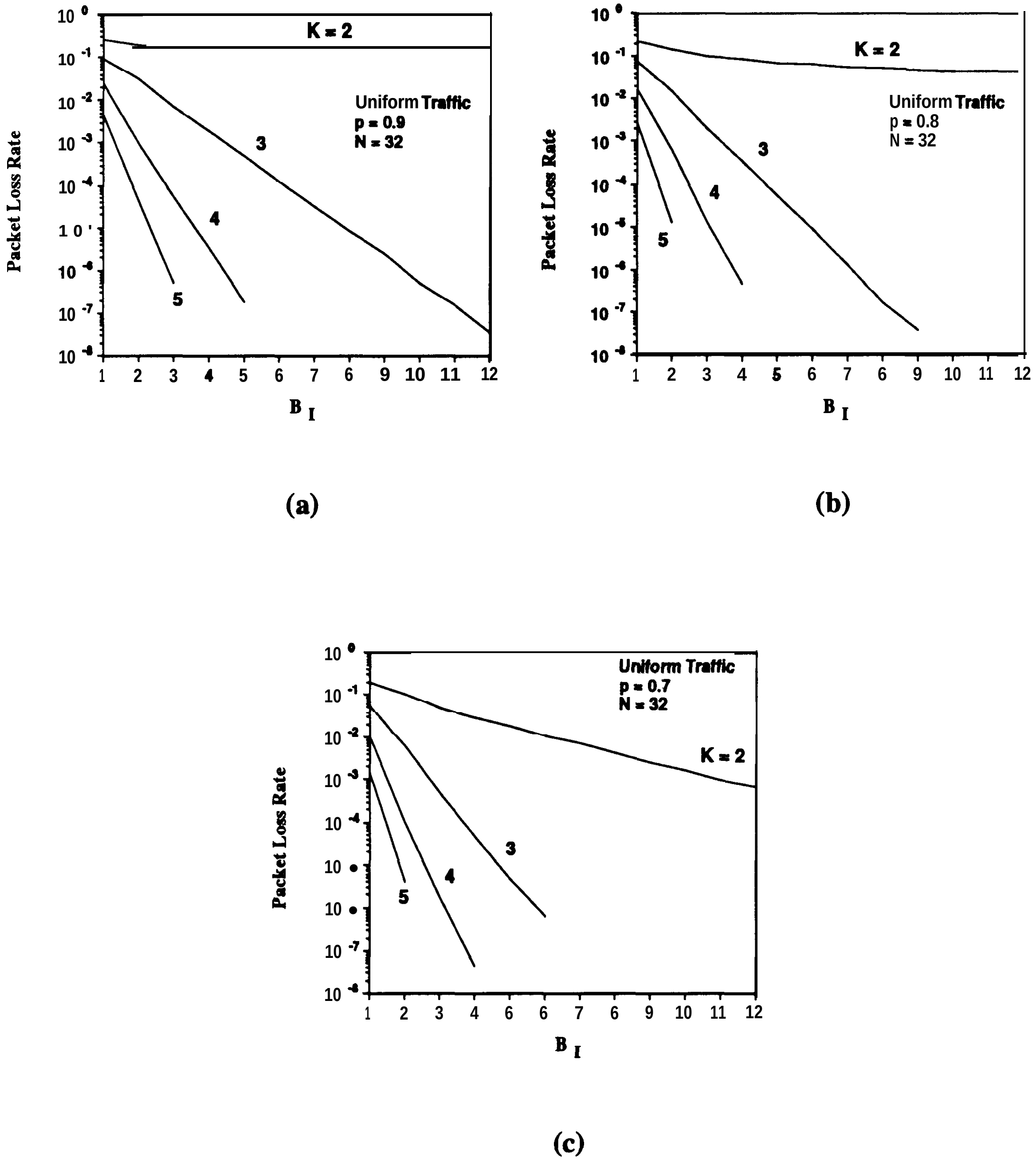
Fig. 3.8. Packet loss rate in the input buffers of the MSM switching fabric with input buffers and sequential dispatching, under uniform traffic at full load, $M = 1$, N = 32 (obtained by simulation).

of 3 banyans in parallel is necessary to achieve low packet loss rates with small input buffer sizes. With 3 banyans in parallel, 15 input buffers are necessary for 10^{-6} packet loss rate. Using more banyans, the required buffer size for a desired packet loss rate decreases. With 4 banyans, 6 input buffers offer a 10^{-7} packet loss rate. With K = 5, 4 input buffers suffice for the same loss rate. The packet loss rate in the switch under uniform traffic for lower loads is shown in Fig. 3.9. With load = 0.7, a minimum of 2 banyans is required to achieve low packet loss rates with small input buffer sizes. As anticipated above, the performance of the MSM is similar to that of the TBSF with recirculation.

b) M > 1

We have studied the design of MSM switches with $M > 1$ in order to achieve a desired packet loss rate under uniform traffic. Here, we show the performance of switches of several sizes for different choices of the design parameters $M, R, N, K,$ and B_I.

The packet loss rate in the input buffers of a 128 x 128 MSM switch with $M = 4$ and N = 32, for different R, as a function of the total buffer size per input controller B_I, under uniform traffic at full load, is shown in Fig. 3.10. For G > 1, the memory is shared among the group queues (the case of partitioned memories is studied below). As mentioned above, as a rule of thumb, $K = 3M$ banyans are necessary to obtain very low packet loss rates with small input buffer sizes. Of course, as more banyans are used, the required input buffer size to achieve a desired packet loss rate decreases. For example, with K = 12, 22 buffers per input controller are necessary for a packet loss rate lower than 10^{-6}. With K = 13, 14 buffers suffice for the same packet loss rate. For G > 1, the number of banyans necessary to achieve low packet loss rates with small input buffer sizes is roughly $K_g = 3R$. In general, as the number of groups increases, the required buffer size to achieve a given packet loss rate increases slightly; alternatively, more banyans have to be used. For

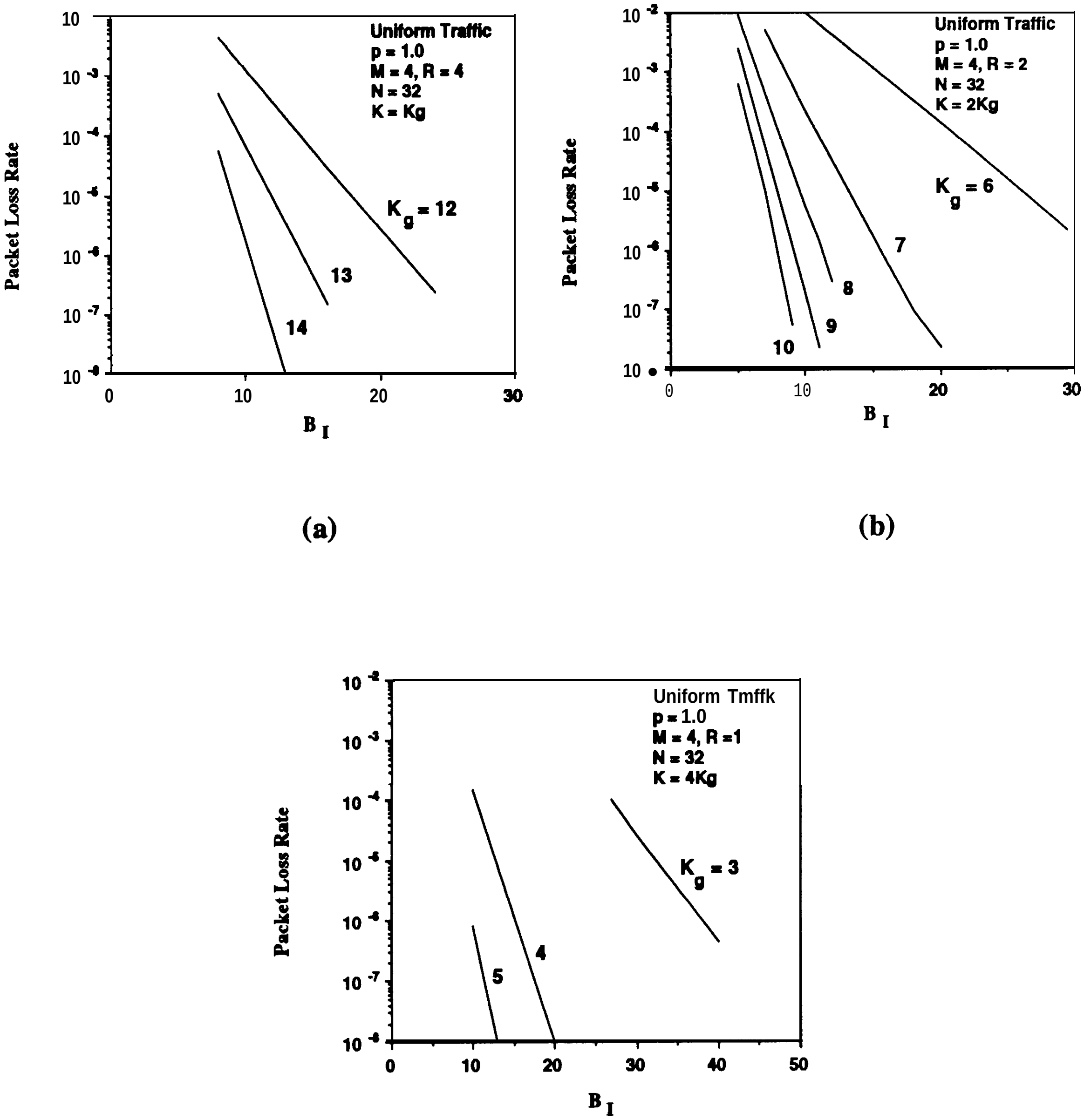


(a)

(b)

(c)

Fig. 3.9. Packet loss rate in the input buffers of the MSM switching fabric with input buffers and sequential dispatching, under uniform traffic and various loads, *M* = 1, N = 32; a) *p* = 0.9; b) *p* = 0.8; c) *p* = 0.7 (obtained by simulation).



(a)

(b)

(c)

Fig. 3.10. Packet loss rate in the input buffers of a 128 x 128 MSM switch, under uniform traffic conditions, $p = 1$, $N = 32$; a) $M = 4$, $R = 4$; b) $M = 4$, $R = 2$; c) $M = 4$, $R = 1$ (obtained by simulation).

example, for the 128 x 128 switch we could use 2 groups of 6 banyans with 31 input buffers, or 4 groups of 3 banyans with 38 input buffers for a packet loss rate below 10”.

For a 256 x 256 MSM switch, several choices of the design parameters are possible; namely, we can choose M = 8, N = 32, or M = 4, N = 64, or even M = 8, N = 64, and use only every other inputs and outputs of the banyan networks. The packet loss rate in the input buffers of a switch with M = 8 and N = 32 is depicted in Fig. 3.11 for different R. With R = 8, 24 banyans and B_I = 30 packet buffers give a 10^{-7} packet loss probability. With R = 4, for the same $K = 2K_g$ = 24 banyans (i.e., with two groups of K_g = 12 banyans each), 44 buffers are needed for the same loss rate; conversely, for the same B_I = 30, two groups of 13 banyans are required. An alternative design choice for the 256 x 256 MSM is N = 64, M = 4; the corresponding packet loss rate in the input buffers is shown in Fig. 3.12. With N = 64, since the throughput of a single banyan network of such size at full load is 0.36 (instead of 0.40 as for N = 32), the switch requires basically the same number of banyans of a switch with N = 32 and the same M and R (i.e., a 128 x 128 switch in this case), but larger input buffers for a given packet loss rate. The former parameter choice [N, M] = [32, 8], for the switch of size 256 requires approximately twice as many banyan networks but half the number of input and output components than those required by the latter parameter choice [64, 4] (with the former choice of parameters, the input and output buffers need about twice the memory bandwidth required by the latter choice). Given that input controllers capable of handling M = 8 lines are practical, the clear design choice to minimize the number of hardware components is M = 8 and N = 32. However, given that 64×64 banyan networks are also feasible, the actual solution of choice for a 256 x 256 switch would use the 64 x 64 networks (since presumably the cost of a 64 x 64 banyan chip is the same as a 32 x 32 chip), and connect the input components with M = 8 to every other input in the banyans (and similarly connect the output components to every other output in the banyan). It is straightforward (by applying a similar approach to that presented

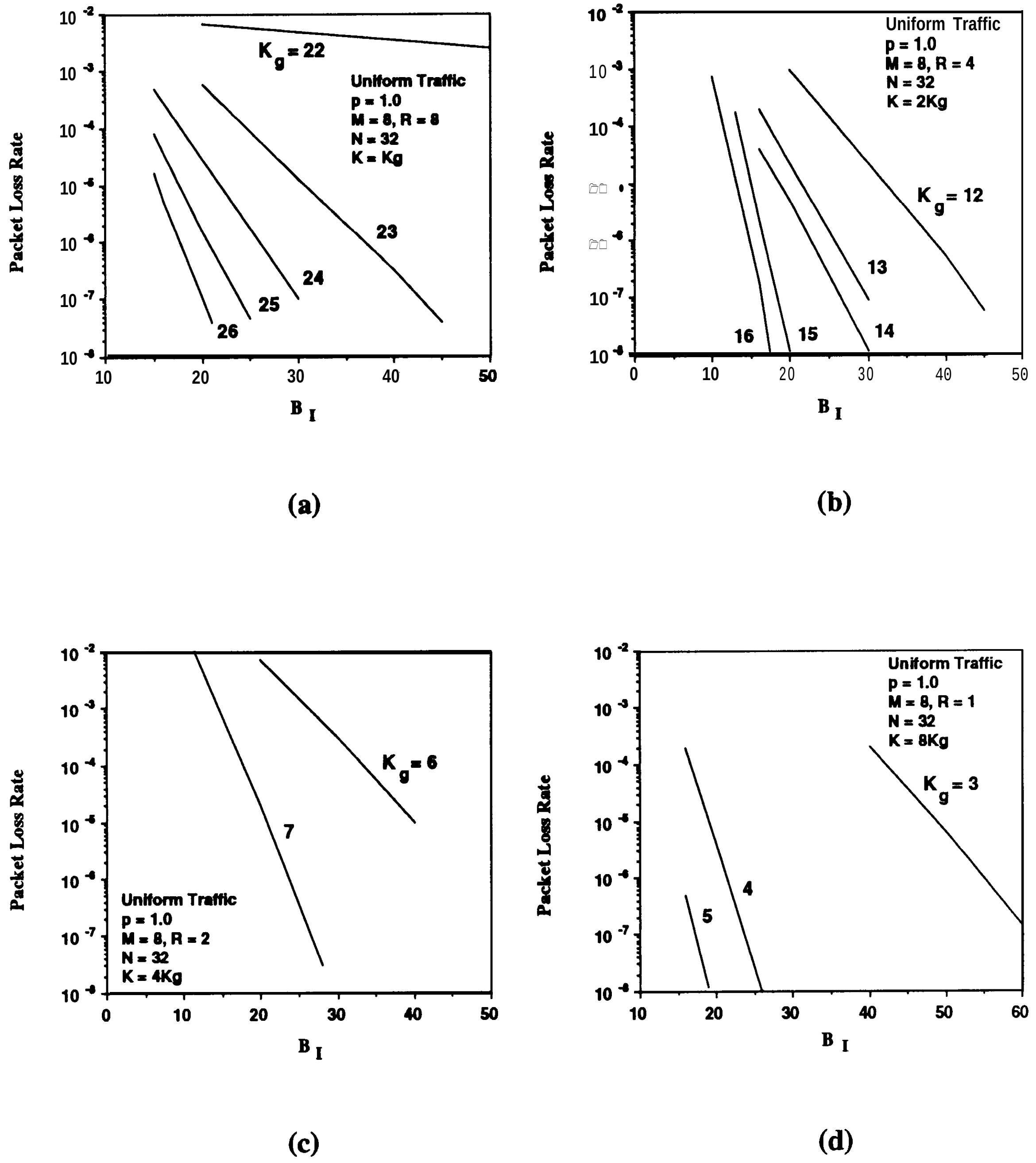


Fig. 3.11. Packet loss rate in the input buffers of a 256 x 256 MSM switch, under uniform traffic conditions, $p = 1, N = 32$; a) $M = 8, R = 8$; b) $M = 8, R = 4$; c) $M = 8, R = 2$; d) $M = 8, R = 1$ (obtained by simulation).

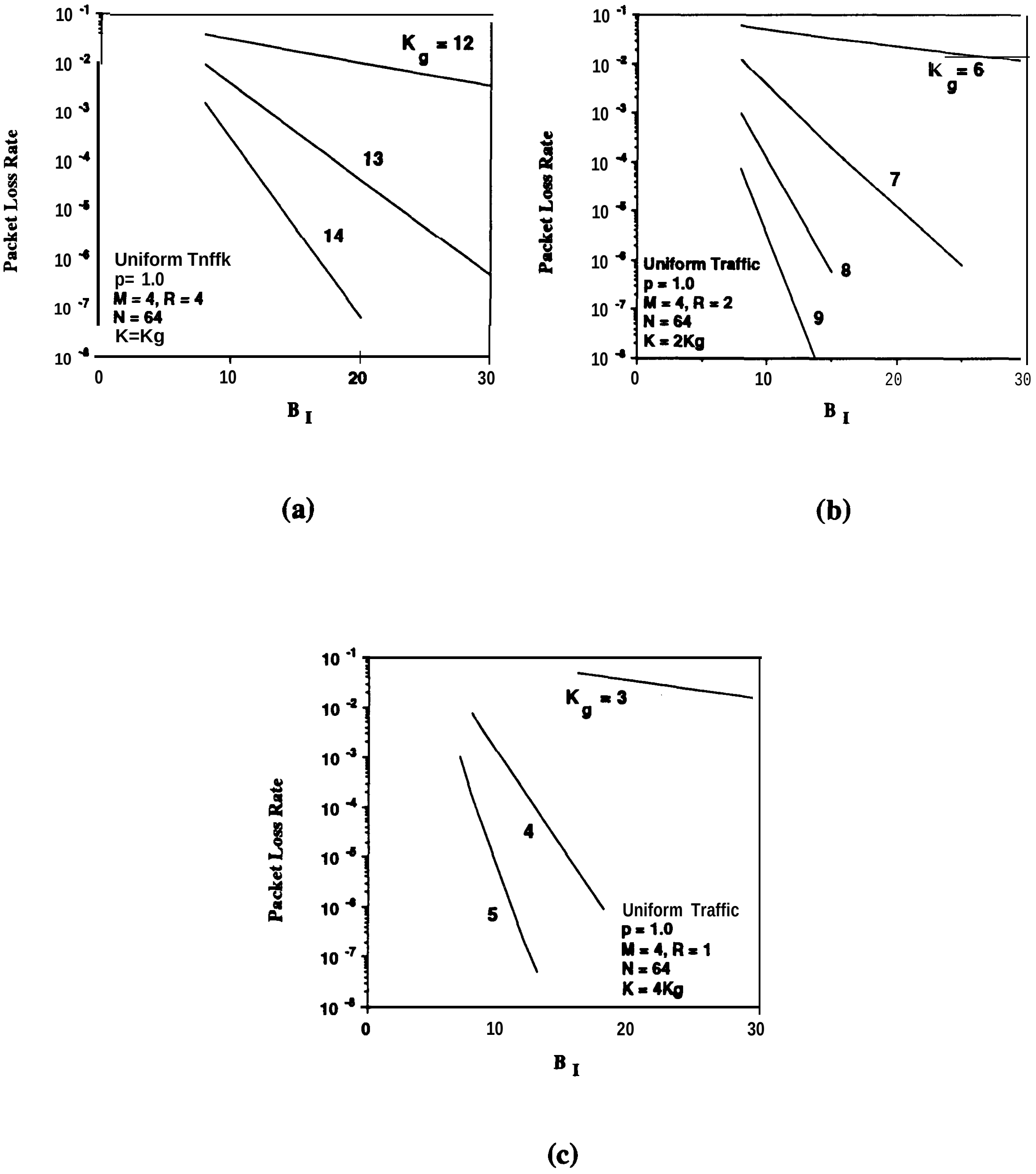


Fig. 3.12. Packet loss rate in the input buffers of a 256 x 256 MSM switch, under uniform traffic conditions, $p = 1$, $N = 64$; a) $M = 4$, $R = 4$; b) $M = 4$, $R = 2$; c) $M = 4$, $R = 1$ (obtained by simulation).

in [Pat81]) to derive that the throughput of the banyan network in this configuration is 0.54. Consequently, the minimum number of banyans necessary to achieve low packet loss rates at full load is, as a rule of thumb, $2M$ (as opposed to $3M$ needed by using banyans of size 32), as shown in Fig. 3.13 for $M = R = 8$.

As mentioned above, for $G > 1$, accommodating the group queues in partitioned input buffers rather than in shared memories is a practical solution to reduce the complexity of the input components (in fact, with partitioned memories and sequential dispatching, the buffers are sequentially-accessed FIFO memories, instead of random-access memories shared by the queues). In Fig. 3.14, we compare the input buffer requirements for shared and partitioned input buffers in a MSM of size 256, with $M = 8$ and N = 32 and different G, under uniform traffic at full load. For G = 2, partitioned buffers need only modest increases in the buffer sizes per port with respect to shared buffers, since the benefit due to sharing is moderate at small G. For larger G, the sharing effect is of course more marked, but the required buffer sizes per port with partitioned buffers remain always easily manageable. For a given G, the benefit of sharing decreases as K_g increases, since the variance of the group-queue lengths decreases. From a practical point of view, implementing partitioned input buffers often greatly reduces the limitations on K_g, so that if the penalty in buffer size is a concern, increasing K_g slightly is a viable option to reduce the buffer size.

3.2.2. Correlated Traffic Patterns

Since the throughput of a banyan network is very sensitive to the input traffic pattern, one possible cause of concern in the MSM switching fabric is the fact that the packets, during their sequential scanning of the banyans, enter every banyan in the sequence from the same input. Under traffic patterns exhibiting correlation (in the space or in the time domain) among packets, this may lead to recurrent conflicts between the same

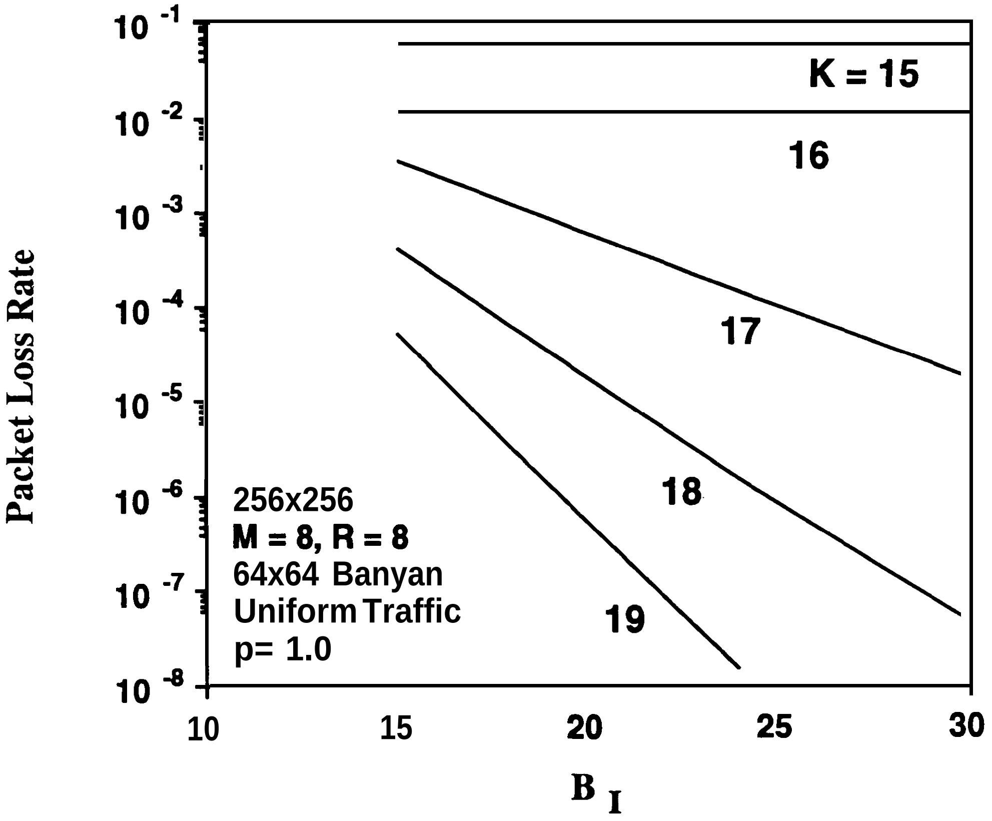
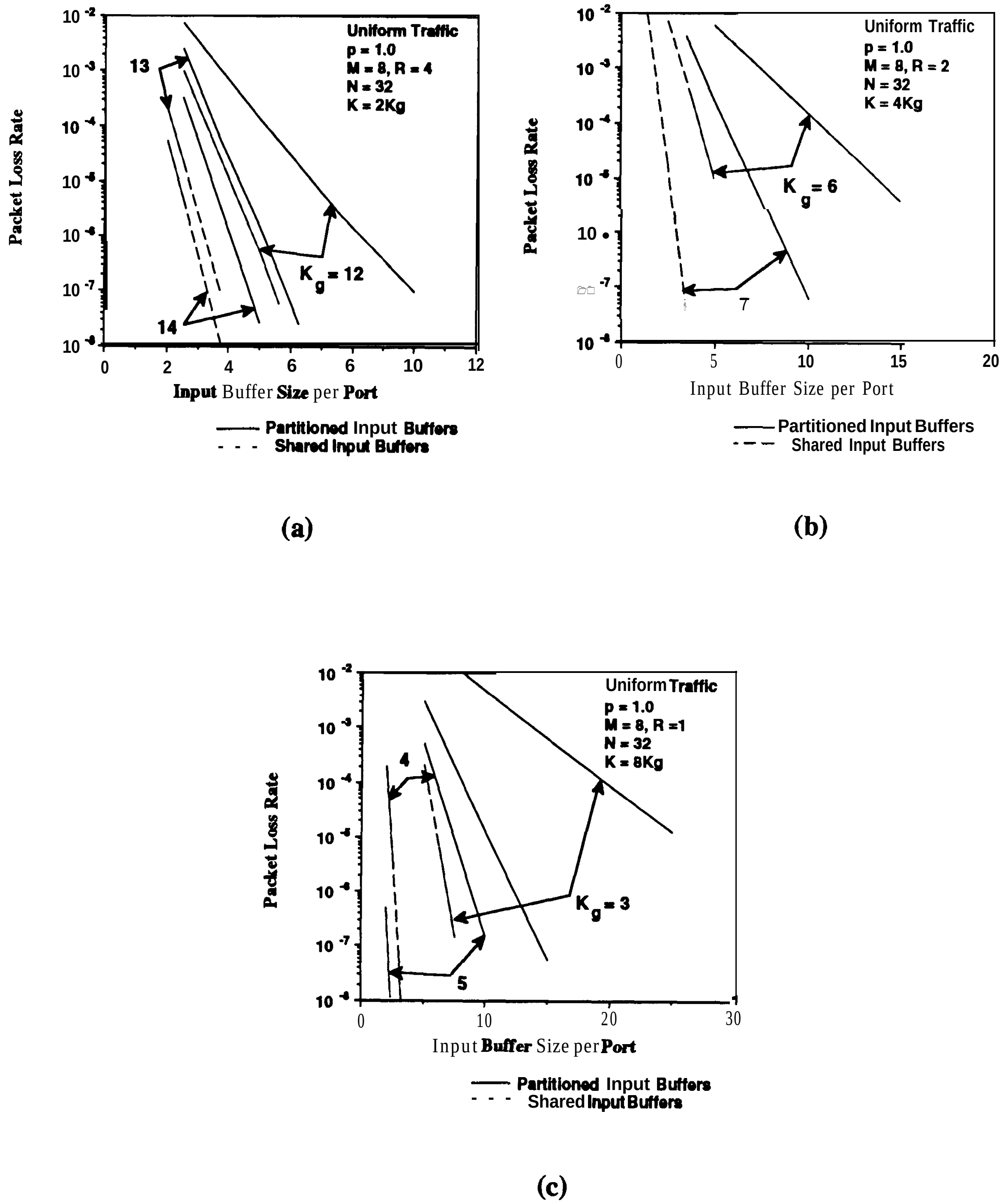


Fig. 3.13. Packet loss rate in the input buffers of a 256 x 256 MSM switch, under uniform traffic conditions, p = 1, ***M*** = 8, ***R*** = 8; the input and output components are connected to every other inputs and outputs, respectively, of 64 x 64 banyan networks (obtained by simulation).



(a)

(b)

(c)

Fig. 3.14. Packet loss rate in the input buffers of a 256 x 256 MSM switch with G > 1, for shared and completely partitioned input buffers, under uniform traffic conditions, p = 1, $M = 8$, $N = 32$; a) $G = 2$ ($R = 4$); b) $G = 4$ ($R = 2$); c) $G = 8$ ($R = 1$) (obtained by simulation).

packets maintained over consecutive banyans and/or several slots, and cause degradation in throughput. Here, we characterize this problem and propose a simple solution for it.

Traffic patterns exhibiting correlation in the *space* domain show dependencies in the distribution of output ports requested. We are concerned about specific traffic patterns for which the limitation on throughput is primarily due to contention on the internal links of the banyans, as opposed to output conflicts. A realistic scenario where this situation may occur is that of ***communities of interest***, where packets from specific subsets of inputs (referred to as ***input communities***) request to connect only with specific subsets of outputs (referred to as ***output communities***). In a banyan network, internal contention is at its worse when the paths connecting the inputs to the requested outputs use the smallest number of interconnection links within the network. Thus, if inputs and outputs belonging to the corresponding communities are connected by a minimum number of internal links, maximum internal contention results **[Tur88].** (An example of such a situation is illustrated in Fig. 3.15 for N = 16.) Even in absence of output conflicts, since packets can only use a small portion of the routing capacity, the maximum throughput attained in a banyan is significantly lower than that obtained under uniform traffic, and decreases as the network size increases. For example, for N = 32, the maximum throughput is reduced from 0.4 under uniform traffic to 0.25, since only one fourth of the internal links is used. For N = 64, the maximum throughput is reduced from 0.36 under uniform traffic to only 0.125, since only one eight of the internal links is used **[Tob91].** Consequently, the MSM switch, when offered this kind of “bad patterns”, shows throughput degradation with respect to the switch under uniform traffic, and more banyans have to be provided to achieve low packet loss rates. For example, considering a community-of-interests scenario where packets from each input are uniformly destined to any output in the corresponding output community, we show in Fig. 3.16 that, at full load, an MSM of size 32 x 32 (N = 32, ***M* =**

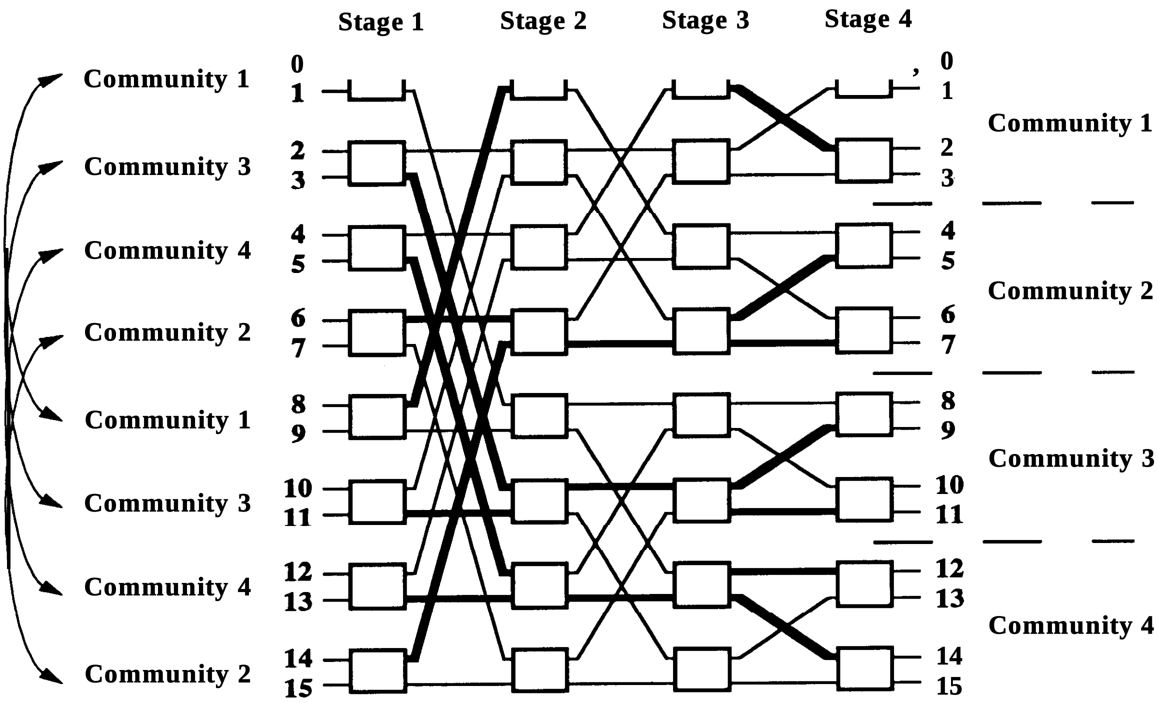


Fig. 3.15. Maximum internal contention in a 16 x 16 banyan network, and community-of-interest scenario leading to the congestion shown.

R = 1) with input buffers requires at least 6 banyans to achieve packet loss rates below 10” (rather than 3 banyans required under uniform traffic).

The MSM switch shows throughput degradation also under traffic patterns exhibiting correlation in the time domain, in particular under the bursty traffic pattern. In fact, under bursty traffic conditions, since in the MSM packets enter the banyans from the same input in every slot, and since all packets in a burst contend over the same paths, recurrent conflicts may occur in different banyans and involve not only the same packets, but also successive packets belonging to the same bursts. These conflicts may be maintained over several banyans and several slots, thus leading to throughput degradation. Such degradation is quite evident in Fig. 3.17, where we plot the packet loss rate in the input buffers of a 32 x 32 MSM, obtained by simulation, under uniform traffic and under bursty traffic with average burst lengths *L* = 10 and *L* = 100, at load = 0.9. The degradation is

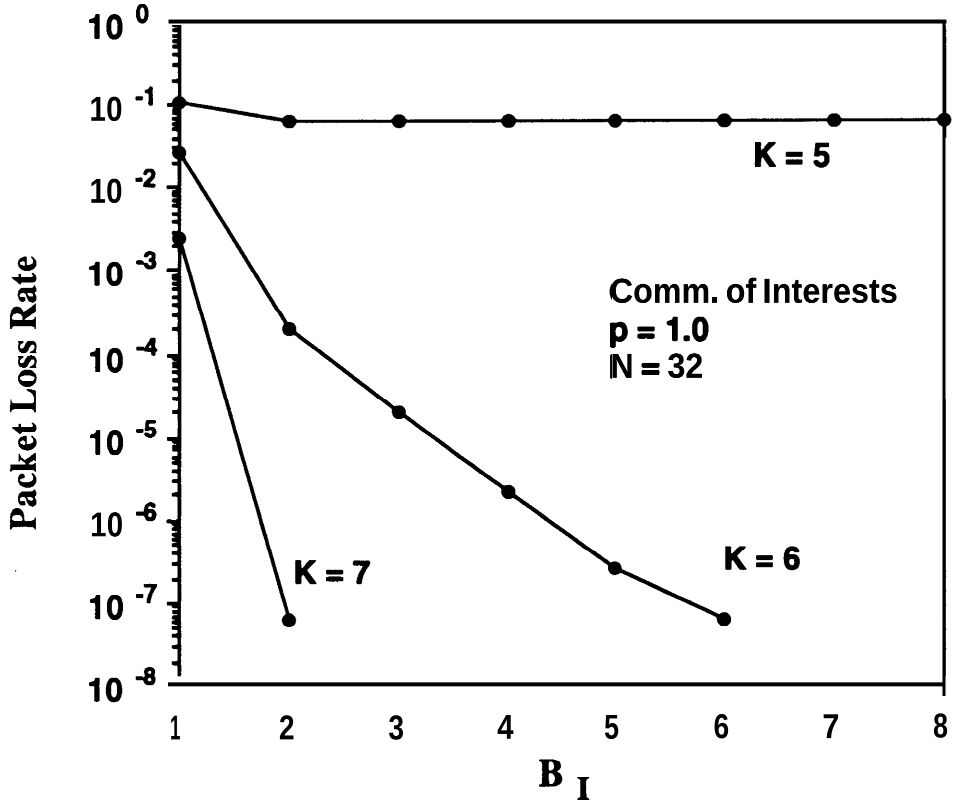


Fig. 3.16. Packet loss rate in the input buffers of an MSM switch, $M = 1$, N = 32 under a community of interest scenario at full load (obtained by simulation).

accentuated for increasing values of the burst length. Again, for long bursts, more than 3 banyans have to be provided to reach low packet loss rates with small input buffer sizes. Similar considerations apply to switches with $M > 1$.

Providing more banyan, although possible, is not desirable. In fact, as mentioned above, there is a limit on the number of banyans that can be used in parallel with FCFS operation of the input buffers (*i.e.*, maintaining the sequence of the packets); thus, a larger number of banyans translates into a stricter limitation on M, and consequently on the switch size (of course, providing more banyans also increases the required amount of resources and complicates the control of the banyans).

