UC Irvine
ICS Technical Reports

Title
Survey of switching techniques in high-speed networks and their performance

Permalink
https://escholarship.org/uc/item/6r54g1n5

Authors
Oie, Yuji
Suda, Tatsuya
Kolson, David
et al.

Publication Date
1989-10-12

Peer reviewed

eScholarship.org Powered by the California Digital Library
University of California

Notice: This Material
may be protected
by Copyright Law
(Title 17 U.S.C.)

SURVEY OF SWITCHING TECHNIQUES IN HIGH-SPEED NETWORKS AND THEIR PERFORMANCE

Yuji OIE

Tatsuya SUDA

David KOLSON

Masayuki MURATA

Hideo MIYAHARA

Technical Report No. 89-37

Department of Information and Computer Science
University of California, Irvine
Irvine, California 92717

Z
699
C3
no. 89-37

Survey of Switching Techniques in High–Speed Networks and Their Performance[*]

Yuji OIE[1], **Tatsuya SUDA**[2], David KOLSON[2], Masayuki MURATA[3], Hideo MIYAHARA[3]

October 12, 1989

1. Department of Electrical Engineering, Sasebo College of Technology, Sasebo 857–11, Japan.

2. **Department of Information and Computer Science, University of California, Irvine, CA 92717, U.S.A.**

3. Department of Information and Computer Sciences, Faculty of Engineering Science, Osaka University, Toyonaka 560, Japan.

Mailing Address:
All the future correspondence should be addressed to **Tatsuya Suda** at the above address.

* : This material is based upon work supported by the National Science Foundation under Grant No. DCI-8602052. This research is also in part supported by the University of California MICRO program.

Abstract

One of the most promising approaches for high speed networks for integrated service applications is fast packet switching, or ATM (Asynchronous Transfer Mode). ATM can be characterized by very high speed transmission links and simple, hardwired protocols within a network. To match the transmission speed of the network links, and to minimize the overhead due to the processing of network protocols, the switching of cells is done in hardware switching fabrics in ATM networks.

A number of designs has been proposed for implementing ATM switches. While many differences exist among the proposals, the vast majority of them is based on self-routing multi-stage interconnection networks. This is because of the desirable features of multi-stage interconnection networks such as self-routing capability and suitability for VLSI implementation.

Existing ATM switch architectures can be classified into two major classes: blocking switches, where blockings of cells may occur within a switch when more than one cell contends for the same internal link, and non-blocking switches, where no internal blocking occurs. A large number of techniques has also been proposed to improve the performance of blocking and nonblocking switches. In this paper, we present an extensive survey of the existing proposals for ATM switch architectures, focusing on their performance issues.

1 Introduction

In recent years, the broadband ISDN (B–ISDN) has received increasing attention as a vehicle to accommodate a wide variety of communication applications [2]. B-ISDN is expected to support diverse applications ranging from low bit rate communications between terminals and host computers, to the broadcasting of high resolution videos.

One of the most promising approaches for B–ISDN is the Asynchronous Transfer Mode (ATM). ATM can be characterized by very high speed transmission links and simple, hardwired protocols within a network. In ATM networks, information (such as voice, data, and video) is divided into fixed length data blocks, called cells, and these cells are asynchronously transmitted through the network. To match the transmission speed of the network links, and to minimize the overhead due to processing of network protocols, the switching of cells is done in the hardware switching fabrics in ATM networks, unlike the traditional packet switching approach, where switching is done by computers running communication processes. VLSI technology is used to implement the switching protocols in hardware rather than software.

A large number of designs has been proposed for implementing ATM switches. While many differences exist among the proposed ATM switch architectures, most of them are based on multi-stage interconnection networks. This is due to the desirable features of multi-stage interconnection networks; features such as a self–routing capability and suitability for VLSI implementation.

ATM switches can be grouped into two major classes: blocking switches and nonblocking switches. In blocking switches, internal blocking of cells occurs within a switch when more than one cell contends for the same internal link. In nonblocking switches, internal blocking will not happen. However, blocking still occurs on the outputs to the switch, when more than one cell is destined for the same output. Note that the blocking due to output contention may also occur in blocking switches. Examples of blocking switches include the Banyan switch and the buffered Banyan switch [14, 16]. The crossbar switch [5] and the Batcher Banyan switch [11] are examples of nonblocking switches.

A large number of techniques has also been proposed to improve the performance of blocking and nonblocking switches. Performance improvement is gained by speeding up switching fabrics, reducing cell loss due to output contention, and/or constructing a switch fabric of multiple switch planes.

In this paper, we categorize and discuss the major switch architectures and their improvement techniques, focusing on the performance issues. The performance measures of our interests are the maximum throughput, the delay time, and the cell loss probability. This paper is organized as follows. In section 2, we summerize the major assumptions and notations used in this paper. Blocking switches are discussed in section 3. The Banyan switch and its improvements are discussed in subsections 3.1 and 3.2, respectively. The buffered Banyan switch (i.e., the Banyan switch with internal buffers) is discussed in subsection 3.3. Section 4 surveys nonblocking switches (subsections 4.1, 4.2 and 4.3) and their improvement techniques (subsections 4.4 through 4.8). Other related research on nonblocking switches are summerized in subsection 4.9. Subsection 4.10 summerizes and compares the performance of a variety of nonblocking switches. Concluding remarks are given in section 5.

2 Assumptions and Notations

Before we start surveying various ATM switch architectures, we summerize assumptions and notations used in this paper. Throughout the paper, we assume switch fabrics of size $N \times N$ (N input ports and N output ports). Input and output channels to a switch are of the same speed. Cells are assumed to be of a constant length, and the channel time is slotted with the slot size being equal to a cell transmission time. All the channels are assumed to be synchronized. Arrivals of cells at each of the N input ports follow a Bernoulli process. Namely, a cell arrives with probability p in a slot, and there is no arrival with probability $1 - p$. Since we use the slot length as the unit of time, p also corresponds to the input traffic load to the input channel.

Uniform traffic refers to the situation where incoming cells are destined to N output ports with a uniform probability of $\frac{1}{N}$. Unless otherwise stated, uniform traffic is assumed in this paper. When all the cells from one input port are going to a particular output port, the traffic is referred to as a point–to–point connection. A hot spot refers to an output port where a heavy concentration of cells is expected to happen.

In some switch architectures, buffers are provided to store cells. Because of the different approaches taken in the design of a switch, there are possible choices for the physical location of the buffers relative to the switch. Buffers may be placed on the inputs to the switch, or on the outputs to the switch, or possibly on both. Queueing of cells may be implemented in a shared buffer common to all the inputs. B_{input} and B_{output} denote the size of a buffer on an input and an output, respectively. B denotes the size of a shared buffer. B_{input}, B_{output} and B may be zero, finite or infinite. Unless otherwise stated, we follow the above assumptions and notations in this paper.

3 Blocking ATM Switches

3.1 Banyan Switches

Banyan switches belong to the class of blocking switches, a major class of multistage interconnection networks. An $N \times N$ Banyan switch is constructed of switching elements which have k inputs and k outputs, i.e., $k \times k$ switching elements, arranged into $\log_k N$ stages. Fig.1 shows an 8×8 Banyan switch with binary switching elements ($k = 2$). Cells are self–routed from input ports to output ports by using the following addressing scheme. For a given destination port address, if k is 2, the ith stage will need only the ith bit of the destination address to route a cell. If this bit is 0, the cell is routed to the upper output, and if this bit is 1, it is routed to the lower one. Fig.1 also shows an example of a cell arriving on input port 1 with destination address 101 (output port 5).

Although the Banyan switch is simple and possesses attractive features (such as modularity which makes the switch suitable for VLSI implementation), it also has some disadvantages. One of its major disadvantages is that it is internally blocking. Namely, cells destined for different output ports may contend for a common link within the switch. (Communication lines between switching elements are referred to as links in this paper.) This results in blocking of all cells that wish to use that link, except for one. Hence, the Banyan switch is referred to as a blocking switch. In Fig.2, a cell going to output port 0 and a cell going to output port 1 contend for the link between the second and third stages. As a result, only one of them (the cell from input port 1 in this example) can actually reach its destination (output port 0). If each switching element has a buffer (input link buffer), blocked cells will be stored for later transmission; otherwise they will be dropped from the switch. It is this internal blocking which causes the degradation in the performance of the Banyan switch.

Kruskal et. al. investigated the effects of the internal blocking on the performance of the Banyan switch [17]. They developed approximate analysis for the Banyan switch assuming that switching elements do not have buffers and that the traffic is uniform. They obtained the probability that cells successfully reaches their destination output ports. That probability is given by

$$p(success) = \frac{2k}{(k-1)m + \frac{2k}{p}}, \tag{1}$$

where m is the number of stages in the Banyan switch, k is the size of a switching element, and p is the input traffic load. Equation 1 turns out to be a very good approximation except when p is large and m is small. From this equation, we can easily see that $p(success)$ decreases sharply as the number of stages m increases. Therefore, the size of the Banyan switch impacts heavily on the performance of the switch.

3.2 Improvements to Banyan Switches

In the above we saw that the internal blocking can occur within the Banyan switch, when cells contend for a common link in the switch, and that it limits the throughput of the Banyan switch. (Throughput of a switch is the average number of cells coming out from an output port per slot.) One can improve the limited throughput of the Banyan switch by reducing or eliminating the internal blocking. Dilated switches and replicated switches reduce the internal blocking by providing multiple paths between any input and output port pairs. In the following we discuss these switch architectures. This subsection also briefly refers to another switch architecture, the Batcher Banyan switch, in which the internal blocking is completely eliminated. The Batcher Banyan switch has a sorting network which precedes the Banyan switch. A sorting network orders the incoming cells according to the destinations in their ascending order, and feeds them into the Banyan switch, making the switch internally nonblocking. The Batcher Banyan switch is discussed again in detail in section 4.

3.2.1 Multipath Banyan Switches

By replacing each link which connects switching elements in the Banyan switch by d (distinct) parallel links, we obtain the d–dilated Banyan switch. Another way to provide multiple paths is to construct a switch from a multiple, say d, of Banyan switch planes in parallel. This architecture is called the d–replicated switch. These two switch configurations provide d multiple paths between any input and output port pairs, and thus reduce the probability of the internal blocking.

These multipath Banyan switches have been analyzed in [17, 32], assuming that switching elements do not have any buffers. For the d dilated Banyan switch, it was shown that the cell loss probability decreases when the dilation d increases [17, 32]. Dilation of between 4 and 8 is shown to be sufficient enough to reduce the cell loss probability to a very small value. The d–dilated Banyan switch becomes a nonblocking switch, when $d = k^{\lceil m/2 \rceil}$, where k is the size of a switching element, and m is the number of stages in the Banyan switch. The d–replicated switch was shown to provide a similar performance to that of a d–dilated switch [17].

Adding a supplementary switch plane to the Banyan switch can also provide multiple paths between input and output port pairs. Anido et. al. [1] considered a multipath switch constructed from two Banyan switches: one act as a routing network, while the other acts as a switching network. They discussed several path selection algorithms and their performance.

3.2.2 Batcher Banyan Switches

The Banyan switch possesses an interesting characteristic. If all of the incoming cells are in ascending order relative to their output addresses, the switch is guaranteed to prevent cells from being internally blocked. To guarantee that the cells will be in ascending order requires only that there be some type of sorting network preceding the Banyan switch. In the Starlite switch [11], a Batcher network is used as the sorting network. This switch configuration is called the Batcher Banyan switch. The Batcher Banyan switch belongs to a major class of multi–stage interconnection networks, the class of nonblocking switches. This class of switches is discussed in detail in section 4.

3.3 Banyan Switches with Internal Buffers (Buffered Banyan Switches)

In the above we saw that the internal blocking limits the throughput of the Banyan switch, and that the multipath and the Batcher Banyan switches improve the throughput by either reducing or eliminating the internal blocking. There exists another switch architecture, the buffered Banyan switch, which improves the throughput by storing contending cells in buffers at switching elements and retransmitting them at later time.

3.3.1 Buffered Banyan Switches

If each switching element within the Banyan switch has a buffer to store blocked cells, cell loss will be decreased, and the throughput will be improved. Such a switch is called the buffered Banyan switch (Fig.3). In the buffered Banyan switch, each switching element is a $k \times k$ switch with a buffer (input link buffer) on each of its input links. The buffered Banyan switch has input buffer controllers (IBCs) in front of the first stage. The IBC forwards the head cell in the input port buffer to the switch, when there is space in the input link buffer of the corresponding switching element at the first stage of the switch.

Once cells are fed into the switch, they are controlled by "back pressure". A cell advances to a switching element at the next stage, if there is space in the input link buffer associated with the destination switching element. If cells from different input link buffers of a switching element are going to the same output link, one cell is randomly chosen and advanced to the next stage, and the others remain in their input link buffers. When congestion happens and the input link buffers start filling up, the back pressure mechanism propagates from the output ports towards the input ports in the buffered Banyan switch. Thus, cell loss does not occur within this switch; it can only occur at the inputs to the switch. (With an infinite input buffer size, no cell loss occurs.)

