On the Multiple Shared Memory Module Approach to ATM Switching

Sherry X. Wei†
School of Electrical Engineering
Purdue University
West Lafayette, Indiana 47907

Vijay P. Kumar
AT&T Bell Laboratories
Holmdel, New Jersey 07733

Abstract

Shared memory based ATM cell switching has been shown to be more efficient in memory utilization than other buffering mechanisms, and is finding its way into several practical implementations. The maximum bandwidth of the shared memory modules limits the number of input and output ports of the switch to a relatively small number, especially when the line speeds are high (2.4 Gbps). In this paper, we propose a new approach to building large ATM switches based on shared memory modules. Shared memory modules are proposed to be placed in parallel, with every input and output port having access to every one of the switch modules. The advantage of this approach over previously proposed schemes is that it permits global sharing of the total buffer space. We study some basic issues in such an arrangement, such as the necessary and sufficient conditions for optimal (best delay-throughput) performance, and the minimum number of modules required for the switch to have the optimal performance while permitting sharing of the entire buffer space among all the input and output ports and while preserving packet sequencing for any virtual channel. A centralized control algorithm which yields optimal performance is also proposed.

1 Introduction

Asynchronous Transfer Mode (ATM) is gaining acceptance as a suitable technique for realizing Broadband ISDN (B-ISDN) networks and services. Much work has been done in the area of ATM cell switching architectures [1,2]. It is necessary to perform this switching at a very high rate since the input lines to the switch operate at 150 Mbps, 600 Mbps or 2.4 Gbps. ATM switching architectures proposed and implemented incorporate a high degree of parallelism and hardware-based control to achieve such switching speeds.

Certain general principles and techniques have evolved over the last few years in the area of high speed switching. Most of the switch architectures found in the literature involve one or more of these techniques [1,2]: self-routing banyan multistage networks; nonblocking batcher-banyan networks; multiple cell accepting and "knockout" principle; and memory efficient shared memory based switching.

Among the switching techniques, shared memory based approach is finding its way into many practical implementations [3,4,5]. A conceptual view of the shared memory switching principle is given in Figure 1. Essentially, all the input lines have access to the Write Bus of a single shared memory module, and all the output lines to the Read Bus. A control mechanism, which is not shown in the figure, exists to direct the incoming cells to empty locations in the memory, to indicate to the output lines where cells meant for them reside in the memory, and to update the list of empty locations when cells are read out of the memory by the outputs.

The main advantage of shared memory based switching is that, to provide a specified level of service (measured, for instance, by the cell loss rate), it requires only a fraction of the amount of memory required by other buffering schemes. However, the memory bandwidth, which depends on the word length (which is limited to the cell size) and the memory cycle time, can become a limitation. For nonblocking operation of a switch with N inputs, it is necessary to write into and read out of the memory N cells within the time it takes to transmit one cell on an input line or an output line. This implies that the bandwidth of the memory module must be at least as much as the sum of the bandwidths of the incoming lines and the outgoing lines. For instance, with ATM cells of 53 octets, input lines of 2.4 Gbps, and memory cycle time of 5 nanoseconds, the maximum number of lines that can be handled by one switch module is 17.

Researchers have been looking for ways to construct large ATM switches by using the shared memory switch as building block. It was proposed in [5] that large ATM switches be built by arranging the shared memory modules in a nonblocking multistage interconnection network configuration. Multistage networks have two disadvantages. First, it is not possible to achieve complete sharing of the buffer space. Secondly, even nonblocking multistage networks can experience destination blocking which can propagate backwards resulting in nonoptimal delay-throughput performance. In [6,7] a "Growable switch" architecture was proposed, which consists of a column of "small" shared memory switch modules preceded by a memory-less distribution network. This ar-

†Currently visiting AT&T Bell Labs, Holmdel, NJ.

CH3133-6/92/0000-0116 $3.00 © 1992 IEEE

chitecture, too, does not allow sharing of the memory among all the outputs, but limits it to a group of outputs.

In this paper, we explore an alternative approach to build large shared memory switches. Several memory modules are used in parallel, and each of the input lines and each of the output lines is provided access to each of the memory modules. This architecture will be called Multiple Shared Memory module switch (MSM) architecture. A conceptual diagram of the MSM switch architecture is given in Figure 2. This architecture has the potential to increase the memory bandwidth, memory efficiency and also to provide fault tolerance in a natural way.