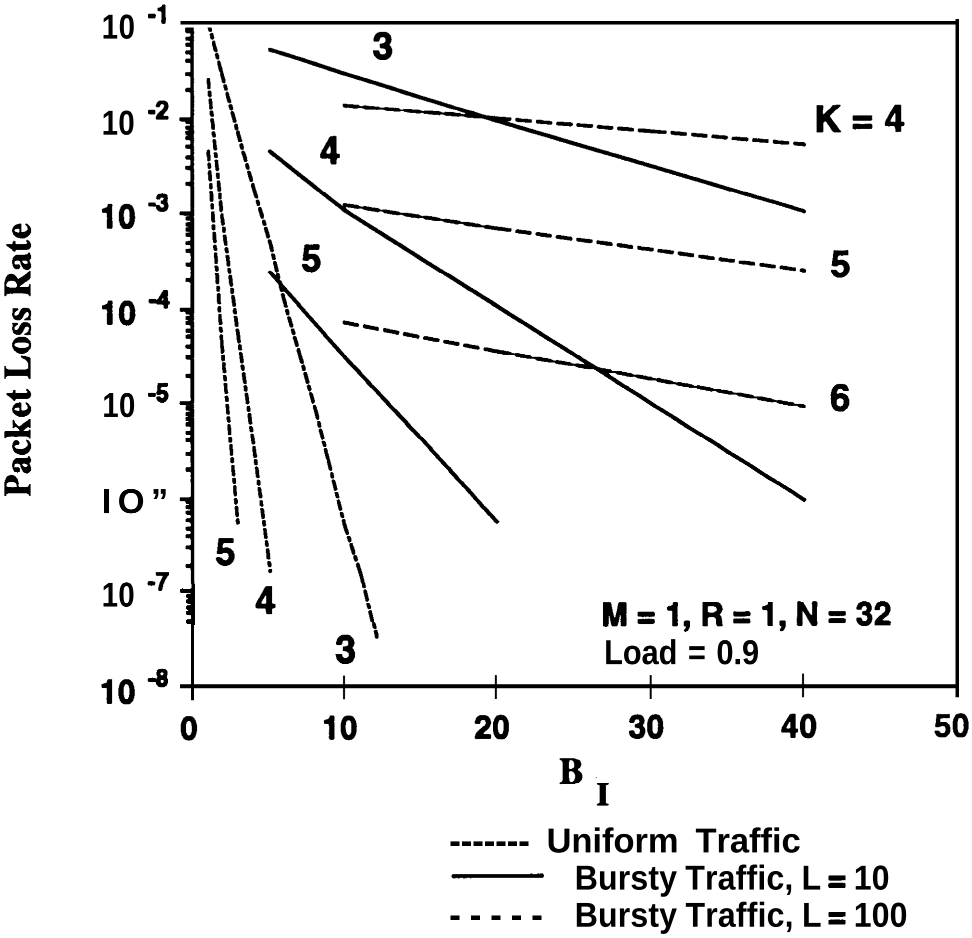


Fig. 3.17. Packet loss rate in the input buffers of an MSM without randomizers by pairs in front of the banyan networks, under uniform traffic and under bursty traffic with average burst length $L =$ 10 and L = 100, load = 0.9, M = 1, N = 32 (obtained by simulation).

An effective solution to overcome the throughput degradation under correlated traffic patterns is to randomize the input patterns of different banyans, so as to reduce the likelihood of persistent “bad patterns” and recurrent conflicts. One possibility is to provide a ***full randomization network*** in front of each banyan, which provides randomization over ***all*** inputs of the banyan **[Tur88].** The resulting concatenation of the randomization network and the banyan network is basically a Benes network in which the first half performs one of the *$N!$* random permutations of an input pattern, and the second half is the banyan routing network. The complexity of such a network is twice the complexity of the banyan network. In particular, the number of stages is doubled; consequently, in the MSM, the duration of the route set-up phase T_s in each routing network is also doubled. Since the limitation on the maximum number of banyan networks that can be placed in parallel is $K_{\max} = T / T_s$, doubling T_s halves $K_{\max}$, and again translates into a stricter limitation on M. In addition, we also note that to implement a Benes network of size equal to that of the largest Banyan network that can implemented on a single chip, two chips must be used, since the Benes requires twice the area of the banyan. Implementing the network in two chips rather than on a single chip not only complicates the synchronization of the network, but also makes more difficult to take full advantage of the higher clock rate that is achievable during the transmission phase, due to signal degradation on the chip-to-chip interconnections.

Our objective is therefore to devise a randomization network that achieves the effect of a full randomization network with lower complexity. In the limit, if such a network consists of a ***single*** stage, the impact on T_s (and consequently on $K_{\max}$) resulting from its addition is minimal. Furthermore, we note that in practice, given the typical aspect ratios of VLSI designs of banyan networks, such additional stage can be usually fitted rather easily on the same chip of the banyan network (thus, the additional hardware complexity resulting from the use of a single-stage randomization network is negligible).

In this section, we show that, indeed, most of the benefits of full randomization over all inputs are obtained at a much lower cost by randomizing packets only between properly selected ***pairs*** of inputs. Such a randomization network can be realized using a single stage of switching elements. Our discussion is organized as follows. After introducing some basic terms and notations, we first characterize a full randomization network, in order to define a performance measure that can serve as an objective function in the design of a single-stage randomization network. We then present a procedure to construct a single-stage randomization network that achieves most of the benefits of the full randomization network. Finally, we describe the performance of the modified MSM resulting from the addition of such single-stage network under various traffic conditions.

a) Basic Terms and Notations

Here, we introduce some useful notations and definitions. Given an N × N network, we define as *packet vector* $V = [V_0, \dots, V_{N-1}]$ a set of N packets whereby packet V_i (V_i =0 ,..., N-l) isatlinei(i=O ,..., $N-1$)[3]. The vector B = [0,1,..., $N-1$] is called the ***base vector.*** A ***permutation on a vector V*** is a mapping of ***V*** which performs a permutation of the positions of its N packets[4]. An ***exchange operation on a vector V*** exchanges the position of two packets in the vector.

In a banyan network, we define as ***distance of collision*** $d_C(i,$ j) (or simply ***distance*** throughout this discussion) ***between two inputs i, j (*** $i = 0, \dots, N-1$; j = 0 ,..., N − 1*),* as the unique stage s (s = 1,. ..,*n* where *n* = log, N) in the network in which packets coming from the two inputs may collide with each other[5]. (In fact, in a banyan, packets from two inputs at distance *k* may only encounter each other at 2^{k-1} switching elements in stage *k,* as determined by the specific banyan topology adopted.) For example, referring to the banyan topology of Fig. 3.18.a [Wu80], input 0 has distance 1 from input 1, since collisions between packets from input 0 and 1 may only happen at a single switching element at stage 1; similarly, input 0 has distance 2 from input 8 and input 9, since when

[3] **In this discussion, we will be mostly interested in the *positions* of the packets in the vector. For our purpose, therefore, a packet vector is a collection of objects (packets), each uniquely identified by a *tug* V_i. While the identification tags are unique, the destination addresses of the individual packets are generic (hence, for example, their addresses are not necessarily unique).**

[4] ***Note that* a *permutation on a vector as defined* here is a permutation of *the positions* of the packets in the vector. This should not *be confused with* what *is* referred *in other* contexts *as permutation,* namely a set of packets with unique address destinations.**

[5] **We should note that if we defme $d_C(i, i) = 0$ (i = 0,.... N -1), *the distance of collision* is indeed a metric; it satisfies nonnegativity ($d_C(i, j) \geq 0$, *awl* $d_C(i, j) = 0$ *if* and only if $i = j$), *symmetry* ($d_C(i, j) = d_C(j, i)$), and triangle-inequality conditions ($d_C(i, j) \leq d_C(i, k) + d_C(k, j)$).**

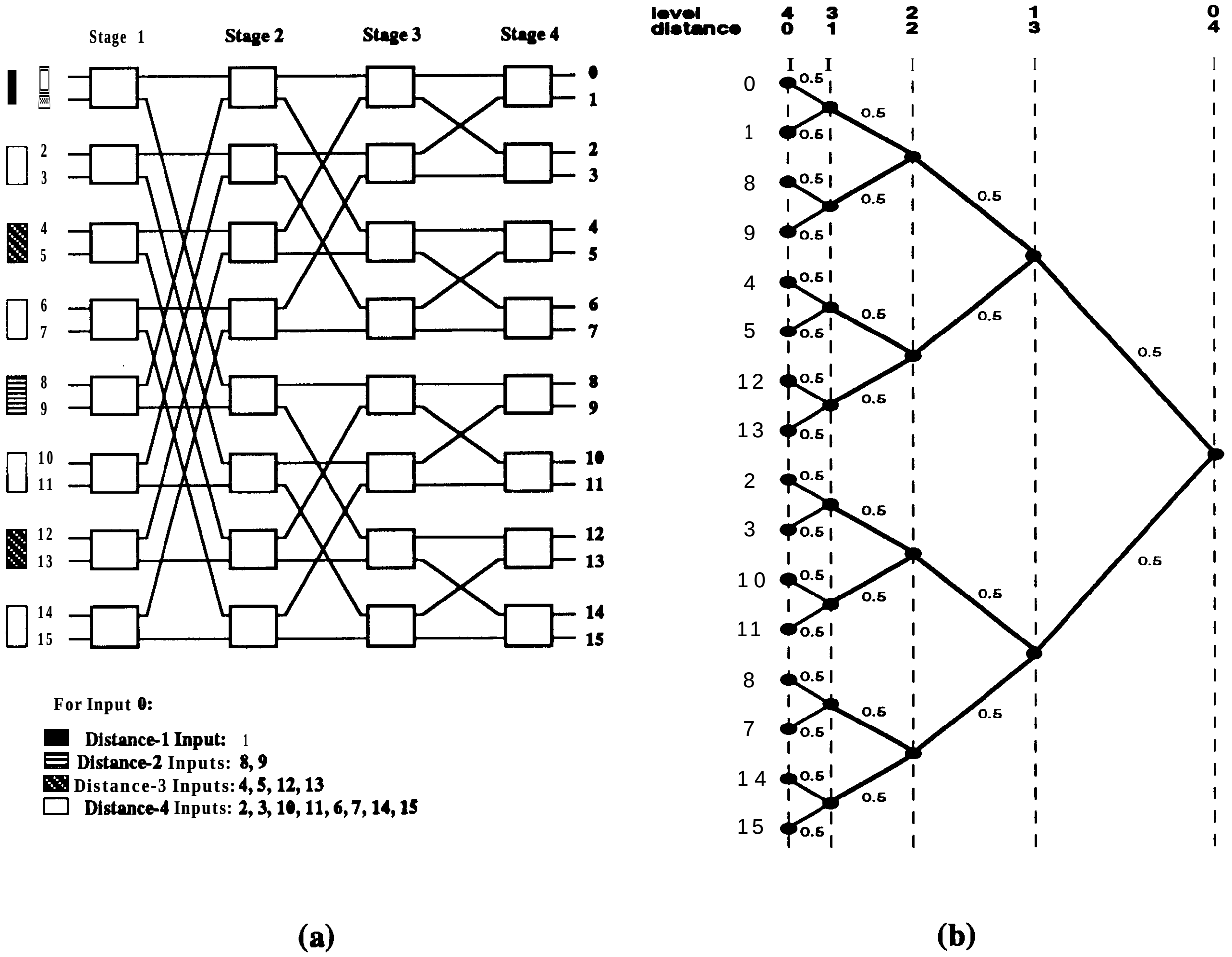


Fig. 3.18. *Distance of collision* between two inputs in a banyan network; a) distance of collision between input 0 and all the other inputs in a 16 x 16 banyan network; b) Binary tree showing the distance between any inputs in the network.

packets from input 0 have conflicts with packets from input 8 or input 9, they collide at two switching elements at stage 2; and so forth.

The distance between any pair of inputs is conveniently computed by using an *n*-level binary tree in which the N inputs correspond to the leaves of the tree, every vertex at level n- 1 joins pairs of inputs at distance 1, and every vertex at level n-k (k = 2,. ..,n) joins pairs of **subtrees** whose leaves are at distance k. The resulting tree for N = 16, for the banyan topology under consideration, is depicted in Fig. 3.18.b. In such a graph, by construction, the distance between any inputs i, j is the distance associated with the root of the smallest **subtree** containing them (alternatively, $d_C(i,$ j) is given by the length of the shortest path from i to j, when the weight of each edge is equal to 0.5).

In a banyan network, for a given input vector V, we then loosely refer as *distance* $d_{c,V}(V_i,$ *Vi> between two packets* as the distance between the two inputs from which the packets enter the banyan network. In a vector V, there are N(N – 1) / 2 possible pairs of packets (because of symmetry, pairs *(i, j)* and (j, *i)* are indistinguishable). For each packet, there are 2^{k-1} packets at distance k ($k = 1,...,n$); accordingly, the total number of pairs of packets at distance k is $2^{k-2}N$. For example, for N = 16 and N = 32, the number of pairs at a given distance are:

N= 16

distance	number of pairs
1	8
2	16
3	32
4	64
	120

$N = 32$

distance	number of pairs
1	16
2	32
3	64
4	128
5	256
	496

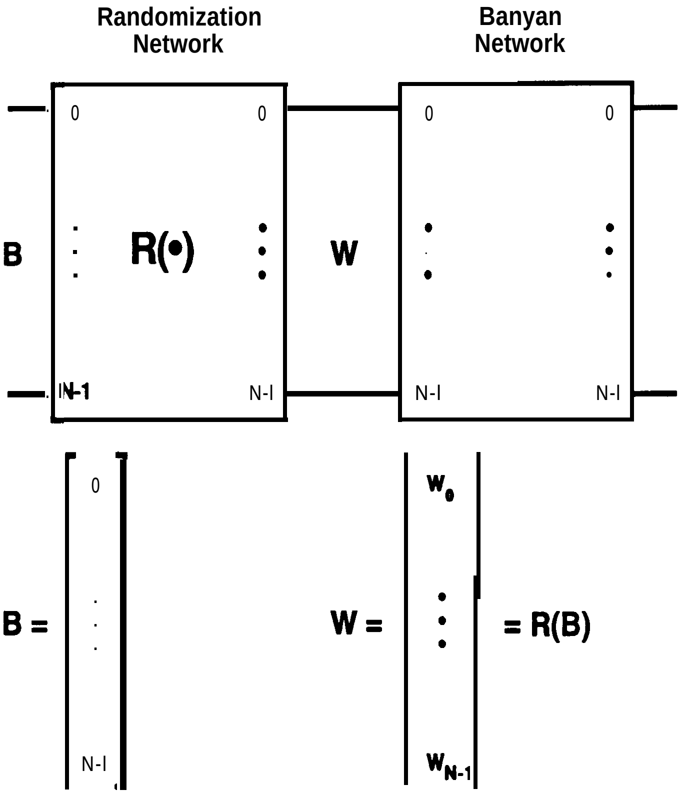


Fig. 3.19. Randomization network in front of a banyan network and related notation.

The *average distance* $D_{c,N}$ over all possible pairs of packets is therefore:

$$D_{c,N} = \frac{2}{N(N-1)} \sum_{all\ pairs(i,j)} d_{c,V}(i,j) = \frac{1}{N-1} \sum_{k=1}^{n} k2^{k-1} = \frac{(n-1)N+1}{N-1} \qquad (2)$$

independent of V. For example, for N = 16, $D_{c,N} = 3.267$; for N = 32, $D_{c,N} = 4.161$.

In this discussion, we are interested in the system depicted in Fig. 3.19, which consist of a ***randomization network*** placed in front of a banyan network The randomization network performs a random permutation on the base vector ***B*** denoted by $W = R(B)$, with corresponding probability p_W. Several randomization networks and their corresponding sets S of all possible outcomes can be defined. In particular, as mentioned above, ***full randomization over all inputs*** $R_f(B)$ performs each of the N! possible permutations of the N packets with equal probability ($S_f = \{W \mid W \text{ is a permutation on } B\}$).

b) Full Randomization and Single-Stage Randomization

With no randomization, assuming that all input patterns are ***equally likely***, the likelihood of conflicts between packets in the banyan decreases as their distance increases. Packets from inputs at distance 1 collide if they are destined to the same half of the banyan network outputs; packets from inputs at distance 2 collide if they are destined to the same fourth of banyan outputs; packets from inputs at maximum distance ***n*** collide only if they are destined to the same output. Accordingly, the ***probability of conflict*** $p_c(i, \mathrm{j})$ between two packets ***i, j*** as a function of their distance $d_{c,B}(i, j)$ **is** equal to:

number of pairs	$p_c(i,j)$	$d_{c,B}(i,j)$
N / 2	1/2	1
N	1/4	2
2N	1/8	3
.		
2^{k-2}N	$1/2^k$	k

A ***full randomization network*** maps each input pattern into one its *N!* permutations. At the outputs of the full randomization network, the probability $p_{R_f,k}(i, j)$ that two packets ***i*** and j are at distance ***k*** **is** equal to:

$$p_{R_f,k}(i,j) = \frac{2^{k-1}}{N-1} \tag{3}$$

for any pair of packets ***(i, j).*** For example, for N = 16:

	distance				
	1	2	3	4	# of pairs
$p_{R_f,k}(i,j)$	$\frac{1}{15}$	$\frac{2}{15}$	$\frac{4}{15}$	$\frac{8}{15}$	120

and for N = 32:

	distance					
	1	2	3	4	5	# of pairs
$p_{R_f,k}(i,j)$	$\frac{1}{31}$	$\frac{2}{31}$	$\frac{4}{31}$	$\frac{8}{31}$	$\frac{16}{31}$	496

Therefore, the ***probability of conflict*** $p_{c,R_f}(i,j)$ between two packets ***i, j*** at the ***output*** of the randomization network is:

$$p_{c,R_f}(i,j) = \sum_{k=1}^{n} \frac{1}{2^k} p_{R_f,k}(i,j) = \sum_{k=1}^{n} \frac{1}{2^k} \frac{2^{k-1}}{N-1} = \frac{n}{2(N-1)} \qquad (4)$$

for any pair of packets *(i,* j). For example, for N = 16, $p_{c,R_f}(i,j) = 4/30$ (note: $1/8 < 4/30 < 1/4$); for $N = 32$, $p_{c,R_f}(i,j) = 5/62$ (note: $1/16 < 5/62 < 1/8$). Thus, the effect of the full randomization network is to equalize the probability of conflict for any pair of packets in ***B***. The full randomization network minimizes the maximum probability of conflict for any pair of packets *(i,* j). With respect to the case with no randomization, those pairs that were more likely to conflict have their probability of conflict reduced; clearly, this is obtained by increasing the probability of conflict for some other pairs.

A randomization network ***R*** can also be characterized in terms of ***distance***. With no randomization, the distance between packets at the inputs of the banyan network is ***deterministic***. With randomization, the distance between packets at the inputs of the banyan network is a ***random*** quantity. For a given pair of packets ***(i, j)*** in ***B***, the ***mean distance*** $\bar{d}_{c,R}(i,j)$ ***over all possible outcomes*** of the randomization for a pair of packets *(i,* j) ***in B*** is:

$$\overline{d}_{c,R}(i,j) = \sum_{w\in S} p_w d_{c,W}(i,j) = \sum_{k=1}^{n} k p_{R,k}(i,j) \tag{5}$$

where $p_{R,k}(i,j) = P\{W \mid d_{c,W}(i,j) = k\}$ is the probability that *i* and j are at distance *k* at the output of the randomization. The mean $\overline{D}_c$ ***over all pairs*** of packets of the mean distance $\overline{d}_{c,R}(i,j)$:

$$\overline{D}_c = \frac{2}{N(N-1)} \sum_{all\ pairs(i,j)} \overline{d}_{c,R}(i,j) = \frac{2}{N(N-1)} \sum_{all\ pairs(i,j)} \sum_{W\in S} p_W d_{c,W}(i,j) =$$

$$= \frac{2}{N(N-1)} \sum_{W\in S} p_W \sum_{all\ pairs(i,j)} d_{c,W}(i,j) = \frac{1}{N-1} \sum_{k=1}^{n} k 2^{k-1} = D_{c,N} \tag{6}$$

independent of ***R***. Similarly, the variance $\sigma^2_{c,R}$ ***over all pairs*** of packets of the mean distance $\overline{d}_{c,R}(i,j)$ over all ***W*** is:

$$\sigma^2_{c,R} = \frac{2}{N(N-1)} \sum_{all\ pairs(i,j)} \left(\overline{d}_{c,R}(i,j) - \overline{D}_c\right) \tag{7}$$

With ***full randomization***, the mean distance over all possible outcomes is:

$$\overline{d}_{c,R_f}(i,j) = \sum_{k=1}^{n} k p_{R,k}(i,j) = \sum_{k=1}^{n} k \frac{2^{k-1}}{N-1} = \overline{D}_c \tag{8}$$

for any pair *(i, j)*. (For N = 16, $\overline{d}_{c,R_f}(i,j) = 3.267$; for N = 32, $\overline{d}_{c,R_f}(i,j) = 4.161$.) Hence, the variance $\sigma^2_{c,R_f} = 0$. **We** note that:

$$\overline{d}_{c,R_f}(i,j) = 2n \, . \, p_{c,R_f}(i,j) - 1 \tag{9}$$

thus, for a pair (i, *j*), there is a direct relation between the mean distance $\overline{d}_{c,R}(i,j)$ and the probability of conflict $p_{c,R}(i,j)$. In other words, minimizing the maximum probability of conflict p_{c,R_f} (i, j) for all pairs (i, j) is equivalent to maximizing the minimum mean distance $\overline{d}_{c,R}(i, j)$for all pairs (i, j). (Note that, since $\overline{d}_{c,R_f}(i, j) = \overline{D}_c$ for all pairs (i, j), and since the sum over all pairs of the distance is a constant, full randomization indeed gives the optimum minimum mean distance for all pairs.)

Although achieving the optimum requires a network performing full randomization over all inputs, it is reasonable to expect that, by properly restricting the set S of possible outcomes, most of the advantage of full randomization could be obtained by less complex networks performing only partial randomization. As mentioned above, our objective is to construct a randomization network that can be implemented using a ***single*** stage of switching elements. In such a network, each packet can only be randomized over two inputs of the banyan, and the only degree of freedom in the design is the selection of such inputs for every packet (***i.e.***, the design of the topology interconnecting the randomization stage with the banyan network). Here, we show that by properly constructing such a topology, the single-stage randomization network can achieve performance very close to that attained with full randomization.

Given our objective of designing an interconnection topology, the mean distance, being a “topological” measure, constitutes a more convenient measure than the probability of conflict to be used for defining our design objectives. In order to construct a single-stage network that performs as close as possible to a full randomization network, we therefore choose the following design objectives:

i) design a single-stage randomization network that maximizes the minimum mean distance $\overline{d}_{c,R}(i, j)$ between any pair of packets (i, j), and

ii) among single-stage randomization networks yielding the same minimum mean distance, choose the one that minimizes the variance $\sigma^2_{c,R}$ over all pairs of packets of the mean distance (thus making the mean distance of any pair as close as possible to the mean distance over all pairs of packets $\overline{D}_c = D_{c,N}$).

We now discuss the construction of the single-stage randomization network. *Without loss of generality*, we impose that a packet at position i ($i = O,\ldots, N-1$) in the input vector B of the randomization network is randomized between *input i itself* and some other input[6] (this means that $W = B \in S$). Since the network consists of a single stage of $N/2$ switching elements, this means that it suffices to consider randomization networks that perform randomization only between properly selected (disjoint) pairs of inputs (*i.e.*, which randomly execute up to $N/2$ disjoint predefined exchange operations on the input vector B); we refer to such single-stage randomization networks as *randomizers by pairs* R_p. Then, the design of the randomization network simply consists of selecting which pairs of inputs should be coupled together to satisfy the design objectives. (The number of distinct ways to divide N inputs into $N/2$ disjoint couples[7] is, of course, $(N-1)\cdot(N-3)\cdot(N-5)\cdot\ldots\cdot 1$.) To identify the couples, we observe the following:

a) The distance between packets arriving at two inputs that are coupled together is deterministic, due to symmetry (in fact, the packets arriving at the two inputs can only exchange their position as a result of the randomization, leaving their distance unaffected). This suggests that, in order to maximize the minimum mean distance for all pairs, we should couple together *inputs at maximum distance n* (*i.e.*, $p_{R_p,n}(i, j) = 1$ for all i, j that are coupled); in fact, it is straightforward to realize that in this case, $p_{R_p,n}(i, j) = 1/2$ for all i, j that are *not* coupled (*i.e.*, for *any* pair of packets (i, j) arriving at inputs that are not coupled,

[6] In fact, it can be shown that any single-stage randomization network can be reduced to one that satisfies this condition by simply *relabeling the* inputs.

[7] In the following discussion, we denote as a *couple* a *pair* of inputs that are matched together (*i.e.*, connected to the same switching element in the single-stage randomization network) so that an exchange operation can be randomly executed on packets arriving at them; this is to avoid any confusion with a generic *pair* of packets *i, j* for which a mean distance can be calculated.

the probability that at the output of the randomizer the two packets are at maximum distance, and therefore may collide only in case of output conflict, is 1/2), which is the best possible case obtainable with only exchange operations.

b) Having imposed this restriction on how couples are selected, the $p_{R_p,k}(i,j)$ $(k=1,\ldots,$n)forallpairs(i,j)(i=O,...,N-1; j=O,...,N-1)canonlyassumea small number of possible values, as listed in Fig. 3.20 for N = 16 and N = 32 (in fact, pairs that are not coupled are connected to two distinct switching elements that, together, can assume only four possible states with equal probability: ***bar/bar***, ***bar/cross***, ***cross/bar***, and ***cross/cross***). Among these possible values, the actual selection of the desired probabilities for each pair cannot be arbitrary, but must be compatible with the fact that they have to be generated only by randomly using N / 2 exchange operations, and must satisfy

$$\sum_{all\ pairs(i,j)} p_{R_p,k}(i,j) = 2^{k-2} N \text{ for } k=1,\ldots,n \qquad (10)$$

since 2^{k-2} N is the number of inputs at distance k.

By inspection of Fig. 3.20, to maximize as much as possible the mean distance between any pair, the couples should be formed as follows. An input i_1 should be coupled with an input j_1 at maximum distance; input i_2 at distance 1 from i_1 should be coupled with an input j_2 at maximum distance from i_2, and as distant from j_1 as possible; input i_3 at distance 2 from i_1 should be coupled with input j_3 at maximum distance from i_3, and as distant as possible from j_1 and j_2; this should continue until all inputs have been coupled. It is evident that for any network of meaningful size, keeping track of the distance of all possible pairs becomes rapidly unmanageable. Fortunately, this problem of identifying the best selection of couples of inputs, and constructing a corresponding randomizer by pairs in a network of size N, can be conveniently solved by transforming it into a ***matching***

$N = 16$

	distance				
$p_{R_p,k}(i,j)$	**1**	**2**	**3**	**4**	**mean distance** $\bar{d}_{c,R_p}$
	1/2	0	0	1/2	2.5
	1/4	1/4	0	1/2	2.75
	1/4	0	1/4	1/2	3
	0	1/2	0	1/2	3
	0	1/4	1/4	1/2	3.25
	0	0	1/2	1/2	3.5
	0	0	0	1	**4**

(a)

$N = 32$

	distance					
$p_{R_p,k}(i,j)$	1	2	3	4	**5**	**mean distance** $\bar{d}_{c,R_p}$
	1/2	0	0	0	1/2	3
	1/4	1/4	0	0	1/2	3.25
	1/4	0	1/4	0	1/2	3.5
	1/4	0	0	1/4	1/2	3.75
	0	1/2	0	0	1/2	3.5
	0	1/4	1/4	0	1/2	3.75
	0	1/4	0	1/4	1/2	4
	0	0	1/2	0	1/2	4
	0	0	1/4	1/4	1/2	4.25
	0	0	0	1/2	1/2	4.5
	0	0	0	0	1	**5**

(b)

Fig. 3.20. Possible **values** of $p_{R_p,k}(i, j)$ for a randomizer by pairs in which only inputs at maximum distance are coupled together; a) N = 16; b) N = 32.

problem in the n-level binary tree that describes the distance between the inputs in the banyan network, which has been introduced above (see Fig. 3.18). In such a tree, a *matching* (*i.e.*, a couple) is formed by adding an edge connecting the two leaves of the tree corresponding to the inputs to be coupled together.

The best matching of all inputs can be found using the following construction. First, the leaves are labeled with an $(n-1)$-digit binary label. The d-th leftmost digit ($k = 1,\ldots,n-1$) in an input's label indicates the branch to which the input belongs in the subtree corresponding to distance $n - d + 1$ (*i.e.*, the subtree with root at level d- 1), as illustrated in Fig. 3.2 1 and 1.22 for N = 16 and N = 32, respectively (the digit is equal to 0 for the upper branch, and to 1 for the lower branch). Then, the problem is simply one of finding a matching of each input such that:

i) each leaf is matched with an input whose label has a different leftmost digit (as required to couple each input with an input at maximum distance from it), and

ii) in each subtree corresponding to distance s (s = I, ... n-2), the leaves are matched with inputs whose labels have the (s + 1)-leftmost digits forming unique sublabels.

Examples of complete matching satisfying these two conditions are shown in Fig. 3.21 and 3.22, for N = 16 and N = 32, respectively. Once the complete matching is found, the $p_{R_p,k}(i,j)$ $(k = 1,.$ **J $n)$ for any pair $(i,$ j) that is not coupled can be derived from the graph by identifying x, the matching input of i, and y, the matching input of j, and setting:

$$p_{R_p,n}(i,j) = 1/2; \qquad (11)$$

$p_{R_p,k_1}(i,j) = 1/4$ where $k_1 = d_c(i,$ j) , if the labels of i and j have the same most-significant digit, or $k_1 = d_c(i,y)$, if the labels of i and j have different most-significant digit; (12)

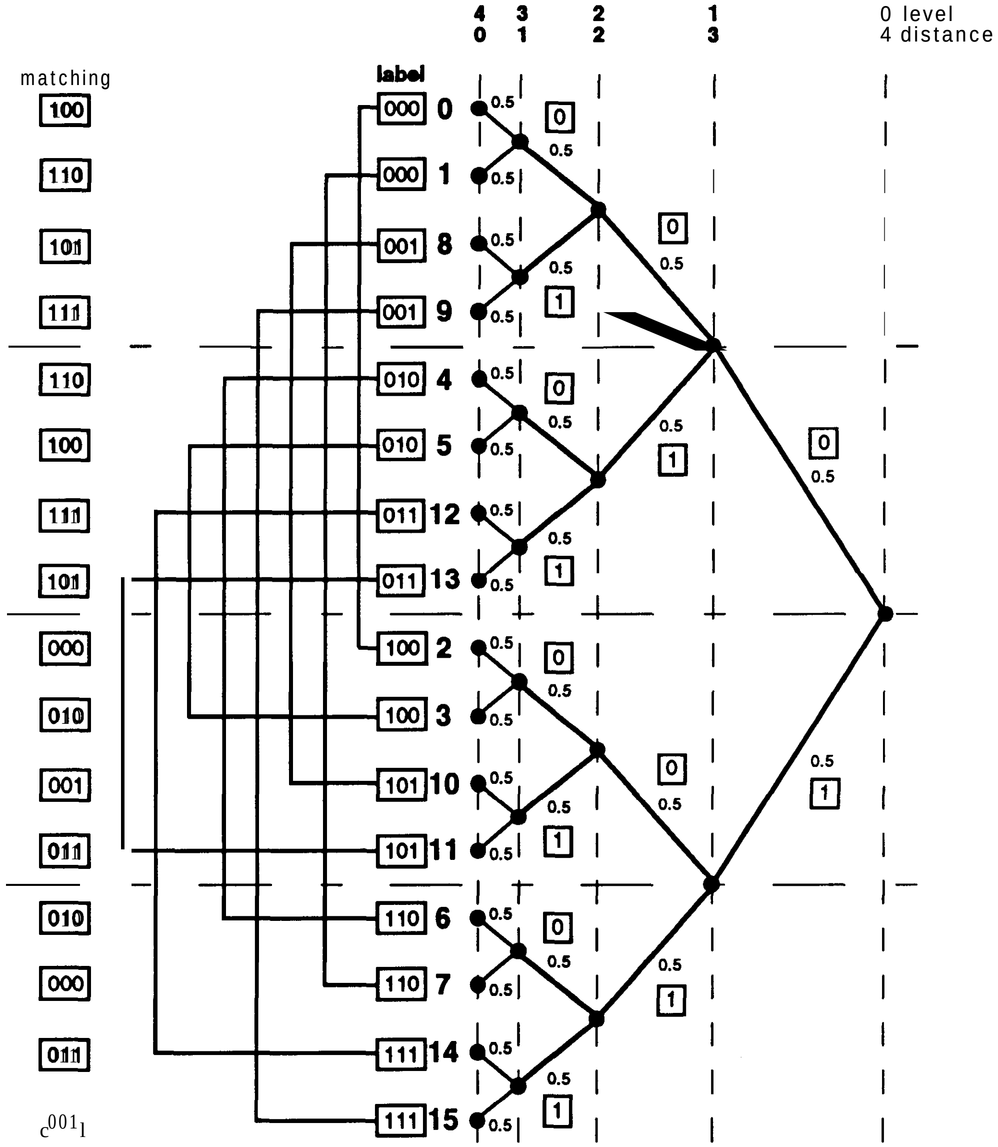


Fig. 3.21. Best matching of inputs in the binary tree representing the distance between inputs in a banyan network, N = 16.

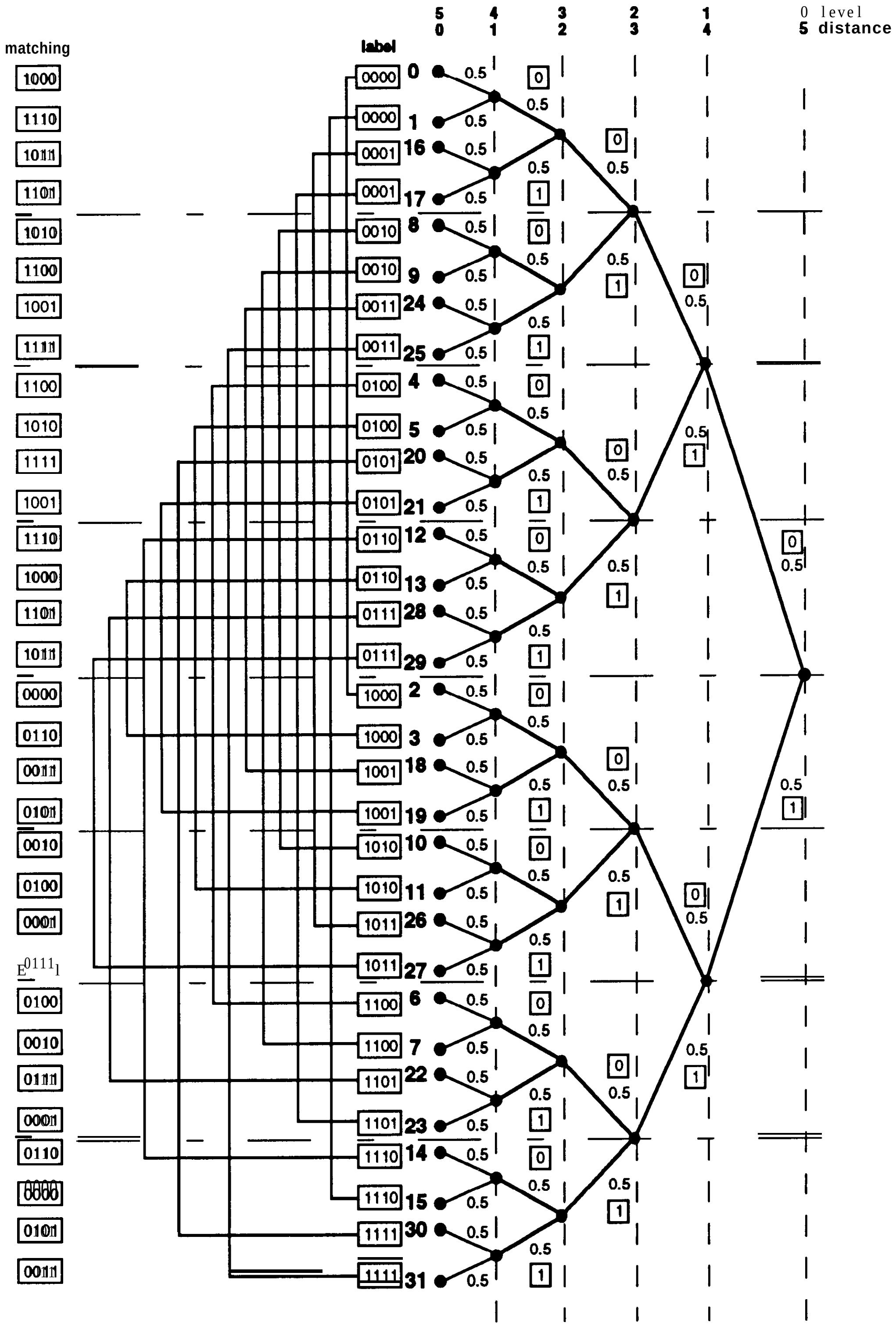


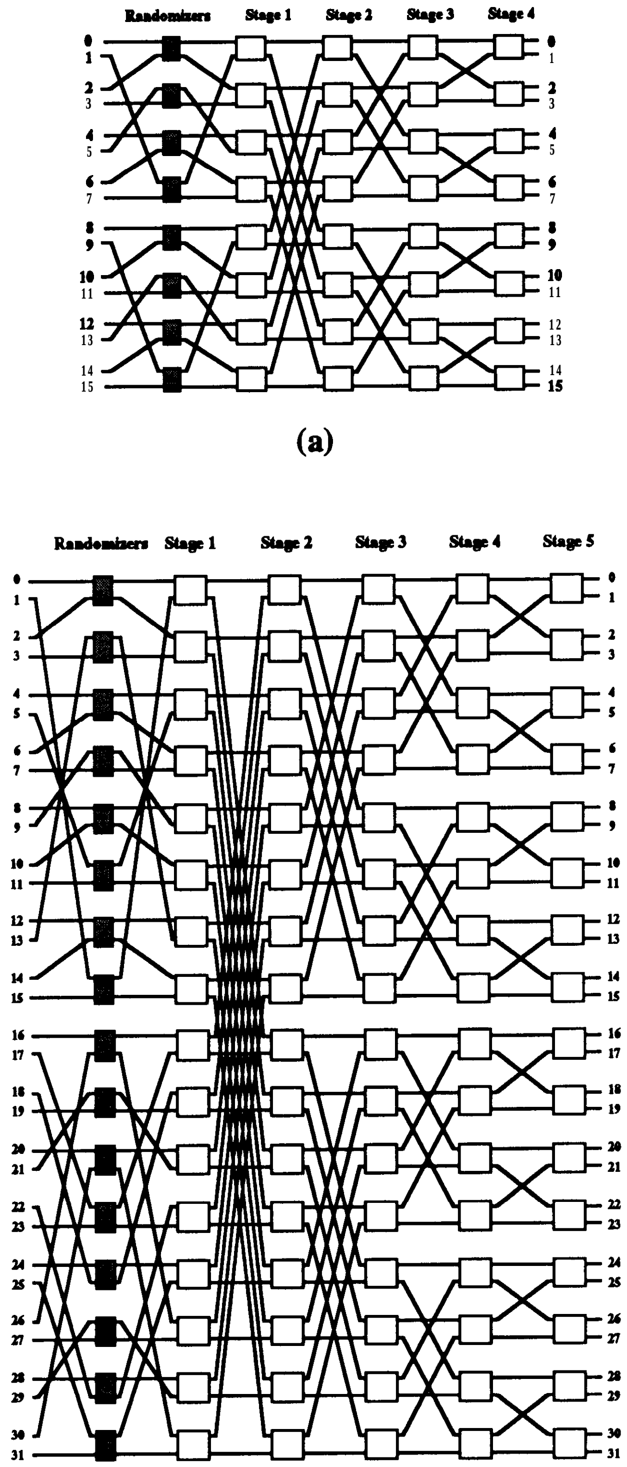
Fig. 3.22. Best matching of inputs in the binary tree representing the distance between inputs in a banyan network, N = 32.