Jenq [14] analyzed a single–buffered binary Banyan switch ($k = 2$), where each input link of a switching element has a buffer size equal to one cell. He assumed buffers of infinite capacity on the input ports to the switch, and showed that the maximum throughput is approximately 0.45 for large values of N, an inadequate throughput value.

One approach to improve this limited throughput is to use input link buffers of larger capacity. Kim et. al. [16] analyzed a multiple–buffered binary Banyan switch constructed of binary switching elements with input link buffer space for more than one cell. They assumed that cells in an input link buffer are served on an FIFO basis. It was shown that the multiply buffered approach achieves a minor improvement in performance as compared to that of a single–buffered approach. This suggests that the performance bottleneck of the multiple–buffered switch lies, not in the limited buffer capacity, but in the contention among the head cells. This is referred to as head of the line blocking (*HOL* blocking) and describes the following phenomenon. When more than one cell contends for the same output link of a switching element (output contention), only one cell may pass through the switching element; the others are blocked and stored in the input link buffers. When the head of the line cell is blocked due to output contention, all the cells in the same input link buffer are blocked, if cells are served on an FIFO basis within the buffer. The *HOL* blocking will be discussed again in subsection 4.5 in more detail.

In the two papers discussed above [14, 16], the buffered Banyan switch is constructed of binary switching elements. In [3, 32], it was shown that the buffered Banyan switch constructed of switching elements of size greater than two results in a higher throughput. It is interesting to note that, in case of nonblocking switches, it has been shown that the maximum throughput decreases, as the size of switching elements increases [15]. For instance, a nonblocking switch constructed of 2×2 switching elements achieves the maximum throughput of 0.75, while the switch constructed of 4×4 switching elements results in smaller throughput of 0.68.

The performance of the buffered Banyan switch under non-uniform traffic has been analyized. Wu [35] investigated the effects of existence of a point–to–point connection on the performance of the single–buffered Banyan switch through simulation. (Refer to section 2 for the definition of a point–to–point connection.) Wu showed that the maximum throughput of the buffered Banyan switch decreases significantly, when there exists a point–to–point connection in addition to uniform traffic. Therefore, the buffered Banyan switch does not favor non–uniform traffic.

Kim et. al. [16] showed that, if all the inputs to the switch are of a point–to–point connection type, single–buffered and multiple–buffered Banyan switches result in almost the same throughput. They also showed that, for the mixture of a point–to–point connection and uniform traffic, the multiple–buffered Banyan switch results in throughput improvement of 10% to 15% over the single–buffered Banyan switch, depending on traffic load from a point–to–point connection. This improvement, however, becomes negligible as the size of the switch becomes large.

3.3.2 Buffered Banyan Switches with Bypass Queueing

In the above subsection, we observed that the *HOL* blocking can happen on the input link buffers of the switching element in the buffered Banyan switch. The *HOL* blocking occurs when the *HOL* cell is blocked due to output contention, and if cells are served on an FIFO basis within the input link buffer. This observation leads to the following technique called "bypass queueing" in order to improve the performance of the buffered Banyan switch.

Bubenik and Turner [3] proposed the bypass queueing discipline. In bypass queueing, when the *HOL* blocking occurs cells in that particular input buffer "bypass" the *HOL* cell and sequentially join competition for the available output links until some cell wins competition. This bypass queueing discipline is a variation of the window selection policy to be discussed in subsection 4.5.

Shiomoto et. al. [29] provided an exact analysis for a 2×2 switching element assuming bypass queueing and obtained its throughput–average delay performance. They showed that bypass queueing (on a 2×2 switching element) attains a maximum throughput close to 1.0. This is a significant increase from 0.75, the throughput of a 2×2 switching element with FIFO input link buffers. They also analyzed the Banyan switch comprised of binary switching elements assuming bypass queueing, and showed that the maximum throughput is achieved when the size of an input link buffer at a switching element is 8.

Bubenik et. al. [3] studied the multiple–buffered Banyan switch of size $N \times N$ constructed of binary switching elements. They showed, through simulation, that bypass queueing achieves the maximum throughput of 0.63, when N is large. They also showed that the maximum throughput of the multiply buffered Banyan switch constructed of 4×4 switching elements is nearly 0.75, if a bypass queueing discipline is assumed at input link buffers. The buffered Banyan switches constructed of switching elements of larger size achieves higher throughput.

4 Nonblocking ATM Switches

The Banyan switches that we studied in the previous section belong to the class of blocking switches, where the internal blocking of cells may occur when they contend for a common link within a switch. In this section, we focus on another major class of multistage interconnection networks, the class of nonblocking switches, where the internal blocking does not occur. The Batcher Banyan switch discussed briefly in subsection 3.2.2 is an example of the switches in this class. The crossbar switch is another example. In this section, we describe possible architectures for nonblocking switches, investigate their performance, and discuss various techniques to improve that performance.

4.1 Architectures for Nonblocking ATM Switches

There are two major architectures to implement nonblocking switches: crossbar switch based architecture and Banyan switch based architecture. We will review each switch architecture in detail in this subsection.

4.1.1 Crossbar Switches

The crossbar switch has N–inputs and N–outputs with a fully connected topology; that is, there are N^2 cross points within the switch. Since it is always possible to setup a connection between any arbitrary input and output pair, internal blocking will not happen in the crossbar switch. Note that, although blocking will not occur inside the switch, it may still occur on the outputs to the switch, when more than one cell is destined for the same output.

The architecture of the crossbar switch has some advantages. First, it uses a simple two–state cross–point switch (open and connected state) which is easy to implement. Also, the modularity of the switch design allows easy expansion. One can build a larger switch by simply adding more cross–point switches. Lastly, this switch design provides for a low latency as compared to the Banyan type switches, because it has the smallest number of connecting points between an arbitrary input and output pair. One disadvantage to this design, however, is the fact that it uses the maximum number of crosspoints (cross-point switches) needed to implement an $N \times N$ switch.

The Knockout Switch is a nonblocking switch based on the crossbar design [8, 36]. It has N inputs and N outputs and consists of a crossbar type (crossbarlike) switch with a bus interface module at each output (Fig.4). Since each input has a direct connection to each output, cells will not interfere with one another; internal blocking does not occur. However, since more than one cell may go to the same output port, output contention may result. The purpose of the bus interface is to resolve this conflict. The bus interface has three components. They are: the cell filter, the concentrator, and the shared buffer (Fig.5). The functions of these components are described below.

In the Knockout switch, each bus interface sees all cells from the input ports. The function of the cell filter is to separate cells destined for this particular output and those destined for other outputs in the switch. The cell filter does this by setting an activity bit in the cell's header to logical one, if that cell is destined for this output. The activity bit is set to logical zero otherwise.

At the beginning of each slot, all of the cell filters are initially open. That is, they will allow data bits from the input buses to pass into the concentrator. As the cell headers pass through these filters, there is a bit-by-bit comparison performed between the destination address (found in the cell header) and the particular output's address. When the filter finds a cell destined for another output, it sets the activity bit of that cell's header to logical zero, otherwise the activity bit is set to logical one. Thus, at the end of the slot, all cells destined for a particular output will be in the concentrator associated with that output.

The name "concentrator" comes from the function it performs. Specifically, the concentrator provides for an N to L concentration, where L is the number of separate buffers in the shared buffer. There are L outputs from the concentrator into the shared buffer. If some number of cells, k ($k \leq L$), arrive for a particular output, then they will appear on the outputs 1 to k of the concentrator. When k ($> L$) cells arrive, then $k - L$ cells will be dropped.

The work of the concentrator is essentially analogous to a tournament of N players competing for L prizes, where the prizes represent the output ports. Initially all of the cells are able to compete for the first of the L concentrator outputs, where there will be one winner. All of the losers ($N - 1$) then compete for the second output. This process, all of the losers of the previous stage competing for the concentrator output in this stage, continues L times. After the winner is selected in stage L, the remaining losers are dropped.

If k cells ($k \leq L$) are destined for a particular output, there will be $L-k$ cells which have an activity bit set to zero emerging from the concentrator. Otherwise, all of the packets emerging from the concentrator are destined for this particular output. The shared buffer then receives these L cells, but only stores those which have activity bit set to one circularly into L buffers. In this way the concentrator has reduced the size of the buffer space from a multiple of N down to a multiple of L. This reduces the overall switch complexity, thus decreasing the cost of fabrication.

The advantages to this design include a low latency and no internal blocking. These are due to the use of a crossbar like switch which provides the smallest number of connection points between an arbitrary input and output ports. In addition, this switch is self–routing. The bus interface allows only those packets which are destined for the particular output to be transmitted on that output trunk. Lastly, this switch was designed with modularity in mind so that it will allow for modular growth without complex changes.

4.1.2 Batcher Banyan Switches

In the previous subsection, we studied nonblocking switches based on the crossbar design. In this section, we discuss nonblocking switches based on the Banyan switch design.

In the Banyan switch, the internal blocking is possible. In the worst case, it is possible that $\sqrt{N}$ cells will contend for a center switch point, where N is the number of input ports. This happens when all of the incoming cells are in descending order relative to their destination address. Clearly, this is not a desirable characteristic for an ATM switch. However, the Banyan switch possesses an interesting feature. If all of the incoming cells are in ascending order relative to their destination addresses, the switch is guaranteed to be internally nonblocking. To guarantee that the cells be in ascending order requires only that there be some way to sort the incoming cells before they are fed into the Banyan switch. There are several implementations of sorting networks, of which the most frequently used is the Batcher network. This switch configuration, the Banyan switch preceded by the Batcher sorting network, is called the Batcher Banyan switch (Fig.6).

In the Batcher Banyan switch the incoming cells first enter a sorting network, which takes the cells and sorts them into ascending order according to their output addresses. In this way the cells can be sent to their destinations with no internal blocking.

The design of a Starlite switch [11] is based upon the Batcher Banyan switch. The input cells go into a sorting network, called the Sort–to–Destination network, where they are arranged in ascending order according to their destination addresses. The output of the Sort–to–Destination network then goes into a Banyan switch, called the Expander, where the relative ordering of the cells is changed to an absolute ordering (and cells are delivered to the output ports).

The Starlite switch can support multicast services, in addition to regular one–to–one communication. In order to yield multicasting capabilities, a copy network is placed between the inputs and the Batcher Banyan switch (Fig.7). The first stage of the copy network is a sort–to–copy network, which takes as input both source and copy cells. The output from the sort–to–copy network is then those cells ordered by their source addresses. Note that copy cells and source cells from the same input will be adjacent to one another. The copy network then copies the information field from the source cells into its corresponding copy cells. These cells are then sent into the Batcher Banyan switch.

Regardless of whether it is a crossbar based or a Batcher Banyan based switch, output contention still occurs in a nonblocking switch, when more than one cell is destined for the same output. When this happens, cells which lose contention are dropped from a switch, if the switch does not have any buffering discipline. In order to minimize the cell loss, buffering of cells is necessary. Buffers may be placed on the inputs to the switch, or on the outputs to the switch, or possibly on both. Queueing of cells may be implemented in a shared buffer common to all the input/output ports. In the following, we first study the performance of nonblocking switches without buffers. We then investigate various buffering schemes and techniques to improve the performance of nonblocking switches.

4.2 Nonblocking Switches Without Input Buffers

In this subsection, we study the performance of the simplest nonblocking switch, a nonblocking switch without any buffers. In this case, when output contention happens, only one cell is successfully transferred to the destination output port, and the remaining cells are dropped from the switch.

In [26], Patel analyzed an $N \times M$ nonblocking switch (the cross–bar switch) in the context of interconnecting multiprocessors, and obtained the cell loss probability for both of N = finite and $N = \infty$ cases. It is assumed that the speed of a switch is equal to that of the input channels; a switch transfers at most one cell per slot from each of the N input ports. Patel showed that, for the case of $N = M = \infty$, the probability $p(success)$ that a cell wins an output contention is given by $p(success) = \frac{1-e^{-p}}{p}$, where p is the input traffic load. (See section 2 for the definition of p.) The numerator $1 - e^{-p}$ is the throughput of the switch. The cell loss probability is given by $p(loss) = 1 - p(success)$. The throughput takes its maximum when p is 1 (and $N = \infty$), and its value is $1 - e^{-1} = 0.632$. It is noteworthy that the maximum throughput of 0.632 is achieved at the large expense of the cell loss at inputs; 36.8% of incoming cells are dropped when p is one. This level of cell loss is not acceptable for ATM networks.