Our goal is to design a switch such that

1. The switch yields the best delay-throughput performance.

2. Memories (or modules) are completely shared among all the input and output ports to achieve memory efficiency.

3. Packets belonging to a given virtual circuit depart from the switch in the order of their arrival.

In this paper, we focus on the study of the underlying theories in achieving the above goal. We present three results: first, we provide a theorem that gives necessary and sufficient conditions for a MSM switch to have optimal performance in terms of throughput and delay. We then derive the minimum number of modules required to achieve such performance. Finally a centralized control scheme which yields the optimal performance is given as an example.

2 Preliminaries

Consider a situation where packets[1] arriving on N input lines need to be switched to N output lines. (We later extend our study to the asymmetric case where the number of input lines is not equal to the number of output lines.) Each packet has an output line[2] to be switched to, according to the information stored in the header of the packet. Each input or output line has a line speed of R_1 packet/sec. We are given a set of switch modules, where each module is a shared buffer memory switch which can read and write R_2 packet/sec. In the ATM environment, all packets are of the same length (53 bytes). It is assumed that interfaces exist at the inputs and the outputs which assure that packets enter and exit the switch in a synchronous manner. Therefore the switch we study here is a synchronous one.

In the rest of the paper, we refer to the individual shared memory modules in the switch simply as *modules*.

[1] In the rest of the paper, the term "packet" is used synonymously with the term "cell".

[2] We do not consider multicasting here. The results in this paper can later be extended to multicasting.

2.1 Line Speed *vs.* Module Speed

We call the time taken to transmit a packet at an output port the *packet slot time* (or *slot* for short). From the above problem formulation, a slot time is $1/R_1$, which is also the time taken for the MSM switch to accept a packet, since both input and output lines have the same line speed. Let a *minislot* be the time taken to read and write a packet for each module, i.e., a mini-slot time is $1/R_2$. If $1/R_2 < 1/R_1$, the slot can be further divided into $R = \frac{R_2}{R_1}$ minislots (For simplicity, we assume that R is an integer. The results in this paper can be extended to non-integer values of R in a straight forward manner). Only one packet can be written into and/or read out of each module during a minislot, so at most R packets can be accepted by a module during one slot.

Suppose a MSM switch has K modules. Then, within this switch, a packet can be identified by the following three-tuple,

$$(t, s, k)$$

where

t – the slot during which the packet is to be sent out.

s – the minislot during which the packet is read out, and $1 \leq s \leq R$.

k – the module where the packet is stored, and $1 \leq k \leq K$.

When a packet enters the MSM switch, it is assigned a three-tuple that determines in which module the packet is to be stored and when this packet should be transmitted. When packets are transmitted, all the packets whose three-tuples have the same t value are sent out during the same slot.

For a slot period t, there are KR distinct three tuples. On the other hand, for an N-input, N-output MSM switch, the maximum number of packets sent out during a slot should be N. It follows that

$$K\ R \geq N,$$

that is,

$$K \geq \frac{N}{R}$$

Note that this same number of modules are also sufficient to accept N packets within a slot time, as the input lines and output lines have the same speed.

For example, if $R = N$, i.e., a switch module is N times faster than an input/output line, then only one switch module is needed to handle N inputs/outputs, and all the tuples are of the form of $(t, s, 1)$. This case corresponds to a single shared memory switch which has been studied previously [3]. On the other hand, if $R = 1$, then N switch modules must be required from the above inequality and in this case, all the tuples are of the form of $(t, 1, k)$.

In this paper, we refer to the method of assigning the three tuples (t, s, k) to incoming packets as the *switch scheme*. Without loss of generality, we shall focus on situations where $R \leq N$.

While the above inequality gives the minimum number of modules needed in order to meet line speed requirement, a MSM switch with $\frac{N}{R}$ modules does not necessarily yield the best performance, due to the potential for contention among the incoming packets for a module and among the outgoing packets for an output link. This problem is addressed in the next subsection.

2.2 Control Issues in a MSM Switch

Since we use a set of switch modules in a parallel form to construct a MSM switch, we have avoided the internal blocking and multistage packet delay problems that exist in a multistage switch. However, a MSM switch has its own problems arising from its architecture. These are input contention, output contention, packet sequencing and memory sharing.