$p_{R_p,k_2}(i,j) = 1/4$ where $k_2 = d_c(x, y)$, if the labels of i and j have the same most-significant digit, or $k_1 = d_c(x, j)$, if the labels of i and j have different most-significant digit; and (13)

$p_{R_p,k}(i,j) = 0$ for all other k. (14)

By construction, this procedure yields the desired randomizer by pairs. It should be noted that, since there exists more than one complete matching of the N inputs satisfying the two conditions above, different randomizer by pairs could be generated by this procedure; it can be shown, however, that all the networks obtained by connecting any of these randomizers with the banyan are topologically isomorphic. The randomizers by pairs corresponding to the matchings of Fig. 3.21 and 3.22 are shown in Fig. 3.23.a and 3.23.b, for N = 16 and N = 32, respectively. The mean distance over all possible outcomes for every pair of packets (i, j) is shown in Fig. 3.24. For N = 16, the minimum mean distance is equal to 3, as opposed to 3.267 obtained with full randomization, and the variance over all pairs of the mean distance is σ^2_{c,R_p} = 0.078. For N = 32, the minimum mean distance is equal to 3.75, instead of 4.161 obtained with full randomization, and the variance is σ^2_{c,R_p} = 0.103. Since the minimum mean distance and the variance are very close to the absolute optimum represented by full randomization, the randomizer by pairs can offer most of the advantage that is achievable by using a randomization network in front of the banyan networks, as it is shown in the sequel.

From a practical point of view, the randomizers by pairs can be simply implemented by adding, in front of each banyan network, an extra stage of switching elements with the activity-bit-resetting capability disabled, and by providing an additional randomly-generated bit in the address field of the headers to control them.



(a)

(b)

Fig. 3.23. Banyan networks augmented with randomizers by pairs corresponding to the matchings of Fig. 3.21 and Fig. 3.22; a) N = 16; b) N = 32.

$N = 16$, $\overline{D}_c = 3.267$

	distance					
σ^2_{c,R_p} = 0.078	1	2	3	4	# of pairs	mean distance $\overline{d}_{c,R_p}$
$p_{R_p,k}(i,j)$	1/4	0	1/4	1/2	32	3
	0	1/2	0	1/2	16	3
	0	1/4	1/4	1/2	32	3.25
	0	0	1/2	1/2	32	3
	0	0	0	1	8	4
					120	

(a)

$N = 32$, $\overline{D}_c = 4.161$

	distance						
σ^2_{c,R_p} = 0.103	1	2	3	4	5	# of pairs	mean distance $\overline{d}_{c,R_p}$
$p_{R_p,k}(i,j)$	1/4	0	0	1/4	1/2	64	3.75
	0	1/4	1/4	0	1/2	64	3.75
	0	1/4	0	1/4	1/2	64	4
	0	0	1/2	0	1/2	32	4
	0	0	1/4	1/4	1/2	128	4.25
	0	0	0	1/2	1/2	128	4.5
	0	0	0	0	1	16	5
						496	

(b)

Fig. 3.24. Mean distance over all possible outcomes for every pair of packets *(i, j)*, for the randomizer by pairs R_p of Fig. 3.23; a) N = 16; b) N = 32.

c) The MSM Switching Fabric with Randomizers by Pairs under Uniform Traffic

Under uniform traffic, the traffic is uniformly distributed over all output ports, so that no unfavorable traffic pattern is sustained over time. Consequently, the effect of persistent conflicts between packets occurring in different networks, due to the fact that packets enter each network from the same input, is marginal. Nevertheless, some repeated conflicts obviously take place, and the addition of the randomizers by pairs gives a small yet observable advantage in performance. The benefit becomes quantitatively more significant for large values of $\boldsymbol{M}$. In general, however, we should note that, since in any case the buffer requirements are fairly modest under uniform traffic, the reductions in buffer size that can be obtained with the addition of the randomizers are not very significant. Still, a characterization of the switch under uniform traffic is useful to show the effectiveness of the randomizers by pairs, and compare them with full randomization.

The packet loss rate in the routing fabric of the MSM with no input buffers under uniform traffic at full load, for $\boldsymbol{N} = 32$, obtained by simulation, is shown in Fig. 3.25. With banyan networks augmented by randomizers by pairs, 9 banyans suffice to achieve 10^{-7} packet loss rate, instead of 10 banyans needed in the MSM without randomizers. As shown in Fig. 3.25, with the addition of the randomizers, the MSM achieves basically the same performance as the TBSF, where the effect of recurrent conflict is not incurred, since banyans upstream in the series effectively act as randomization networks for following banyans in the series [Tob91].

The effect of the addition of the randomizers on the performance of a 32 × 32 MSM with input buffers and sequential dispatching, $\boldsymbol{M} = 1$, under uniform traffic at full load, obtained by simulation, is shown in Fig. 3.26. The benefit is more visible for low values of $\boldsymbol{K}$, and decreases as more banyans are used, since any negative influence of repeated conflicts is diluted in abundant routing capacity. With $\boldsymbol{K} = 3$, with the addition of

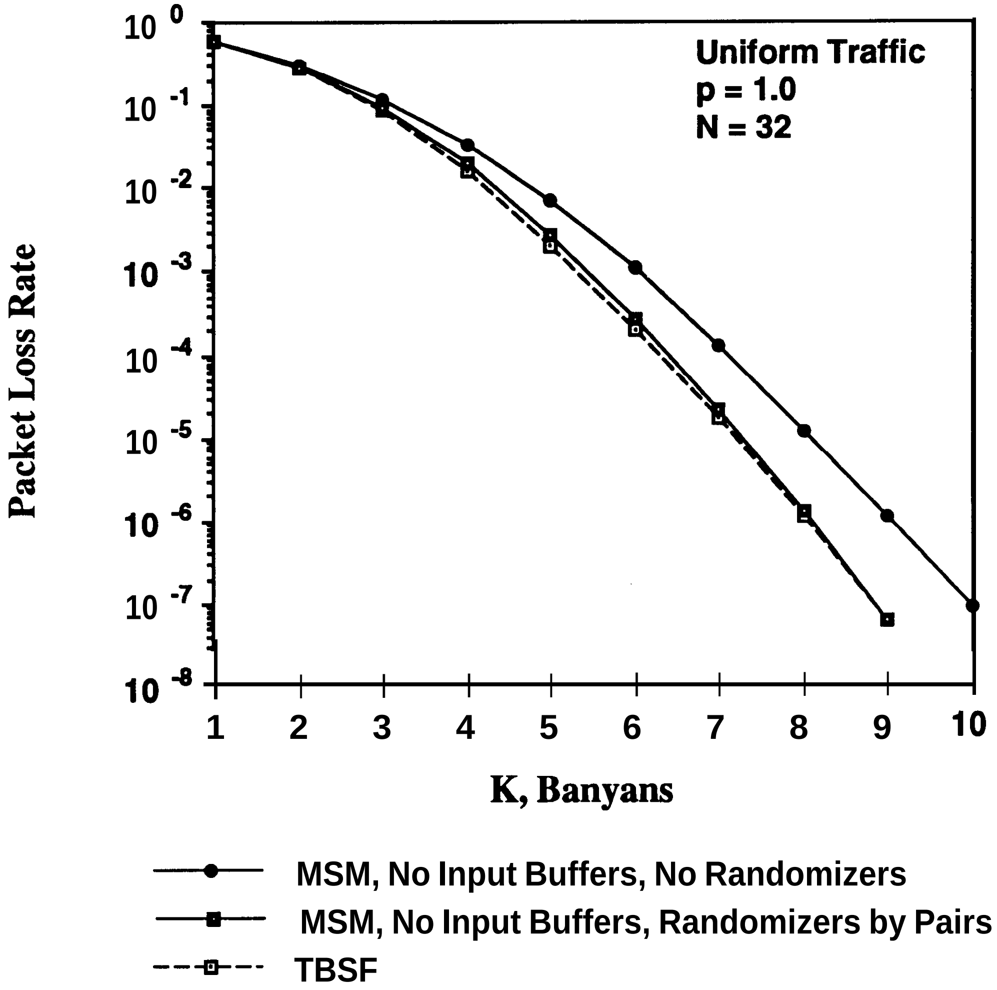


Fig. 3.25. Packet loss rate in the MSM routing fabric with no input buffers, with and without randomizers by pairs in front of the banyan networks, under uniform traffic at full load, N = 32. Also shown, the packet loss rate in the routing fabric of the TBSF, under the same traffic conditions (obtained by simulation).

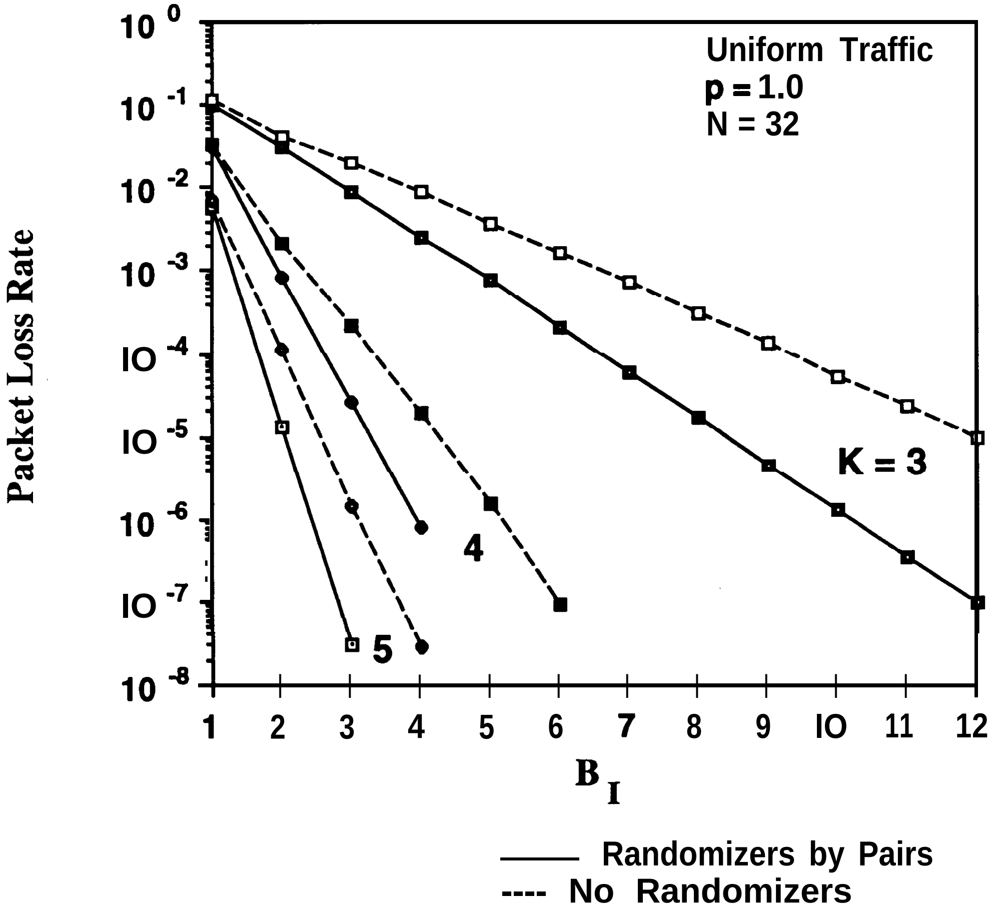


Fig. 3.26. Packet loss rate in the input buffers of the MSM switching fabric for sequential dispatching with and without randomizers by pairs in front of the banyan networks, under uniform traffic conditions, $\boldsymbol{p}$ = 1, $\boldsymbol{M}$ = 1, N = 32 (obtained by simulation).

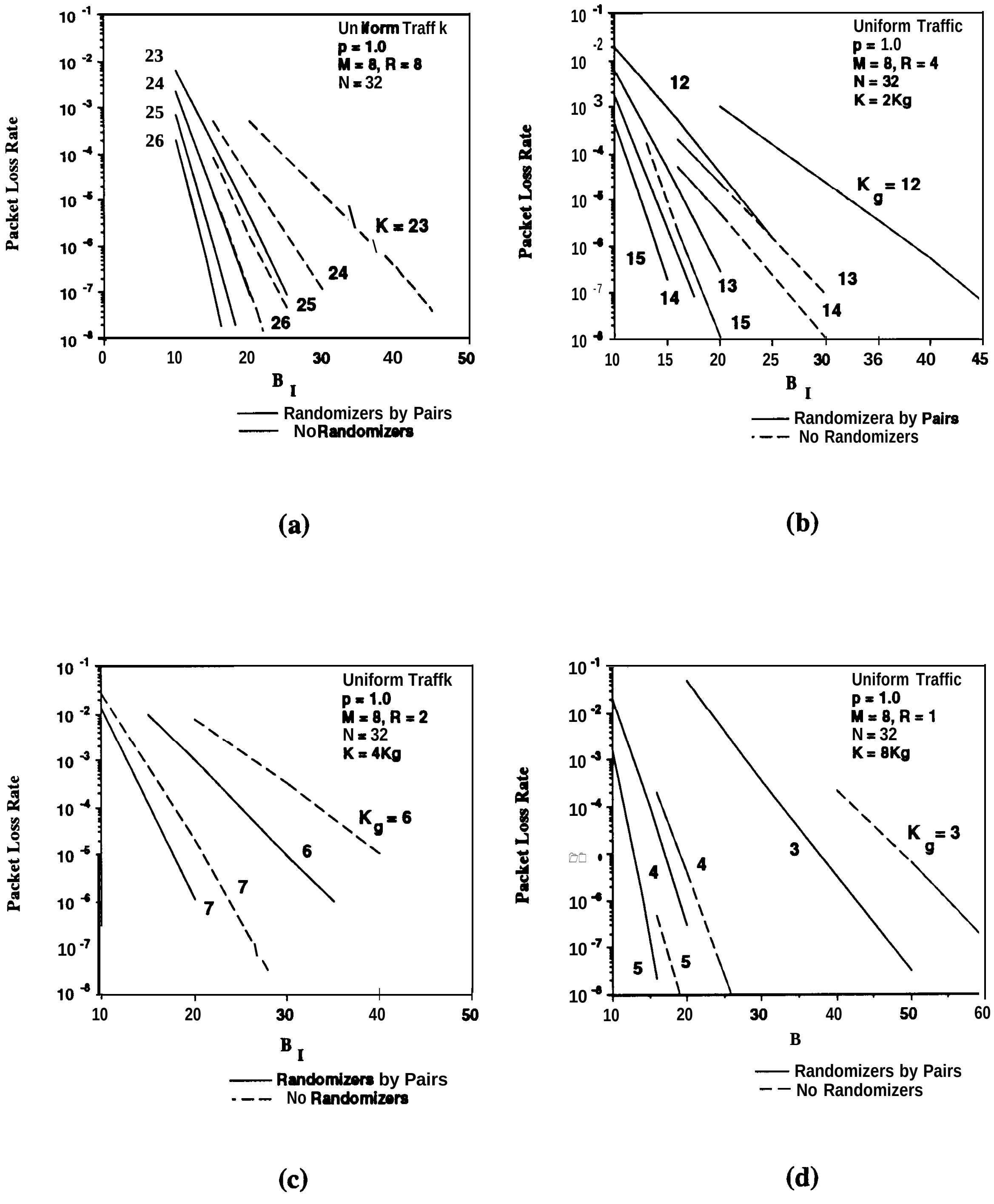
the randomizers, 10 buffers instead of 15 buffers are needed for a packet loss rate below 10^{-6}.

The effect of using randomizers by pairs in front of the banyans is more substantial for large $\boldsymbol{M}$. In Fig. 3.27, we show the packet loss rate in the input buffers of a 256 × 256 MSM, $\boldsymbol{M}$ = 8, with and without randomizers by pairs in front of the banyans, for various $\boldsymbol{R}$, at full load. Again, for a given $\boldsymbol{R}$, the benefit of the randomizers decreases as K increases. For example, for $\boldsymbol{R}$ = 8 and $\boldsymbol{K}$ = 23 banyans, with the addition of the randomizers, the number of buffers necessary for a 10^{-6} loss rate is reduced to 23, as opposed to 38 buffers required without randomizers; with $\boldsymbol{K}$ = 24 banyans, the reduction is 8 buffers. For $\boldsymbol{R = 4}$ and $\boldsymbol{K_g}$ = 12 banyans per group, with the randomizers, 26 buffers rather than 38 are necessary for the same loss rate. Larger reductions in the total number of buffers are observed if partitioned input buffers are used, as shown in Fig. 3.28; for example, again for $\boldsymbol{R}$ = 4 and $\boldsymbol{K_g}$ = 12 banyans per group, with the randomizers a total of 44 buffers are required instead of 68 buffers in the case without randomizers.

As expected, randomizers by pairs offer most of the benefit achievable by randomization. For example, in Fig. 3.29, we show the packet loss rate in the input buffers of a 256 × 256 MSM, $\boldsymbol{M}$ = 8, obtained by simulation, with randomizer by pairs and with full randomizers over all $\boldsymbol{N}$ inputs in front of the banyans, for various $\boldsymbol{R}$, at full load. In all cases, the buffer sizes necessary to meet a certain packet loss rates with randomizers by pairs are very similar to those needed with full randomizers.

d) The MSM Switching Fabric under Traffic Patterns with Space Correlation: Community-of-Interest Scenario

As mentioned above, randomization is truly needed in the MSM to make the switch less prone to congestion under traffic patterns exhibiting correlation in the space and/or in the time domain. With correlation among packets, unfavorable traffic patterns



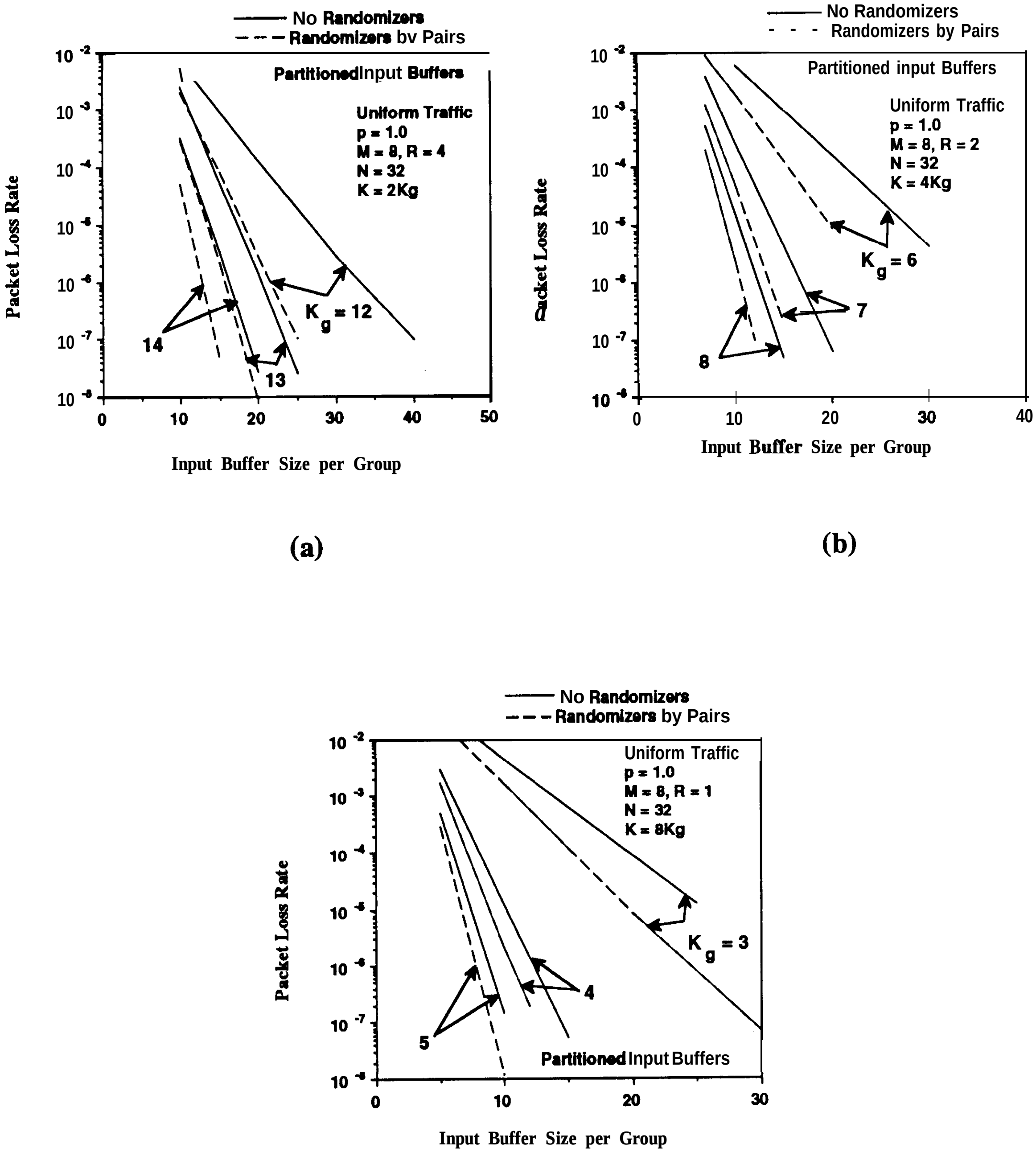
(a)

(b)

(c)

(d)

Fig. 3.27. Packet loss rate in the input buffers of a 256 X 256 MSM switch with and without randomizers by pairs in front of the banyan networks, under uniform traffic conditions, p = 1, N = 32; a) M = 8, R = 8; b) M = 8, R = 4; c) M = 8, R = 2; d) M = 8, R = 1 (obtained by simulation).



(a)

(b)

(c)

Fig. 3.28. Packet loss rate in the input buffers of a 256 X 256 MSM switch with completely partitioned input buffers, with and without randomizers by pairs in front of the banyan networks, under uniform traffic conditions, ***p*** = 1, ***M* = *8*, *G*** > 1, N = 32; a) G = 2 (***R*** = 4); b) G = 4 (***R*** = 2); c) G = 8 (***R*** = 1) (obtained by simulation).

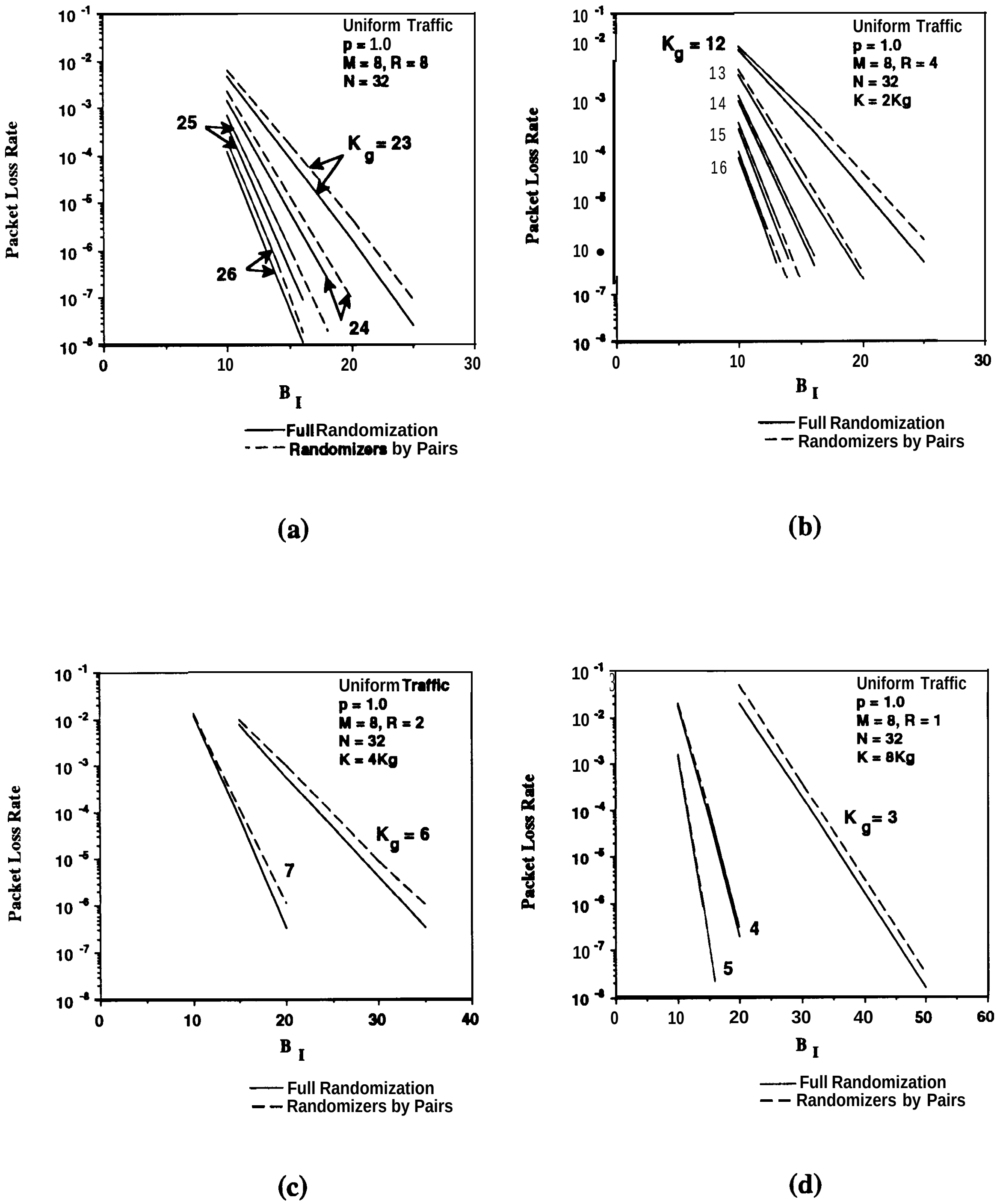


Fig. 3.29. Packet loss rate in the input buffers of a 256 × 256 MSM switch with randomizers by pairs and with full randomizers in front of the banyan networks, under uniform traffic conditions, p = 1, N = 32; a) M = 8, R = 8; b) M = 8, R = 4; c) M = 8, R = 2; d) M = 8, R = 1 (obtained by simulation).

may be offered to the switch and maintained over time, leading to substantial throughput degradation. Under these conditions, the addition of randomizers in front of the banyans effectively breaks the correlation among packets by distributing the traffic in the networks, and reduces internal congestion.

We have shown above that the MSM switch with no randomizers shows throughput degradation under traffic scenarios exhibiting heavy spatial correlation among packets, such as the community-of-interest traffic pattern. Under these traffic conditions, the addition of randomizers by pairs in front of the banyans dramatically improves the performance of the switch. For example, in Fig. 3.30, we plot the packet loss rate in the routing fabric of ***an*** MSM of size 32 × 32 ***with no input buffers,*** with and without randomizers by pairs in front of the banyan networks, under a community-of-interests scenario where packets from each input are uniformly destined to any output in the corresponding output community, at full load. It is interesting to note that, while considerable throughput degradation is observed with no randomizers, with the addition of randomizers the switch performs even slightly better than the TBSF (this latter switch is scarcely prone to degradation under spatially correlated traffic patterns since the first banyans in the series act as randomization networks for the following banyans [Tob91]).

The addition of randomizers by pairs brings substantial performance improvements ***also*** in the MSM ***with input buffers,*** under these traffic conditions. The packet loss rate in a switch of size 32 × ***32*** with randomizers by pairs is shown in Fig. 3.31. With respect to the switch without randomizers (for which the packet loss rate has been shown in Fig. 3.16 above), two less banyans are required to achieve low packet loss rates in the input buffers.

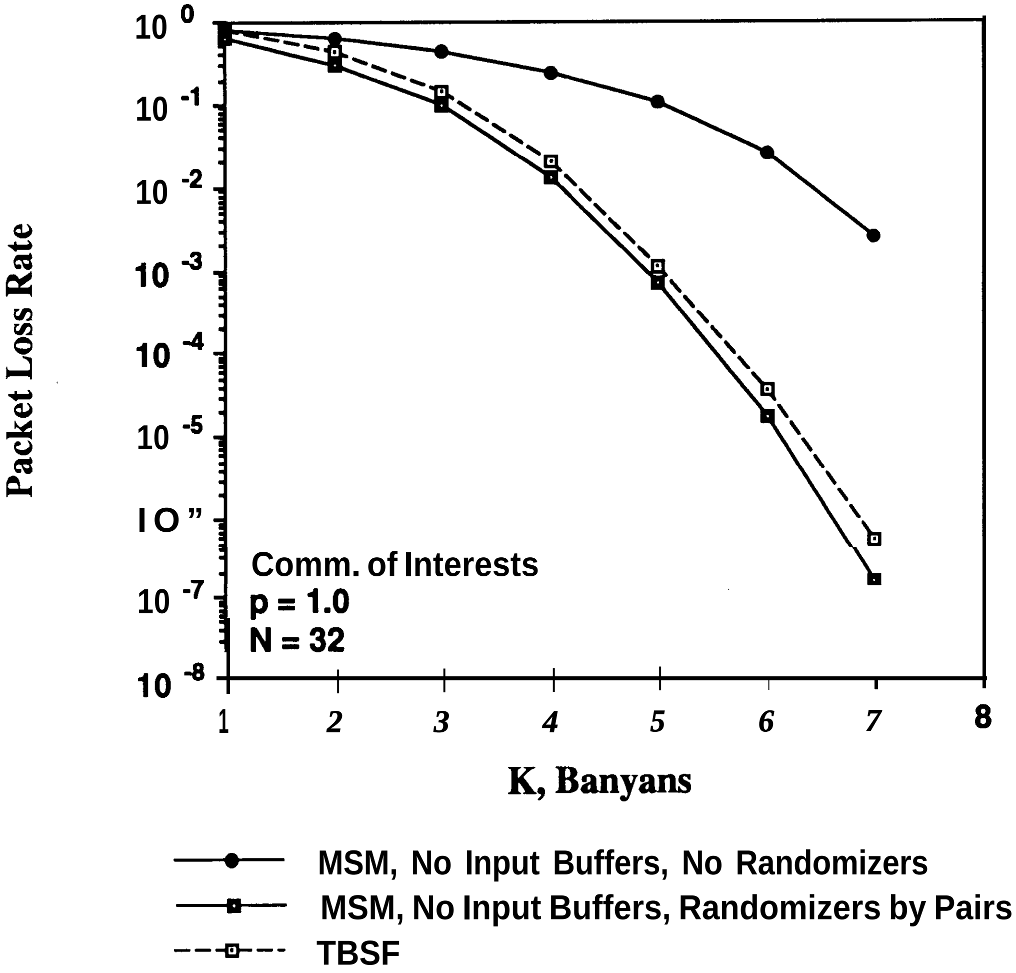


Fig. 3.30. Packet loss rate in the MSM routing fabric with no input buffers, with and without randomizers by pairs in front of the banyan networks, under a community of interest scenario at full load, ***M*** = 1, N = 32. Also shown, the packet loss rate in the routing fabric of the TBSF, under the same traffic conditions (obtained by simulation).

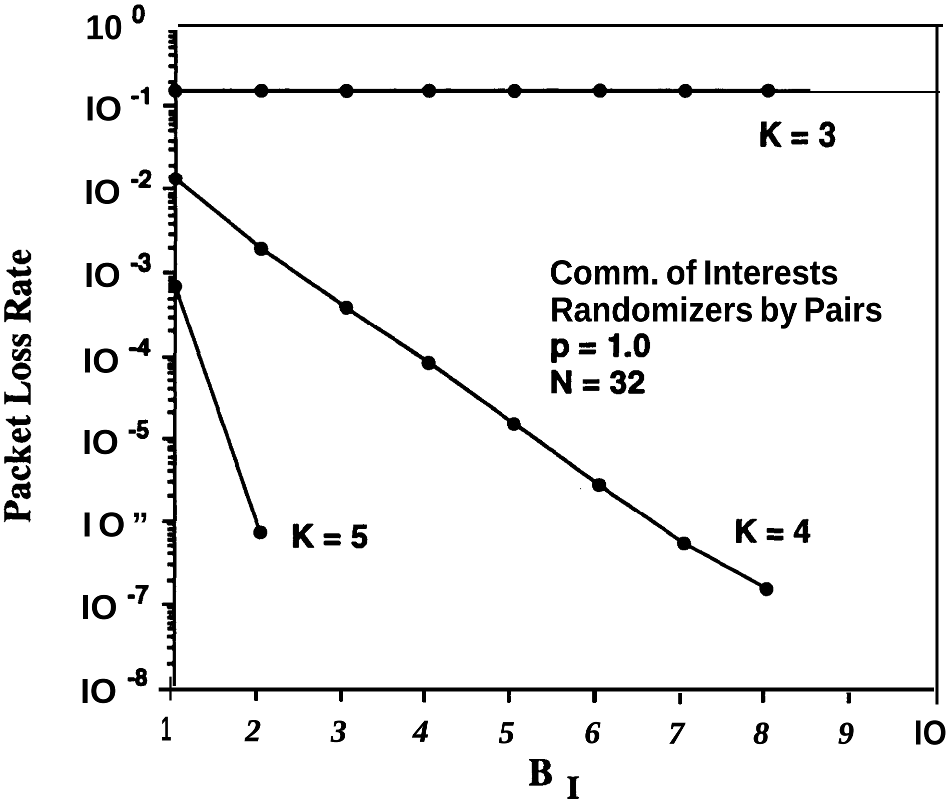
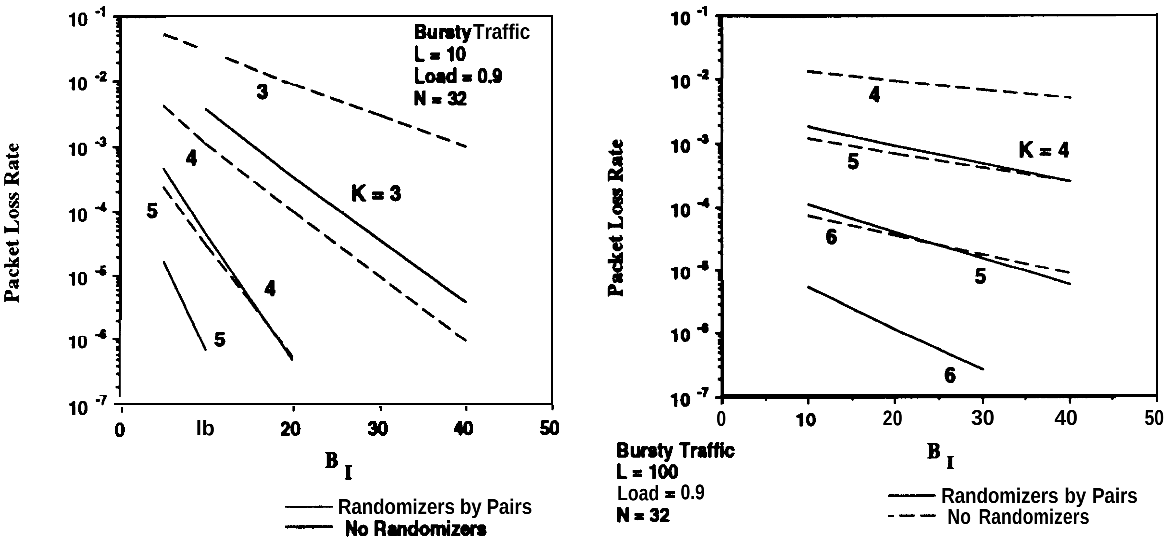


Fig. 3.31. Packet loss rate in the input buffers of an MSM switch with randomizers by pairs in front of the banyans, $M = 1$, N = 32 under a community of interest scenario at full load (obtained by simulation).



(a)

(b)

Fig. 3.32. Packet loss rate in the input buffers of an MSM with and without randomizers by pairs in front of the banyan networks, under bursty traffic, load = 0.9, M = 1, N = 32; a) L = 10; b) L = 100 (obtained by simulation).

e) The MSM Switching Fabric under Traffic Patterns with Time Correlation: Bursty Traffic

The second important traffic pattern under which the MSM without randomizers shows throughput degradation is the bursty traffic pattern. The effect is accentuated for increasing values of the burst length. With the addition of randomizer by pairs in front of the banyans, the likelihood of persistent conflicts is reduced, and the throughput improves. Here, we characterize the impact of the addition of randomizers in MSM switches of different sizes. In this study, we consider the same bursty traffic model described in Section 2.3 above.