4.3 Nonblocking Switches With FIFO Input Buffers

In nonblocking switches without buffers, when output contention happens, cells which lose contention are lost from the switch. This limits the throughput of the switch to 0.632 as we saw in the previous subsection. In this section, we study nonblocking switches with buffers on the input ports (referred to as input buffers) (Fig.8). Cells which lose contention, as well as the newly arrived cells, are stored in input buffers. We assume FIFO discipline to serve cells within an input buffer. Further, in this subsection, we assume that the speed of a switch is equal to that of the input and output channels; a switch transfers at most one cell per slot from each of the N input ports. Note that, when the switch operates at the same speed as the input and output channels, buffering of cells is needed only on the inputs. No buffering is necessary on the outputs. This is because a maximum of one cell per output port gets through the switch, and this throughput matches the speed of the output channel.

An example of a control mechanism for nonblocking switches with FIFO input buffers is the three–phase algorithm proposed by Hui et. al. [12]. This algorithm is used in conjunction with the Batcher Banyan switch to resolve output contention. In the first phase of the algorithm, an input port with cells to transmit sends a request cell with the source and the destination addresses in the cell header. These transmission requests are then fed into the Batcher network where they are sorted. After this sorting, cells which will contend for common destinations are adjacent to one another. Of these contending cells only one of them will be selected.

In phase two, the selected requests will be acknowledged. This is accomplished by sending the winning requests back into the Batcher Banyan switch with the source and destination address fields exchanged. This serves to route the request cell back to the input port which initially sent it. When an input port receives this acknowledgement it has been given the "go ahead" to transmit the cell. Finally, in phase three, transmission of the cell is done. One of the important characteristics of the three–phase algorithm is that cells within an input buffer are sent on an FIFO basis, and thus, cell sequence is retained.

In the following, we consider the performance of nonblocking switches with FIFO input buffers. The following performance discussions apply for both crossbar based and Batcher Banyan based switches. It is assumed in this subsection that, in case of output contention, one of the contending cells is randomly selected and sent to the output port (random selection policy).

The performance of nonblocking switches with FIFO input buffers has been analyzed in [12, 15] under the assumptions described above. For the case of $N = \infty$, Hui and Arthurs [12] showed that the maximum throughput of a switch is 0.586. They also obtained an upper bound on the cell loss probability at input buffers by truncating the queue length distribution for an infinite buffer size. When a buffer for B_{input} cells is placed at each input, the cell loss probability $p(loss)$ is upper bounded by

$$p(loss) < \frac{p(2-p)}{2(1-p)}\left(\frac{p^2}{2(1-p)^2}\right)^{B_{input}}. \qquad (2)$$

We can see from the above equation that the nonblocking switch with FIFO input buffers, for instance, achieves a cell loss probability of less than 10^{-6} at $p = 0.5$ when $B_{input} \geq 20$.

Karol et. al. [15] have obtained the maximum throughput of the switch for both finite N and infinite N, as well as the average delay time for infinite N. Tab.1 is from [15] and shows how the maximum throughput decreases as the number of inputs N increases. It is seen that, as N increases, the maximum throughput decreases to 0.586. Note that the throughput 0.75 for $N = 2$ gives the throughput of a 2×2 switching element, a basic building block of the Banyan switch.

The maximum throughput (0.568) of the nonblocking switch with FIFO input buffers is smaller than that of the nonblocking switch without buffers (0.632). This is due to the head of the line cell blocking (*HOL* blocking). When the head of the line cell (*HOL* cell) is blocked due to output contention, all the cells in the same input buffer are blocked, if cells are served on an FIFO basis within the buffer. This *HOL* blocking severely limits the maximum throughput of the nonblocking switch with input buffers, resulting in a lower throughput than that of the nonblocking switch without buffers.

In order to improve the limited throughput of the nonblocking switch discussed above, a number of improvement techniques has been proposed and investigated. One possible approach is to speed up the switching fabric. Effect of speed up on the performance of a switch is discussed in subsection 4.4.

Another possible approach to improve the performance of the nonblocking switch is to reduce or eliminate the *HOL* blocking. When the *HOL* cell is blocked due to output contention, a cell behind it going to an available output port can be sent instead. This reduces the *HOL* blocking and results in a better throughput performance. Subsection 4.5 explains the window selection discipline, where one of the first w cells in an input buffer is selected and sent prior to the *HOL* cell. Use of a shared buffer also improves the throughput of a switch by eliminating the *HOL* blocking. In the shared buffer switch there is no buffer on the inputs, nor on the outputs. Arriving cells are immediately injected into the switch. When output contention happens, a winning cell goes through a switch, and the losers are stored in a shared buffer common to all of the input ports for later transmission. Since cells which lose output contention are stored in a shared buffer, the queue structure is not retained. Newly arriving cells immediately join in the competition for available outputs. In addition, cells in the shared buffer also have access to the switch. Since more cells are available to select from, it is possible that less outputs will be idle in the shared buffer scheme. Thus, the throughput of the shared buffer switch is slightly better than that of the switch with the window selection discipline. In subsection 4.6, the switch with a shared buffer and its performance are discussed.

The third possible approach to improve the performance of the nonblocking switch is to optimally select one cell among the contending cells and transfer it to the output, instead of selecting a cell randomly. There are a number of possible selection policies. For instance, selecting a cell from the longest queue may improve the switch performance. In subsection 4.7, we discuss the longest queue selection and the priority selection schemes.

A fourth possible technique for improving performance of the nonblocking switch is to construct a switch consisting of multiple switch planes (a parallel switch). Parallel switches are discussed in subsection 4.8. Other related research is summerized in subsection 4.9. Subsection 4.10 summerizes and compares the performance of a variety of nonblocking switches.

4.4 Speedup of Nonblocking Switches With FIFO Input Buffers

In the preceding subsections, we assumed that, when output contention happens, only one of the contending cells is accepted at the output. This section considers the case where up to L ($1 < L \leq N$) contending cells are accepted at an output port per slot. (N is the number of input ports.) L is referred to as the speedup ratio of the switch in the following. This switching speed of cells may be realized by, for instance, the Batcher Banyan switch operating L times faster than the input/output channels. The Knockout switch with an "N to L concentrator" at each output port has the same switching speed [36]. The ATOM switch speeds up the cell switching speed by transmitting arriving bits in parallel through the switch [31]. A number of papers has studied the effect of increasing the cell switching speed on the switch performance. Tab.2 summerizes the mathematical models and the performance measures obtained in the past research. In what follows, we discuss the major results of the past research.

We first consider the ideal case where the switch can transfer the maximum of N cells per slot to an output (i.e., $L = N$). In this case, buffering of cells is not necessary on inputs, because the switch operates fast enough to handle all the cells, even when the cells from all the input ports (N of them) are going to the same output. Queueing of cells occurs only on the outputs (Fig.9). A switch only with output buffers are referred to as the output buffered switch [15, 31].

The performance of the output buffered switch has been investigated. The average delay is analyzed in [15] for infinite buffer case, and the cell loss probability at output buffers is obtained in [10] for finite buffer case. In [15], Karol et. al. assumed uniform traffic—an incoming cell goes to a particular output with the probability $\frac{1}{N}$. Thus, the probability of having k cell arrivals (aggregated from all the inputs) in a slot at a particular output buffer becomes $a_k = \binom{N}{k}(\frac{p}{N})^k(1-\frac{p}{N})^{N-k}$. By observing the number of cells in an output buffer at each slot, they derived the z–transform for the queue length distribution in an output buffer as

$$Q(z) = \frac{(1-p)(1-z)}{A(z)-z}, \qquad (3)$$

where $A(z) = \sum_{k=0}^{\infty} a_k z^k$ is the z–transform for the number of cell arrivals. When N approaches infinity, a_k approaches $\frac{p^k}{k!}e^{-p}$, making the the number of cell arrivals at an output buffer a Poisson process. Thus, for $N = \infty$, the z–transform for the queue length distribution on an output port in the steady state becomes

$$Q(z) = \frac{(1-p)(1-z)}{e^{-p(1-z)} - z}. \tag{4}$$

Interestingly, this $Q(z)$ is same as the z-transform for the queue length distribution in the M/D/1 queue. It is clear that the switch attains the maximum throughput of 1.0, if the speedup ratio of a switch is N (i.e., if the switch transfers the maximum of N cells to a particular output in a slot).

As we saw in the above, the speedup of $L = N$ achieves the highest possible maximum throughput of 1.0. However, it is very difficult and costly to build a very high speed switch of large size due to hardware limitations. Thus, the case of $L < N$ becomes of practical importance, when N is large. When the speedup ratio L is less than N, if k ($> L$) cells are destined for the same output, $k - L$ cells are blocked at inputs. Therefore, queueing occurs not only on the output ports, but also on the input ports. (B_{input} and B_{output} denote the size of an input and an output buffers, respectively.) See Fig.10 for a switch with both input and output buffers. The performance of switches when the speedup ratio is less than N has been analyzed in [36, 24, 25, 8].

Yeh et. al. [36] and Oie et. al. [24, 25] analyzed the cell loss probability on input buffers, assuming infinite capacity buffers on the outputs ($B_{output} = \infty$). In [36], no buffering is assumed to be on the inputs ($B_{input} = 0$), and in [25], infinite buffer capacity is assumed on the inputs ($B_{input} = \infty$). Yeh et. al. [36] obtained a cell loss probability at an input port as follows:

$$p(loss) = \begin{cases} \frac{1}{p}\sum_{k=L+1}^{N}(k-L)a_k & (N < \infty) \\ (1-\frac{L}{p})(1-\sum_{k=0}^{L}\frac{p^k}{k!}e^{-p}) + \frac{p^L}{L!}e^{-p} & (N = \infty). \end{cases} \tag{5}$$

Their analysis showed that a small speedup ratio can achieve a cell loss probability nearly equal to zero; that is, a speedup ratio of $L = N$ is not required to achieve the very small cell loss probability. For example, a speedup of $L = 8$ is sufficient enough to achieve the cell loss probability of less than 10^{-6}, when the input traffic load is 0.9 ($p = 0.9$) and N is infinity.

Oie et. al. [24, 25] analyzed a nonblocking switch, assuming $N = \infty$ and $B_{input} = B_{output} = \infty$, and obtained the maximum throughput as a function of the speedup ratio L. Tab.3 shows the values of the maximum throughput for various values of L from [24]. They also obtained an upper bound on the cell loss probability by truncating the tail of the queue size distribution for the case of $B_{output} = \infty$. Fig.11 shows the upper bound on the cell loss probability at the input buffers as a function of the buffer size for various values of L and p (input traffic load). By comparing their results with Yeh's [36], they showed that a slower switch with input buffers can achieve as good of a performance as a faster switch without input buffers. For instance, Fig.11 shows that, if there is enough buffer space for 9 cells at each input port, a switch with the speedup ratio of 3 achieves a cell loss probability of less than 10^{-6} at the input traffic load of 0.9, while the speedup of $L = 8$ is necessary to achieve the same cell loss rate if a switch does not have input buffers, as we saw in the previous paragraph [36]. They also obtained the cell loss probability at output buffers, assuming a finite buffer space on the outputs. Fig.12 shows the cell loss probability on outputs as a function of B_{output} for various values of L. Both N and B_{input} are assumed to be infinite, and the input traffic load p is 0.9. This figure shows the capacity of an output buffer necessary to satisfy a cell loss requirement. For instance, the capacity of an output buffer required to keep the cell loss probability less than 10^{-6} is 55, if the speedup ratio L is N. For the smaller speedup factors of $L = 3$ and $L = 4$, the buffer capacity required is 50 and 54, respectively, to satisfy the same cell loss requirement. There is not a very large difference among the values of the required buffer capacity for different speedup factors.

Eng [8] introduced 2-level priority classes among cells and analyzed the cell loss probability, assuming a single buffer at the input ports (i.e., $B_{input} = 1$). In his switch, when the speedup ratio L is 4, the cell loss probability is less than 10^{-6} even at a high input traffic load of $p = 0.9$. On the other hand, as Fig.11 shows, in non-priority switches, a buffer for 4 cells is required at an input port in order to satisfy the same cell loss requirement. This priority scheme is discussed again in subsection 4.7 in more details.

The effects of speeding up switch fabrics on switch performance have been studied in other papers. Tab.2 summerizes the models analyzed and the performance measures obtained in the past work.

4.5 Nonblocking Switches with Non–FIFO input buffers

As explained in subsection 4.3, serving cells in an input buffer based on an FIFO discipline leads to *HOL* blocking, limiting the maximum throughput to 0.586, when the speed of a switch is the same as that of the input channels. In this subsection, we discuss an alternative service discipline, the window selection discipline, for serving cells in an input buffer, and investigate its performance. In the window selection discipline, when an output contention occurs, the first w cells in each of the input buffers involved in the contention sequentially contend for access to the switch outputs. The cells at the head of the input buffers contend first for access to the switch outputs. The inputs which still contend then compete with the next cell in each contending queue for access to any remaining idle outputs (i.e., outputs not yet assigned to receive cells). This contention process repeats up to w times at the beginning of each time slot, sequentially allowing the w cells in an input buffer's "window" to contend for any remaining idle outputs, until the input is selected to transmit a cell. Hence, w is referred to as the window size. Different names have been used for this service discipline in different papers. For instance, it is called "priority scheme" in

[12], and "window policy" in [15]. In [3], "bypass queueing discipline" is used to describe the window selection discipline in the context of blocking switches. In this paper, we use "window selection discipline".

Hui et. al. [12] proposed a priority scheme to implement the window selection discipline on the nonblocking switch with FIFO input buffers (i.e., the Batcher Banyan switch). Their priority scheme allows the first w cells in each input buffer to sequentially contend for the idle switch outputs at the beginning of each slot until a cell wins an output contention. Once a cell wins this output contention, it is given priority and no other cells will be assigned to the same output.

Oki Electric Industry Company [20] implements the window selection discipline on the nonblocking switch with FIFO input buffers. The switch has a "Nemawashi" ("negotiation" in Japanese) network followed by a nonblocking switch (i.e., the Batcher Omega switch). The Nemawashi network choses at most one cell from each input buffer in such a way that the selected cells do not cause any output contention in the Batcher Omega Switch. This Nemawashi network can be implemented using the priority scheme proposed in by Hui et. al. in [12]. Masaki et. al. [20] show simulation results on the throughput–average delay performance of the Nemawashi switch. They assumed a 32×32 switch with the window size of 7 ($N = 32$ and $w = 7$) and showed that the maximum throughput increases to approximately 0.9 from 0.586 (the maximum throughput of the nonblocking switch with FIFO input buffer).

Performance study on the nonblocking switch with the window selection discipline is found in [29] and [10]. The throughput–average delay performance of a binary switch ($N = 2$) with the window selection discipline is obtained through an exact analysis in [29]. In [10], Hluchyj et. al. present simulation results on the maximum throughput of a nonblocking switch assuming the window selection discipline. Tab.4 shows the maximum throughput values of a nonblocking switch with the window selection discipline from [10]. This table shows that the window selection discipline is most effective when N is small and w is large. For instance, the maximum throughput is 0.96, very close to 1.0, when $N = 2$ and $w = 8$. Even when N is large, the window selection discipline achieves high throughput. For instance, when $N = 128$ and $w = 8$, the maximum throughput is 0.88. This throughput value is significantly higher than 0.586, the maximum throughput of the nonblocking switch with FIFO discipline. (See subsection 4.3.) However, it should be noted that this window selection discipline can not achieve the throughput of 1.0, even when $N = \infty$ and $w = \infty$ [10]. This is because that the window selection discipline limits each input to send at most one cell into the switch fabric per slot, and as a result, prevents the maximum throughput from reaching 1.0.

Lastly, we note that parallel buffers at the inputs can implement the window selection discipline at the expense of additional control hardware. Fitzpatrick et. al. [9] proposed an $N \times N$ switch where each input port has N buffers in parallel (Fig.13). Buffer i at an input port stores cells going to the output port i ($1 \leq i \leq N$). A controller selects at most one cell per input port, a total of N or less cells from all the input ports, in such a way that no two cells are going to the same output port. This is equivalent to the window selection discipline where the window size w is infinite.

4.6 Nonblocking Switches with a Shared Buffer

So far, we have studied nonblocking switches where each input and/or output port(s) has a dedicated buffer, and surveyed various techniques to improve the performance of the switch. In this section, we investigate a nonblocking switch with a shared buffer. Use of a shared buffer can eliminate the *HOL* blocking and improve the performance of the nonblocking switch.

In the shared buffer switch, there is no buffering done on the inputs, nor on the outputs (Fig.14). Newly arriving cells are immediately injected to the switch. When output contention happens, a winning cell goes through a switch, and the losers are stored in a shared buffer for reentry. Since cells which lost output contention are stored in a shared buffer and a queue is not formed on the inputs, newly arriving cells immediately join in the competition. Because of this, there is no *HOL* blocking. In addition to newly arriving cells, those in the shared buffer can also access a switch, and thus, the throughput of the shared buffer switch is slightly better than that of the switch with the window selection discipline.

One example of the shared buffer switch architecture is the Starlite switch with trap (Fig.15). Refer to subsection 4.1.2 for the description of the Starlite switch (without trap). A module called the trap is added into the Starlite switch configuration. The trap module is placed between the sort–to–destination network and the expander. Its responsibility is to check adjacent cells and determine if there will be any output contention. When contention occurs, one cell will be selected as the winner and will be put into the expander, while the rest of the cells destined for that output will be stored in a shared buffer and then re–routed into the sort–to–destination network. (Note that new input ports are added internally to the switch to be able to re-route cells from the shared buffer, making the switch bigger, as shown in Fig.15.) In a selection of a winner in the trap, priority is not given to older cells, and thus, cells may be delivered out of sequence. (Note that the out of sequence problem does not arise in the three–phase algorithm, since cells within an input buffer are sent on an FIFO basis.)

The performance of the Starlite switch with trap [11] has been analyzed in [7, 10]. In the following, we assume an $N \times N$ Starlite switch with trap with a shared buffer of capacity $N \times B$ (cells). B could be interpreted as the capacity of the shared buffer per input (or per output). $N \times B$ input ports are added internally to the switch for the reentry of cells from the shared buffer, and the same number of output ports are added internally to the switch to store cells which lose output contention. Thus, an $N \times N$ Starlite switch with trap internally consists of an $N(B+1) \times N(B+1)$ switch fabric. The model used to analyze the performance of the Starlite switch with trap (i.e., the nonblocking switch with a shared buffer) is depicted in Fig.14.

Eckberg et. al. [7] studied the Starlite switch with trap and developed an approximate analysis to obtain the cell loss probability at a shared buffer, assuming that the queue length in the shared buffer follows a Gamma distribution. They obtained the capacity of the shared buffer required to satisfy a given cell loss requirement as a function of N (the number of input ports) and p (input traffic load). Furthermore, it is shown that, as N approaches infinity, the value of B (capacity of the shared buffer per output) required to satisfy a cell loss requirement of practical interests approaches its lower bound

$$f(p) = \frac{p^2}{2(1-p)}. \tag{6}$$

As pointed out in [10], this lower bound is same as the average queue length in the M/D/1 system.

Hluchyj et. al., in their study of the performance of the Starlite switch with trap [10], used the N fold convolution of an M/D/1 queue length to approximate the steady state distribution for the queue length of the shared buffer. They showed that the lower bound $f(p)$ on the buffer size required to satisfy a given cell loss requirement is given by the average queue length in the M/D/1 system, confirming the results obtained in [7]. It is also shown that the Starlite switch with a large shared buffer attains the maximum throughput of one.

As we saw in subsection 4.4, a nonblocking switch with the speedup ratio of N (referred to as the output buffered switch in the following) can also achieve the throughput of one. However, the buffer space required in the shared buffer switch to attain the throughput of one is much less than that needed in an output buffered switch. For example, as we saw in subsection 4.4, to satisfy a cell loss probability of less than 10^{-6} at the input traffic load of $p = 0.9$ in the output buffered switch, it is required to have a buffer for 55 cells at each output port. On the other hand, for the shared buffer switch, eq.(6) gives the lower bound on B (the buffer capacity required per output) to satisfy the same cell loss requirement, and the value of B is 5 (cells). The shared buffer switch only needs enough buffer space for 5 cells per output, as opposed to a buffer space for 55 cells per output in the output buffered switch. This decrease in the required buffer capacity is at the expense of an increase in the number of input and output ports internal to the switch. An $N \times N$ switch with a shared buffer of size $B = 5$ internally consists of a $6N \times 6N$ switch, thus, the number of input ports and output ports are six times as many as those in an output buffered switch. This increase in the number of input and output ports is one of the drawbacks of this switch. Another drawback of the shared buffer switch is that cells may be delivered out of sequence because newer cells may win an output contention [12].

4.7 Selection policy for contending cells

When output contention happens, one of the contending cells is selected and transferred to its destination output port. In all of the performance studies mentioned so far in this paper (except in [8]), it is assumed that one cell is randomly selected for transmission. However, random selection of a cell may not always be the most desirable policy. For instance, if real–time cells and non–real–time cells contend for the same output, selecting one of the real–time cells may be more desirable. Eng [8] and Chen et. al. [4] investigated the priority selection policy. Another possible selection policy is to choose a cell from the input buffer which has the longest queue [15]. In this subsection, we discuss these alternative selection policies.

As we briefly discussed in subsection 4.4, Eng proposed a priority selection in the (photonic) Knockout switch [8]. In his scheme, 2 priority levels are assumed; new cells are given the lower priority, and previously blocked cells are given the higher priority. When an output contention occurs, a cell from the higher priority class is selected and transferred to the destination output. Cells which lose contention are stored in the input buffers and join in a contention for available outputs in the next slot. If a cell loses contention twice, it is dropped from the input buffer. Thus, buffer space for one cell is all that is needed at each input, when this priority selection is assumed.

Eng showed that Knockout switch with priority selection can achieve a cell loss probability of less than 10^{-6}, when the input traffic load is 0.9 and the speedup ratio of a switch fabric is 4 (i.e., $p = 0.9$ and $L = 4$) [8]. On the contrary, under the same set of parameter values, a buffer for 4 cells is required at each input to satisfy the same cell loss requirement, if we assume a random selection in the Knockout switch (as shown in Fig.11).

Eng [8] gives a shared buffer switch architecture, where he assumes the priority selection and the speedup of the switch fabric. His $N \times N$ switch internally consists of a $(1+\frac{1}{2})N \times (1+\frac{1}{2})N$ switch, where $\frac{1}{2}N$ inputs are exclusively used for reentry of cells which lost output contention (as in the shared buffer switch). In case of output contention, old cells are given priority (priority selection). He showed that this switch guarantees a cell loss probability of less than 10^{-6} at an input traffic load of $p = 0.9$, if the speedup ratio of the switch fabric is 4. In this switch, cells are delivered in sequence because of the priority selection. Further, he showed that, if the size of the shared buffer is $2N$, no cell loss due to overflow occurs in the shared buffer, since cells are only allowed to retry once. Cells which failed at the first retry are dropped, but the resulting cell loss is small, as we saw in the above paragraph.

Chen et. al. [4] proposed another type of a priority selection. They assumed two types of traffic, real time traffic and data traffic. When output contention occurs, priority is given to cells from real time traffic. If there is more than one cell from the same priority level contending for the same output, one is selected randomly. In addition, they also assumed a priority selection of a cell within an input buffer. If cells from both real time traffic and data traffic exist in the same input buffer, cells from the real time traffic are sent first. Among the cells at the same priority level, FIFO is assumed at an input buffer.

Chen et. al [4] assumed a nonblocking switch with the speedup ratio of 1 ($L = 1$) and analyzed the performance of the switch assuming the priority selection policy described above. Uniform traffic is assumed in the analysis. They obtained the maximum throughput, the cell loss probability and the average delay time. The maximum throughput is given by

$$S = \lambda \quad {}_{x}(\lambda_H) + \lambda_H, \qquad (0 \leq \lambda_H \leq 0.586) \tag{7}$$

where $\lambda_{max}(\lambda_H)$ is the maximum allowed arrival rate of the low priority cells for a given value of λ_H. $\lambda_{max}(\lambda_H)$ is given by the following:

$$\lambda_{max}(\lambda_H) = \frac{(\lambda_H^2 - 6\lambda_H + 4) - \sqrt{-3\lambda_H^4 + 12\lambda_H^3 - 16\lambda_H + 8}}{2(1 - \lambda_H)}. \tag{8}$$

From eqs.(7) and (8), the maximum value of S is obtained as 0.6063 when λ_H is 0.447. This throughput value is larger than 0.586, the maximum throughput of the nonblocking switch with FIFO input buffers when priority selection is not assumed.

In [15], the longest queue selection policy was introduced. Under this policy, when output contention happens, a cell is taken from the queue which has the longest length, and sent to its destination output. Simulation results show that this policy offers smaller delay time than with the random selection policy [15].

4.8 Parallel Switches

In the previous subsections, we first observed that the maximum throughput of a nonblocking switch is 0.586, if the speedup ratio of the switch fabric is 1, and if cells are served on an FIFO basis within each input buffer. We, then, discussed three major classes of techniques to improve the performance of the nonblocking switch; speedup of the switch fabric, techniques to reduce or eliminate the *HOL* blocking, and policies to select a cell from those contending for the same output. As the speed of the switch fabric increases, so does the throughput. By speeding up the switch fabric N times faster, the maximum throughput of one can be achieved. It is, however, difficult to implement a large size switch operating at very high speeds due to the limitations of current hardware technology. The window selection discipline at input buffers reduces the *HOL* blocking and increases the throughput of the nonblocking switch up to 0.88 at the expense of cell scheduling overhead (hardware) on input buffers. Using a shared buffer eliminates the *HOL* blocking and achieves the maximum throughput of one, when the size of a shared buffer is infinitely large. However, as the size of the shared buffer grows, so does the size of the switch. This is because the shared buffer switch is internally implemented using a $(B+1)N \times (B+1)N$ switch, where B is the size of a shared buffer. Again, limitations of current hardware technology put a cap on the size of the switch. By selecting a cell from those which have a higher priority in case of output contention (priority selection), throughput can be improved up to 0.606; not a great improvement.