The packet loss rate in a 32 × 32 MSM with and without randomizers, under bursty traffic with average burst length L = 10, at load = 0.9, obtained by simulation, is shown in Fig. 3.32.a. The randomizers are necessary to keep the required buffer size to achieve low loss rates small. With K = 3 banyans, 46 buffers are needed with randomizers to achieve a 10^{-6} packet loss rate, instead of more than 100 in absence of randomizers. With K = 4 banyans, with the addition of randomizers, the buffer size for the same loss rate is reduced from 40 buffers to 18. It should be noted that even by using the randomizers by pairs, the required buffer size to meet a given packet loss rate is larger than that needed under uniform traffic. As shown below, this is the case also when full randomizers are used in front of the networks, and the effect is more accentuated for small values of K. The degradation, therefore, is due primarily to head-of-the-line blocking caused by increased output contention, rather than by increased internal contention in the banyans. The addition of the randomizers reduces the likelihood of recurrent conflict internally to the networks, but of course is not useful against output conflicts. As the burst size increases, the degradation becomes severe, and eventually more banyans have to be provided in order to achieve low loss rates with small buffers. For L = 100, at load = 0.9, in the 32 × 32 switch with randomizers by pairs, K = 5 banyans and 60 buffers arc necessary to achieve a packet loss rate below 10^{-6}, as shown in Fig. 3.32.b. Of course, for lower traffic loads, the number of banyans and the required buffer size decrease rapidly; for example, as shown in Fig. 3.33, for the same switch at 0.7 load, with L = 100, 4 banyans and 31 input buffers suffice for 10” loss rate.

The fact that the number of banyans necessary to guarantee a desired packet loss rate may have to be increased to accommodate long bursts would become a concern for large switch sizes. Fortunately, as M and R increase, the performance degradation under bursty traffic is reduced and eventually practically disappears. This is because, if the destinations of bursts arriving at different input lines are statistically independent, as the size R of a group of output ports served by an output buffer increases, the number of bursts contending for the output buffers tends to decrease relatively to R, so that

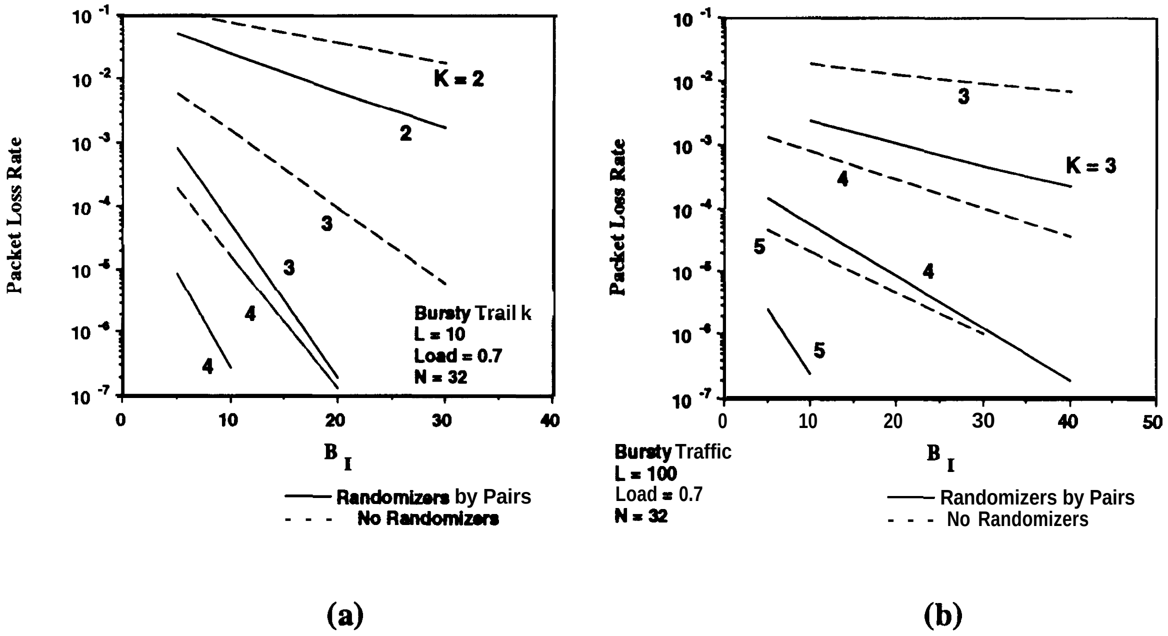
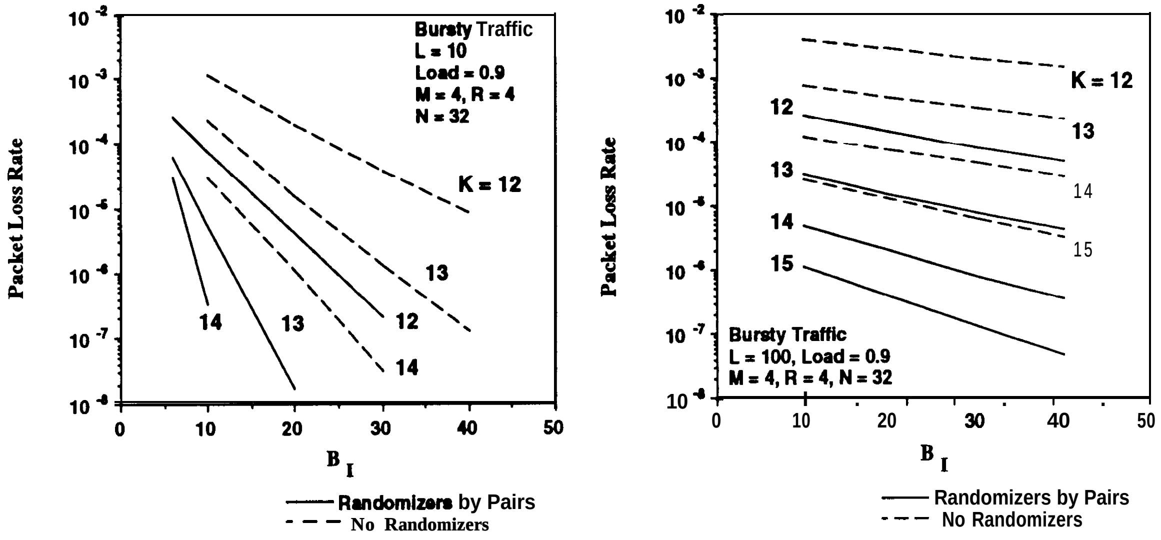


Fig. 3.33. Packet loss rate in the input buffers of an MSM with and without randomizers by pairs in front of the banyan networks, under bursty traffic, load = 0.7, ***M*** = 1, ***N = 32;*** a) ***L*** = 10; b) ***L*** = 100 (obtained by simulation).

eventually the output conflicts can be handled by the same number of banyans required under uniform traffic.

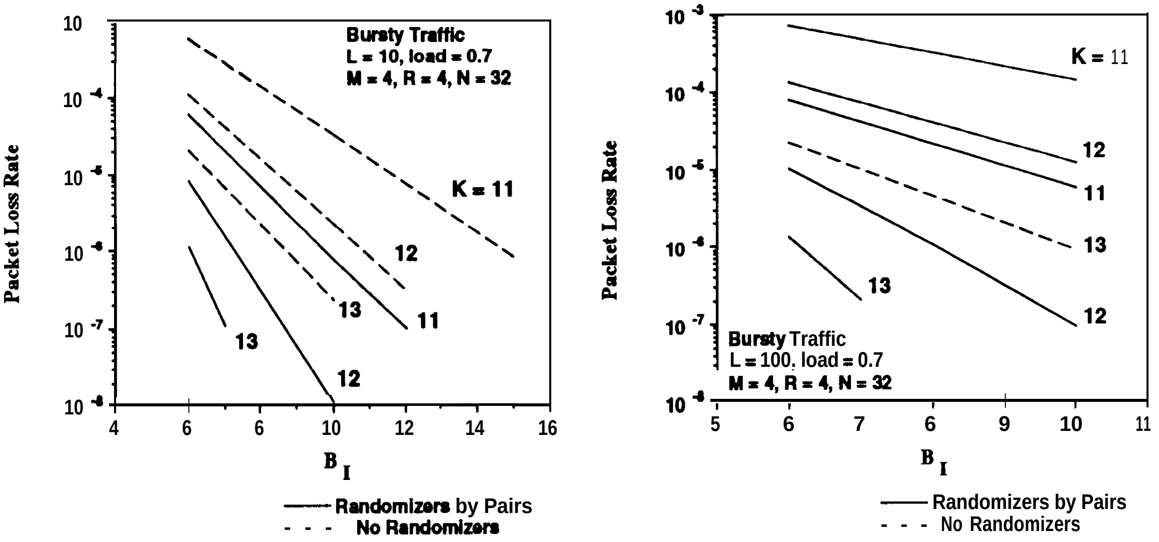
In order to fully characterize this effect, we have studied the performance of MSM switches of several sizes under bursty traffic. In Fig. 3.34, we show the packet loss rate in the input buffers of a 128 × 128 switch, ***M*** = 4, ***R*** = 4, at load = 0.9, obtained by simulation. With ***L*** = 10, with the randomizers, 24 buffers are required for 10^{-6} loss rate with ***K*** = ***3M*** = 12 banyans. For larger burst sizes, substantial degradation is still observable. With ***L*** = 100, and ***K*** = 13 banyans, 100 buffers are required for the same loss rate. Similar considerations apply for the same switch at 0.7 load, as shown in Fig. 3.35. As ***M*** and ***R*** increase further, however, while with no randomizers the performance of the switch is severely affected by bursty traffic, with the addition of the randomizers by pairs the performance becomes insensitive to the burst length, and the same number of banyans



(a)

(b)

Fig. 3.34. Packet loss rate in the input buffers of a 128 × 128 MSM switch, with and without randomizers by pairs in front of the banyan networks, under bursty traffic, load = 0.9, M = 4, R = 4, N = 32; a) L = 10; b) L = 100 (obtained by simulation).



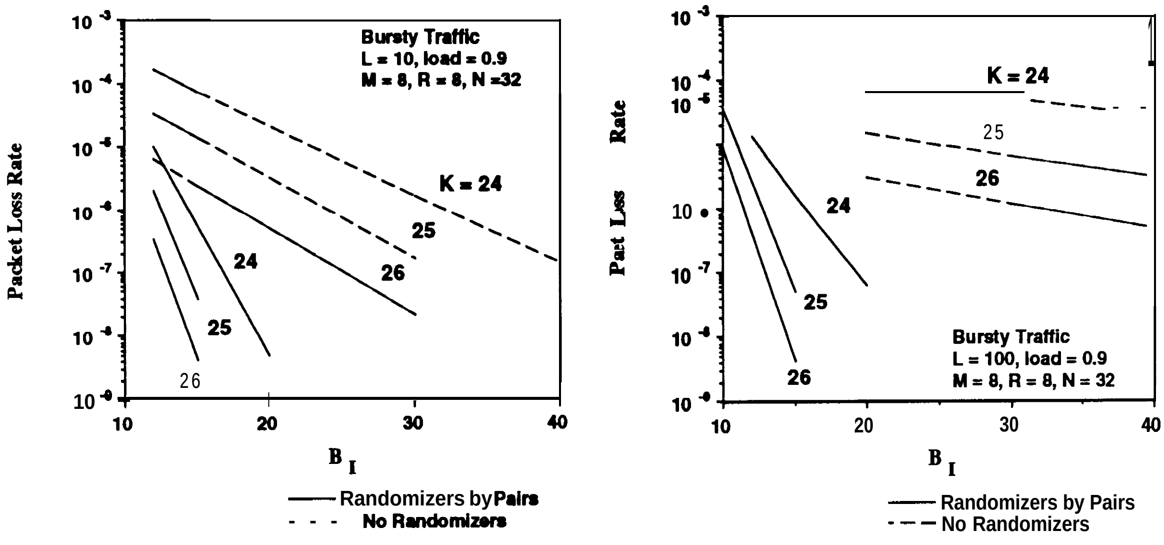
(a)

(b)

Fig. 3.35. Packet loss rate in the input buffers of a 128 × 128 MSM switch, with and without randomizers by pairs in front of the banyan networks, under bursty traffic, load = 0.7, M = 4, R = 4, N = 32; a) L = 10; b) L = 100 (obtained by simulation).

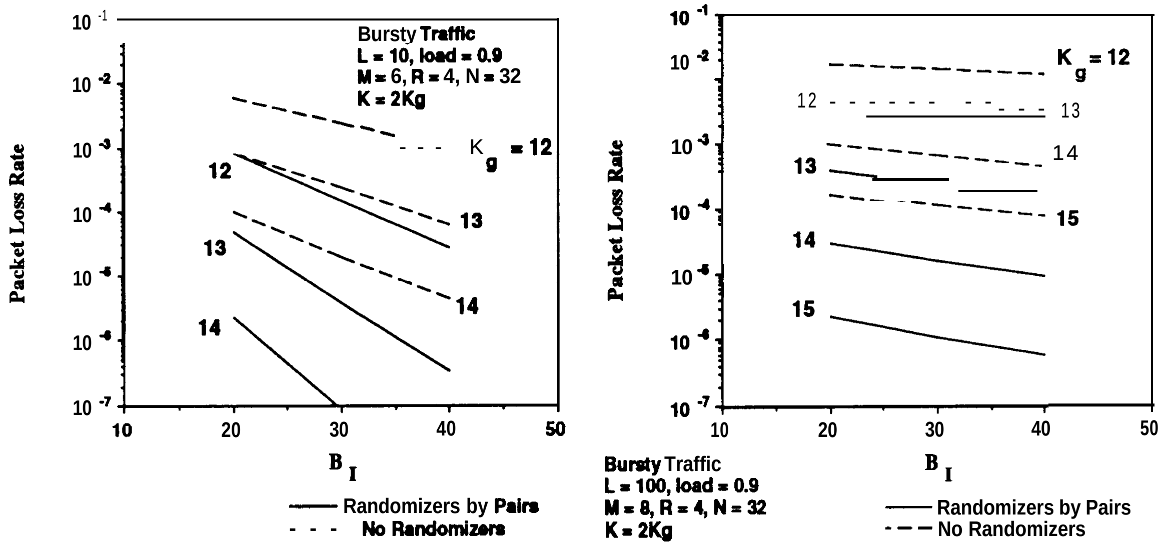
and basically the same buffer size required under uniform traffic are necessary to meet a desired loss rate under bursty traffic. This is substantiated in Fig. 3.36, where we show the performance of a 256 × 256 switch with M = 8, R = 8 under bursty traffic, at load = 0.9. For example, with 24 banyans augmented by randomizers by pairs, 15 buffers are necessary with L = 10 for 10^{-6} packet loss rate. With L = 100 and the same number of banyans, only one additional buffer is needed for the same loss rate. Of course, in the same switch, if R is smaller, then the performance degrades as the burst size increases. For example, for R = 4, the performance of the switch at 0.9 load is shown in Fig. 3.37. With K_g = 13 banyans per group, 36 buffers are needed for 10^{-6} packet loss rate with L = 10; with the same number of banyans and L = 100, the buffer size must be 150 to guarantee the same loss rate. Again, however, it should be noted that if the number of banyans per group K_g is not very large, the addition of a few banyans to achieve the desired performance with bursty traffic is not critical. To further characterize the effect of the output-group size R on performance, the packet loss rate in the input buffers in a 192 × 192 switch, M = 6, R = 6 and in a 224 × 224 switch, M = 7, R = 7, are plotted in Fig. 3.38 and Fig. 3.39, respectively. As M and R increase, the degradation in throughput in the switch with no randomizers is primarily due to recurrent internal conflicts rather than to increased output conflicts, so that the addition of the randomizers obviates for most of the problem, and the performance is less sensitive to the burst length. For example, with R = 6 and $K = 3M$ = 18 banyans, the required buffer size for 10^{-6} packet loss rate is 16 with L = 10, and becomes 36 with L = 100. For R = 7 and $K = 3M$ = 21 banyans, the requirements for the same loss rate are 19 buffers and 27 buffers for L = 10 and L = 100, respectively.

As expected, also under bursty traffic, the randomizer by pairs give almost the full advantage obtainable by randomization of the input patterns of each banyan. For example, as shown in Fig. 3.40, the performance of both a 128 ×128 switch, $M = R = 4$, and a



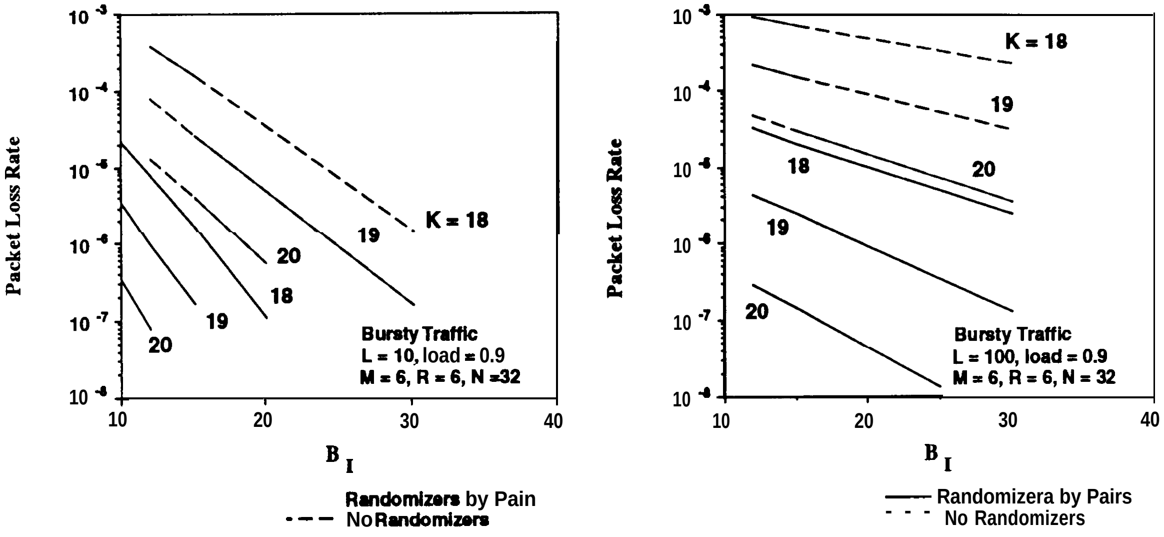
(a) (b)

Fig. 3.36. Packet loss rate in the input buffers of a 256 × 256 MSM switch, with and without randomizers by pairs in front of the banyan networks, under bursty traffic, load = 0.9, M = 8, R = 8, N = 32; a) L = 10; b) L = 100 (obtained by simulation).



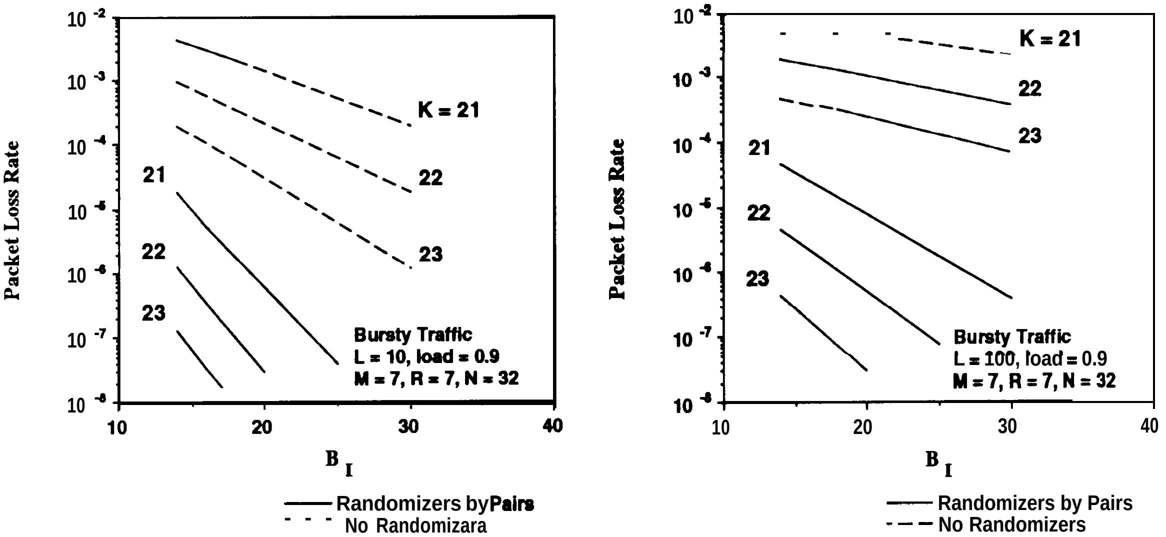
(a) (b)

Fig. 3.37. Packet loss rate in the input buffers of a 256 × 256 MSM switch, with and without randomizers by pairs in front of the banyan networks, under bursty traffic, load = 0.7, M = 8, R = 8, N = 32; a) L = 10; b) L = 100 (obtained by simulation).



(a) **(b)**

Fig. 3.38. Packet loss rate in the input buffers of a 192 x 192 MSM switch, with and without randomizers by pairs in front of the banyan networks, under bursty traffic, load = 0.9, M = 6, R = 6, N = 32; a) L = 10; b) L = 100 (obtained by simulation).



(a) **(b)**

Fig. 3.39. Packet loss rate in the input buffers of a 224 x 224 MSM switch, with and without randomizers by pairs in front of the banyan networks, under bursty traffic, load = 0.9, M = 7, R = 7, N = 32; a) L = 10; b) L = 100 (obtained by simulation).

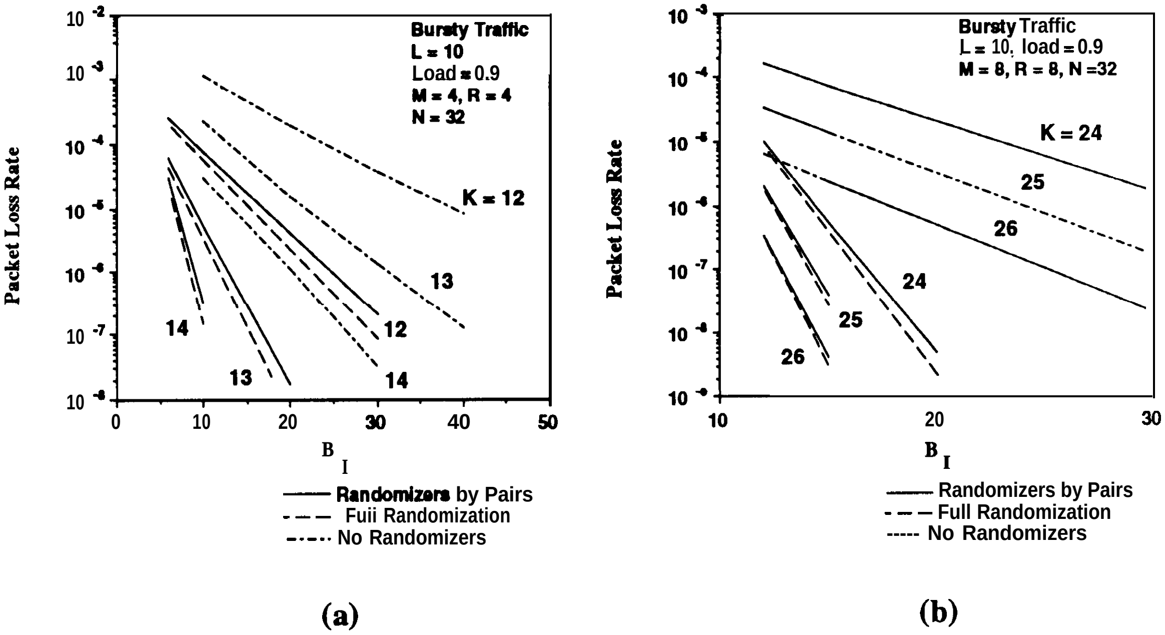


Fig. 3.40. Packet loss rate in the input buffers of an MSM switch with randomizers by pairs and with full randomizers in front of the banyan networks, under bursty traffic, ***L*** = 10, load = 0.9, ; a) *M* = ***8***, *R* = ***8, N = 32;*** b) ***M = 8, R = 8, N = 32*** (obtained by simulation).

256 x 256 switch, *M* = *R* = 8, with randomizers by pairs and with full randomizers in front of the banyan networks are practically comparable under bursty traffic with ***L*** = 10.

3.2.3. The MSM with Input-Memory Bandwidth Limitation

As discussed in Section 3.1.2 above, the required memory bandwidth of the input shared buffer is one of the factors that limit the maximum number of inputs that an input controller in the MSM can have. With *M* input ports per input component and ***K*** banyans in parallel (where ***K*** is about 3***M***), the input memory bandwidth must be adequate to sustain a flow to and from the memory of *M* + ***K*** packets per slot, since as many as ***K*** packets can be retrieved from an input buffer in a single slot. In this section, we observe that a restriction can be imposed on the maximum number of packets ***W*** that can be dispatched by an input controller in a slot without noticeable effect on the performance of the switch, thus alleviating the bandwidth requirements of the input shared buffer. Clearly, a limitation on the number of packets that can be dispatched in a slot, corresponds to a limitation on the bandwidth of the input buffer equal to $(M + W)V$ (hence, we denote an MSM switch with such a restriction as an MSM with *input-memory bandwidth limitation*).

Considering the switch at full load, M packets arrive at the switch per slot. Of course, to satisfy work conservation, the capacity offered by the K banyans must be adequate to sustain an average output flow from the buffer of M packets per slot. Since the buffer sizes (in packets) needed to achieve very low packet loss rates in the input buffers are comparable with K, regardless of the traffic conditions, it is reasonable to expect that the instantaneous variations from the work-conserving flow are actually rather gradual. In fact, in any given slot, although in principle a maximum of K packets can be successfully dispatched by an input controller, the likelihood that a number of packets substantially larger than M are indeed successful is very low because of the heavy blocking effect in the banyans. In other words, although in a slot the instantaneous output capacity of the buffer can be as high as K, we argue that such a capacity is actually never approached because of blocking, and the instantaneous capacity is with high probability close to the average capacity. If this is the case, a limitation on the maximum number of packets W that can be successfully transmitted in a slot, and a corresponding limitation on the bandwidth of the input buffer, should not significantly affect the performance of the switch. Clearly, in a fully loaded switch, a lower bound for W is M + I, in order to have a ratio of input to output capacity of the buffer (*i.e.*, an utilization factor) lower than one. We have studied the performance of the MSM switch with input-memory bandwidth limitation under various traffic conditions, and shown that this lower bound is actually sufficient to guarantee performance very similar to that of the switch with no limitation.

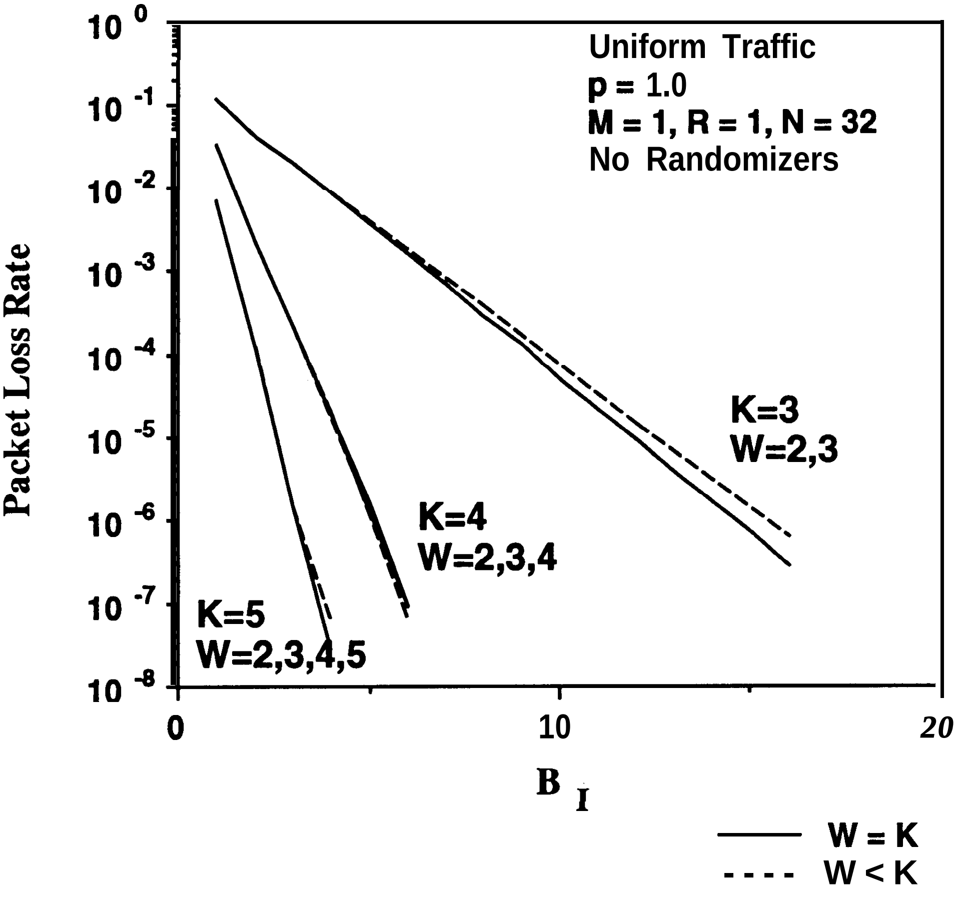


Fig. 3.41. Packet loss rate in the input buffers of the MSM switching fabric with limitation on the number of packets W that can be successfully dispatched in a time slot, $M + 1 \leq W \leq$ K, under uniform traffic at full load, $M = 1$, $N = 32$ (obtained by simulation).

In Fig. 3.41, we show the packet loss rate in the input buffers of a 32 × 32 MSM with $M = 1$, under uniform traffic conditions at full load, for $M + 1 \leq W \leq$ K. No noticeable degradation of the switch performance is induced by the limitation on the number of packets that can be dispatched in a slot. A similar conclusion is valid for the same switch under bursty traffic at 0.9 load, as shown in Fig. 3.42. Actually, the limitation on the number of packets even results in slightly improved performance. This is because

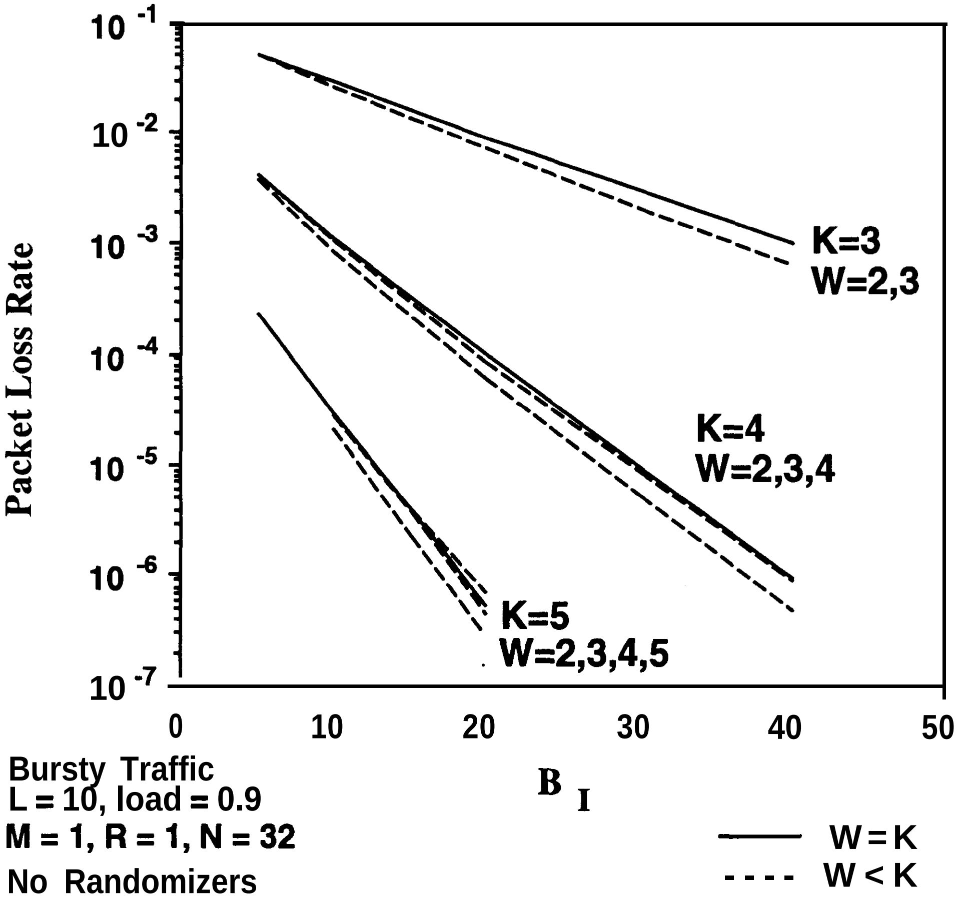


Fig. 3.42. Packet loss rate in the input buffers of the MSM switching fabric with limitation on the number of packets W that can be successfully dispatched in a time slot, $M + 1 \leq W \leq K$, under bursty traffic, $L = 10$, load = 0.9, $M = 1$, N = 32 (obtained by simulation).

the limitation reduces the effect of conflicts between successive packets belonging to the same bursts that continue to collide in different banyans.

Of course, the fact that the input-memory bandwidth can be reduced without affecting the performance of the switch is very relevant from an implementation point of view when M is large. In Fig. 3.43, we show the packet loss rate in the input buffers of a 256×256 MSM with $M = 8$ under uniform traffic at full load, for $M + 1 \leq$ WI K and various K. For $K = 25$ banyans, the limitation on the maximum number of packets that can be dispatched in a slot has no observable influence on performance. For $K = 24$ banyans, no effect is noticeable for $W \geq M+ 2$. For $W = M+ 1$, however, a modest increase in buffer size is required to guarantee the same packet loss rate as in the switch with no limitation. For $K = 23$ banyans, the limitation on the number of packets has negligible effect for $W \geq M + 2$. For $W = M + 1$, a sizable increase in buffer size is required to meet the same loss rate as in the MSM with no limitation. Under bursty traffic conditions, at load = 0.9, the limitation on the number of packets deliverable in a slot has no influence on performance for $W \geq M + 1$, as shown in Fig. 3.44. Similarly to the 32×32 switch, the limitation on the number of packets is even slightly beneficial since it reduces persistent contentions among packets belonging to the same bursts.

From this discussion, we conclude that, for $K \geq 3M$, the maximum number of packets that can be transmitted from an input buffer in a slot can be limited to $W = M + 1$ with no practical degradation in performance. This corresponds to a reduction of almost a factor of two in the required memory bandwidth, which has to be adequate to sustain a flow to and from the memory of only $2M + 1$ packets per slot, as opposed to $M + K \approx 4M$ packets per slot as in the switch with no limitation.

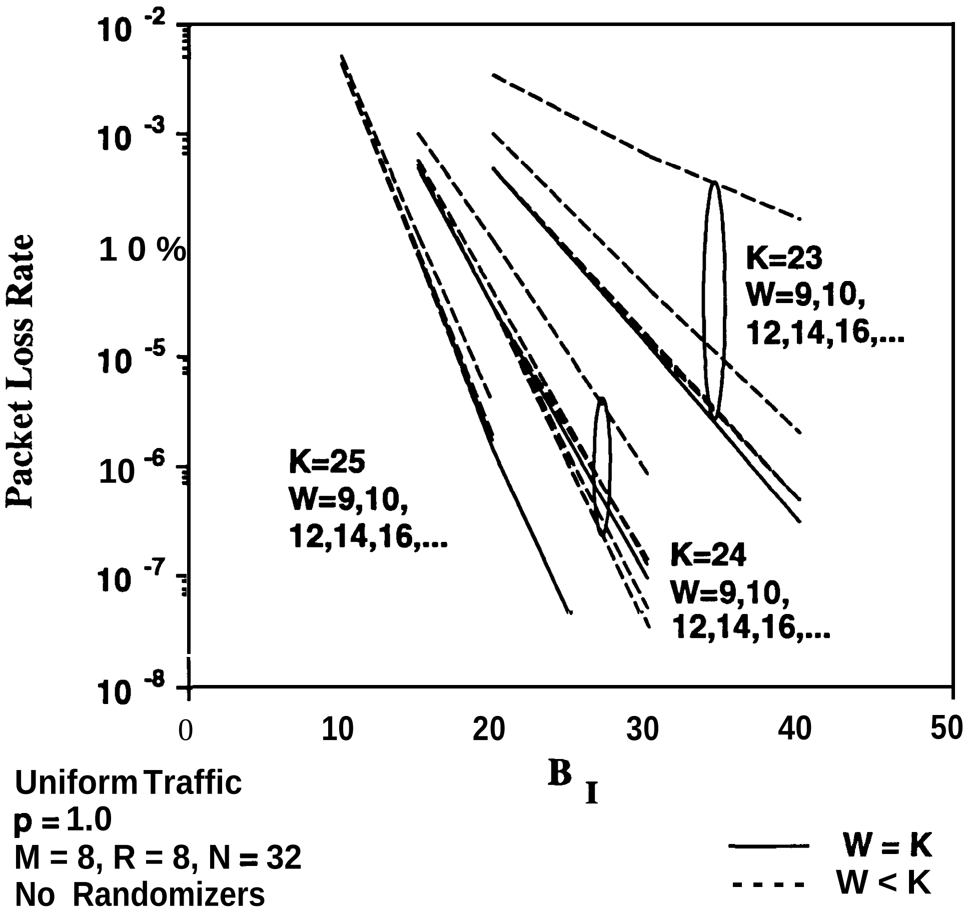


Fig. 3.43. Packet loss rate in the input buffers of the MSM switching fabric with limitation on the number of packets *W* that can be successfully dispatched in a time slot, *M* + 1 ≤ WI K, under uniform traffic at full load, ***M = 8, N = 32*** (obtained by simulation).