The above techniques all try to improve the performance of a single switch plane. Another type of improvement technique exists: the parallel switch. The parallel switch uses, not just one, but some number of switch planes and achieves better performance than the single plane switch. It can be easily seen that using two nonblocking switch planes together will achieve a maximum throughput of 1.0.

Several papers [33, 13, 22] investigate a switch fabric consisting of multiple switch planes in parallel. Beside the obvious advantage of increased reliability, this switch architecture has the following advantages [13]. First, it serves as a method to match the speed of the incoming link when the switch fabric cannot run at the same speed as the incoming link. Second, the throughput of the switch increases to the sum of the throughput of each switch plane.

One possible architecture for the parallel switch is the following. Let's assume that a switch consists of K identical switch planes. Each switch plane has its own input buffers but shares output buffers with other planes (Fig.16). In other words, input port i $(1 \leq i \leq N)$ of switch plane j $(1 \leq j \leq K)$ has its own input buffer. Output port i $(1 \leq i \leq N)$ of switch plane j $(1 \leq j \leq K)$ shares a common output buffer with the other switch planes. Cells arriving from input channel i of the parallel switch are randomly dispatched to one of the switch planes, stored in its input buffer, and sent to their destination output ports. Note that this is not the only possible switch architecture. For instance, switch planes may share input buffers, instead of each having its own dedicated buffer.

To illustrate the performance of the parallel switch, we show some numerical examples below. Let's assume that the switch planes are nonblocking, and that their speedup ratio is L. Assume that the capacity of an input buffer is zero (i.e., no input buffers) and that the capacity of an output buffer is infinite. We consider the limiting case where N is ∞ and show the number of switch planes K^* necessary to satisfy the cell loss requirement of less than 10^{-6} at the input traffic load of $p = 0.9$ for a given value of L. By solving the equation $p(loss) = 10^{-6}$ with respect to p, we can obtain the maximum input load p^* not to violate the cell loss requirement. ($p(loss)$ is given by eq.(5) in subsection 4.4.) By dividing the input traffic load under consideration ($p = 0.9$) by the value of p^*, we can obtain the number of switch planes needed to satisfy the cell loss requirement at the input traffic load of $p = 0.9$. Tab.5 shows the values of p^* and K^* for various values of L. If L is 1, we need an extraordinary number of switch planes (450,000 switch planes). A moderate speedup of switch planes results in a more realistic number of switch planes. For instance, a parallel switch consisting of nine switch planes, each with $L = 4$, attains the loss probability of less than 10^{-6} at $p = 0.9$ without input buffers. If L is 8, a single (nonblocking) switch plane can achieve a cell loss probability less than 10^{-6} at $p = 0.9$, again, without input buffers.

In the switch architecture shown in Fig.16, it is assumed that each switch plane has dedicated input buffers, and incoming cells are randomly assigned to one of the switch planes. One of the disadvantages of this switch architecture is that cells may be delivered out of sequence to output ports. This is because different cells from the same input stream may be assigned to different switch planes. One possible approach to solve the out of sequence problem is to equip common input buffers shared by all the switch planes, and send cells in an input buffer on an FIFO basis (Fig.17). If some number of calls are multiplexed onto one input port it is also possible to assign, not an individual cell, but a call, to a switch plane. All the cells belonging to the same call take the same switch plane, and therefore, cells will be delivered in sequence to their destinations.

4.9 Related Research on Nonblocking Switches

In this section, we consider two research topics so far not addressed in this paper: performance analysis assuming non-uniform input traffic, nonblocking switches with parallel input buffers and parallel service, and multicast switches.

4.9.1 Performance Analysis of Nonblocking Switches Under Non-Uniform Traffic

Yoon et. al. [37] analyized the performance of the Knockout switch assuming the existence of a hot spot. (A hot spot refers to an output port where heavy concentration of cells is expected to happen.) In their model, a fraction h of the incoming cells go to a hot spot, and the rest of the cells are uniformly destined to the N output ports. With this hot spot traffic model, the arrival rate of cells going to a hot spot becomes $hp + \frac{(1-h)p}{N}$ (cells per slot, per an input port). Thus, the probability that k cells arrive at the hot spot (from all the input ports) is given by

$$P_k = \binom{N}{k}(hp + \frac{(1-h)p}{N})^k(1 - hp - \frac{(1-h)p}{N})^{N-k}. \tag{9}$$

Using the above P_k, the cell loss probability is given by

$$p(loss) = \frac{1}{p}\sum_{k=L+1}^{N}(k-L)P_k. \tag{10}$$

From this equation, it can be shown that, in the limiting case of $N = \infty$, the speedup ratio L has to be at least 20 to achieve a cell loss probability of less than 10^{-6}, when the input traffic load p is 0.9 and h is 0.005, a fairly small value of h. On the other hand, as we saw in subsection 4.4, if the traffic is uniform, $L = 8$ is sufficient enough to achieve the same loss probability. We can see that existence of a hot spot significantly reduces the performance of a switch.

In [37], the performance of the Knockout switch is also analyzed assuming the existence of a point-to-point connection (different type of non-uniform traffic). Refer to section 2 for the definition of a point-to-point connection. Assuming that there is a point-to-point connection and that the rest of the inputs to the switch follows a uniform traffic model, the cell loss probability is obtained for the case of $N = \infty$ as follows.

$$p(loss) = (2 - \frac{L}{p})(1 - \sum_{k=0}^{L-1} \frac{p^k e^{-p}}{k!}) + \frac{p^{L-1} e^{-p}}{(L-1)!}. \tag{11}$$

It is shown from the above equation that $L = 9$ is sufficient to achieve the cell loss probability of less than 10^{-6} at $p = 0.9$. Considering the fact that the speedup ratio of 8 is required on the Knockout switch (without input buffer) to satisfy the same cell loss requirement for the uniform traffic case, existence of a point-to-point connection may not significantly reduce the performance of the switch. On the contrary, it is shown in [35] that the degradation in the performance of the Banyan switch due to a point-to-point connection is significant.

Other research on the performance of the nonblocking switch under the non-uniform traffic is found in [19], where the switch has buffers only on inputs and its speedup ratio is assumed to be 1.

4.9.2 Nonblocking Switches with Parallel Input Buffers and Parallel Service

In this section, we consider a switch which allows up to a number B of cells per input to compete for the outputs in parallel. The architecture of this switch is shown in Fig.18. This switch has B buffers in parallel on each input channel. The capacity of a parallel buffer is assumed to be one. Each of the B buffers is connected to the switch, and thus, the $N \times N$ switch internally is of size $NB \times NB$. In every B slots, all the cells at the head of these B parallel buffers are injected into the switch at a time. An output can accept at most B cells in a slot, and thus, if k ($> B$) cells are destined for an output, only B of them are accepted and the rest ($k - B$ cells) are lost.

Hluchyj et. al. [10] analyzed this switch architecture and obtained the cell loss probability and the average delay time. For the case of $N = \infty$, the cell loss probability is given by

$$p(loss) = 1 - \frac{1}{p} + \frac{e^{-Bp}}{Bp} \sum_{k=0}^{B-1} (B - k) \frac{(Bp)^k}{k!}. \tag{12}$$

It was shown in [10] that a very large number of buffers are needed to attain a small cell loss probability. For example, B must be greater than 100 to achieve the cell loss probability of less than 10^{-3} at the input traffic load of $p = 0.85$. Thus, the switch for the practical applications would have a large size. Another drawback of this switch architecture is that cells can be delivered out of sequence. Because an output can only accept B cells, when more than B cells are destined for an output, there is no guarantee which cells are transmitted. This leads to an out of sequence problem.

This switch architecture, used in conjunction with some of the following improvement techniques, may result in switch of more practical value. One may increase the switching speed of the fabric. One may increase the capacity of a parallel buffer so that the cells which lose an output contention are stored and retransmitted later.

4.9.3 Multicast Switches

A generally agreed upon feature of future high-performance networks is the ability to set up one-to-many or many-to-many connections for such applications as teleconferencing, commercial television, and multi-way telephone conversations. A key element of the design of such a system is the multicast switch module, which is responsible for duplicating incoming cells and forwarding them to every output port which belongs to the multipoint connection.

As we saw in subsection 4.1.2, the Starlite switch has multicast capability. Cells are duplicated by a copy network placed between inputs and the Batcher Banyan switch and multicast to the destination output ports (Fig.7).

The Broadcast Packet Switch proposed by Turner [33, 3] is another example of multicast switches. Fig.19 shows the design of a Broadcast Packet Switch. This switch fabric composed of a series of major components: a Copy Network, Broadcast and Group Translators (BGT), a Distribution Network, and a Routing Network. The Routing Network is a self-routing, binary switching network (Banyan network) with buffers at each input port capable of holding two complete cells. Blocking on the Routing Network is reduced by the Distribution Network. The Distribution Network evenly distributes all cells it receives across all its outputs breaking up any "communities of interest" that may exist. The Copy Network and Broadcast and Group Translators are included to accommodate multi-point connections throughout the network.

An alternative Turner's copy network has been proposed by Lee [18]. Lee proposes a non-blocking copy network consisting of a running adder network, a set of dummy address encoders, a concentrator network, and a broadcast Banyan network.

4.10 Summary of the Performance of Nonblocking Switches

Tab.6 summerizes the past research on the performance of the nonblocking switches. In this table, switches are classified according to the following characteristics:

- the selection policy used in case of output contention to choose a cell from those contending for the same output port (random, priority, or longest queue selection policies),

- the switching speed of the fabric (the speedup ratio L), or equivalently, the maximum number of the *HOL* cells that the switch can deliver to an output port per slot $(1 \leq L \leq N)$,

- the physical location of the buffers to store cells which lost an output contention (input buffers, a shared buffer, or no buffer (cell dropping)), and the service discipline to serve cells in a buffer (FIFO, priority, window selection)

- the availability of buffers on the output ports to store cells which won an output contention (output buffer, or no queueing is necessary on outputs)

- the size of the internal switch fabric needed to implement an $N \times N$ switch

Tab.6 also shows the performance of each switch. It is assumed that the value of N is large, and that the input traffic is uniform. If the switch guarantees the cell loss probability of less than 10^{-6} at the input traffic load of $p = 0.9$, it is marked with the symbol ◯. If this cell loss requirement cannot be met, the value of the maximum throughput is shown.

Both the output buffered switch and the switch with a shared buffer can achieve the the maximum throughput of one. The output buffered switch achieves this throughput, when the speedup ratio L is N and the size of the output buffers is infinite. The switch with shared buffer can achieve the throughput of one, when the switch size is infinitely large. Examples of the output buffered switch include the Knockout switch and the ATOM switch. The Starlite switch with trap is an example of the switch with a shared buffer. Each switch architecture has advantages and disadvantages. The switch with a shared buffer requires less buffer capacity on outputs than the output buffered switch. However, in the shared buffer switch, cells may be delivered out of sequence. Furthermore, the size of the switch is six times as big as an output buffered switch. In some applications, the output buffered switch may be slightly favorable, because of the drawbacks inherit in a shared buffer approach. Although the switches indicated by ◯ do not achieve the maximum throughput of one, they offer very good performance. They achieve the cell loss probability of less than 10^{-6} at the input traffic load of $p = 0.9$, features we believe to have practical importance.

Some of the switches cannot satisfy the cell loss requirement of 10^{-6} at the input traffic load of $p = 0.9$. However, one may improve the performance of such switches by applying a combination of some of the improvement techniques described in this paper. For example, implementing the window selection policy on a faster speed switch fabric may improve the switch performance significantly.

5 Concluding Remarks

In this paper, we have surveyed various switch architectures for ATM networks. Surveyed switch architectures include the blocking switches and the nonblocking switches. Improvement techniques to these switch architectures are also discussed.

One of the areas that needs more research attention is the performance evaluation and comparison of switch architectures under integrated service environments. In such environments, different types of network traffic may co-exist in a switch, heavy concentration of traffic may occur, and the uniform traffic assumption may not hold any longer. This area of research is key to the successful application of ATM networks for integrated services.

References

[1] G. J. Anido and A. W. Seeto, "Multipath Interconnection: A technique for Reducing Congestion within Fast Packet Switching Fabrics," *IEEE J. Selected Areas Commun.*, vol.6, pp.1480–1488, Dec. 1988.