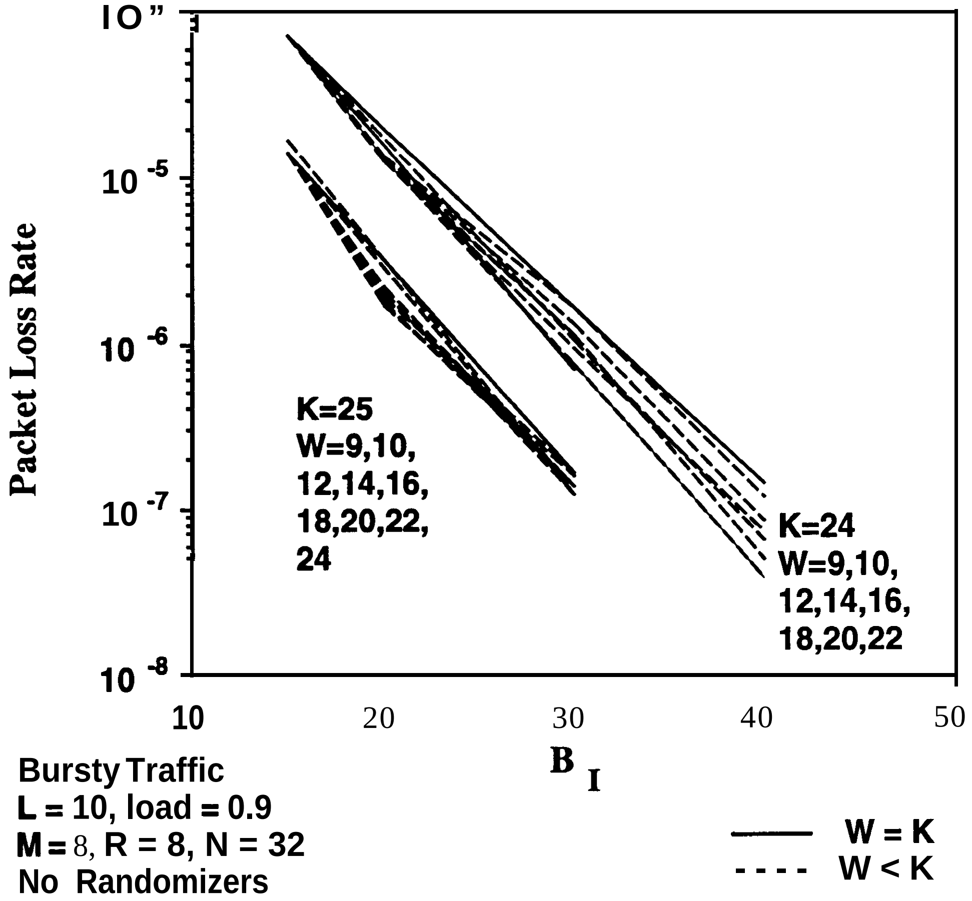


Fig. 3.44. Packet loss rate in the input buffers of a 256 X 256 MSM switching fabric with limitation on the number of packets ***W*** that can be successfully dispatched in a time slot, ***M*** + 1 ≤ ***W*** ≤ K, under bursty traffic, ***L*** = 10, load = 0.9, ***M = 8, N = 32*** (obtained by simulation).

3.2.4. **The MSM with Crossbars**

With large *M*, if the number of networks in parallel in the MSM becomes excessive, an architectural modification of the MSM that can be considered is one that replaces the banyan networks with crossbars. In fact, since the throughput of a crossbar is higher than the throughput of a banyan network, less networks in parallel would be required to provide the necessary routing capacity. The switch operation remains the same as in the original MSM, so that each packet is dispatched by the corresponding input controller to the crossbars in sequence, until it is successfully routed to the requested output buffer by one of the networks. (Self-routing operation of the crossbars is assumed.)

Note that, in this configuration, the MSM directly resembles the Growable Switch [Eng89]. In contrast with the latter architecture, however, the MSM provides buffering at the inputs to increase the utilization of the routing networks, so as to use a relatively small number of networks without requiring intelligent routing to assign the path to the packets. In the following, we show that the use of crossbars reduces the number of routing networks required with a given *M* by about 30% under uniform traffic; the advantage is however less substantial under bursty traffic. Unfortunately, from a practical point of view, we observe that, since the duration of the route set-up phase in crossbars increases linearly with the size of the network, the size of crossbar networks that are realizable with current technology is severely limited. For this reason, crossbars are less suitable than banyan networks to be used as building blocks for the routing fabric of the MSM.

In Fig. 3.45, we show the packet loss rate in the input buffers of a 32 × 32 MSM with crossbars, $M = 1$, under uniform traffic at full load. With 2 crossbars in parallel, 11 packet buffers are necessary for a packet loss rate below 10”. This minimum of two crossbars in parallel is easily anticipated, since the throughput of a single crossbar of size

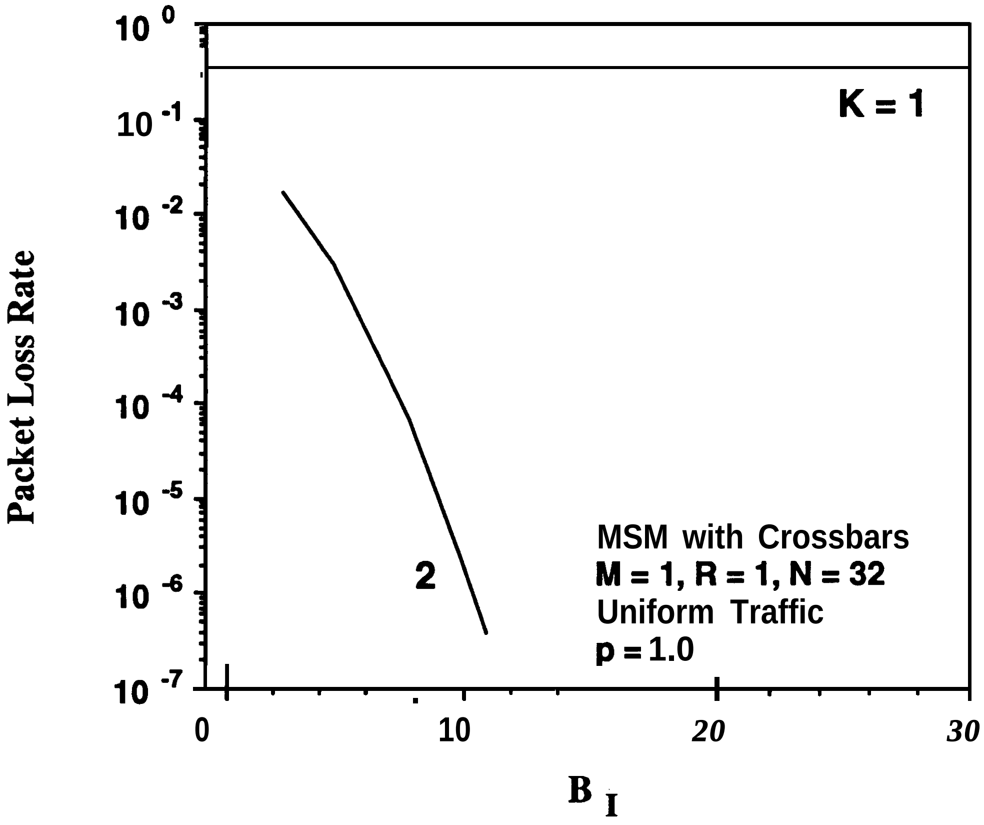


Fig. 3.45. Packet loss rate in the input buffers of the MSM switching fabric with crossbars, under uniform traffic at full load, M = 1, N = 32 (obtained by simulation).

32 under uniform traffic at full load is $1 - (1 - 1/32)^{32} = \boldsymbol{0.638}$, making the total capacity with two networks larger than 1.2. The 20% extra capacity explains the relatively small buffer sizes needed. With $M > 1$, as a rule of thumb, less than $K = 2M$ crossbars in parallel are required to achieve low packet loss rates in the input buffers under uniform traffic at full load. For example, as shown in Fig. 3.46, in a 256 × 256 MSM with crossbars, $M = 8$, a minimum of 14 crossbars in parallel is necessary to guarantee low packet loss rates with manageable buffer sizes. With $K = 14$ crossbars, 61 packet buffers are needed for a loss rate below 10^{-6}. The required number of buffers decreases fairly rapidly if more crossbars in parallel are provided; with $K = 15$ crossbars, 42 buffers are needed for 10” loss rate, with $K = 16$, 31 buffers suffice for the same loss rate.

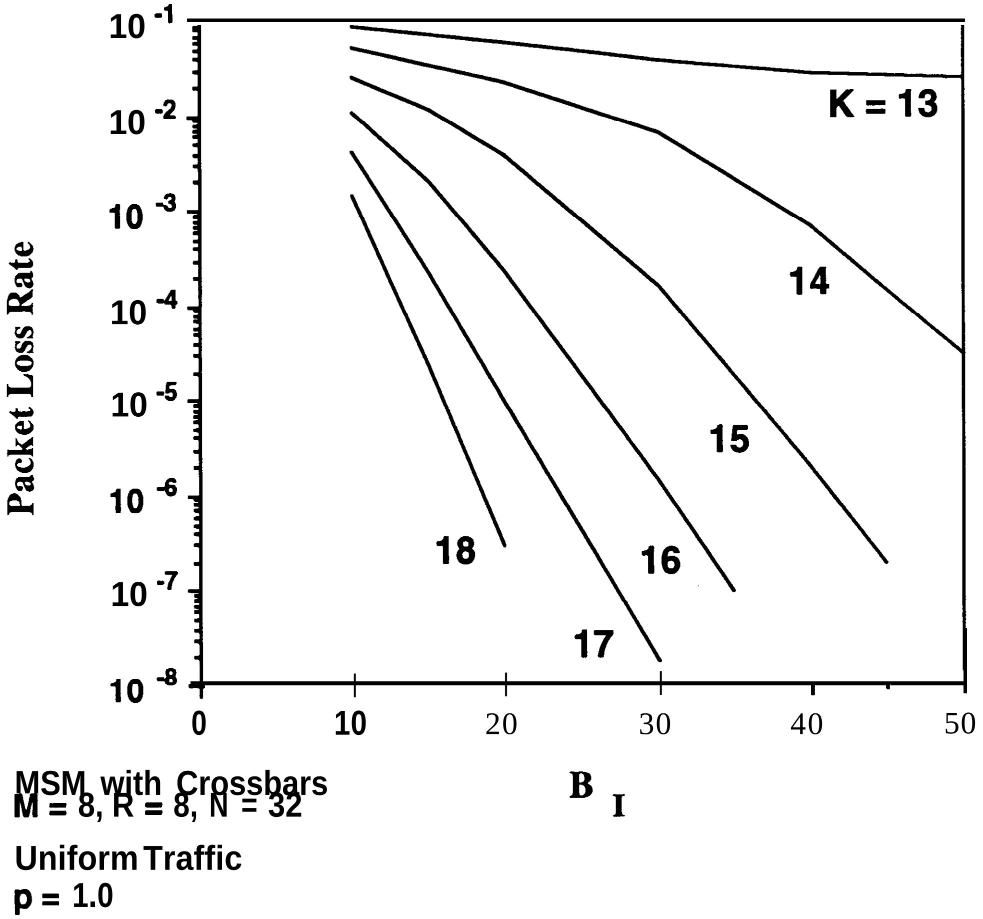


Fig. 3.46. Packet loss rate in the input buffers of a 256 x 256 MSM switch with crossbars, under uniform traffic at full load, ***M*** = 8, ***R*** = 8, N = 32 (obtained by simulation).

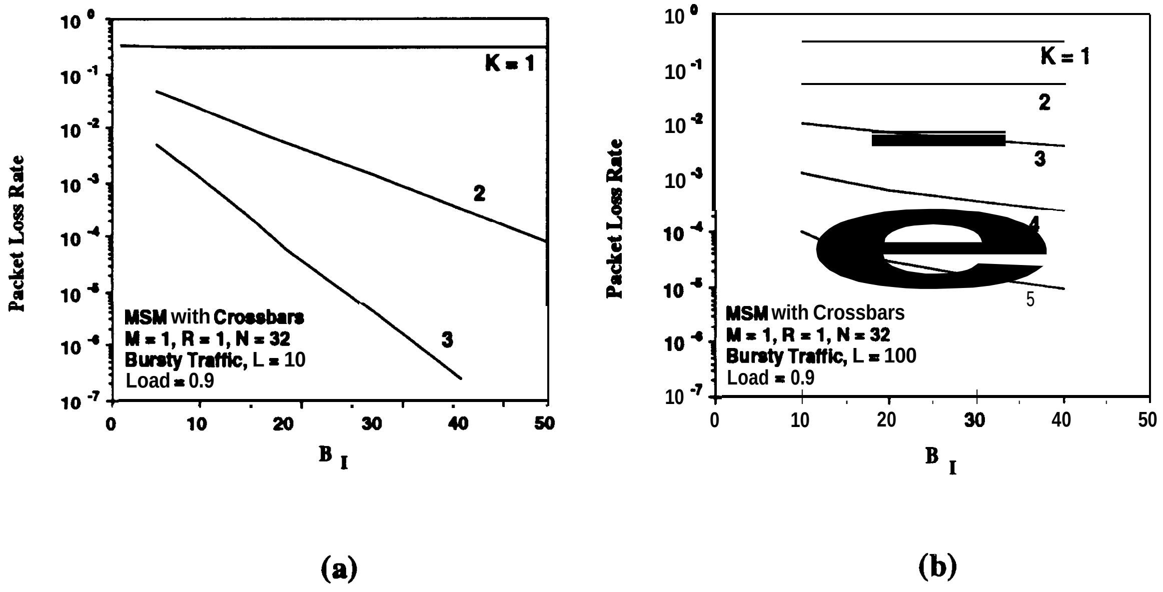
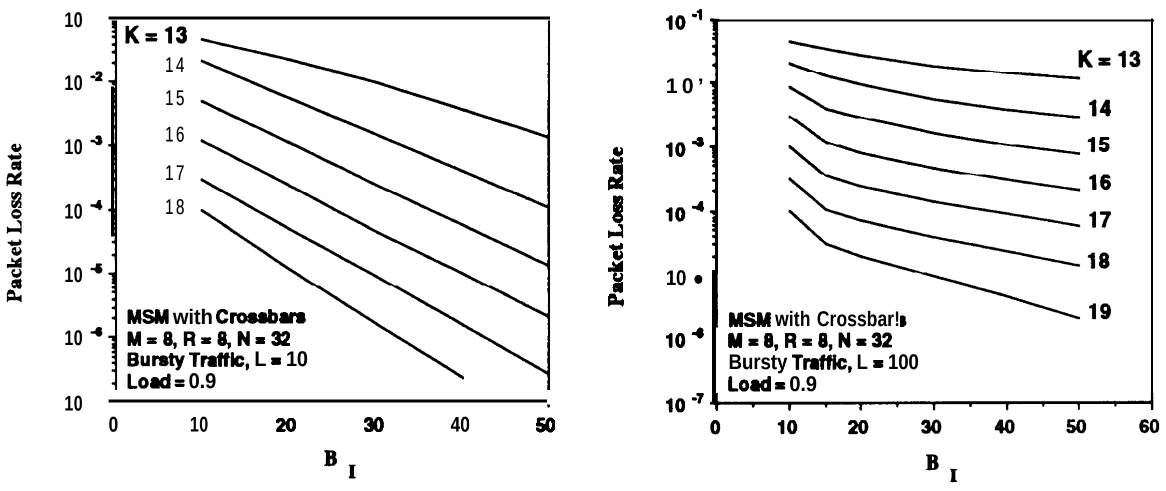


Fig. 3.47. Packet loss rate in the input buffers of the MSM switching fabric with crossbars, under bursty traffic, load = 0.9, M = 1, N = 32; a) L = 10; b) L = 100 (obtained by simulation).

Under bursty traffic, head-of-the-line blocking in the input buffers increases, significantly affecting the performance. Consequently, larger buffers and/or more networks in parallel are needed to meet a desired packet loss rate. The degradation is more and more noticeable as the average burst length increases. For example, in a 32 x 32 MSM, with 2 crossbars in parallel, about 80 buffers are necessary to achieve a packet loss rate below 10^{-6} under bursty traffic with average burst length L = 10 at 0.9 load, as shown in Fig. 3.47.a; with K = 3, 35 buffers are needed for the same loss rate. With L = 100 at the same load, as shown in Fig. 3.47.b, at least 4 crossbars in parallel are required to achieve low loss rates with manageable buffer sizes. With K = 4 crossbars, more than 150 buffers are needed for 10^{-6} loss rate; with K = 5 crossbars more than 80 buffers are necessary. The packet loss rate in the input buffers of a 256 x 256 MSM, $M = 8$, $N = 32$, is shown in Fig. 3.48.a, under bursty traffic with L = 10, at load = 0.9. At this load, a minimum of 13 crossbars in parallel is required. With K = 13 crossbars, more than 120 packet buffers are



(a) (b)

Fig. 3.48. Packet loss rate in the input buffers of a 256 x 256 MSM switch with crossbars, under bursty traffic, load = 0.9, ***M* = 8, *R* = 8, *N* = 32;** a) *L* = 10; b) *L* = 100 (obtained by simulation).

necessary to guarantee 10” loss rate; for the same packet loss rate, the required buffers sizes are 90, 70, and 54 buffers for ***K*** = 14, 15, and 16 crossbars, respectively. With *L* = 100, as shown in Fig. 3.48.b, 200 buffers are needed for 10^{-6} loss rate with 16 crossbars in parallel. ***K*** = 18 crossbars are necessary to meet the same loss rate with a buffer size smaller than 100 packets. With ***K*** = 19 crossbars, the required buffer size is 60.

Similarly to the case of the MSM with banyan networks, also here the number of crossbars in parallel ***K*** must satisfy:

$$K \leq T / T_s \qquad (15)$$

in order for the sequential scanning of the crossbars to be completed within a slot duration ***T.*** This constitutes a major limitation on the number of crossbars that can be used in parallel; in fact, the duration of the set-up phase $\boldsymbol{T_s}$ in crossbars is significantly longer than that in banyan networks, as now explained. (Note that, in the case G $>$ 1, (15) becomes $K_g \leq T / T_s$, and the limitation is on the number of crossbars K_g in parallel per group.)

From a practical point of view, to realize the crossbars to be used in the MSM, either a self-routing crossbar network or a memory interchange may be implemented on a chip. With the former solution, during the route set-up phase, a header must traverse, in the worst case, 2N $-$ 1 crosspoints in the network in order to reach its destination, as opposed to log, N switching elements in case of banyan networks. (Note that in a self-routing crossbar, in order to permit all input lines to gain fair access to the crosspoints, the delay incurred by packets from each input line to reach each crosspoint must be equalized.) Thus, the duration of the set-up phase increases linearly with the size of the crossbar (in fact, T_s is proportional to the number of crosspoints that have to be set), and becomes rapidly very large. Self-routing operation of a crossbar requires knowledge of the complete destination address of the packets at each crosspoint; this would require to incur a minimum of (log, N)-bit delay to set each crosspoint. Lower delay per crosspoint may be achieved by using a (2 N $-$ 1)-bit local switching address, where each bit represents the requested state of a crosspoint; in this way, a minimum of l-bit delay per crosspoint may be incurred. Even in this case, a crossbar of size 32 x 32 would require a T_s more than ten times longer than a banyan network of the same size (this of course assumes that the delay incurred to set a switching element in the banyan network is also 1 bit). Neglecting any other delay that may be incurred in T_s (e.g., to generate and transmit the feedback signal back to the controllers), and assuming for simplicity that the same clock rate is used in route set-up and transmission phases, an upper bound on the number of 32 x 32 crossbars that can be used in parallel with ATM packets is 7 ((53 x 8 + 63) / 63).

An alternative solution for the realization of the crossbars is a memory interchange. Such a component comprises a common memory of N packet buffers, one for each of its N outputs. In each slot, inputs are served sequentially, and the incoming packets are written in the memory locations corresponding to the desired output destinations. To resolve output conflicts, for each memory location, a busy flag prevents that a packet overwrites a memory location that has already been occupied in the current slot. For each packet, after the write operation has been performed, a signal indicating whether the write operation has been successful or not is fed back to the corresponding input controller. Fair service of the inputs is guaranteed by servicing them in round-robin fashion. Once packets have been written in the desired memory locations, they are read from the memory and transmitted to the corresponding output buffers. Unfortunately, a severe limitation on the number of crossbars that can be used in parallel comes from the required memory bandwidth; in fact, since all the write operations have to be completed within T_s, *in* order to use $K = T / T_s$ crossbars of size N × N in parallel, the required memory bandwidth is equal to $K \, . \, N \cdot V$. For example, with ATM packets at 155.52 Mb/s data rate, by using a 20-ns cycle-time memory with maximum possible parallelism in the memory access equal to 424 bits (53 octets x 8 bits), with 32 x 32 crossbars only K = 4 crossbars in parallel can be used $(424 / (32 \times 155.52 \times 10^6 \cdot 20 \times 10^{-9}))$. From these considerations, it is evident that, with crossbars, as M increase, G $>$ 1 must be used to keep the required number of crossbars per group low.

3.2.5. The MSM Output Buffers

At the low packet loss rates for which fast packet switches are designed $(< 10^{-6})$, the traffic that reaches the output buffers is virtually equal to the incoming traffic destined to the corresponding output port. The slight decrease in traffic due to packet loss in the input buffers is insignificant so far as to determine the output buffer size B_o required to achieve a desired packet loss rate in the buffer. Consequently, the output buffer requirements can be studied by assuming infinite input queues. In the MSM, as in any switch achieving output buffering, the output-buffer requirements are highly dependent on the traffic conditions and on the degree of memory sharing in the buffers.

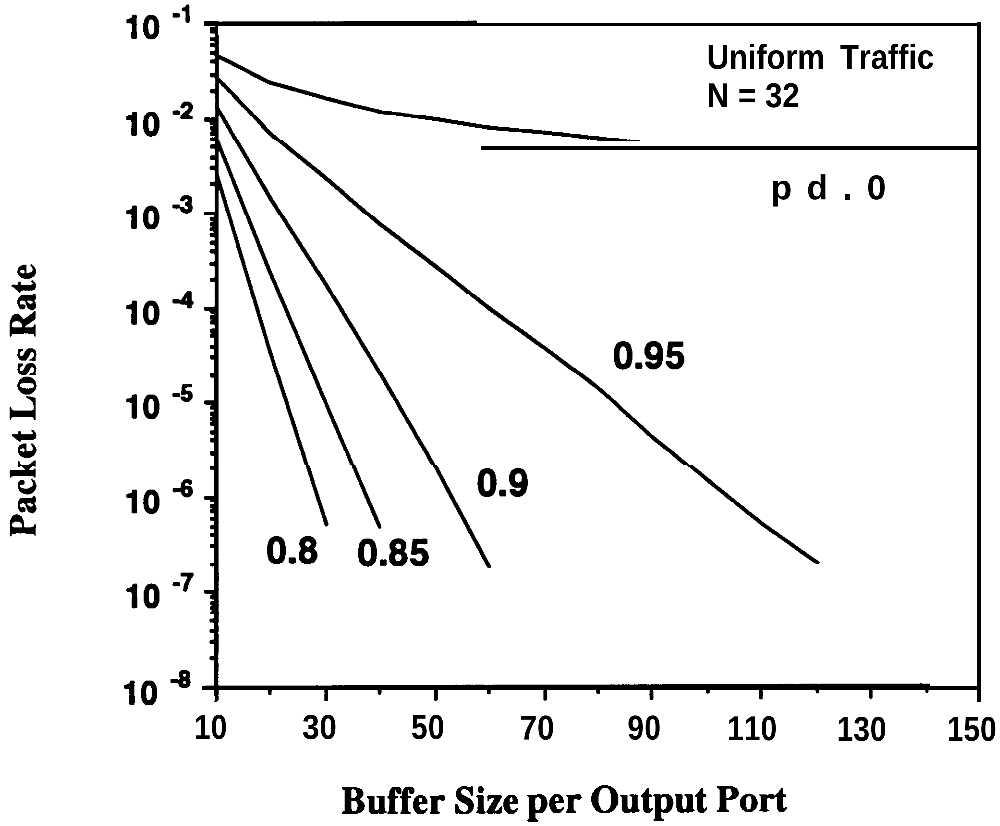


Fig. 3.49. Packet loss rate in the output buffers of a 32 x 32 MSM under uniform traffic conditions, for various loads, $\boldsymbol{R} = 1$, N = 32 (obtained by simulation).

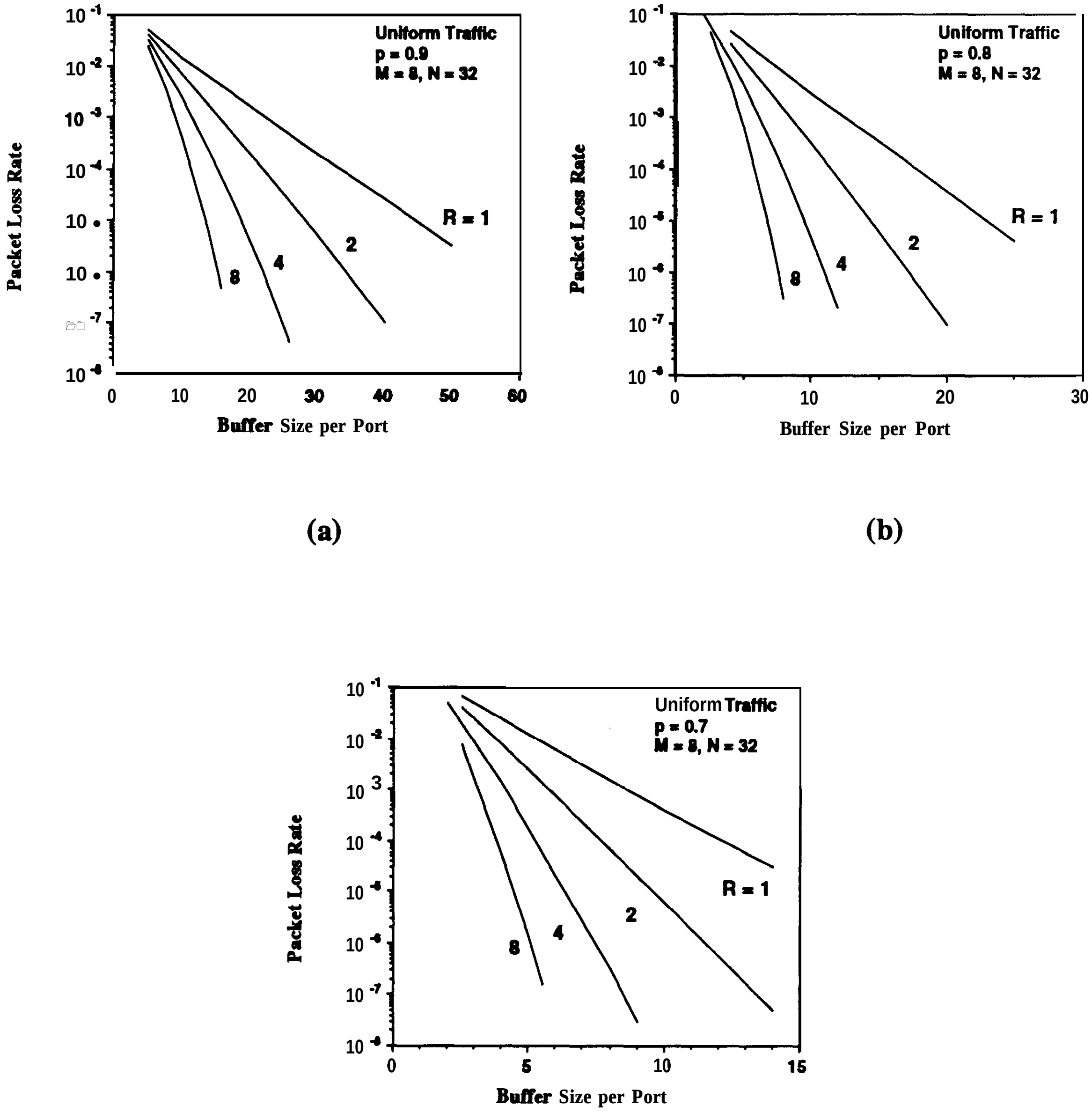
a) Uniform Traffic

The output-buffer requirements in an output buffer switch under uniform traffic conditions have been extensively studied by several authors [Hlu87,Hlu88,Eck88], for both completely-partitioned and shared memories. The packet loss rate in the output buffers of a 32 ×32 MSM, $\boldsymbol{M} = \boldsymbol{R} = 1$, under uniform traffic and various loads p, obtained by simulation, is shown in Fig. 3.49. Of course, these results are in very good agreement with those obtained by analysis in [Hlu87,Hlu88] for a generic output buffer switch. Under these traffic conditions, the required buffer sizes are relatively small. At loads as high as 0.95, 110 packet buffers per port are sufficient to guarantee packet loss rates lower than 10^{-6}. Assuming 53-byte ATM cells, this translates in memory sizes of less than 50 Kbit. The buffer requirements decrease rapidly as the load decreases. At 0.9 load, 51 packet buffers are necessary to achieve 10” loss rate; at 0.8 load, 28 buffers are required for the same loss rate.

With $R > 1$, each output buffer is shared among R ports. As well known [Hlu87,Hlu88], the required buffer size per port decreases significantly as the memory is shared among more and more ports. In Fig. 3.50, the buffer requirements per output port in a 256 × 256 MSM with $M = 8$ and N = 32, obtained by simulation, are shown for R ranging from 1 to 8, at various loads p. For $p = 0.9$, 56 packet buffers are necessary to achieve a packet loss rate below 10” with completely-partitioned buffers $(R = 1)$; a substantial reduction in buffer size per port is attained for $R = 2$; in this case, 35 buffers per port suffice for the same packet loss rate. The reduction continues for larger R, albeit it is progressively of a lesser extent; for $R = 4$, 23 packet buffers per port, and for $R = 8$, 15 buffers per port are required for a loss rate below 10”. As with completely-partitioned buffers, also with shared buffers the buffer size per port is of course highly dependent on the load; at $p = 0.7$, 20 packet buffers per port are required for $R = 1$, and 12, 8, and 5 buffers per port are necessary for $R = 2, 4, 8$, respectively, for a 10” loss rate. In general, in each output buffer component, the buffer requirements to accommodate uniform traffic are fairly small, for both completely-partitioned and shared buffers. Assuming ATM cells, with $R = 1$, a memory of less than 25 Kbit is required in each single-port output buffer to sustain a 0.9 load; for $R = 8$, a total memory of 50 Kbit in each 8-port output buffer component is necessary at the same load.

b) Bursty Traffic

Under bursty traffic, the size of the output buffers required to guarantee a desired packet loss rate is substantially larger than that needed under uniform traffic conditions



(a)

(b)

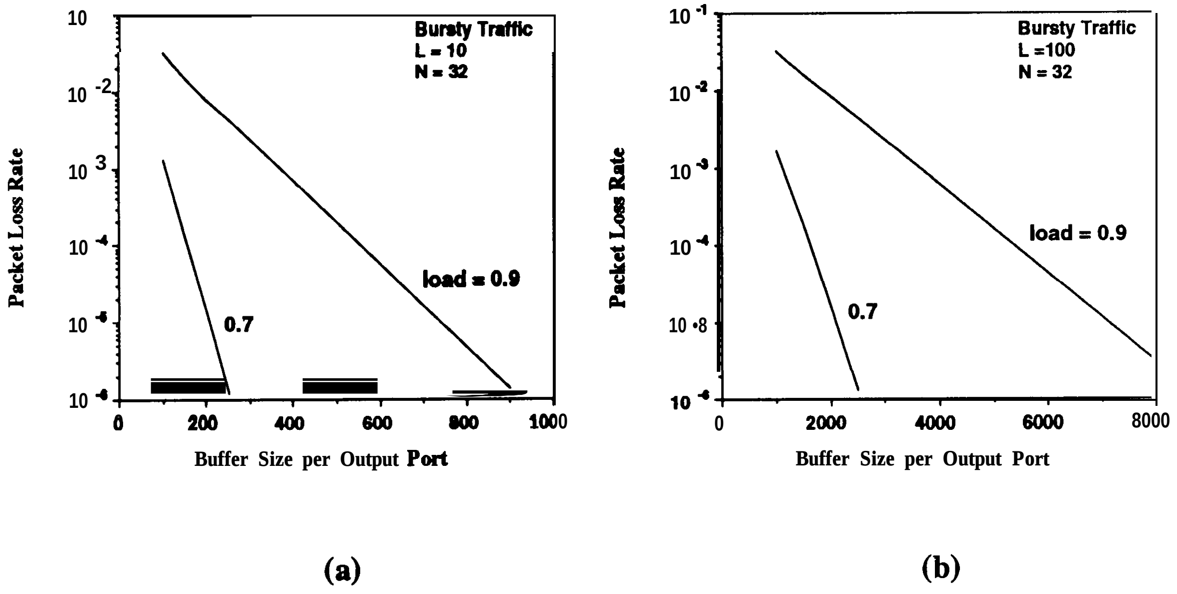
(c)

Fig. 3.50. Packet loss rate in the output buffers of a 256 × 256 MSM under uniform traffic conditions, for different ***R, M = 8;*** a) ***p*** = 0.9; b) ***p*** = 0.8; c) ***p = 0.7*** (obtained by simulation).

[End90,Lie90]. We have studied the output buffer requirements in the MSM switching fabric using the same bursty traffic model described in Section 2.3 above. The packet loss rate in the output buffers of a 32×32 MSM with M = R = 1, at loads equal to 0.9 and 0.7, obtained by simulation, is shown in Fig. 3.5 1.a and 1.5 l.b, for average burst lengths L = 10 and L = 100, respectively. At load = 0.9, for L = 10, more than 920 packet buffers per port are necessary to achieve a loss rate below 10^{-6} ; the increase in buffer size is quite dramatic respect to the 51 packet buffers required under uniform traffic conditions at the same load. The required buffer size for a desired packet loss rate increases approximately linearly with the average burst size. For L = 100, 9000 buffers are necessary for a 10” loss rate. Again, the buffer requirements are very much dependent on the load. At load = 0.7, with L = 10,255 packet buffers are necessary for the same loss rate; with L = 100, 2600 buffer arc needed.

Also under bursty traffic conditions, memory sharing substantially decreases the required output buffer size per port to meet a given packet loss rate. To illustrate the effect of sharing, in Fig. 3.52 we plot the packet loss rate in the output buffers of a 128×128 MSM with M = R = 4, for average burst sizes L = 10 and L = 100. At 0.9 load, with L = 10,440 packet buffers per port suffice for 10^{-7} loss rate. The increase in buffer requirements with the burst size is still approximately linear; with L = 100, 4600 buffers per port are necessary for the same packet loss rate. At 0.7 load, 110 and 1100 packet buffers are required for L = 10 and L = 100, respectively.

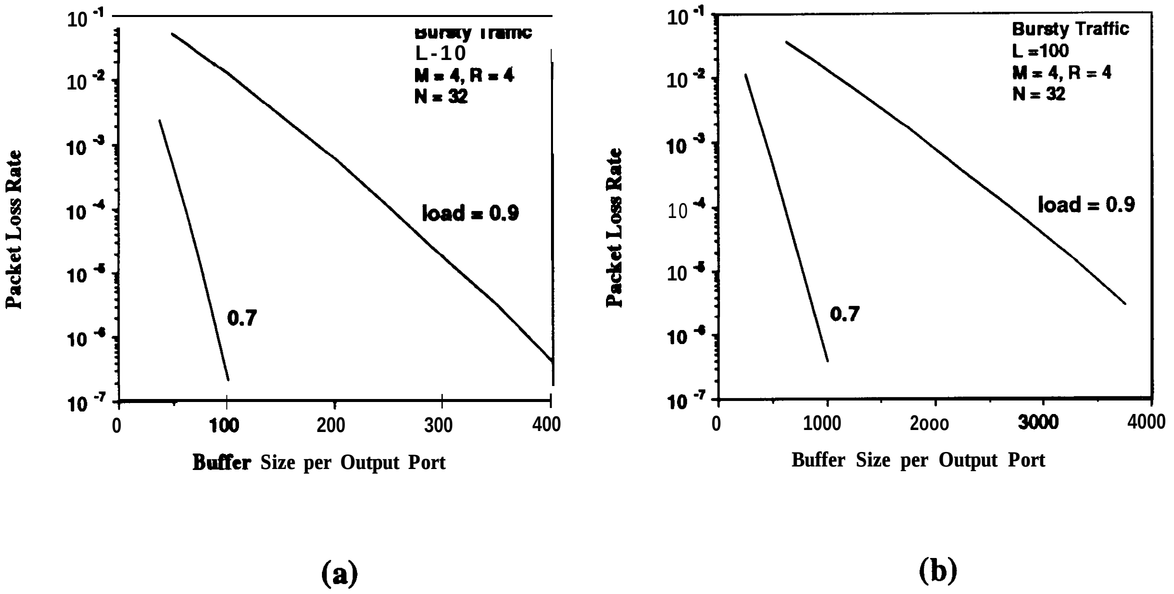
In each output-buffer component, the local buffer requirements to sustain bursty traffic conditions are clearly quite large. With ATM cells, in the 32×32 switch, the required memory size in each single-port output-buffer component is approximately 4 Mbit, to accommodate bursts with average size of 100 packets at a 0.9 load. In the 128×128 switch, each four-port output component requires a memory of about 8 Mbit to sustain the same traffic.



(a)

(b)

Fig. 3.51. Packet loss rate in the output buffers of an MSM, under bursty traffic, for 0.9 and 0.7 load, $\boldsymbol{N}$ = 32; a) L = 10; b) L = 100 (obtained by simulation).