[2] T. T. Bradley and T. Suda, "Survey of Unified Approaches to Integrated–Service Networks," submitted *IEEE Computers.*

[3] R. G. Bubenik and J. S. Turner, "Performance of a Broadcast Packet Switch," *IEEE Trans. Commun.*, vol.37, pp.60–69, Jan. 1989.

[4] J. S.- C. Chen and R. Guerin, "Input Queueing of an Internally Non–Blocking Packet Switch with Two Priority Classes," *Proc. INFOCOM'89*, pp.529–537, Ottawa, Apr. 1989.

[5] C. Clos, "A Study of Non–blocking Switching Networks," *Bell System Tech. J.*, vol.32, pp.406–424, 1953.

[6] C. Day, J. Giacopelli and J. Hickey, "Application of Self–Routing Switches to LATA Fiber Optics Networks," *Proc. ISS'87*, pp.A7.3.1–A7.3.5, 1987.

[7] A. E. Eckberg and T.–C. Hou, "Effects of Output Buffer Sharing on Buffer Requirements in an ATDM Packet Switch," *Proc. INFOCOM'88*, pp.5A.4.1–5A.4.8, New Orleans, 1988.

[8] K. Y. Eng, "A Photonic Knockout Switch for High–Speed Packet Networks," *IEEE J. Select. Areas Commun.*, vol.6, pp.1107–1116, Aug. 1988.

[9] G. J. Fitzpatrick and E. A. Munter, "Input–Buffered ATM Switch Traffic Performance," *Proc. Multimedia'89*, No.4.2, Ottawa, April 1989.

[10] M. G. Hluchyj and M. J. Karol, "Queueing in High–Performance Packet Switching," *IEEE J. Select. Areas Commun.*, vol.SAC–6, pp.1587–1597, Dec. 1988.

[11] A. Huang and S. Knauer, "Starlite: A Wideband Digital Switch," *Proc. GLOBE-COM'84*, pp.5.3.1-5.3.5, Nov. 1984.

[12] J. Y. Hui and E. Arthurs, "A Broadband Packet Switch for Integrated Transport," *IEEE J. Select. Areas Commun.*, vol.5, pp.1264–1273, Oct. 1987.

[13] J. Y. Hui, "Resource Allocation for Broadband Network," *IEEE J. Select. Areas in Commun.*, vol.6, pp.1598-1608, 1988.

[14] Y-C. Jenq, "Performance Analysis of a Packet Switch Based on a Single–Buffered Banyan Network," *IEEE J. Select. Areas Commun.*, vol.SAC–1, pp.1014–1021, Dec. 1983.

[15] M. J. Karol, M. G. Hluchyj, and S. P. Morgan, "Input versus Output Queueing on a Space–Division Packet Switch," *IEEE Trans. Commun.*, vol.COM–35, pp.1347–1356, Dec. 1987.

[16] H. S. Kim and A. Leon–Garcia, "Performance of Buffered Banyan Networks under Nonuniform Traffic Paterns," *Proc. INFOCOM'88*, pp.4A.4.1–4A.4.10, New Orleans, March 1988.

[17] C. P. Kruskal and M. Snir, "The Performance of Multistage Interconnection Networks for Multiprocessors," *IEEE Trans. Computer*, vol.32, pp.1091–1098, Dec. 1983.

[18] T. T. Lee, "Nonblocking Copy Network for Multicast Packet Switching," *IEEE J. Select. Areas Commun.*, vol.6, pp.1455–1467, Dec. 1988.

[19] S.-Q. Li and M. J. Lee, "A Study of Traffic Imbalances in a Fast Packet Switch," *Proc. INFOCOM'89*, pp.538–547, Ottawa, Apr. 1989.

[20] T. Masaki, et. al., "A Study on a Switch for High Speed Packet Switching," (in Japanese) *IECEJ*, SE87–132, 1987.

[21] R. Melen and J. S. Turner, "Nonblocking Networks for Fast Packet Switching," *Proc. INFOCOM'89*, pp.548–557, Ottawa, Apr. 1989.

[22] P. Newman, "A Fast Packet Switch for the Integrated Services Backbone Network," *IEEE J. Select. Areas in Commun.*, vol.6, pp.1468-1479, Dec. 1988.

[23] H. Obara and T. Yasushi, "High Speed Transport Processor for Broad–Band Burst Transport System," *Proc. ICC'88*, pp.29.5.1–29.5.6, Philadelphia, June 1988.

[24] Y. Oie, M. Murata, K. Kubota and H. Miyahara, "Effect of Speedup in Nonblocking Packet Switch," *Proc. ICC'89*, Boston, June 1989.

[25] Y. Oie, M. Murata, K. Kubota and H. Miyahara, "Effect of Speedup in Nonblocking Packet Switch," under preparation, 1989.

[26] J. K. Patel, "Performance of Processor–Memory Interconnections for Multiprocessors," *IEEE Trans. Comput.*, vol.30, pp.771–780, Oct. 1981.

[27] G. M. Parulkar and J. S. Turner, "Towards a Framework for High Speed Communication in a Heterogeneous Networking Environment," *Proc. INFOCOM'89*, pp.655–667, Ottawa, Apr. 1989.

[28] E. P. Rathgeb, T. H. Theimer and M. N. Huber, "Buffering Concepts for ATM Switching Networks," *Proc. Globecom'88*, pp.39.3.1–39.3.5, Florida, Nov.-Dec. 1988.

[29] K. Shiomoto, M. Murata, Y. Oie and H. Miyahara, "Performance Analysis of Banyan Network Applied for ATM Switch," under preparation, 1989.

[30] Y. Sun and M. Gerla, "SFPS: A Synchronous Fast Packet Switching Architecture for Very High Speed," *Proc. INFOCOM'89*, pp.641–646, Ottawa, Apr. 1989.

[31] H. Suzuki, H. Nagano, T. Suzuki, T. Takeuchi, and S. Iwasaki, "Output–buffer Switch Architecture for Asynchronous Transfer Mode," to appear in *Proc. ICC'89*, Boston, June1989.

[32] T. Szymanski and S. Shaikh, "Markov Chain Analysis of Packet–Switched Banyans with Arbitrary Switches, Queue Sizes, Link Multiplicities and Speedups," *Proc. INFOCOM'89'*, pp.960–971, Ottawa, Apr. 1989.

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

[34] H. Uematsu and R. Watanabe, "Architecture of a Packet Bsed on Banyan Switching Network with Feedback Loops," *IEEE J. Select. Areas Commun.*, vol.6, pp.1521–1527, Dec. 1988.

[35] L.J. Wu, "Mixing Traffic in a Buffered Banyan Network," *Proc. 9th ACM Data Commun. Symp.*, pp.134–139, Sept. 1985.

[36] Y.-S. Yeh, and M. G. Hluchyj, and A. S. Acampora, "The Knockout Switch:A Simple, Modular Architecture for High–Performance Packet Switching," *IEEE J. Select. Areas Commun.*, vol.SAC–5, pp.1274–1283, Oct. 1987.

[37] H. Yoon, M. T. Liu and K. Y. Lee, "The Knockout Switch under Nonuniform Traffic," *Proc. Globecom'88*, pp.49.5.1–49.5.7, Florida, Nov.–Dec. 1988.

List of Figures and Tables

Fig.1 8×8 Banyan switch with binary switching elements

Fig.2 Example of internal blocking in the Banyan switch

Fig.3 8×8 buffered Banyan switch

Fig.4 Knockout switch

Fig.5 Bus Interface of the Knockout switch

Fig.6 Batcher Banyan switch

Fig.7 Starlite switch

Fig.8 Nonblocking switch with input buffers

Tab.1 Maximum throughput of a nonblocking switch with FIFO input buffers

Tab.2 Past research on the effects of speedup on the performance of a nonblocking switch

Fig.9 Nonblocking switch with output buffers

Fig.10 Nonblocking switch with input and output buffers

Tab.3 Maximum throughput of a nonblocking switch with input and output buffers ($N = \infty$)

Fig.11 Upper bound on the cell loss probability at input buffers ($N = \infty$)

Fig.12 Cell loss probability at output buffers ($N = \infty$, $p = 0.9$)

Tab.4 Simulation results on the maximum throughput of a nonblocking switch with window selection discipline

Fig.13 Nonblocking switch with N parallel buffers

Fig.14 Nonblocking switch with a shared buffer

Fig.15 4×4 Starlite switch with trap

Fig.16 Parallel switch with two switch planes (dedicated input buffers)

Tab.5 Input rate p^* and the number of switch planes K^*

Fig.17 Parallel switch with two switch planes (common input buffers)

Fig.18 Nonblocking switch with B parallel buffers

Fig.19 Broadcast Packet Switch

Tab.6 Performance of a nonblocking (single plane) switch (large N and uniform traffic)