(a)

(b)

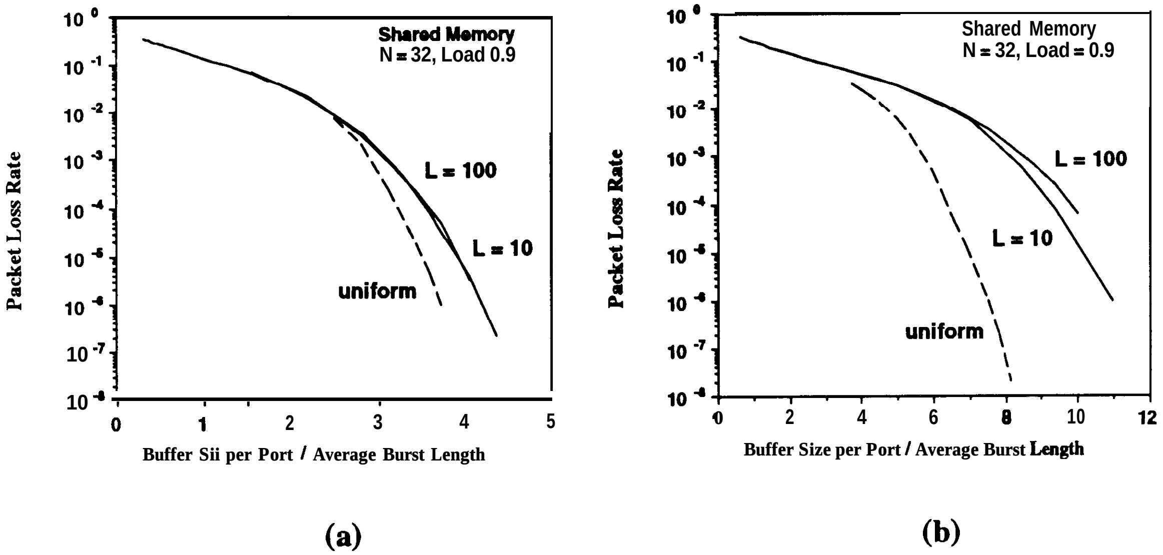
Fig. 3.52. Packet loss rate in the output buffers of a 128 x 128 MSM, under bursty traffic, for 0.9 and 0.7 load, $\boldsymbol{M}$ = 4, R = 4, $\boldsymbol{N}$ = 32; a) L = 10; b) L = 100 (obtained by simulation).

It is evident that given the approximately linear increase of the buffer size with the average burst length, the local output buffer requirements to sustain realistic traffic conditions represent a serious design and implementation concern, regardless of the degree of memory sharing used. It is important to note that these considerations do not pertain only to the MSM, but apply in general to any switch achieving output buffering, regardless of its architecture.

4. The Memory/Space/Memory Switching Fabric with Backpressure Under Bursty Traffic

In a switch achieving output buffering, under bursty traffic conditions, the buffer size per port required to meet a desired packet loss rate increases approximately linearly with the average burst length. This behavior is observed regardless of the architecture of the switch, and is independent of whether shared buffers or completely partitioned buffers are used. For example, for bursts with geometrically distributed lengths and uniformly distributed destinations, this linear increase in buffer size is clearly shown in Fig. 4.1 and Fig. 4.2, where we plot the packet loss rate as a function of the required buffer size per port normalized to the average burst length *L* for a generic switch of size 32 x 32 with shared buffers and partitioned buffers, respectively, at loads equal to 0.8 and 0.9. Similar results can be obtained for switches of different sizes.As the burst size increases, the buffer requirements become quite large. For example, in case of partitioned buffers, the required buffer size per port to achieve a packet loss rate below 10” is approximately $91 \cdot L$ packet buffers at 0.9 load; with ATM cells, a memory of about 0.5 MBytes per port is therefore necessary to accommodate bursts with average length of 100 packets; in case of shared buffers, the buffer size per port to meet the same packet loss rate is roughly 11. *L*, making the total memory necessary to handle bursts with average length of 100 packets in a 32 x 32 shared-buffer switch about 2 MBytes; (as a comparison, the current realization of such a switch with the largest storage capacity has a memory of about 0.25 MBytes [Kuw89,Koz91]). It should be noted that a NO-packet burst contains a user payload of less than 5 KBytes; thus, in realistic traffic conditions, much longer bursts may be expected, and much larger buffer sizes would be necessary to handle them.

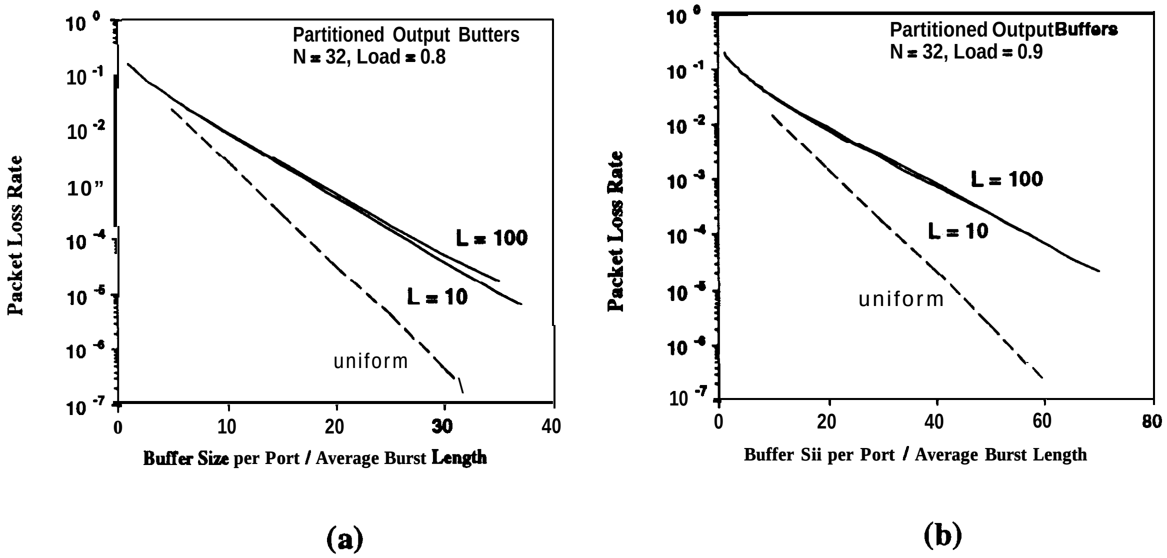
It is quite apparent from these examples that, given the linear increase of the required buffer size with the average burst length, the buffer size in each buffer component



(a)

(b)

Fig. 4.1. Packet loss rate vs. buffer size per port normalized to the burst length *L* (in packets) in an output buffer switch of size 32 x 32 with shared buffers, under bursty traffic; a) load = 0.8; b) load = 0.9 (obtained by simulation).



(a)

(b)

Fig. 4.2. Packet loss rate vs. buffer size per port normalized to the burst length *L* (in packets) in an output buffer switch of size 32 x 32 with partitioned buffers, under bursty traffic; a) load = 0.8; b) load = 0.9 (obtained by simulation).

(referred to as the ***local*** buffer size) necessary to guarantee a desired packet loss rate under realistic traffic conditions is a critical design issue for any switch, and may represent a serious limitation on the switch size that can be realized, or on the traffic load that can be sustained by the switch. Accordingly, there is a strong motivation to investigate architectural solutions that can reduce the local buffer requirements, as well as the aggregate buffer requirements (i.e., the sum of the buffer requirements in all buffer components) in the switches.

In switches achieving output buffering, each output buffer must be sized to accommodate all bursts destined to that output that may simultaneously arrive at all inputs; as a result, large buffers at each output are necessary. At a given time, however, while some output buffers may experience congestion, others may be nearly empty; consequently, a large portion of the buffer capacity is not used, and the average buffer occupancy of the buffers is low. In contrast, in switches with input and output buffers (as in the case with the MSM), by appropriately regulating the flow of packets from the input queues to the output queues, it may be possible to reduce congestion in the output buffers by having part of the bursts wait at their respective input buffers instead of letting them "collide" at the outputs. In this way, we can more evenly divide the bursts in different buffers and balance the buffer occupancy, thus making better use of the available buffer capacity; as a result, we expect to be able to use similar buffer sizes at the inputs and at the outputs, and reduce both local and aggregate buffer requirements. The switch operates as follows. Packets in the input queues are served according to a certain order, and contend for their requested output buffers. If an output buffer cannot accept more packets, it applies ***backpressure*** to the incoming flow of packets, so that packets are held in their respective input buffers until congestion in the output buffer is relieved.

Existing studies have analyzed the effect of backprcssure in switches with input and output queueing under uniform and Poisson traffic conditions, for a wide variety of assumptions, including fixed or exponentially distributed packet sizes, slotted or unslotted systems, infinite or finite input and output queues, and limited or unlimited switching capacity [Ili90a,Ili90b,Ili9 1 ,Ili92,Gup9 1 ,Bru90]. These studies, however, since they have been limited to Poisson or uniform traffic conditions, have not identified any distinctive improvements in terms of buffer requirements as a result of the use of backpressure from output to input queues. This is due to the fact that, under these traffic conditions, even without backprcssure, buffers are rather equally occupied; as a result, the required buffer sizes are in general rather small, and there are no buffers that are experiencing heavy congestion while others may be relatively free (i.e., the variance in buffer occupancy is low at any time). By using backpressure, output queueing can be traded for input queueing, but given the fact that buffers arc already evenly utilized in the absence of backpressure, the total buffer requirements are not reduced. The relevant traffic scenario, ***i.e.,*** bursty traffic, in which the use of backpressure may be very advantageous (since the buffer occupancy without backpressure is markedly uneven), has not been considered in literature so far. Indeed, the focus of this section is to study the effect of backpressure in a switch with input and output queues, such as the MSM, under bursty traffic.

To use backpressure to control the flow of packets, and thus control the input and output buffer sizes, we need to define: i) the ***conditions*** under which packets should or should not be accepted in the output buffers, ii) the ***buffer-management strategy*** of the input queues in order to decide in which order packets have to be served, and iii) the ***arbitration policy*** to select which packets should be accepted in case that an output buffer has capacity only for a subset of the packets that contend for it.For backpressure to be a viable and attractive solution at large switch sizes, it is essential that the conditions according to which backpressure is applied and relinquished, the strategy according to which packets in the input buffers are served, and the arbitration policy do not depend on global knowledge of buffer occupancy throughout the switch, but rather can be implemented in a distributed fashion.

Given the presence of input controllers with shared buffers and given the simple single-stage banyan-based structure of its routing fabric, the Memory/Space/Memory switching fabric constitutes a convenient structure in which to implement backpressure. As discussed in detail in the previous section, the MSM is essentially an output-buffer type switch; the input buffers are in general rather small, and are necessary to distribute the traffic to the banyan networks and cope with internal blocking in the banyans; it is only the output buffer that increases linearly with the burst size. A suitable condition to apply backpressure to the flow of packets is the current available storage capacity in the output buffers: a packet is accepted in the output buffers only if there exists storage capacity to accommodate it; if no buffer space is available, the packet is held in its corresponding input buffer. The arbitration policy is intrinsically performed by the sequential operation of the banyan networks. The input controllers must then implement a buffer-management strategy to determine in which or&r packets are served, and to establish when a packet that has been blocked by the requested output buffer being full should be dispatched again.

In general, the flow of packets from input to output queues is limited by the available switching capacity and by backpressure. Provided that sufficient switching capacity is available, only the limitation due to backpressure is in effect. As the size of the output buffers is reduced, more and more packets have to be held in the input buffers because of backpressure; thus, the size of the input buffers must increase. If the input queues arc served with a FCFS discipline, head-of-the-line (HOL) blocking at the input queues also increases as the output buffer size decreases; in fact, packets that are blocked at the inputs by a full output buffer, prevent packets queued behind them that may be destined to output buffers with available buffer capacity to reach those buffers. (It should be noted that, as long as the output buffers are sufficiently large, HOL blocking does not limit the maximum throughput of the switch, but requires to increase the size of the input buffers.) Due to HOL blocking, as the size of the output buffers is decreased, the size of the input buffers increases more rapidly. Under bursty traffic, backpressure-induced HOL blocking causes congestion in the input buffers; as a result, both large output buffers, to reduce HOL blocking, and large input buffers, to accommodate the blocked packets, must be used, making the total buffer size even larger than in the case without backpressure.

In order to take advantage of backpressure, therefore, HOL blocking must be overcome. This is achieved by implementing a service discipline of the input queues other than FCFS. One possibility is to simply bypass the HOL packet once it is blocked to serve a following packet in the queue. Under bursty traffic, however, if a packet is blocked, the likelihood that the following packet is also destined to the same output buffer is very high; thus, simple HOL bypassing does not overcome HOL blocking. In order to solve the problem, a different buffer-management strategy of the input buffers is necessary. Specifically, packets can be organized according to the output buffer to which they are destined, and served accordingly. Once a packet is blocked, all packets destined to the same output buffer are known to be also blocked, and packets destined to other buffers are served. By serving the queue of packets destined to each output buffer with FCFS discipline, packet sequence for each virtual circuit is maintained.

With such input-buffer management strategies, the expected benefits of backpressure are achieved; namely, we show that we can achieve important reductions in total buffer requirements (e.g., more than 70% reduction under bursty traffic with bursts of average length equal to 100 packets), equal size of the input and output buffers, and sublinear increase of the buffer requirements with the burst size. These buffer-management strategies can operate in a distributed fashion since they are only based on local knowledge available at each input buffer, and are therefore implementable regardless of the switch size.

Although we study the effect of backpressure under bursty traffic in the specific context of the MSM, we note that these results are in general valid for any switch with input and output queueing and backpressure.

This section is organized as follows. First, we describe how backpressure can be implemented in the MSM switching fabric. Then, we describe several possible buffer-management strategies for the input buffers that operate in a distributed fashion, and study their performance under bursty traffic with different average burst lengths.

4.1. Backpressure in the Memory/Space/Memory Switching Fabric

4.1.1. Architecture

As mentioned above, given its architecture comprising input buffers and a single-stage banyan-based routing network, the MSM switching fabric has a convenient structure to implement backpressure. A suitable limiting factor for the packet flow is the available storage capacity in the output buffers in a given slot. A packet is accepted in an output buffer only if there is sufficient storage capability; otherwise backpressure is applied and the packet is held in its corresponding input buffer. Consequently, no loss occurs in the output buffers. This backpressure mechanism is rather easily implemented in the architecture by a simple modification of the feedback mechanism already in place in the banyans.

With backpressure, the switch operation in every slot works as follows. During the set-up phase, a header is successful in a banyan only if:

i) it is properly routed by the banyan, and

ii) the requested output buffer has storage capacity available for the corresponding packet.

A feedback signal that distinguishes the case of a header that has lost contention in the banyan from the case of a header that has been properly routed but has been blocked by a full output buffer is sent back to the input controllers. The feedback signal should therefore consist of two bits; namely, the activity bit ***a*** of an header after it has been routed through the banyan ***(a =*** 1, for a properly routed header, ***a*** = 0, for a misrouted header) and the status s of the corresponding output buffer (s = 1, for an available buffer, s = 0, for a full buffer). From an implementation point of view, this requires a simple modification of the banyan network functionality. Since the feedback signal is generated by a switching element in the last stage of a banyan, information on the occupancy of the corresponding output buffer must be provided to that switching element. We note that, since with sequential dispatching the set-up phases in the banyans are staggered in time, the status of the available capacity of each output buffer can be updated after each route set-up phase (taking into account the packet that may have been routed by that banyan), and the information made available to the corresponding switching element in the last stage of the next banyan in the sequence, so that a proper feedback signal is generated. Then, the input controllers, after receiving the feedback signal, dispatch a header that has been misrouted to the following banyans in the sequence, while they hold for the rest of the slot a header that has been blocked by a full output buffer, and avoid unnecessary contention in the remaining banyans in the sequence.

It is evident that the selection of which packets are accepted by an output buffer that can only accommodate a subset of the packets that are destined to it in a slot ***(i.e., the*** arbitration policy) is intrinsically performed by the usual operation of the banyan networks in parallel. (Given the sequential operation of the banyans, there is no ambiguity of which packets are selected.) With the simple operation of the banyans as described in the previous section, the selection is random, but other policies could be realized by modifying the switching algorithm performed in the switching elements. (For example, we could introduce priorities for packets that have spent a certain amount of time in the input queues, so that they can prevail when contending with “younger” packets in the banyans.)

4.1.2. Buffer Management Strategies

As mentioned above, the buffer-management strategy used in the input buffers plays an essential role in handling HOL blocking induced by backpressure, and in making it possible to achieve the desired benefits in terms of buffer requirements. We have investigated different buffer-management strategies for the input queues which use only the local knowledge available in each input component of the MSM.

Each input component has only information regarding packets arriving at its own M input lines. If packets are just multiplexed in the shared buffer in the order in which they arrive and served independently from one another ***(i.e.,*** they are not organized in any way, such as according to their virtual circuit or their destination), then the only information that can be used in order to decide whether to further dispatch or hold a packet is the information obtained from the feedback signal after that packet has been dispatched to a banyan. On the other hand, if packets are distinguished by the virtual circuit to which they belong, and the input queues are organized accordingly, then the feedback information obtained for a packet applies to all the packets of its connection during a given time slot. In general, it is sufficient to distinguish packets by the output buffer to which they are destined, thus grouping together packets belonging to all connections with the same output buffer destination. The clear advantage is that as soon as a packet is blocked by the requested output buffer being full, all packets destined to that output buffer are known to be blocked, and do not need to be served. From these considerations, we group the possible buffer-management strategies in two categories:

i) ***single-queue*** buffer-management strategies which consider packets as being independent from one another, and

ii) ***destination-by-destination*** buffer-management strategies, which distinguish packets by the output buffer to which they arc destined, and serve the input queues accordingly.

Single-queue strategies do not need any modification in the organization of the input queues with respect to the switch without backpressure. We have considered two such strategies; namely, ***FCFS*** and ***HOL bypassing. In the*** following, we show that, with both these strategies, HOL blocking at the input queues is quite substantial under bursty traffic, so that the benefits resulting from the use of backpressure are limited.

Destination-by-destination strategies require a proper organization of the input queues, resulting in significant modifications with respect to the switch without backpressure. However, they are very advantageous under bursty traffic, because they avoid serving packets that are doomed to be blocked by a buffer which is known to be full, and use the routing capacity for packets that may be destined to other output buffers with available capacity, thus overcoming HOL blocking. In the following, we consider two such strategies, referred to ***as Exhaustive Until Blocked*** and ***One Per Destination,*** and show that with them we can achieve the large reductions in buffer requirements expected from the use of backpressure.

We have studied the performance of the MSM switching fabric with backpressure using these buffer-management strategies for the input buffers, under bursty traffic conditions. For simplicity, in this discussion, we only consider MSM switches with G = 1 ***(i.e.,*** $M = R$***).*** Similar results could be obtained for G **>** 1. We also assume that the banyan networks are augmented with randomizers by pairs. (In the previous section, we have shown that the addition of the randomizers by pairs makes the performance of the switch more robust under correlated traffic patterns, and brings negligible additional complexity in the design.) It should be noted that the results reported here on buffer requirements in the switch with backpressure do not pertain exclusively to the MSM switching fabric, but apply to any switch using input and output queueing and backpressure.

In this study, we have used the same bursty traffic model considered in the previous section in the characterization of the MSM without backpressure. To serve as a reference, we recall that , for example, in a 32 x 32 MSM without backpressure, $M = 1$, $N = 32$, with banyans augmented by randomizers by pairs, under bursty traffic at 0.9 load, $L = 10$, the following buffer sizes are required to guarantee a packet loss rate below 10^{-6} in the whole switch (this assumes that we size the buffers in order to achieve equal packet loss rate in the input and in the output queues):

Number of Banyans	B_i	B_o	Total Buffer Size per Port
3	50	980	1030
4	20	980	1000
5	10	980	990

With $L = 100$, the required buffer sizes :o meet the same loss rate in the switch are:

B_i ımber of Banyans		B_o	Total Buffer Size per Port
350 3		9400	9750
4	135	9400	9535
5	66	9400	9466
6	26	9400	9426

For M > 1, B_I and B_o denote the buffer size in each input and output component, respectively (*i.e.*, per M ports). For example, we recall that in a 128 x 128 MSM switch, A.4 = 4, $N = 32$, the buffer sizes per component (i.e., per 4 ports) necessary to achieve 10^{-6}

packet loss rate in the switch, with $\boldsymbol{L} = 10$, at 0.9 load, are as follows (again, this assumes that buffers are sized so as to guarantee equal loss rate at the input and at the output):

Number of Banyans	$\boldsymbol{B_i}$	$\boldsymbol{B_o}$	Total Buffer Size per 4 Ports
12	108	1600	1708
13	56	1600	1656
14	40	1600	1640

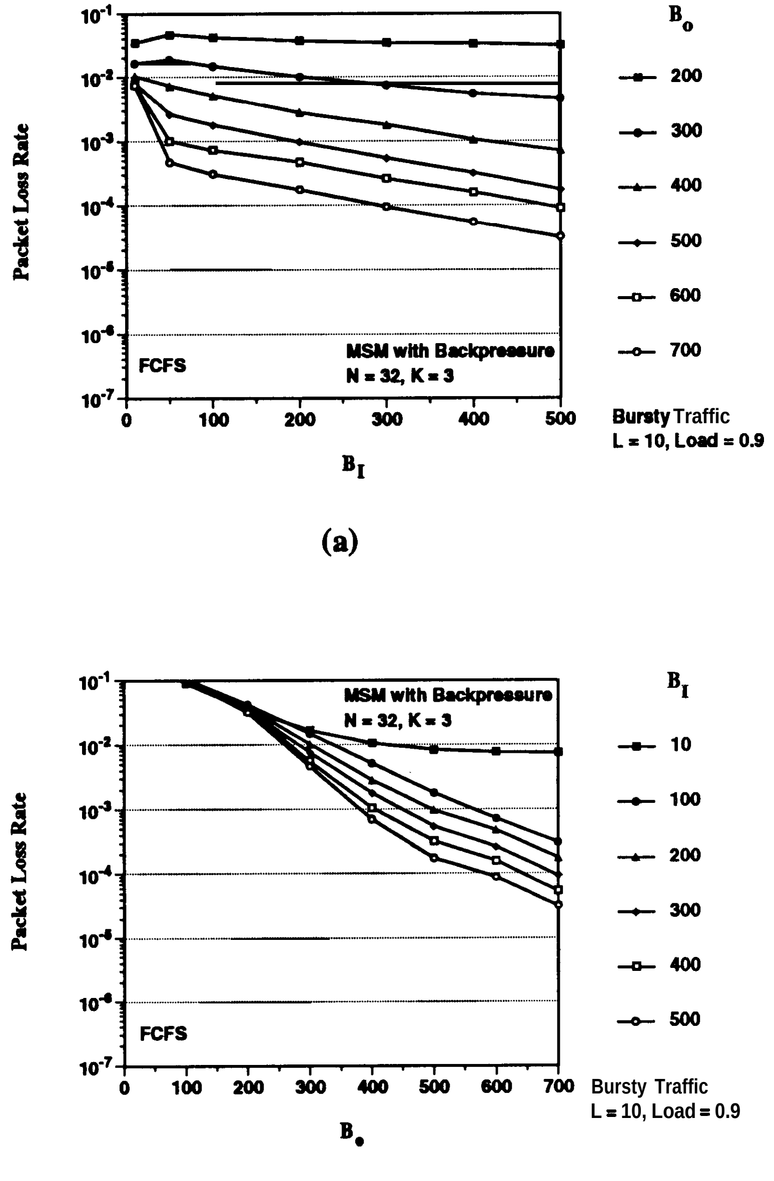
With $\boldsymbol{L} = 100$, the required buffer sizes per component are:

Number of Banyans	$\boldsymbol{B_i}$	$\boldsymbol{B_o}$	Total Buffer Size per 4 Ports
12	480	17200	17680
13	280	17200	17480
14	136	17200	17336

a) Single-Queue Buffer Management: FCFS

With FCFS, an input controller, after dispatching an header, if it receives a negative feedback signal specifying that the requested output buffer is full, holds the header in the corresponding input buffer, and does not dispatch any other header to the remaining banyans in the sequence during that slot.

With backpressure, no packet loss rate occurs at the output buffers, as mentioned above. The packet loss rate in a 32 × 32 MSM switch with backpressure, $\boldsymbol{M} = 1$, $\boldsymbol{N} = 32$, $\boldsymbol{K} = 3$ banyans, with input buffers served with FCFS discipline, under bursty traffic with average burst length $\boldsymbol{L} = 10$ packets, at 0.9 load, is shown in Fig. 4.3, as a function of the input and output buffer size per port. Under these traffic conditions, HOL blocking at the input queues caused by packets that cannot proceed to the output buffers because of no available storage capacity is substantial. This is evidenced by the fairly flat curves of the



(a)

(b)

Fig. 4.3. Packet loss rate in the MSM switching fabric with backpressure, with FCFS input buffers, M = 1, N = 32, K = 3, under bursty traffic conditions with average burst size L = 10, load = 0.9; a) packet loss rate vs. the input buffer size, for different output buffer sizes; b) packet loss rate vs. the output buffer size, for different input buffer sizes (obtained by simulation).

packet loss rate as a function of the input buffer size, for a fixed output buffer size, shown in Fig. 4.3.a. Due to HOL blocking, large buffers at both inputs and outputs are necessary to achieve low packet loss rates. For example, to achieve a packet loss rate below 10^{-6}, possible choices of the buffer size per port at the input and at the output are:

B_i	B_o	Total Buffer Size per Port
400	900	1300
700	700	1400
950	600	1550

As the output buffer size B_O decreases, the input buffer size B_I has of course to increase to accommodate the larger volume of packets experiencing backpressure. The increase in B_I, however, is more rapid than the decrease in B_O, so that the total required buffer size increases for decreasing *B*,. In general, the total buffer requirements are ***larger*** than those in the switch without backpressure; this is true even for large output buffer sizes approaching the values necessary without backpressure. This is due to HOL blocking caused by those packets that without backpressure would have been discarded upon arrival at a full buffer and that are instead stored at the inputs. As long as the number of banyans in parallel is adequate to provide sufficient capacity to sustain the offered traffic load, backpressure is the only limiting factor on the packet flow from the input to the output buffers. Therefore, as shown in Fig. 4.4, the buffer requirements basically do not depend on K. It is interesting to note that this holds even for large buffer sizes, when backpressure is presumably applied relatively seldom. Only for small input buffer sizes that are not sufficient to sustain the offered traffic with a given K, the available routing capacity limits the throughput of the switch, and the packet loss rate rapidly saturates at the value achieved without backpressure; this is evident, for example, in the plot of the packet loss rate for $B_I = 10$, shown in Fig. 4.3.b for $K = 3$. The same effect is noticeable in Fig. 4.4, where the packet loss rate for $B_I = 10$ shows a marked dependency on K. (A

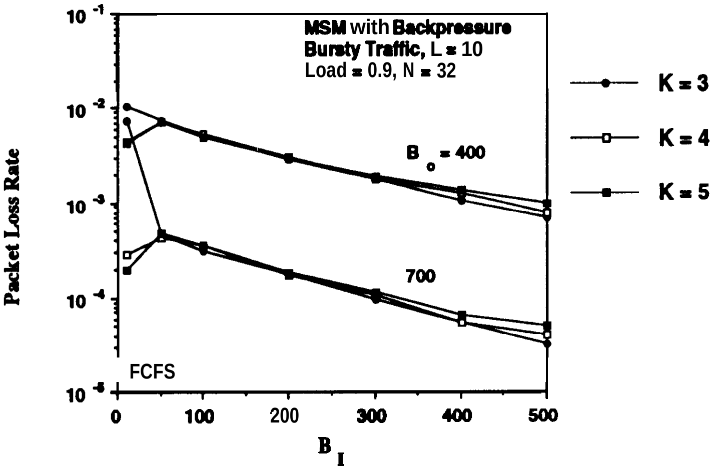


Fig. 4.4. Packet loss rate vs. the input buffer size in the MSM switching fabric with backpressure, with FCFS input buffers, M = 1, N = 32, for different K, under bursty traffic conditions with average burst size L = 10, load = 0.9, for different output buffer sizes (obtained by simulation).

second order, yet quite intriguing effect observed in Fig. 4.4 is that the curves of the packet loss rate become even slightly worse for increasing values of K, when backpressure is the limiting factor on the packet flow; a possible explanation for this marginal effect is that the increased volume of packets that reaches the output buffers with larger K presumably allows a few more packets coming from moderately busy input queues to use the available capacity in the output buffers, preempting packets from input queues that are almost full from being stored in those buffers.)

As the burst size increases, the increase in the required buffer size is ***sublinear,*** as shown in Fig. 4.5, where we plot the packet loss rate in the switch with K = 3 versus the output buffer size normalized to the burst length, for different input buffer sizes also normalized to the burst length, for L = 10 and L =100. In fact, for large burst sizes, the benefit of keeping part of the bursts at the inputs instead of letting them congest the output buffers becomes more and more noticeable. The packet loss rate in the switch with K = 3

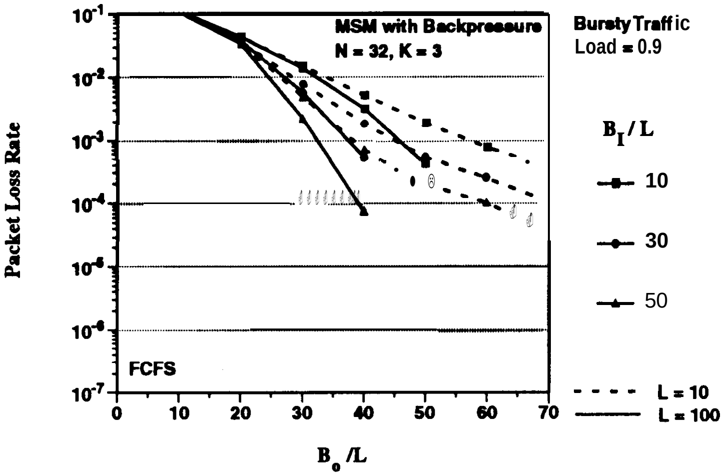


Fig. 4.5. Packet loss rate vs. the output buffer size normalized to the burst length in the MSM switching fabric with backpressure, with FCFS input buffers, $M = 1$, $N = 32$, $K = 3$, under bursty traffic conditions with average burst size $L = 10$ and $L = 100$, load = 0.9, for different input buffer sizes normalized to the burst length (obtained by simulation).

banyans under bursty traffic with $L = 100$ packets, at 0.9 load, is shown in Fig. 4.6, as a function of the input and output buffer size per port. For example, possible choices of the buffer sizes per port for a 10” packet loss rate are:

B_i	B_o	Total Buffer Size per Port
9500	4000	13500
5500	5000	10500
700	6000	6700
700	7000	7700
700	8000	8700

We observe an interesting behavior of the packet loss rate. Indeed, large output buffers are necessary to reduced HOL blocking at the input induced by backpressure; however, for increasing values of the output buffer size, the packet loss rate for a given

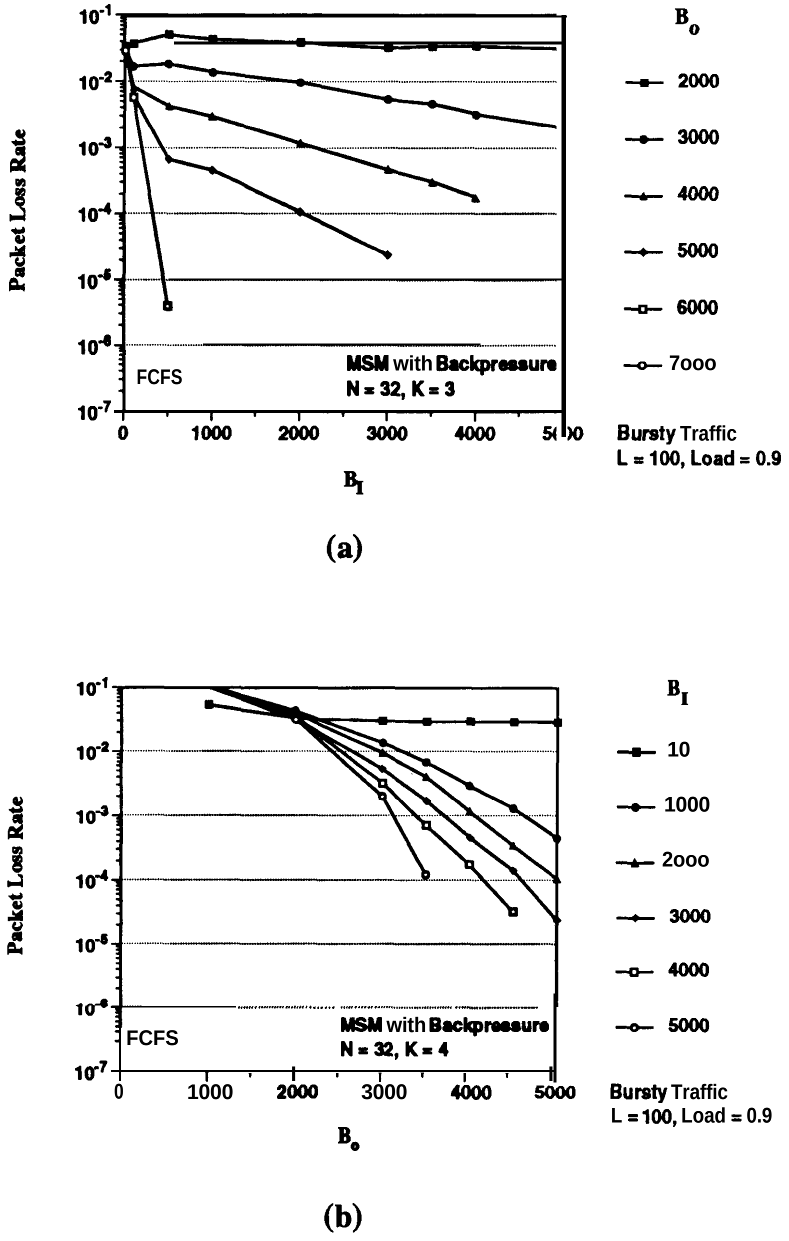


Fig. 4.6. Packet loss rate in the MSM switching fabric with backpressure, with FCFS input buffers, $\boldsymbol{M}$ = 1, $\boldsymbol{N}$ = 32, $\boldsymbol{K}$ = 3, under bursty traffic conditions with average burst size $\boldsymbol{L}$ = 100, load = 0.9; a) packet loss rate vs. the input buffer size, for different output buffer sizes; b) packet loss rate vs. the output buffer size, for different input buffer sizes (obtained by simulation).

input buffer size improves more and more rapidly; this is evidenced for example by the diverging curves of the packet loss rate for different B_I as a function of B_O in Fig. 4.6.b, in contrast with the almost parallel corresponding curves reported in Fig. 4.3.b for L = 10. Eventually, a ***threshold*** output buffer size is reached, for which the packet loss rate for a given input buffer size improves abruptly and becomes ***independent*** of further increases in **B,,** as clearly shown in Fig. 4.6.a, where the curves of the packet loss rate for B_O = ***6000*** buffers and B_O = 7000 buffers overlap. With output buffer sizes equal to the threshold, low packet loss rates are achieved with relatively small input buffer sizes. For example, with B_O = 6000,700 input buffers are necessary to achieve a packet loss rate below 10^{-6}. Quite surprisingly, the value of the output buffer size at this threshold is significantly ***smaller*** than the output buffer size necessary to achieve comparable packet loss rates without backpressure ***(i.e.,*** 9400 packet buffers per port); the size of the input buffers is however only moderately larger than that needed without backpressure, yielding a noticeable reduction in total buffer size.

In order to further characterize the behavior of the packet loss rate and the threshold effect, in Fig. 4.7 we show the packet loss rate in the switch as a function of the input buffer size, for different K. We make the following observations:

i) For output buffer sizes much smaller than the threshold output buffer size B_O = 6000 (such as, for example, B_O = 2000 and B_O = 4000), the packet loss rate essentially does not depend on K, thus indicating that backpressure is indeed the only limiting factor on packet flow. For these buffer sizes, because of HOL blocking, the total buffer requirements are significantly larger than those without backpressure.

ii) For larger sizes of the output buffers (closer, but still below the threshold), such as B_O = 5000 in Fig. 4.7, the packet loss rate does not depend on K only for relatively small input buffer sizes (up to B_I = ***2000);*** then, for larger values of B_I, it

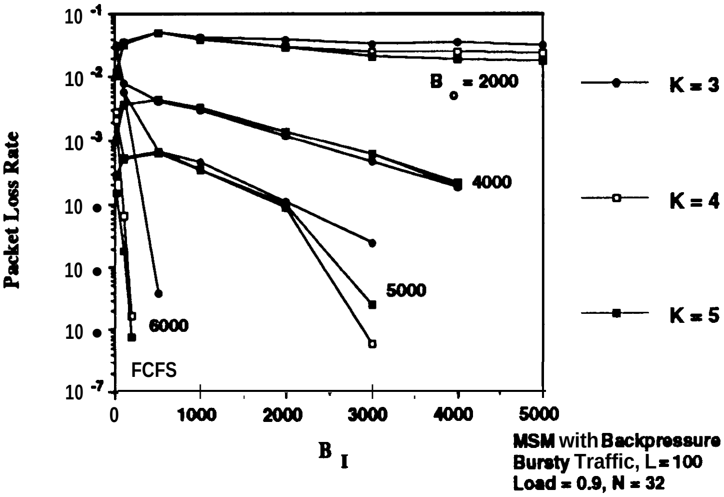


Fig. 4.7. Packet loss rate vs. the input buffer size in the MSM switching fabric with backpressure, with FCFS input buffers, $\boldsymbol{M}$ = 1, $\boldsymbol{N}$ = 32, for different $\boldsymbol{K}$, under bursty traffic conditions with average burst size $\boldsymbol{L}$ = 100, load = 0.9, for different output buffer sizes (obtained by simulation).

shows a marked dependency on $\boldsymbol{K}$. This suggests that, as the input buffer size increases, there is a sharp transition from backpressure being the limiting factor on the packet flow to routing capacity being the limiting factor. Still, even for these buffer sizes, the total buffer requirements are larger than those without backpressure for $\boldsymbol{K}$ = 3 and $\boldsymbol{K}$ = 4, and only slightly smaller for $\boldsymbol{K}$ = 5.

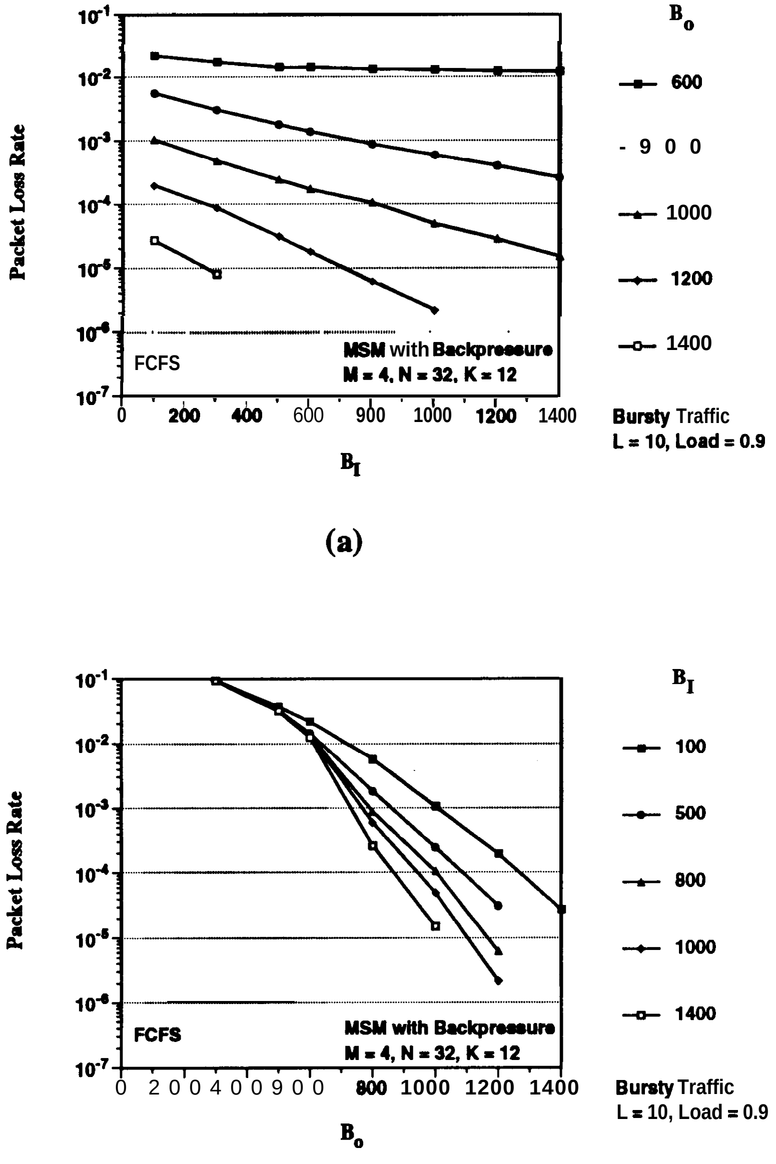
iii) For output buffer sizes at or near the threshold, the required input buffer size to achieve the threshold decreases with $\boldsymbol{K}$. For example, with $\boldsymbol{B_O = 6000}$, the required input buffer size to meet a packet loss rate below 10^{-6} is 700 with K = 3, and reduces to fewer than 250 buffers with $\boldsymbol{K}$ = 4. This marked dependency on $\boldsymbol{K}$ of the input buffer size indicates that backpressure is no longer the only limiting factor on packet flow; on the contrary, the major limitation is the available routing capacity in the switch. The fact that the output buffer size at the threshold is significantly smaller than the output buffer size required without backpressure suggests that, although backpressure is no longer the main limitation on the packet flow and HOL blocking at the input is not important, enough backpressure is still applied to significantly affect the tail of the distribution of the output queue lengths, and visibly reduce the output buffer size needed.

It is important *to* note that *the threshold* in the output buffer size exists regardless of the buffer-management strategy used, and the values of the buffer sizes at the threshold depend only moderately on the specific algorithm, as shown in the remaining of this discussion. This gives additional support to the fact that, as the threshold is reached, HOL blocking has no impact on performance and the limitation becomes the routing capacity of the switch.

We have also studied MSM switches with $M > 1$, with backpressure and FCFS input buffers. For example, in Fig. 4.8 we show the packet loss rate in a 128 × 128 switch, $M = 4$, $N = 32$, $K = 12$ banyans, as a function of the buffer size per input component B_I and the buffer size per output component B_o, under bursty traffic with $L = 10$, for $K = 12$ and for different K, respectively. Possible choices of the buffer sizes per component for a 10^{-6} packet loss rate are:

B_i	B_o	Total Buffer Size per 4 Ports
100	1700	1800
600	1400	2ooo
1200	1200	2400

Again, the total required buffer size with this burst size is larger than in the switch without backpressure. Actually, for $M > 1$, HOL blocking at the input queues becomes even more important, since a packet blocked by a full output buffer blocks all packets from M inputs.



(a)

(b)

Fig. 4.8. Packet loss rate in a 128 × 128 MSM switching fabric with backpressure, with FCFS input buffers, M = 4, N = 32, K = 3, under bursty traffic conditions with average burst size L = 10, load = 0.9; a) packet loss rate vs. the input buffer size, for different output buffer sizes; b) packet loss rate vs. the output buffer size, for different input buffer sizes (obtained by simulation).

Finally, we should note that, since backpressure rather than switching capacity is the limiting factor on the packet flow from input to output buffers, FCFS in the MSM with backpressure can be implemented in a simpler way than the one described above, without noticeable effect on performance. Specifically, the input controllers, rather than holding a header that has been blocked by a full output buffer, can dispatch it to the following banyans in ***the*** sequence ***regardless*** of whether in the previous network it was misrouted or blocked at the output. In this way, unnecessary contention in the banyans occurs and some switching capacity is wasted, but this does not have observable impact on switch performance, since switching capacity is not the limiting factor on the packet flow (at the threshold, where the switching capacity comes into play, the effect of additional contention is so small that is also not noticeable). With this simplification, no modification of the input controllers is necessary with respect to the switch without backpressure; in addition, the feedback signal ***f*** is still a single bit, as in the switch without backpressure (however, the feedback signal should now also take into account the status of the corresponding output buffer s; therefore ***f*** = (a and s), where ***a*** is the activity bit of the header after it has been routed by the banyan).

b) Single-Queue Buffer Management: HOL Bypassing

In order to overcome the HOL blocking effect caused by backpressure, one possibility may be to bypass the HOL packet once it is blocked by a full output buffer to serve the following packets in the queue. With HOL bypassing, an input controller holds an header that has been blocked by a full buffer, but dispatches the header of the next packet in the input queue to the next banyan. In order to maintain the sequence of the packets, in each slot, packets are served starting from the top of the input queue.

HOL bypassing, requires an important modification of the input memory with respect to FCFS. In fact, with FCFS, the input shared buffer is a sequentially-accessed

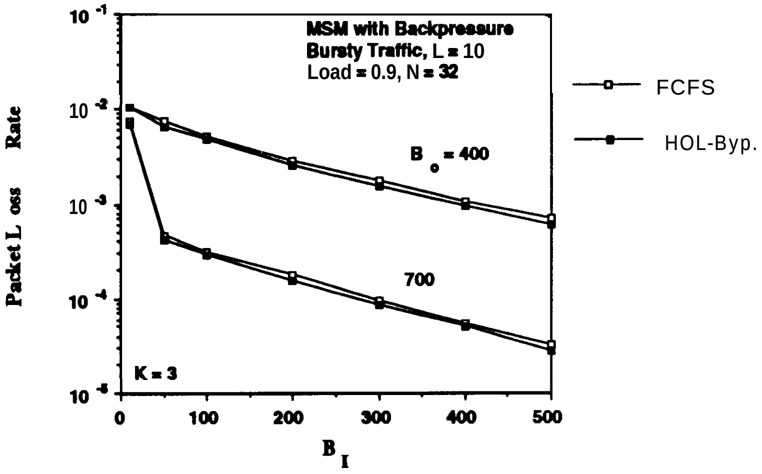


Fig. 4.9. Packet loss rate vs. the input buffer size in the MSM switching fabric with backpressure, with Algorithms FCFS and HOL bypassing, $\boldsymbol{M}$ = 1, $\boldsymbol{N}$ = 32, $\boldsymbol{K}$ = 3, under bursty traffic conditions with average burst size $\boldsymbol{L}$ = 10, load = 0.9, for different output buffer sizes (obtained by simulation).

memory (for G = 1, and for G > 1 with partitioned buffers for each group). On the contrary, with HOL bypassing, the shared-buffer becomes a randomly-accessed memory, and a linked list of the packets must be managed to retain their order in the queue. From an implementation point of view, the additional complexity involved is rather significant.

We have studied the performance of the MSM switch with backpressure with HOL bypassing in the input queues, and shown that the buffer requirements are essentially the same as those of the switch with FCFS input buffers. For example, the packet loss rate as a function of the input buffer size, for constant output buffer sizes, in the 32 x 32 MSM, $\boldsymbol{K}$ =3, with FCFS and HOL bypassing are compared in Fig. 4.9 and Fig. 4.10, for $\boldsymbol{L}$ = 10 and $\boldsymbol{L}$ = 100, respectively. The reason why the simple bypassing a packet in the input queue that is blocked by a full output buffer to serve the next packet in the queue (*i.e.*, without attempting to know whether the following packet is destined to a different output buffer), has negligible impact on switch performance is that, under bursty traffic

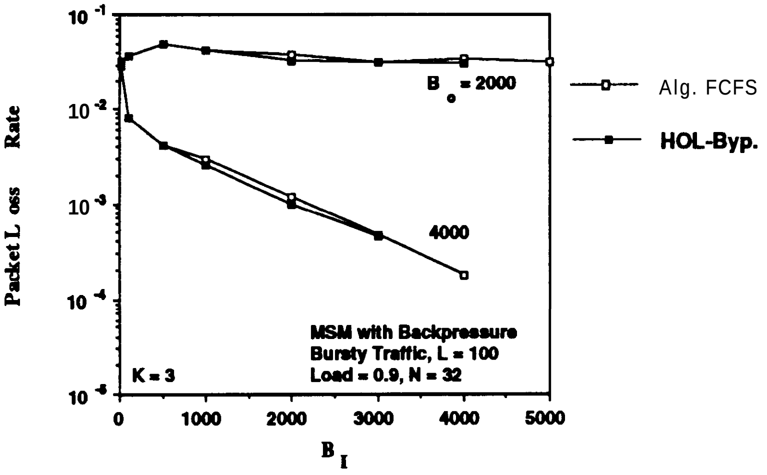


Fig. 4.10. Packet loss rate vs. the input buffer size in the MSM switching fabric with backpressure, with Algorithms FCFS and HOL bypassing, $\boldsymbol{M}$ = 1, $\boldsymbol{N}$ = 32, $\boldsymbol{K}$ = 3, under bursty traffic conditions with average burst size $\boldsymbol{L}$ = 100, load = 0.9, for different output buffer sizes (obtained by simulation).

conditions, successive packets in the queue are very likely to be destined to the same output buffer. Only with more "intelligent" bypassing, as it is possible with destination-by-destination strategies described below, can HOL blocking be overcome.

In summary, from this discussion, we conclude that, under single-queue buffer-management strategies, for small burst sizes, the switch with backpressure does not achieve any performance advantage over the switch without backpressure. For large burst sizes, however, the advantage of holding part of the bursts at the inputs instead of letting them collide in the output buffers becomes more visible. In particular, there are values of the output buffer size (or more precisely there is a "threshold" of the output buffer size) for which HOL blocking at the inputs induced by backpressure is negligible, but the control applied by backpressure on the packet flow is still sufficient to substantially reduce congestion at the outputs. With output buffers sized at the threshold value, fairly small input buffer sizes are necessary to achieve low packet loss rates, leading to noticeable reductions in buffer requirements with respect to the switch without backpressure. From an implementation point of view, to achieve such improvement, only a rather simple modification of the MSM switch without backpressure is required. Finally, we have shown that simple HOL bypassing is not sufficient to overcome HOL blocking under bursty traffic, and more "intelligent" bypassing must be used, as described in the sequel, in order to obtain the full benefits that are expected from the use of backpressure.

c) Destination-by-Destination Buffer Management

In order to operate destination-by-destination strategies, a more complex organization of the input buffers than what necessary with single-queue strategies is required. Within the input queues, packets have to be organized according to their output buffer destination. Consequently, as many as N active linked lists must be managed in each input shared buffer, one per each output-buffer destination to which at least one packet in the queue is destined. The required buffer organization, depicted in Fig. 4.11, consists of a queue Q_i^D of active linked lists corresponding to the desired destinations of packets in input buffer i, and a queue Q_i^P of packets waiting to be routed; queue Q_i^D consists of N memory locations; if active, location j ($j = 1, \ldots . N$) in Q_i^D contains the address of the packets in Q_i^P at the head and tail of the linked list of packets destined to output buffer j; the active locations in Q_i^P are linked together to form the queue. In Q_i^P, the chains of memory addresses to form the linked lists are implemented by storing, together with each packet, a pointer to the next packet in the linked list. Packets in Q_i^P are managed on a destination-by-destination basis: when a packet is blocked by a full output buffer, all packets in its linked list are known to be blocked for that slot; hence, packets belonging to a different link list are served. By serving packets in each linked list using FCFS discipline, sequence is maintained for each virtual circuit (accordingly, given their FCFS operation, the linked lists are implemented with single pointers only).

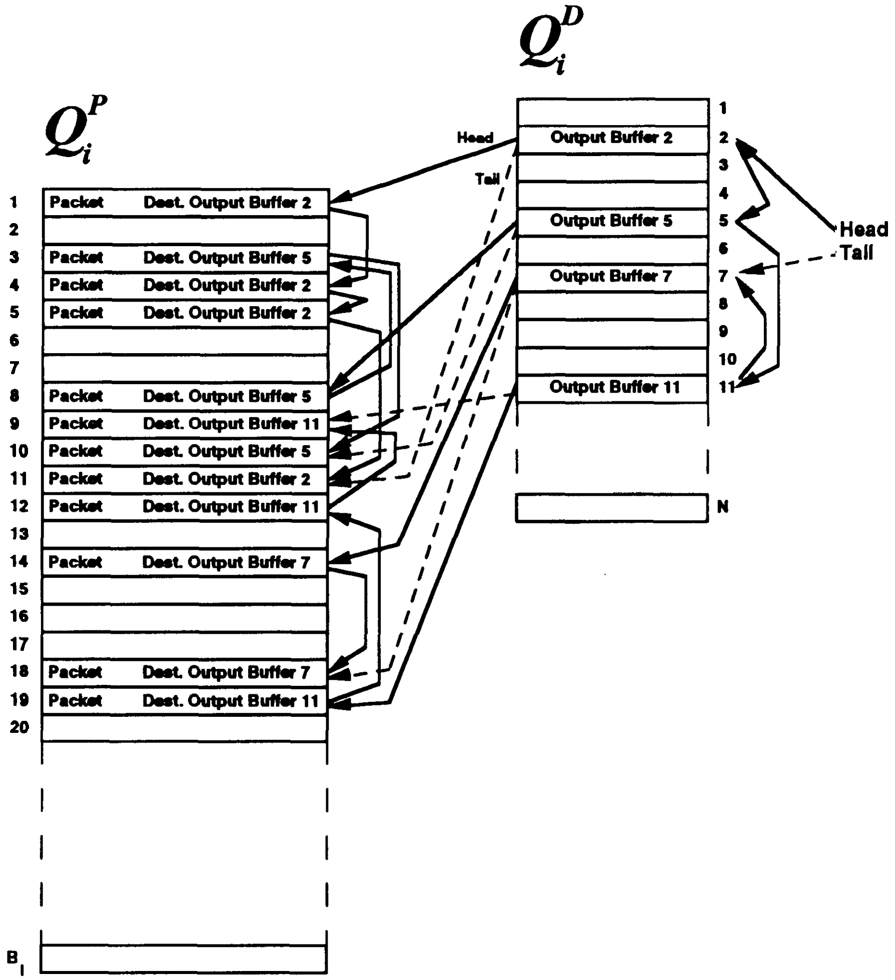

Q_i^D
Q_i^P
1
2 Output Buffer 2
3
4
5 Output Buffer 5
6
7 Output Buffer 7
8
9
10
11 Output Buffer 11
N
Head
Tail
Head
Tail
1 Packet Dest. Output Buffer 2
2
3 Packet Dest. Output Buffer 5
4 Packet Dest. Output Buffer 2
5 Packet Dest. Output Buffer 2
6
7
8 Packet Dest. Output Buffer 5
9 Packet Dest. Output Buffer 11
10 Packet Dest. Output Buffer 5
11 Packet Dest. Output Buffer 2
12 Packet Dest. Output Buffer 11
13
14 Packet Dest. Output Buffer 7
15
16
17
18 Packet Dest. Output Buffer 7
19 Packet Dest. Output Buffer 11
20
B_I

Fig. 4.11. Organization of an input shared-buffer to operate a destination-by-destination buffer-management strategy. In the buffer as many as N linked lists are maintained, one per each output buffer.

With this buffer organization, ***switching*** is actually performed in the input buffers to sort packets according to their destination buffers. From a hardware perspective, the implementation and management of the N linked lists increases substantially the complexity of the input buffers. As shown in the sequel, the design trade-off with respect to the switch without backpressure is between the implementation complexity of the memory control and the required buffer size to handle highly-bursty traffic conditions. It is important to note that, while packets in the queues are organized according to their requested output buffer, they are still processed independently from one another during dispatching and routing. Thus, aggregating packets by their output destination is a pure buffer-management issue, not a capacity allocation and routing matter; (this is sharply different from, for example, routing packets on a virtual-circuit-by-virtual-circuit basis in multistage configurations of identical switching modules, which has been described in Section 2.1 above).

We have considered and evaluated two different algorithms for the management of the queues. They differ from each other in the way Q_i^D and each linked list in Q_i^D are served.

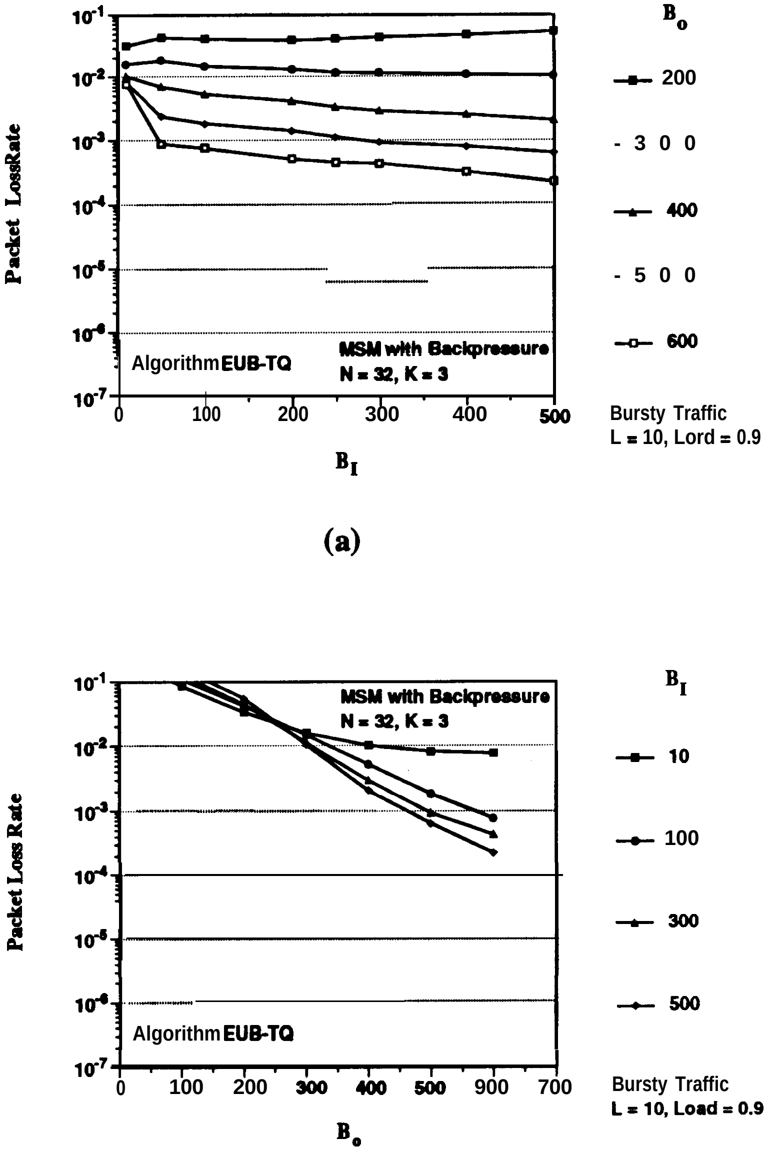
Algorithm 1: Exhaustive Until Blocked (EUB). Packets in a linked list in Q_i^D are served as long as i) there are packets available, and ii) packets are not blocked by the requested output-buffer. We have considered two variations of this algorithm: in ***Exhaustive Until Blocked — Top of the Queue (EUB-TQ),*** in each slot the linked lists are served starting from the ***top*** of Q_i^D ; in ***Exhaustive Until Blocked — Round Robin (EUB-RR),*** in each slot the linked ***lists are*** served in a ***round-robin*** fashion, starting from the list that was being served at the end of the previous slot.

The idea behind the Exhaustive-Until-Blocked algorithm is that, once a packet is successful in a slot, the likelihood that another packet destined to the same output buffer can also be successful may be higher (since the successful packets may have found no contention in its route to the output buffer); consequently, in case of a successful packet, it may be advantageous to keep serving its linked list, rather than moving to another one for which no knowledge is available. Clearly, Exhaustive Until Blocked — Top of the Queue may lead to unfair service of the linked lists. Exhaustive Until Blocked — Round Robin is an obvious way to solve the unfairness problem. An alternative algorithm is one that serves at most a single packet per linked list:

Algorithm 2: One per Destination (OPD). For a linked list in Q_i^D ***, one*** packet is served until it is properly routed by the banyans or until it is blocked by the requested output-buffer; then the following list in Q_i^D is served. We have considered two variations of this algorithm: in ***One per Destination — Top of the Queue (OPD-TQ),*** in each slot the linked lists are served starting from the ***top*** of Q_i^D. In ***One per Destination — Round Robin (OPD-RR),*** in each slot the linked lists are served in a ***round-robin*** fashion.

The One-Per-Destination algorithm is expected to serve the linked lists as evenly as possible; on the other hand, it may also lead to higher contention in the banyans and blocking at the outputs than what experienced by using the Exhaustive Until Blocked algorithm. One Per Destination — Top of the Queue may still induce unfairness in serving the linked lists; One Per Destination — Round Robin is the obvious solution for the problem.

The packet loss rate in a 32 x 32 MSM switch with K = 3 banyans and input buffers operated under the ***Exhaustive Until Blocked — Top of the Queue*** algorithm is plotted in Fig. 4.12 as a function of the input and output buffer size, under bursty traffic with L = 10, at 0.9 load. Because of unfairness (since the lists are always served starting from the top of the queue of active destinations Q_i^D), some packets may experience blocking due to packets belonging to lists that are ahead of theirs in Q_i^D; consequently, the



(a)

(b)

Fig. 4.12. Packet loss rate in the MSM switching fabric with backpressure, with algorithm Exhaustive Until Blocked — Top of the Queue, M = 1, N = 32, K = 3, under bursty traffic conditions with average burst size L = 10, load = 0.9; a) packet loss rate vs. the input buffer size, for different output buffer sizes; b) packet loss rate vs. the output buffer size, for different input buffer sizes (obtained by simulation).

buffer requirements are large. For example, to achieve a packet loss rate below 10^{-6}, possible choices of the buffer size per port at the input and at the output are:

B_i	B_o	Total Buffer Size per Port
100	1350	1450
300	1200	1500
500	1100	1600
700	1050	1750

In general, the buffer requirements with L = 10 are even larger than those of FCFS (and substantially larger than those of the switch without backpressure). Some marginal improvement in buffer requirements is obtained by using more banyans, as shown in Fig. 4.13; in fact, with larger K, the linked lists have more chances of being served, but since K is in any case much smaller than the number of active lists, the effect is negligible. Similar results are obtained by using the ***One Per Destination — Top of the Queue*** algorithm. Although in this case at most one packet per slot is served for any active destination, unfairness is still experienced, given the small values of K.

For larger burst sizes, the performance is very similar to that obtained with FCFS. For example, in Fig. 4.14, we show the packet loss rate versus input and output buffer sizes in the switch with K = 3 banyans, under the One Per Destination — Top of the Queue algorithm, for L = 100 at 0.9 load. Again we note the threshold effect in the output buffer size for B_o = 6000. The packet loss rate shows some dependency on the number of banyans, since with larger K the linked lists have more chances to be served. However, for buffer sizes much smaller than the threshold, the reductions in the required buffer size are marginal, as shown in Fig. 4.15.a, since blocking at the inputs remains the main limiting factor. Near the threshold, as shown in Fig. 4.15.b, the behavior of the curves of the packet loss rate as a function of the input buffer size is rather interesting. For example,

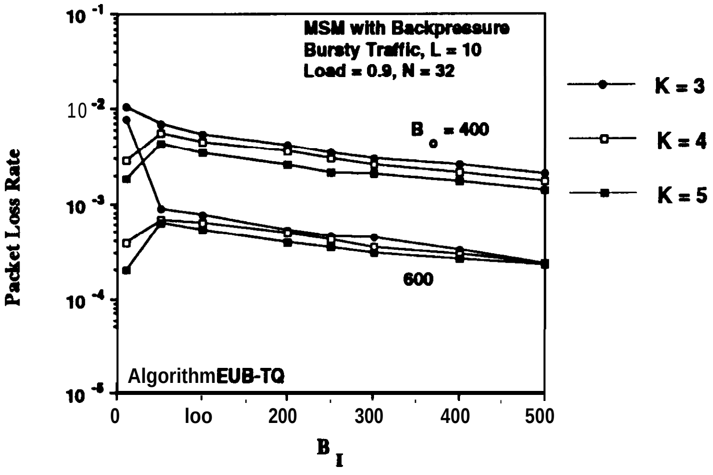


Fig. 4.13. Packet loss rate vs. the input buffer size in the MSM switching fabric with backpressure, with algorithm Exhaustive Until Blocked — Top of the Queue, M = 1, N = 32, for different K, under bursty traffic conditions with average burst size L = 10, load = 0.9, for different output buffer sizes (obtained by simulation).

considering the curve for B_O = 5000, for K = 4 and K = 5, we note the following: for small input buffer sizes (< 100 packet buffers), packets are discarded at the input, so that no blocking occurs, and the limitation on the packet flow is the available routing capacity; thus, the curves of the packet loss rate are very close to those at the threshold; then, for increasing values of the input buffer size, there is a region in which blocking comes into play, and the packet loss rate has even a positive slope; eventually, for further increase of B_I, the queues are large enough to handle blocking, and the packet loss rate improves rapidly. At the threshold, the buffer requirements are the same as those with FCFS for any K, since switching capacity rather than backpressure is the limiting factor on the packet flow.

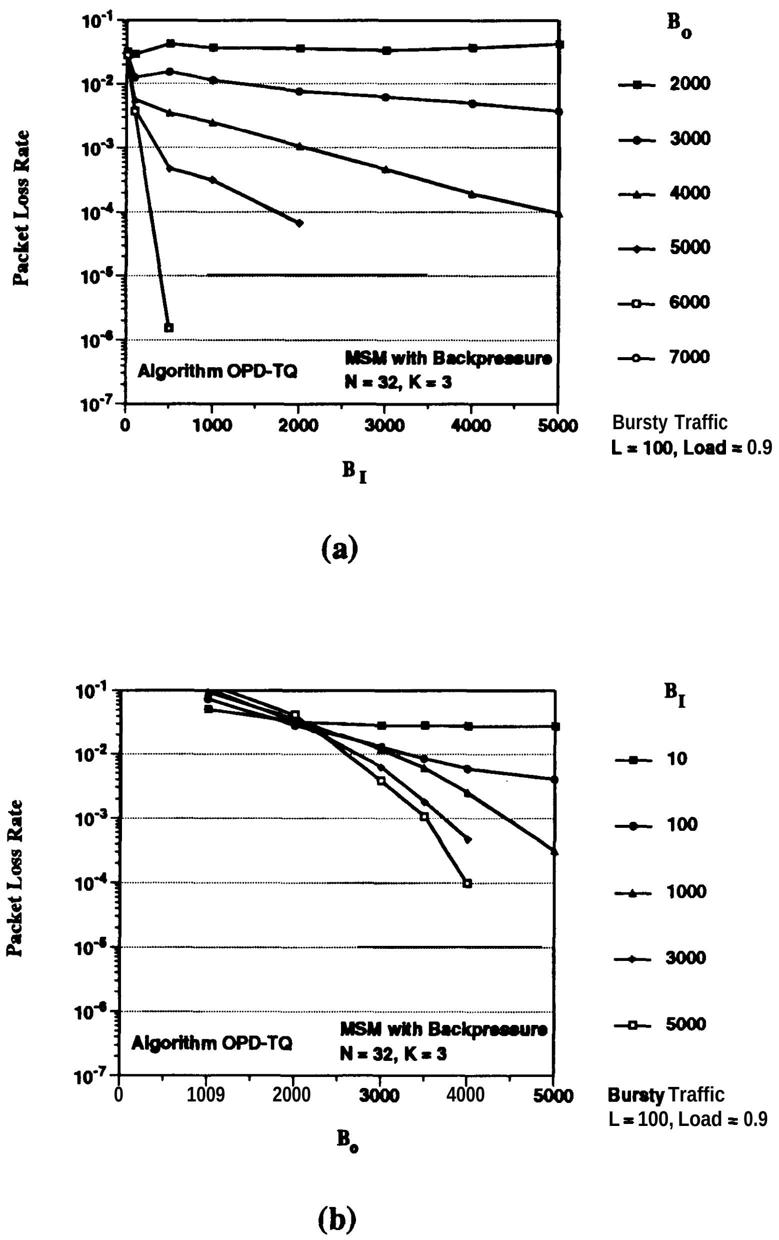
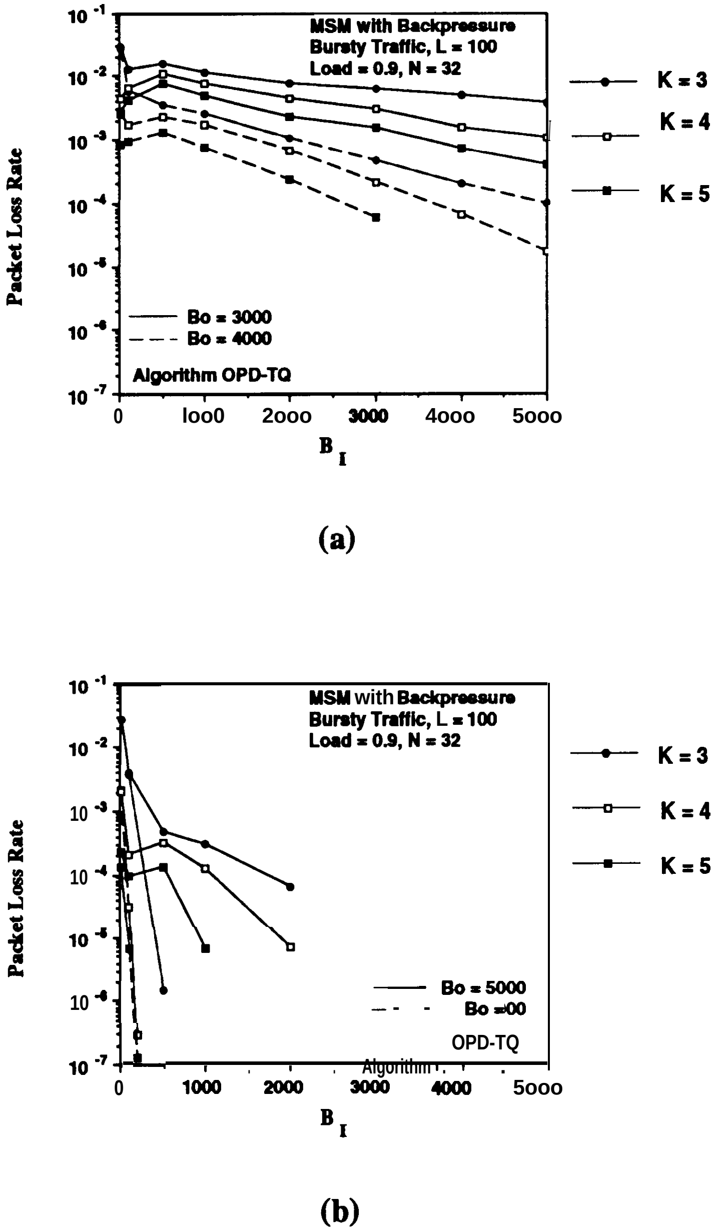


Fig. 4.14. Packet loss rate in the MSM switching fabric with backpressure, with algorithm One Per Destination — Top of the Queue, $M = 1$, $N = 32$, $K = 3$, under bursty traffic conditions with average burst size $L = 100$, load = 0.9; a) packet loss rate vs. the input buffer size, for different output buffer sizes; b) packet loss rate vs. the output buffer size, for different input buffer sizes (obtained by simulation).



(a)

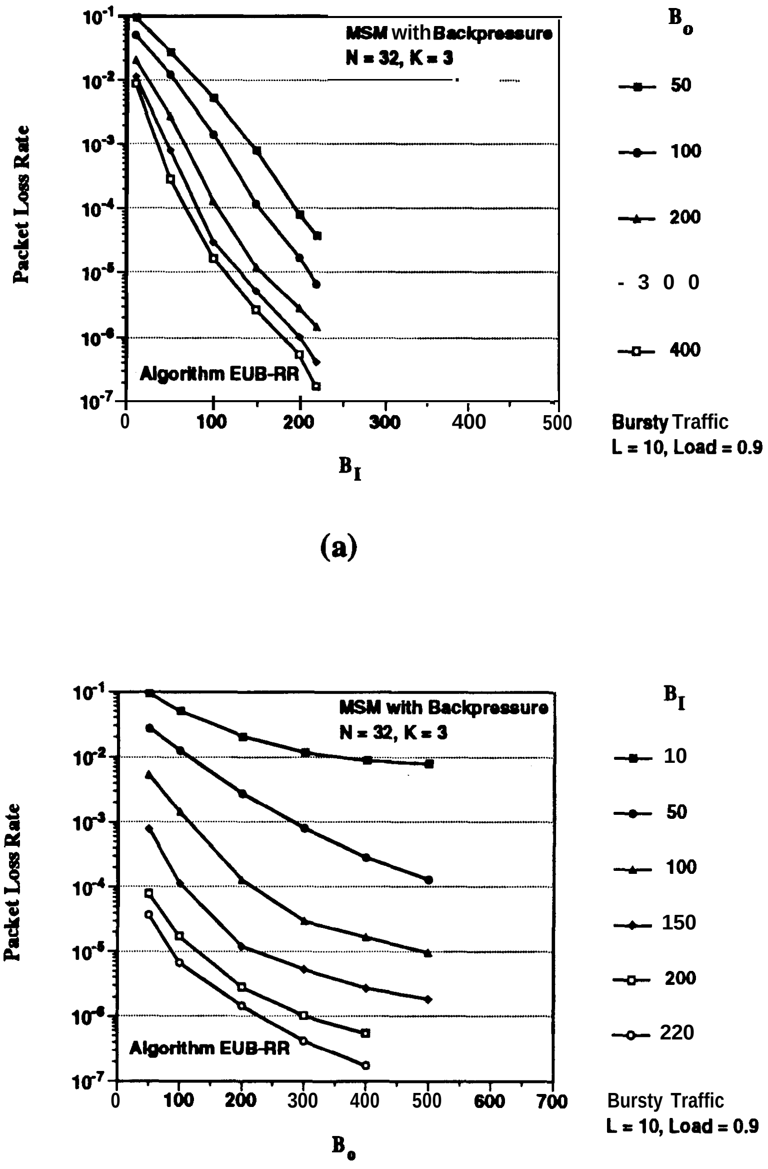
(b)

Fig. 4.15. Packet loss rate vs. the input buffer size in the MSM switching fabric with backpressure, with algorithm One Per Destination — Top of the Queue, $\boldsymbol{M}$ = 1, $\boldsymbol{N}$ = 32, for different $\boldsymbol{K}$, under bursty traffic conditions with average burst size $\boldsymbol{L}$ = 100, load = 0.9; a) for output buffer size equal to 3000 and 4000 packet buffers; b) for output buffer size equal to 5000 and 6000 packet buffers (obtained by simulation).

With all these algorithms ***(i.e.,*** FCFS, HOL bypassing, Exhaustive Until Blocked — Top of the Queue, and One Per Destination — Top of the Queue), blocking at the input (either due to HOL blocking or to unfairness) is a major problem when the output buffer size is smaller than the threshold; consequently, improvements in buffer requirements with respect to the switch without backpressure are only observed with burst sizes for which the threshold effect occurs ***(i.e.,*** long bursts), and are limited to the output buffer size given by the value of the threshold. Fortunately, blocking at the inputs can be obviated, and buffer requirements in the switch improved dramatically, by using algorithms Exhaustive Until Blocked — Round Robin or One Per Destination — Round Robin. With these algorithms, by serving the linked lists corresponding to the various destinations in a round-robin fashion, we remove the unfairness that led to blocking with algorithms Exhaustive Until Blocked — Top of the Queue and One Per Destination — Top of the Queue, and achieve the advantage of managing the queues on a destination-by-destination basis.

In Fig. 4.16, we show the packet loss rate in a ***32*** x ***32*** MSM switch as a function of the input and output buffer sizes for K = 3, and L = 10, with the input queues operated under the ***Exhaustive Until Blocked — Round Robin*** algorithm. We note the steep slope of the curves of the packet loss rate for fixed output buffer size as a function of the input buffer size. For example, to achieve a 10” packet loss rate, possible choices for the buffer sizes are:

B_i	B_o	Total Buffer Size per Port
100	900	1000
150	700	850
200	300	500
220	240	460
260	100	360
300	50	350



(a)

(b)

Fig. 4.16. Packet loss rate in the MSM switching fabric with backpressure, with algorithm Exhaustive Until Blocked — Round Robin, M = 1, N = 32, K = 3, under bursty traffic conditions with average burst size L = 10, load = 0.9; a) packet loss rate vs. the input buffer size, for different output buffer sizes; b) packet loss rate vs. the output buffer size, for different input buffer sizes (obtained by simulation).