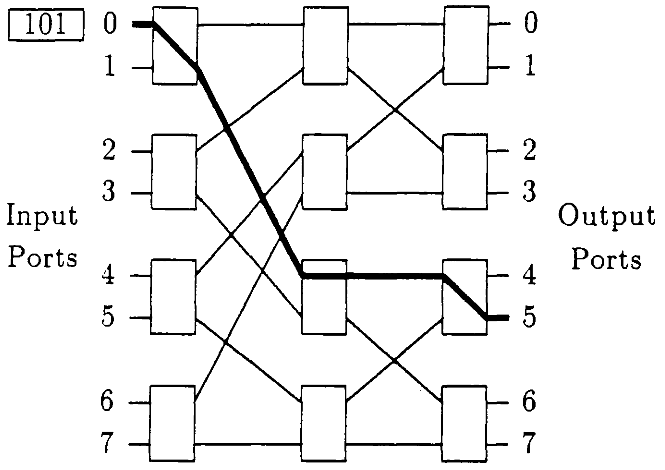

Fig.1 8 × 8 Banyan switch with binary switching elements

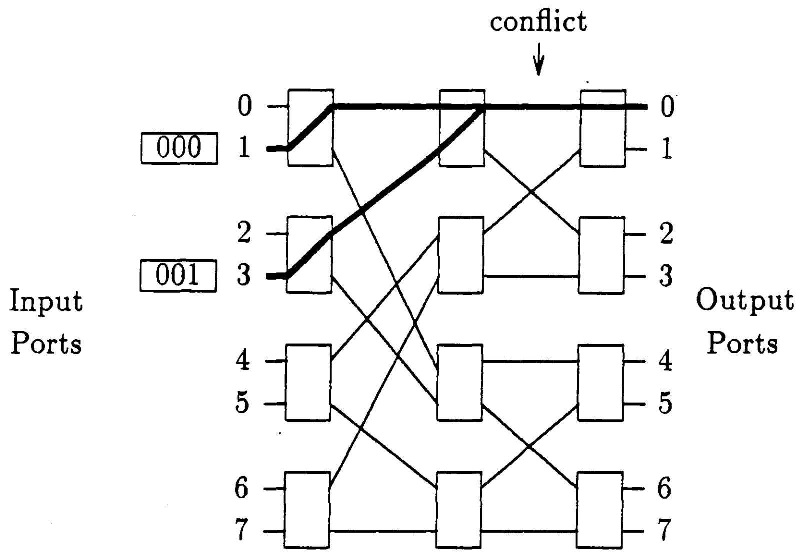

Fig.2 Example of internal blocking in the Banyan switch

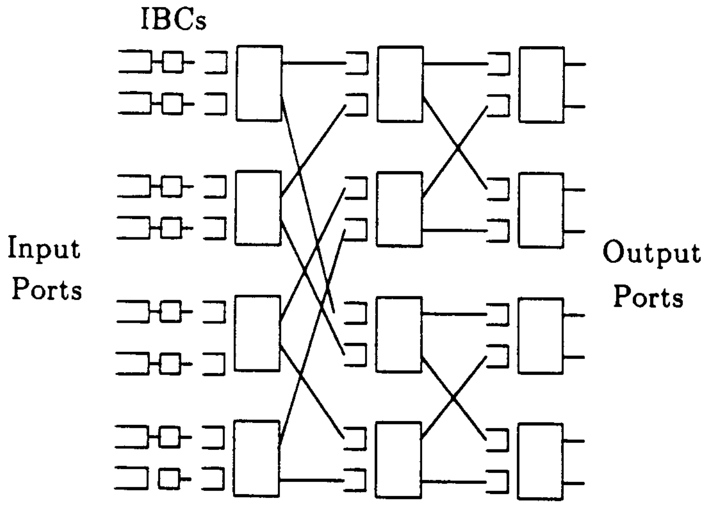

Fig.3 8 × 8 buffered Banyan switch

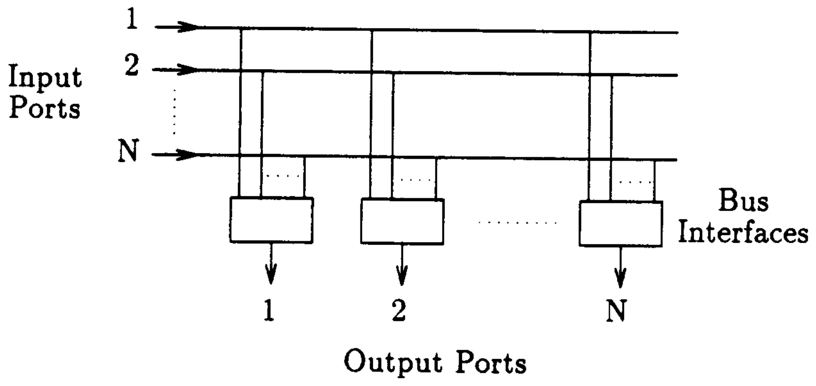

Fig.4 Knockout switch

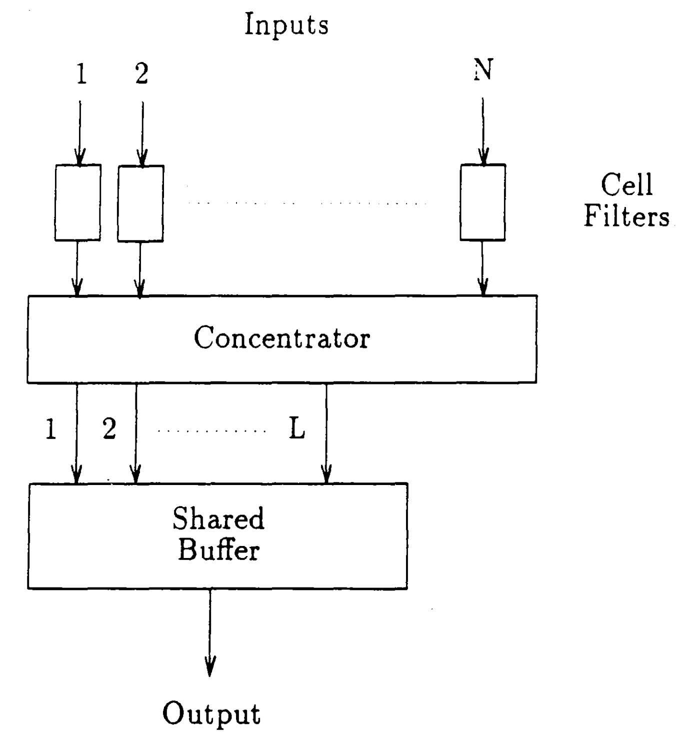

Fig.5 Bus Interface of the Knockout switch

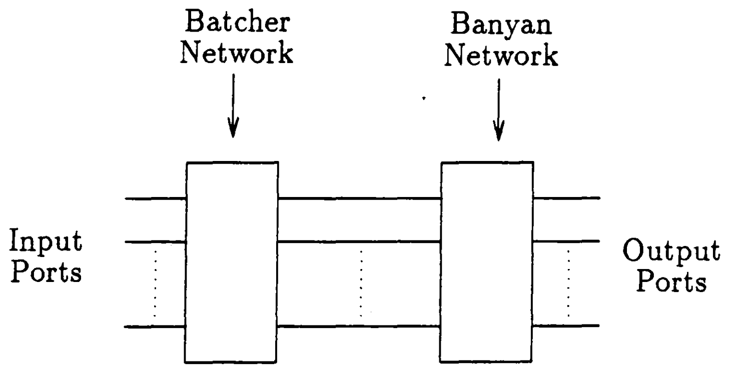

Fig.6 Batcher Banyan switch

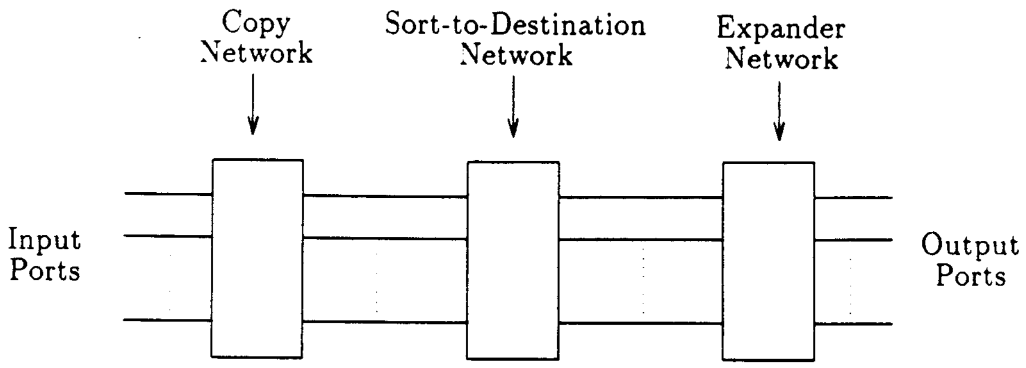

Fig.7 Starlite switch

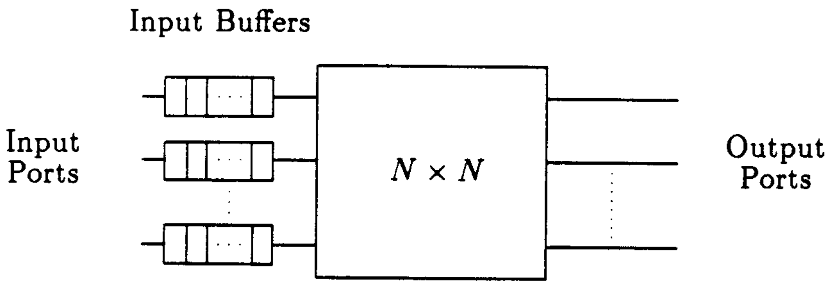

Fig.8 Nonblocking switch with input buffers

N	Maximum Throughput
2	0.7500
3	0.6825
4	0.6553
5	0.6399
6	0.6302
7	0.6234
8	0.6184
∞	0.5858

Tab.1 Maximum throughput of a nonblocking switch with FIFO input buffers

Analytic model for switch fabric				Performance measures obtained	References
L	N	B_{input}	B_{output}		
1	∞	∞	NQ	upper bound on $p(loss)$ on inputs	Hui [12]
				average delay	Karol [15]
	$\leq \infty$	∞	NQ	maximum throughput (see Tab.1)	
N	$\leq \infty$	NQ	∞	average delay	
	$\leq \infty$	NQ	$< \infty$	$p(loss)$ on outputs	Hluchyj [10]
$1 < L < N$	$\leq \infty$	0	∞	$p(loss)$ on inputs	Yeh [36]
	$\leq \infty$	0	$< \infty$	$p(loss)$ on outputs	Oie [24, 25]
	∞	∞	∞	maximum throughput (see Tab.3)	
				upper bound on $p(loss)$ on inputs	
	∞	∞	$< \infty$	$p(loss)$ on outputs	
	$\leq \infty$	1	∞	$p(loss)$ on inputs	Eng [8]

NQ : Queueing does not occur.

$< \infty$: finite value

$\leq \infty$: both finite and infinite values

Tab.2 Past research on the effects of speedup
on the performance of a nonblocking switch

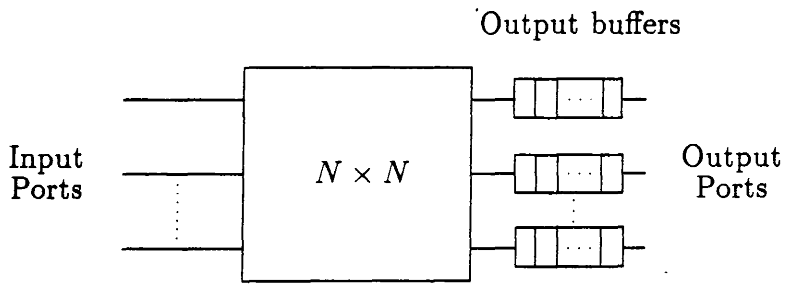

Fig.9 Nonblocking switch with output buffers

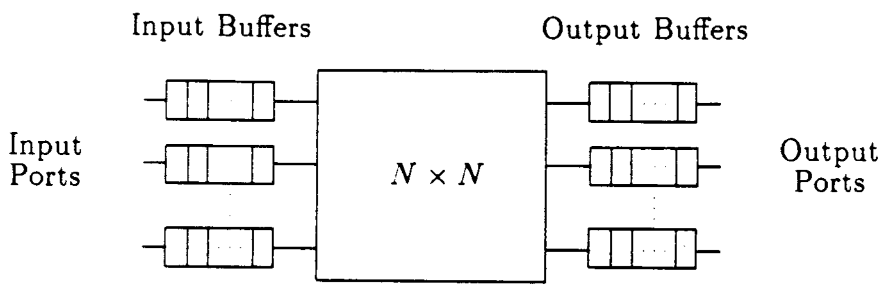

Fig.10 Nonblocking switch with input and output buffers

L	Maximum Throughput
1	0.5858
2	0.8845
3	0.9755
4	0.9956
5	0.9993
6	0.9999
∞	1.0000

Tab.3 Maximum throughput of a nonblocking switch with input and output buffers ($N = \infty$)

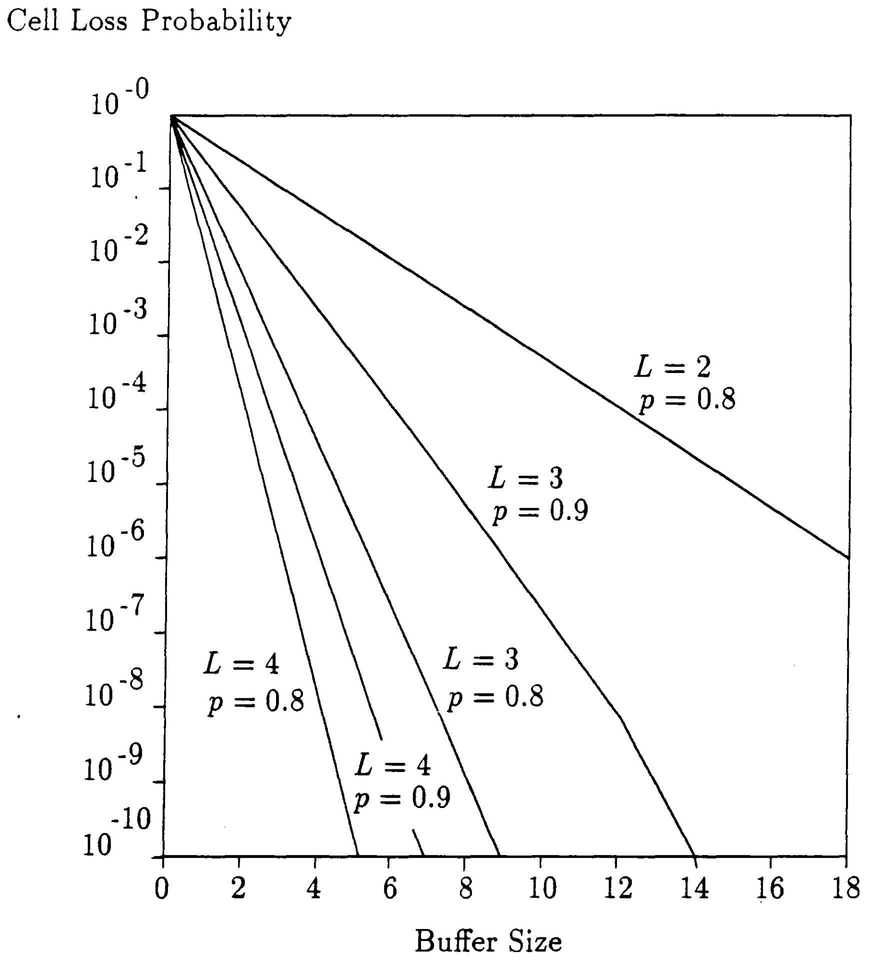

Fig.11 **Upper bound on the cell loss probability at input buffers ($N = \infty$)**

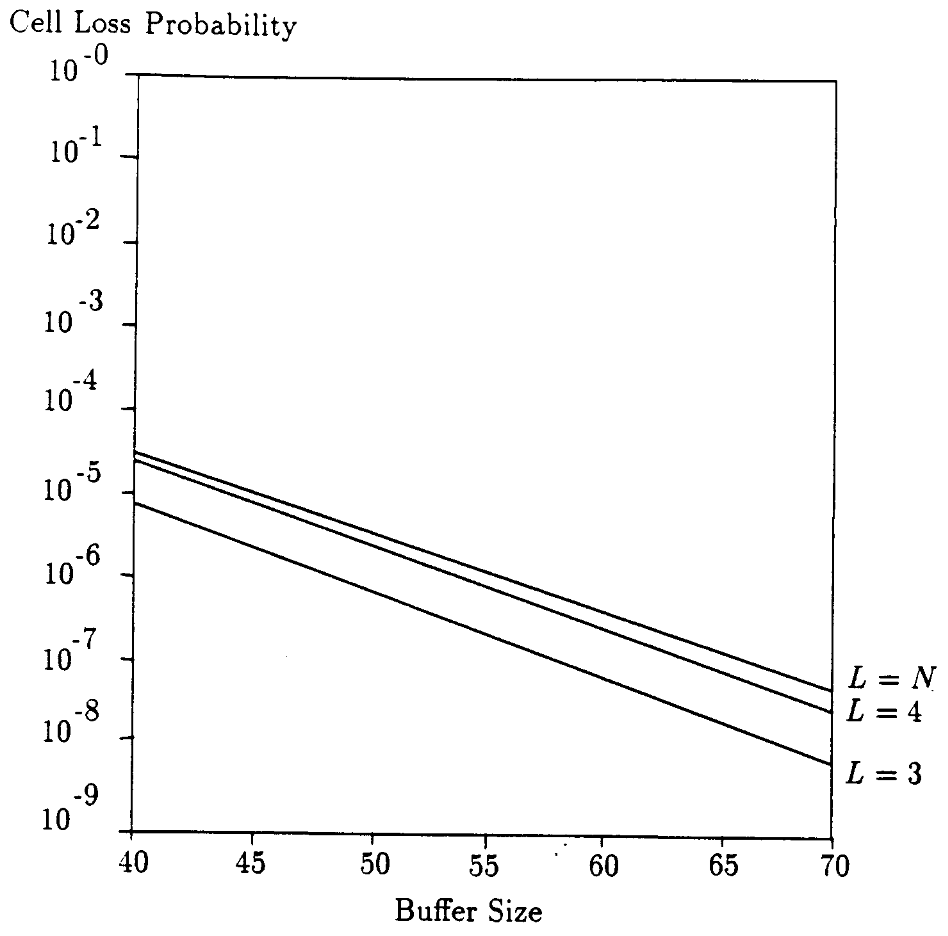

Fig.12 Cell loss probability at output buffers ($N = \infty$, $p = 0.9$)

N	Window size w							
	1	2	3	4	5	6	7	8
2	0.75	0.84	0.89	0.92	0.93	0.94	0.95	0.96
4	0.66	0.76	0.81	0.85	0.87	0.89	0.91	0.92
32	0.59	0.70	0.76	0.80	0.83	0.85	0.87	0.88
128	0.59	0.70	0.76	0.80	0.83	0.85	0.86	0.88

Tab.4 Simulation results on the maximum throughput of a nonblocking switch with window selection discipline

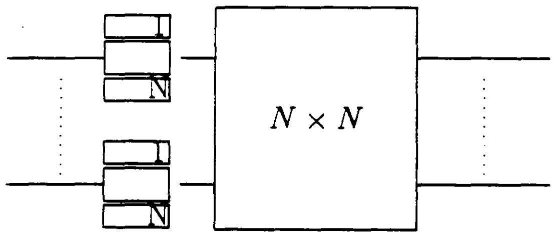

Fig.13 Nonblocking switch with N parallel buffers

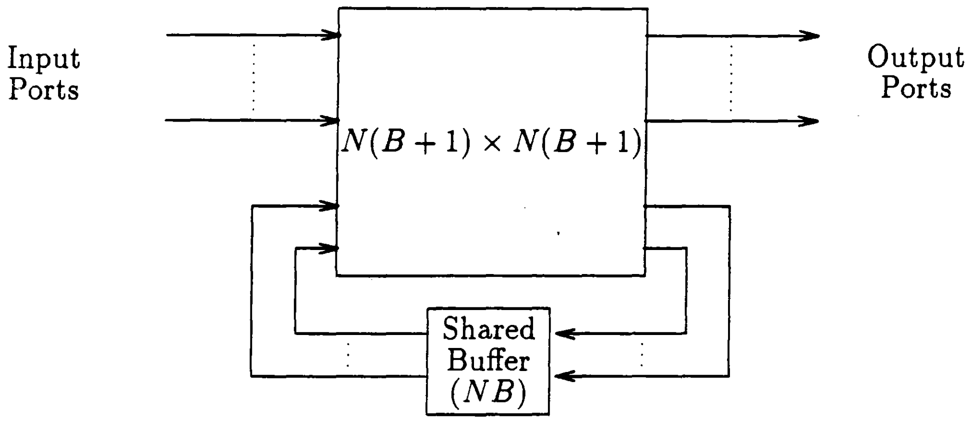

Fig.14 Nonblocking switch with a shared buffer

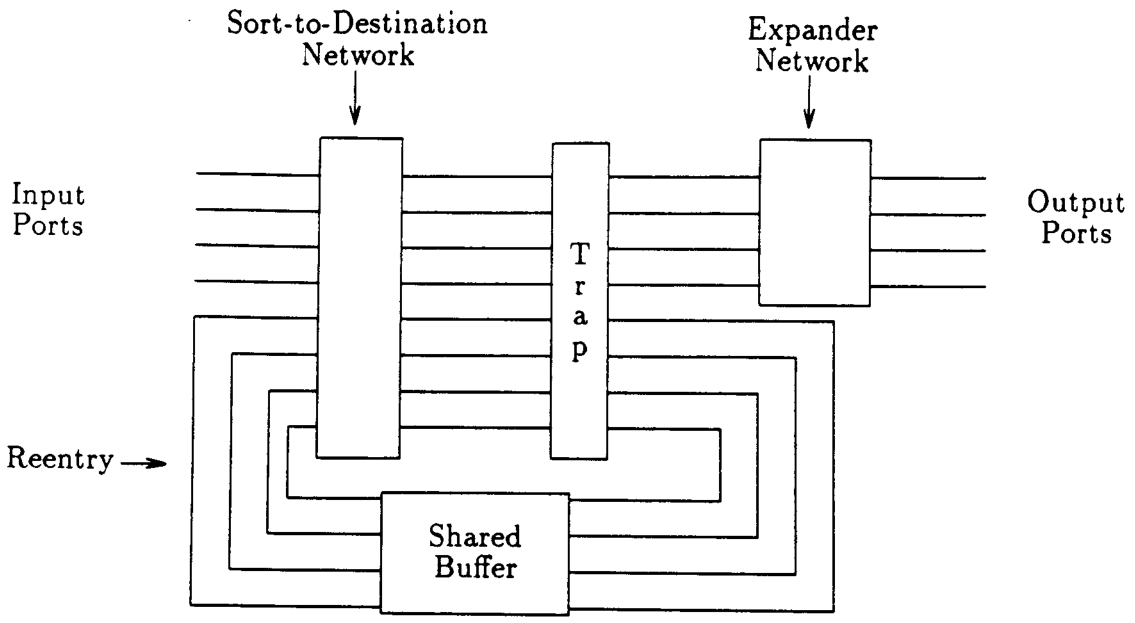

Fig.15 4 × 4 Starlite switch with trap

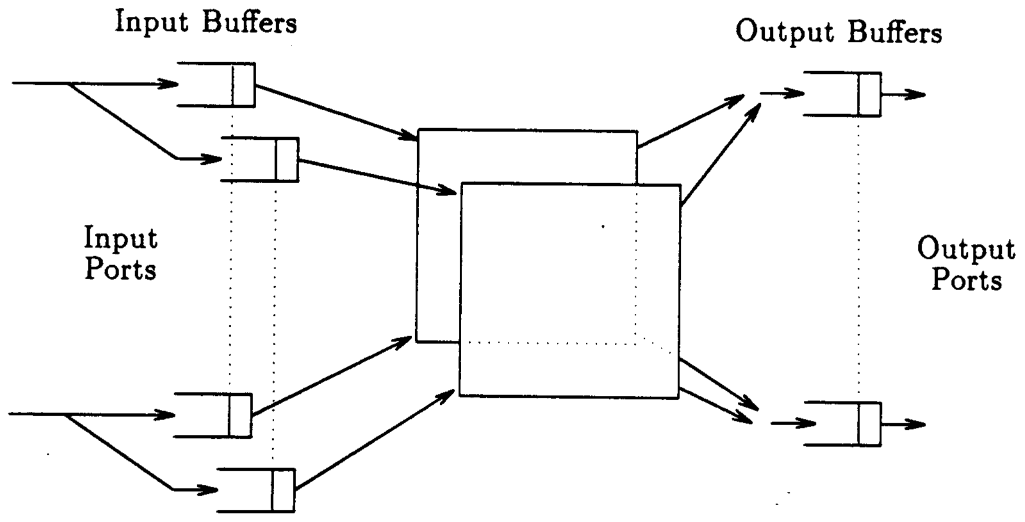

Fig.16 Parallel switch with two switch planes (dedicated input buffers)

L	p^*	K^*
1	0.000002	450000
2	0.002451	369
3	0.029013	32
4	0.106534	9
5	0.243505	4
6	0.437178	3
7	0.681421	2
8	0.969856	1

Tab.5 Input rate p^* and the number of switch planes K^*

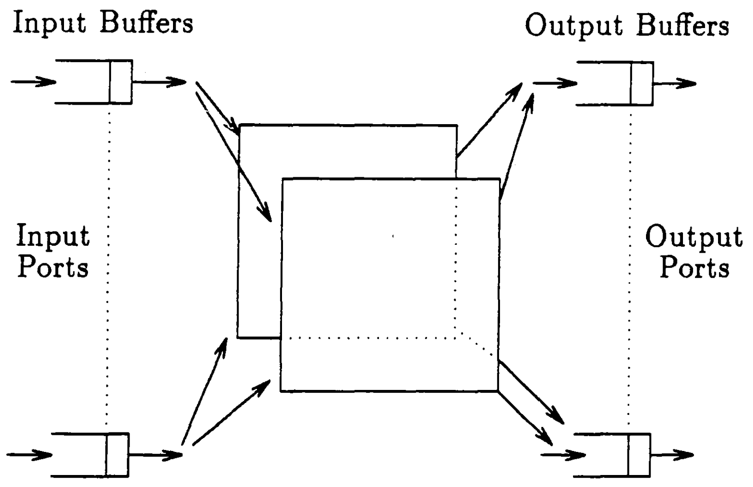

Fig.17 Parallel switch with two switch planes (common input buffers)

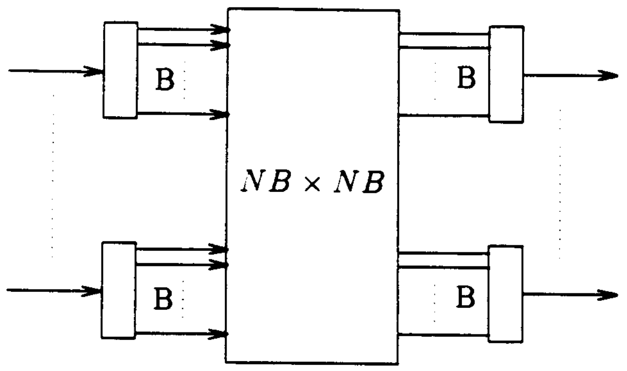

Fig.18 Nonblocking switch with B parallel buffers

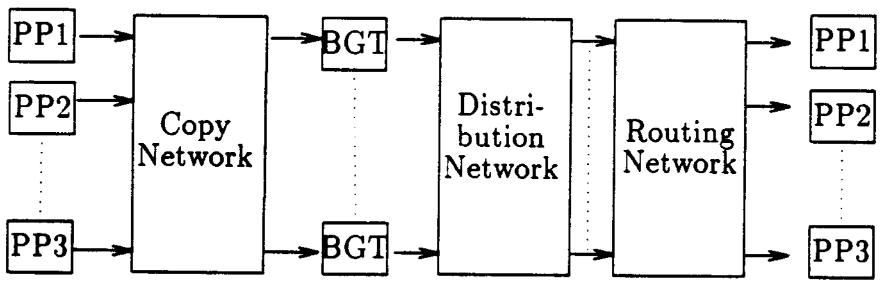

Fig.19 Broadcast Packet Switch

Contention Resolution		Buffering		Size	Performance	Ref.
selection	speed	losers	winners			
random	1	input (FIFO)	no queue	N	0.586	[15]
longest queue	1	input (FIFO)	no queue	N	$> 0.586^{(*)}$	[10]
priority	1	input (priority)	no queue	N	0.606	[4]
random	1	dropped	no queue	N	0.632	[26]
random	1	dropped	output	$100N$	$0.85^{(**)}$	[10]
random	1	input (window)	no queue	N	0.88	[12, 20, 10]
random	1	shared	no queue	$6N$	○	[7, 10]
random	3	input (FIFO)	output	N	○	[24]
priority	4	shared	output	$2N$	○	[8]
random	8	dropped	output	N	○	[36]
unnecessary	N	no losers	output	N	○	[15]

* : lower bound on the maximum throughput

** : This switch achieves the cell loss probability of less than 10^{-3} at $p = 0.85$.

Tab.6 Performance of a nonblocking (single plane) switch (large N and uniform traffic)