In general, the minimum total buffer size is obtained by using larger input buffers than output buffers; this is because many packets can be held in the input buffers by using backpressure without inducing HOL blocking, due to intelligent bypassing of blocked packets in the input queues. With the packet flow heavily shaped by backpressure, the bursts wait in their respective input buffers, rather than congesting the outputs, leading to more even occupancy of the buffers and smaller buffer requirements. The reduction in total buffer requirements with respect to the switch without backpressure is above 65%. Furthermore, we note that with backpressure the total buffer size depends on B_o rather weakly for a relatively large range of ***B*,.** Consequently, since we are interested in reducing not only the total but also the local buffer requirements, we can use almost equal buffer sizes at both inputs and outputs and still benefit of considerable reductions in buffer requirements. As expected, the packet loss rate for given buffer sizes does not depend on K, as shown in Fig. 4.17, since backpressure is indeed the only limiting factor on the packet flow.

The packet loss rate for K = 3, and L = 10, with the input queues operated with algorithm ***One Per Destination — Round Robin*** is shown in Fig. 4.18 as a function of the input and output buffer sizes. For example, to achieve a 10^{-6} packet loss rate, possible choices for the required buffer sizes are:

B_i	B_o	Total Buffer Size per Port
150	900	1050
200	800	1000
270	300	570
340	100	440
400	50	450

Perhaps surprisingly, Algorithm One Per Destination — Round Robin yields slightly worse performance than Algorithm Exhaustive Until Blocked — Round Robin, as compared

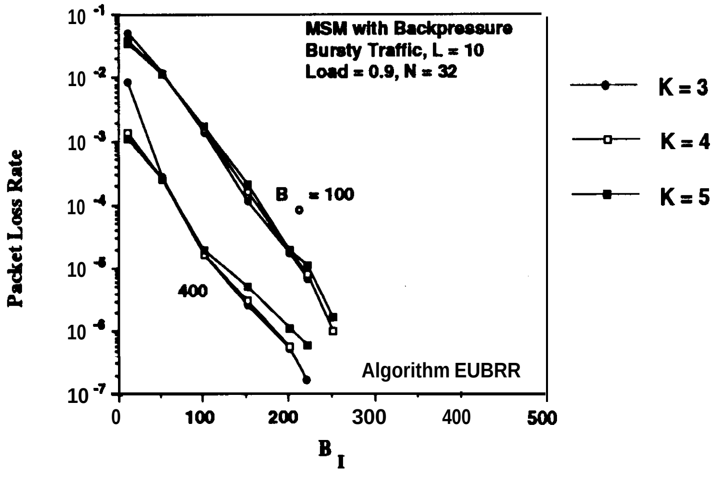


Fig. 4.17. Packet loss rate vs. the input buffer size in the MSM switching fabric with backpressure, with algorithm Exhaustive Until Blocked — Round Robin, *M* = 1, ***N*** = 32, for different ***K*,** under bursty traffic conditions with average burst size ***L*** = 10, load = 0.9, for different output buffer sizes (obtained by simulation).

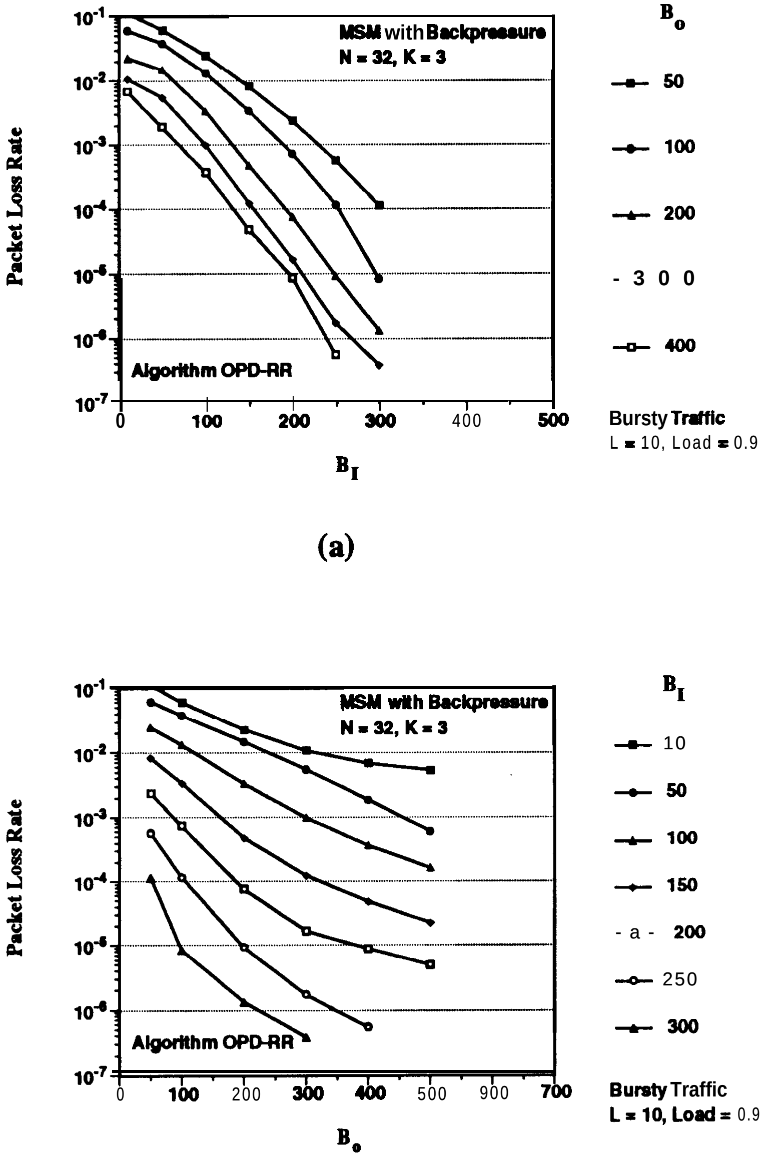


Fig. 4.18. Packet loss rate in the MSM switching fabric with backpressure, with algorithm One Per Destination — Round Robin, M = 1, N = 32, K = 3, under bursty traffic conditions with average burst size L = 10, load = 0.9; a) packet loss rate vs. the input buffer size, for different output buffer sizes; b) packet loss rate vs. the output buffer size, for different input buffer sizes (obtained by simulation).

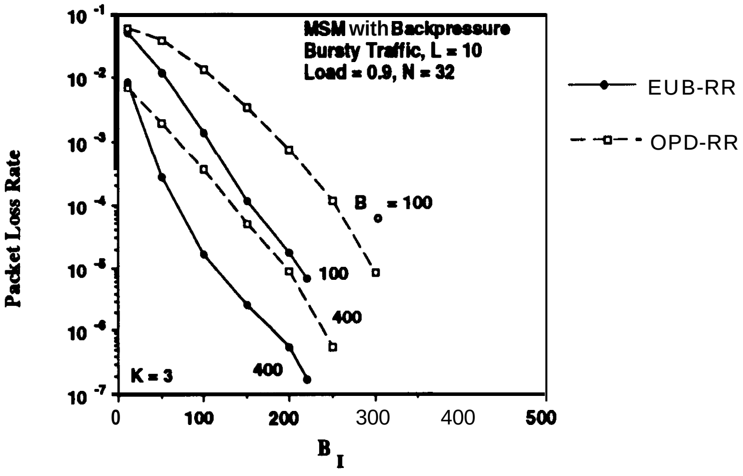


Fig. 4.19. Packet loss rate vs. the input buffer size in the MSM switching fabric with backpressure, with algorithms Exhaustive Until Blocked — Round Robin and One Per Destination — Round Robin, $\boldsymbol{M}$ = 1, $\boldsymbol{N}$ = 32, $\boldsymbol{K}$ = 3, under bursty traffic conditions with average burst size $\boldsymbol{L}$ = 10, load = 0.9, for different output buffer sizes (obtained by simulation).

in Fig. 4.19, where we show the packet loss rate for the two algorithms as a function of the input buffer size, for different output buffer sizes. However, with algorithm One Per Destination — Round Robin, the packet loss rate for given buffer sizes moderately improves with the number of banyans, as shown in Fig. 4.20, and eventually, for larger $\boldsymbol{K}$, the two algorithm have very similar performance. This indicates that with Algorithm One Per Destination — Round Robin, since only one packet per list is served in a slot, there is more contention in the banyans than with Algorithm Exhaustive Until Blocked — Round Robin, and therefore more banyans are necessary to provide sufficient routing capacity.

As the burst size increases, since the benefit of using backpressure increases with the burst length, the increase in buffer requirements is markedly sub-linear, as shown in Fig. 4.2 1, where we plot the packet loss rate in the switch with $\boldsymbol{K}$ = 3 banyans, operated with ***Exhaustive Until Blocked-Round Robin,*** versus the output buffer size normalized to the

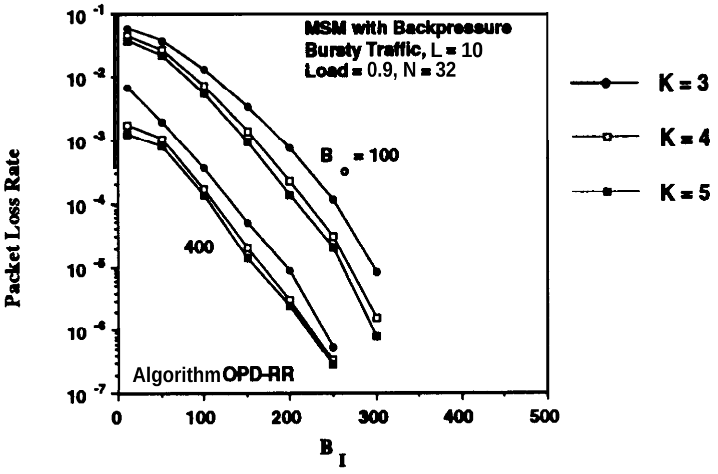


Fig. 4.20. Packet loss rate vs. the input buffer size in the MSM switching fabric with backpressure, with algorithm One Per Destination — Round Robin, M = 1, N = 32, for different K, under bursty traffic conditions with average burst size L = 10, load = 0.9, for different output buffer sizes (obtained by simulation).

burst length, for different input buffer sizes also normalized to the burst length, for L = 10 and L =100.The packet loss rate in the switch for L = 100 as a function of input and output buffer sizes are reported in more detail in Fig. 4.22. In this case, the required buffer sizes for a 10^{-6} packet loss rate are for example:

B_i	B_o	Total Buffer Size per Port
2000	800	2800
1800	1000	2800
1600	1500	3100
1250	2500	3750
730	5000	5730
650	6000	6650

Choosing the buffer sizes appropriately, the reduction in the total required buffer size is above 70% with respect to the switch without backpressure, and almost 60% with respect

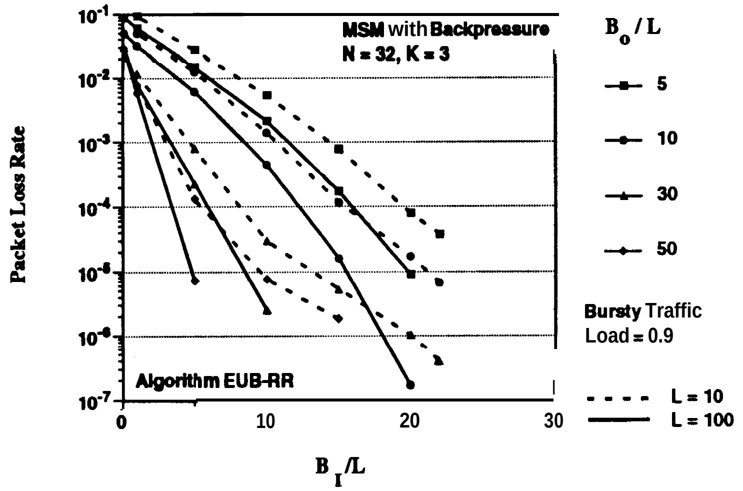
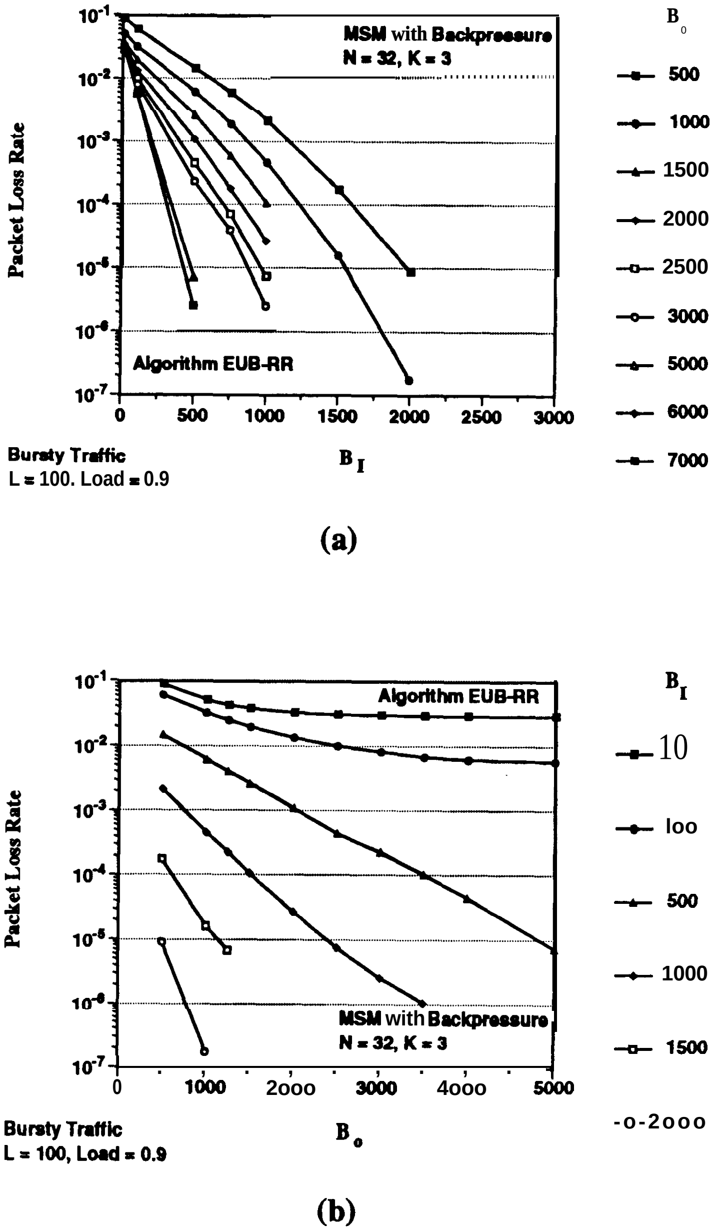


Fig. 4.21. Packet loss rate vs. the input buffer size normalized to the burst length in the MSM switching fabric with backpressure, with algorithm Exhaustive Until Blocked — Round Robin, ***M*** = 1, ***N*** = 32, ***K*** = 3, under bursty traffic conditions with average burst size ***L*** = 10 and ***L*** = 100, load = 0.9, for different output buffer sizes normalized to the burst length (obtained by simulation).

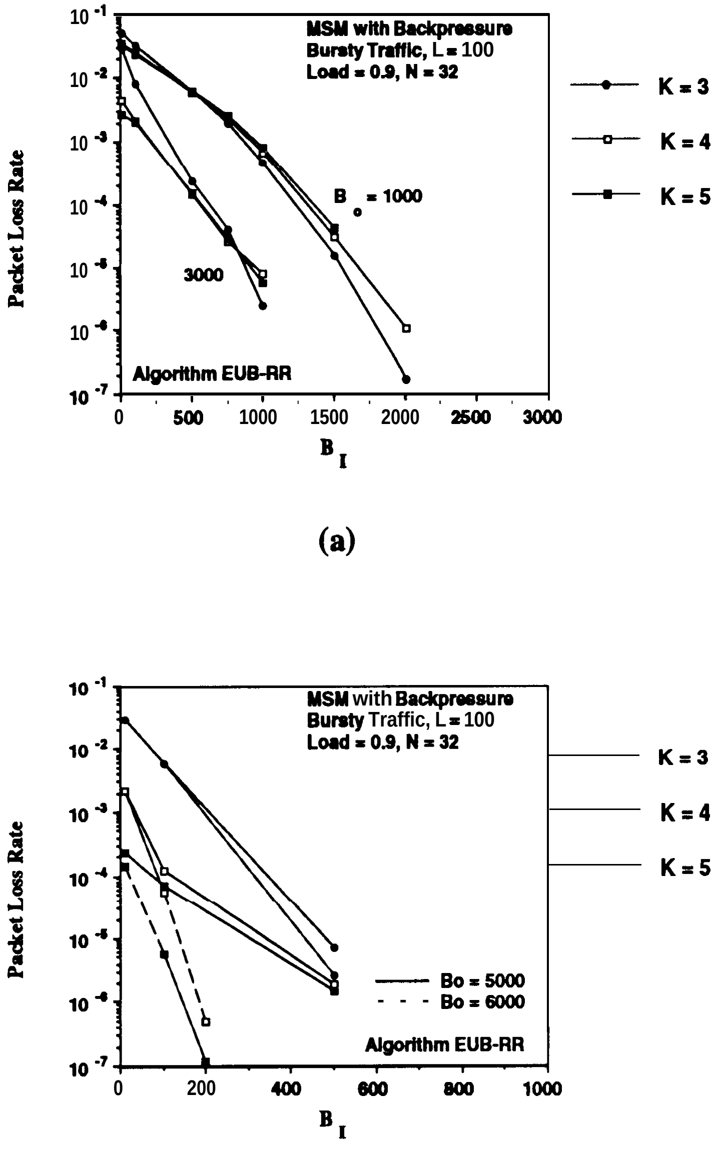
to the switch with backpressure operated with single-queue buffer-management strategies or with Algorithms Exhaustive Until Blocked — Top of the Queue and One Per Destination — Top of the Queue. Furthermore, here the buffers can be evenly distributed at the inputs and at the outputs, thus balancing the local buffer requirements. For large output buffer sizes ***(B, ≥ 6000),*** in a similar way as with all the previous algorithms, there exist a threshold buffer size above which the packet loss rate becomes independent of the input buffer size. Note that the required buffer sizes for this threshold are basically the same as with the previous algorithms; as anticipated above, this confirms that, at the threshold, the available switch capacity is the main limiting factor on the packet flow, regardless of the backpressure algorithm used. Accordingly, the packet loss rate depends on the number of banyans ***K*** only near the threshold, as shown in Fig. 4.23.



(a)

(b)

Fig. 4.22. Packet loss rate in the MSM switching fabric with backpressure, with algorithm Exhaustive Until Blocked — Round Robin, $M = 1$, $N = 32$, $K = 3$, under bursty traffic conditions with average burst size L = 100, load = 0.9; a) packet loss rate vs. the input buffer size, for different output buffer sizes; b) packet loss rate vs. the output buffer size, for different input buffer sizes (obtained by simulation).



(a)

(b)

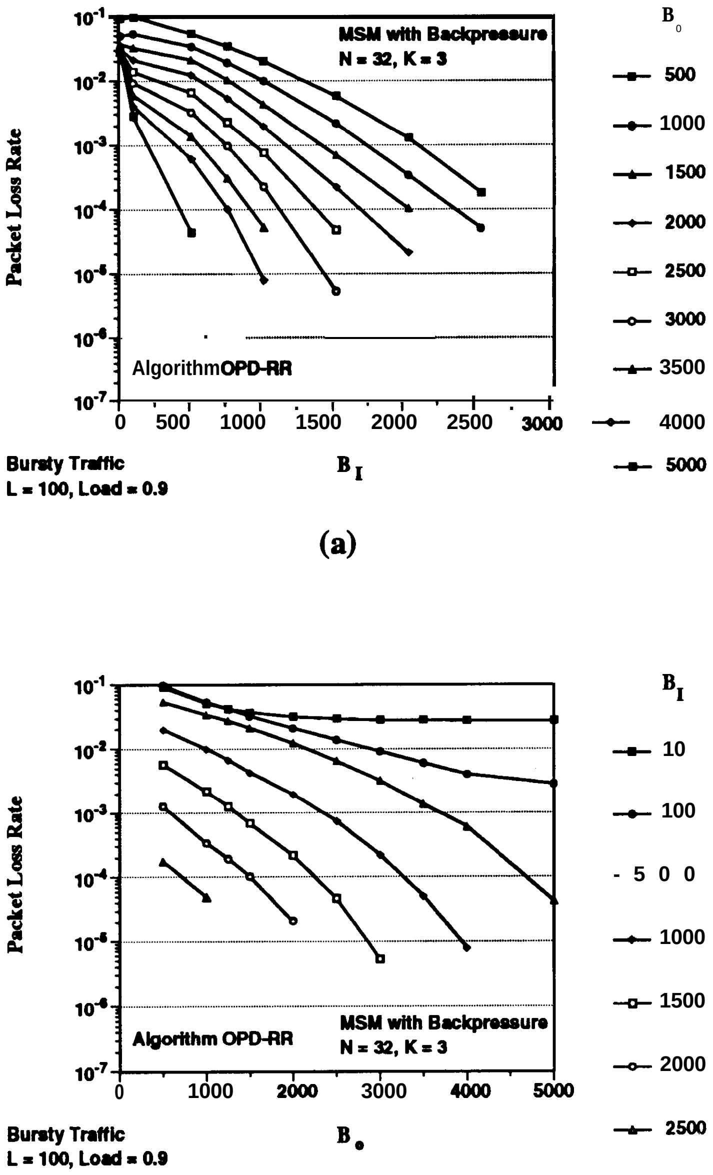
Fig. 4.23. Packet loss rate vs. the input buffer size in the MSM switching fabric with backpressure, with algorithm Exhaustive Until Blocked — Round Robin, M = 1, N = 32, for different K, under bursty traffic conditions with average burst size L = 100, load = 0.9; a) for output buffer size equal to 1000 and 3000 packet buffers; b) for output buffer size equal to 5000 and 6000 packet buffers (obtained by simulation).

The packet loss rate ***in the*** switch with ***Algorithm One Per Destination — Round Robin, $K = 3$,*** is shown in Fig. 4.24, for $L = 100$. Similar considerations to those made in the case $L = 10$ are valid; namely, with $K = 3$, the buffer requirements with this algorithm are slightly worse than those obtained with Algorithm Exhaustive Until Blocked — Round Robin, as compared in Fig. 4.25; however, the buffer requirements improve for larger K, as shown in Fig. 4.26, and become quite similar to those achieved with the other algorithm.

Algorithms ***Exhaustive Until Blocked — Round Robin*** and ***One Per Destination — Round Robin*** offer of course considerable reductions in buffer requirements also for A4 > 1. For example, in Fig. 4.27 we show the packet loss rate in a 128 x 128 MSM switch, $M = 4$, N = 32, $K = 12$, operated with ***Algorithm Exhaustive Until Blocked-Round Robin,*** as a function of the buffer size per input component B_I and per output component **B,.** For example, to achieve a packet loss rate below 10^{-6}**,** the required buffer sizes per components are:

B_i	B_o	Total Buffer Size per 4 Ports
50	1200	1250
100	900	1000
150	700	850
300	500	800
400	400	800

By properly selecting the buffer sizes, the reduction with respect to the switch without backpressure is above 50%. Again, we note that with backpressure we can size input and output buffers equally, thus improving the local buffer requirements dramatically.



(a)

(b)

Fig. 4.24. Packet loss rate in the MSM switching fabric with backpressure, with algorithm One Per Destination — Round Robin, $\boldsymbol{M}$ = 1, N = 32, $\boldsymbol{K}$ = 3, under bursty traffic conditions with average burst size $\boldsymbol{L}$ = 100, load = 0.9; a) packet loss rate vs. the input buffer size, for different output buffer sizes; b) packet loss rate vs. the output buffer size, for different input buffer sizes (obtained by simulation).

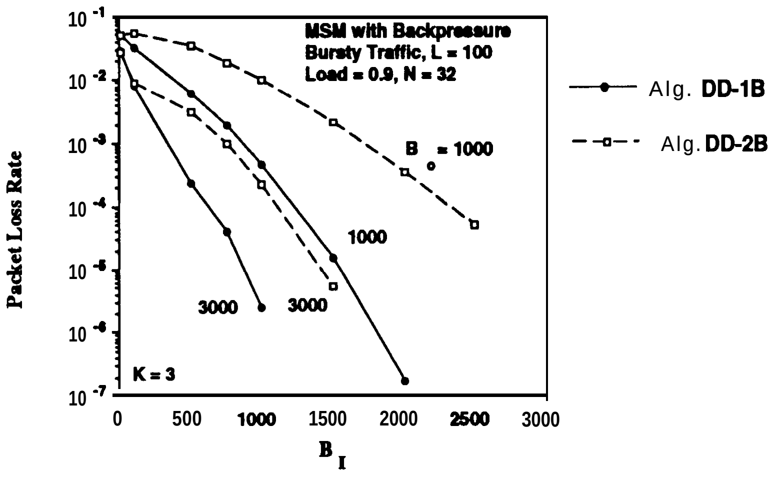


Fig. 4.25. Packet loss rate vs. the input buffer size in the MSM switching fabric with backpressure, with algorithms Exhaustive Until Blocked — Round Robin and DD-2B, *M* = 1, N = *32, K* = 3, under bursty traffic conditions with average burst size *L* = 100, load = 0.9, for different output buffer sizes (obtained by simulation).

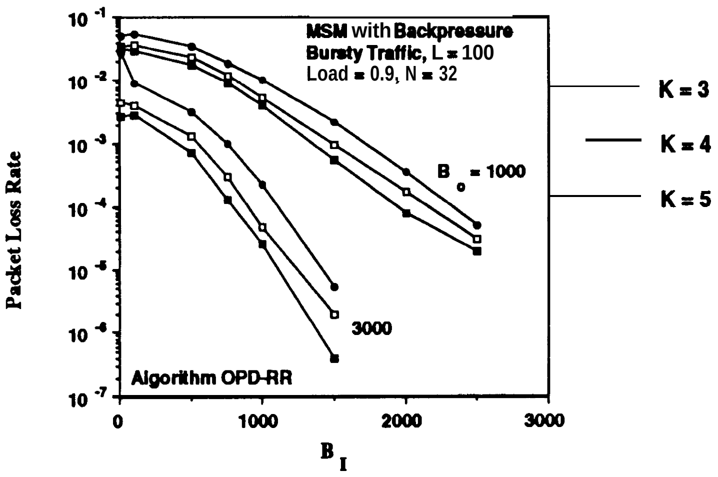
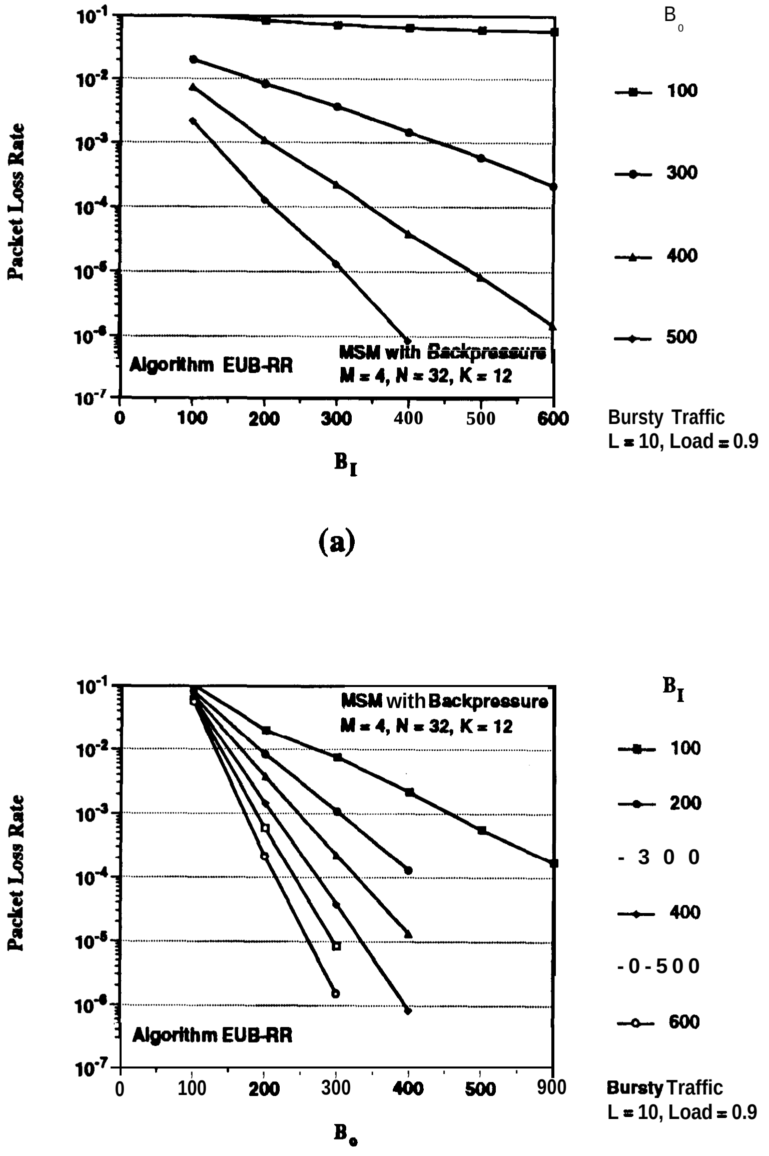


Fig. 4.26. Packet loss rate vs. the input buffer size in the MSM switching fabric with backpressure, with algorithm One Per Destination — Round Robin, *M* = 1, N = 32, for different *K*, under bursty traffic conditions with average burst size *L* = 100, load = 0.9, for different output buffer sizes (obtained by simulation).



(a)

(b)

Fig. 4.27. Packet loss rate in a 128 x 128 MSM switching fabric with backpressure, with algorithm Exhaustive Until Blocked — Round Robin, ***M* = 4, *N* = 32, *K* = 3,** under bursty traffic conditions with average burst size ***L*** = 10, load = 0.9; a) packet loss rate vs. the input buffer size, for different output buffer sizes; b) packet loss rate vs. the output buffer size, for different input buffer sizes (obtained by simulation).

References

[Ban91] T. R. Banniza ***et al.,*** "Design and Technology Aspects of VLSI's for ATM Switches," ***IEEE Jour. Select. Areas Comm.,*** SAC-9, pp. 1255- 1264, Oct. 199 1.

[Ben65] V. E. Benes, ***Mathematical Theory of Connecting Networks and Telephone Traffic,*** Academic Press, New York, 1965.

[Bru90] G. Bruzzi and A. Pattavina, "Performance Evaluation of an Input-Queued ATM Switch with Internal Speed-up and Finite Output Queues," *Proc. GLOBECOM'90,* paper 801.5, San Diego, CA, Dec. 1990.

[Chi93] F. M. Chiussi and F. A. Tobagi, "Implementation of a Three-Stage Banyan-Based Architecture with Input and Output Buffers for Large Fast Packet Switches," ***Tech. Rep. No. CSL-TR-93-577,*** Stanford University, Stanford, CA, June 1993..

[CCI90] CCITT Study Group XVIII, Revised Draft Recommendation I.21 1, Geneva, May 9-25, 1990.

[Clo53] C. Clos, "A Study of Non-Blocking Switching Networks," ***Bell Syst. Tech. Jour.,*** pp. 406-424, Mar. 1953.

[Eck88] A. E. Eckberg and T.-C. Hou, "Effects of Output Buffer Sharing on Buffer Requirements in an ATDM Packet Switch," *Proc. INFOCOM'88,* paper 5A.4, New Orleans, Louisiana, March 1988.

[End90] N. Endo ***et al.,*** "Traffic Characteristics Evaluation of a Shared Buffer ATM Switch," ***Proc. GLOBECOM '90,*** paper 905.1, San Diego, CA, December 1990.

[Eng89] K. Y. Eng, M. J. Karol, and Y. S. Yeh, "A Growable Packet (ATM) Switch Architecture: Design Principles and Applications," *Proc. GLOBECOM'89,* paper 32.2, Dallas, TX, Nov. 1989.

[Gup91] A. K. Gupta and N. D. Georganas, "Analysis of a Packet Switch with Input and Output Buffers and Speed Constraints," ***Proc. INFOCOM '91,*** paper 7A.2, Bal Harbour, FL, April 1991.

[Hlu873 M. Hluchyj and M. Karol, "Queueing in High-Performance Packet Switching," ***IEEE Jour. Select. Areas Commun.,*** SAC-5, 8 , pp. 1274-1292, Oct. 1987.

[Hlu88] M. Hluchyj and M. Karol, "Queueing in Space-Division Packet Switching," ***Proc. INFOCOM '88,*** paper 4A.3, New Orleans, Louisiana, March 1988.

[Hua84] A. Huang and S. Knauer, “Starlite: A Wideband Digital Switch,” *Proc. GLOBECOM '84,* pp. 121-125, Atlanta, GA, Nov. 1984.

[Ili90a] I. Iliadis and W. E. Denzel, “Performance of Packet Switches with Input and Output Queueing,” *Proc. ICC'90,* paper 316.3, Atlanta, GA, April 1990.

[Ili90b] I. Iliadis, “Head of the Line Arbitration of Packet Switches with Input and Output Queueing,” *Proc. Fourth Znt. Conf. Data Commun. Syst. Perform.*, pp. 85-98, Barcelona, Spain, June 1990.

[Ili91] I. Iliadis, “Performance of a Packet Switch with Shared Buffer and Input Queueing,” *Proc. Thirteenth Znt. Teletraffic Congress,* pp. 911-916, Copenhagen, Denmark, June 199 1.

[Ili92] I. Iliadis, “Performance of a Packet Switch with Input and Output Queueing under Unbalanced Traffic,” *Proc. ZNFOCOM '92,* paper 5D.4, Florence, Italy, May 1992.

[JSA91] Special Issue on Large Scale ATM Switching Systems for B-ISDN, eds. W. E. Stephens *et al., IEEE Jour. Select. Areas Comm.,* SAC-9, Oct. 1991.

[Koz91] T. Kozaki *et al.,* “32 × 32 Shared Buffer Type ATM Switch VLSI’s for B-ISDN’s,” *IEEE Jour. Select. Areas Comm.,* SAC-9, pp. 1239-1247, Oct. 1991.

[Kuw89] H. Kuwahara *et al.,* “A Shared Buffer Memory Switch for an ATM Exchange,” *Proc.* ICC ‘89, paper 4.4, Boston, MA, June 1989.

[Lia89] K.-Q. Liao, X. Dziong, and L. G. Mason, “Dynamic Link Bandwidth Allocation in an Integrated Services Network,” *Proc.* ICC ‘89, paper 3 1.5, Boston, MA, June 1989.

[Lie90] S. C. Liew, “Performance of Input-Buffered and Output-Buffered ATM Switches Under Bursty Traffic: Simulation Study”, *Proc. GLOBECOM '89,* paper 905.2, San Diego, CA, Dec. 1990.

[Mar90] W. S. Marcus, “A CMOS Batcher and Banyan Chip Set for B-ISDN Packet Switching,” *IEEE Jour. Solid-State Circ.*, 25, pp. 1426-1431, Dec. 1990.

[Mel89] R. Melen and J. S. Turner, “Nonblocking Networks for Fast Packet Switching,” *Proc. ZNFOCOM '89,* pp. 548- 557, Ottawa, Ont., Canada, 1989.

[New88] P. Newman, “A Fast Packet Switch for the Integrated Services Backbone Network,” *IEEE Jour. Select. Areas Comm.*, SAC-6, pp. 1468- 1479, Dec. 1988.

[Noj87] S. Nojima *et al.,* “Integrated Services Packet Network Using Bus Matrix Switch,” *IEEE Jour. Select. Areas Comm.,* SAC-5, pp. 1284-1292, Oct. 1987.

[Oda90] T. Oda and Y. Watanabe, "Optimal Trunk Reservation for a Group with Multislot Traffic Streams," ***IEEE Trans. Commun., 38,*** pp. 1078-1084, July 1990.

[Pat81] J. H. Patel, "Performance of Processor-Memory Interconnections for Multiprocessors," ***IEEE Trans. Comp., C-30,*** pp. 77 l-780, Oct. 198 1.

[Sar91] K. W. Sarkies, "The Bypass Queue in Fast Packet Switching," ***IEEE Trans. Commun., 39,*** pp. 766-774, May 1991.

[Sho91] Y. Shobatake ***et al.,*** "A One-Chip Scalable 8 x 8 ATM Switch LSI Employing Shared Buffer Architecture," ***IEEE Jour. Select. Areas Comm.***, SAC-9, pp. 1248-1254, Oct. 1991.

[Suz89] H. Suzuki ***et*** al., "Output-Buffer Switch Architecture for Asynchronous Transfer Mode," ***Proc. ICC*** '89, paper 4.1, Boston, MA, June 1989.

[Tob90a] F. A. Tobagi, "Fast Packet Switch Architectures for Broadband Integrated Services Digital Networks," ***IEEE Proceedings,*** vol. 78, 1, Jan. 1990.

[Tob90b] F. A. Tobagi and T. C. Kwok, "Fast Packet Switch Architectures and the Tandem Banyan Switching Fabric," ***Proc. NATO Workshop on Architecture and Performance Issues of High-Capacity Local and Metropolitan Area Networks,*** Sophia Antipolis, France, June 25-27, 1990.

[Tob91] F. A. Tobagi, T. Kwok, and F. M. Chiussi, "Architecture, Performance and Implementation of the Tandem Banyan Fast Packet Switch," ***IEEE Jour. Select. Areas Comm., SAC-9,*** pp. 1173-l 193, Oct. 1991.

[Tur86] J. S. Turner, "Design of an Integrated Services Packet Network," ***IEEE Jour. Select. Areas Comm.***, SAC-4, pp. 1373-1380, Nov. 1986.

[Tur88] J. S. Turner, "Design of a Broadcast Packet Switching Network," ***IEEE Trans. Commun.,*** 36, pp. 734-743, June 1988.

[Wan92] W. Wang, "Architectural Design and Performance Analysis of Large-Scale Asynchronous Transfer Mode (ATM) Switches," ***Ph.D. Thesis,*** Stanford University, Stanford, CA, Aug. 1992.

[Wu80] C.-L. Wu and T.-Y. Feng, "On a Class of Multistage Interconnection Networks," ***IEEE Trans. Computers,*** pp.694-702, Aug. 1980.

[Yeh87] Y.-S. Yeh, M. Hluchyi, and A. Acampora, "The Knockout Switch: A Simple, Modular Architecture for High Performance Packet Switching," ***IEEE Jour. Select. Areas Comm.***, SAC-5, pp. 1274-1283, Oct. 1987.