In a mini-slot, if two or more packets from different input ports attempt to enter the same module, then *input contention* occurs; Similarily, if two or more packets stored in different modules attempt to depart from the same output port during a slot, an *output contention* occurs. These input/output contentions cause *head-of-line* (HOL) blocking at both input/output ports which, if not resolved properly, result in an extra packet delay and degrade the system throughput. It is well known that HOL blocking at the input ports alone reduces the switch throughput to 58% [9], so with HOL blocking at both input/output ports as in our MSM switch, the switch throughput would be even less. If packets belonging to the same virtual channel are not sent out in the order in which they arrive, an *out-of-sequence* problem occurs, which is not acceptable for an ATM network. Also, to maximize the memory efficiency we must ensure that *memory sharing* takes place properly.

It is not difficult to solve these four problems individually, but since each of them requires a solution which usually conflicts with another, it is non-trivial to solve all of them simultaneously. For example, if we let each input line have a dedicated module to write packets in, we will not have input contention problem. However, we will have output contention and lose memory sharing. (This case degenerates to the input buffering scheme. [9]) If, on the other hand, each output has a dedicated module to read packets out per mini-slot, then the output contention problem is eliminated, but we will have input contention and also lose memory sharing among the outputs. Or if packets from each input line are written to different modules in a round robin fashion and packets in each module are read out to different output lines in a round robin fashion, we will not have either contention problem, but packets can be out of sequence. It can be seen that some kind of control or communication among modules must be introduced to coordinate the operation of the MSM switch. The challenge therefore, is to design a control scheme that eliminates these problems and guarantees optimal switch performance.

3 Optimal MSM Switch

3.1 General Aspects

Since queueing is unavoidable in a packet switch, it is clear that a switch, from the system performance viewpoint, can be treated as a queueing system in which each output port is a server and its customers are packets that are destined for this port. Optimizing the switch performance is therefore equivalent to optimizing the performance of the underlying queueing system.

In general, such optimization should include two parts: the optimization of the queueing scheme (or discipline) assuming infinite buffer space and the optimization of the buffer management strategies assuming a given queueing discipline and a finite buffer space. The former part involves the design of system architectures with control mechanisms so that certain optimum criterion, for instance, the "shortest time" it takes before a packet in the queue is processed, can be satisfied. A typical study of this is [9]. The latter part is concerned with the design of policies, for a given system architecture, that allocate a finite buffer space to packets or users, and, especially, deal with situations when the buffer is full.

Many studies [11,12,13] have shown that buffer management policies have significant effect on the system performance. While this part of optimization is undoubtedly important, it can be separated from the first one. It is only when we have an optimal queueing system that an optimal buffer management strategy for a system makes sense. In this paper, we concentrate on the study of the first part and assume that there is infinite amount of buffer space.

Our criterion of an optimal switch scheme is derived from the performance viewpoint. It is well known that a queueing system, in which all customers (packets) have the same length, has the best delay-throughput if it is work conserving. Applying this result to a switch, we have the definition for an *the optimal switch scheme.*

Let $p(P_j)$ be the output port for which a packet P_j is destined, $Q_{p(P_j)}$ be the queue of output port $p(P_j)$, and $q_{p(P_j)}$ be the queue length of this queue when scheduling P_j.

Definition 1 *1. A packet P_j arriving in slot T is scheduled optimally if this packet can be accepted during slot T and scheduled to be sent out during slot $(T + q_{p(P_j)} + 1)$.*
2. A switch scheme is optimal if its underlying queueing system has the best delay-throughput performance.

It follows that a switch scheme is optimal if every packet can be scheduled optimally. The reason is straight forward. With such scheduling, the underlying queueing system is work conserving because the only delay a packet experiences in the switch is the queueing delay of its own queue, by definition 1. Note that this result holds for packet switches of all types and architectures.

It is worthwhile to distinguish between nonblocking packet switches and optimal packet switches. A non-blocking switch may not have an optimal switch scheme and an opti-

mal switch scheme is not unique. For example, the Batcher-Banyan [8] and the crossbar architectures are non-blocking, but they do not always have the best delay-throughput performance. If a switch built on either of these architectures employs input buffering, then the head-of-line blocking phenomenon will occur. However, if they are used in conjunction with output queueing [9], then an optimal delay-throughput performance results. The shared memory switch (shared buffering) [3] is also an optimal switch scheme, but it has a completely different architecture.

In the case of a single shared memory switch, the realization of the optimal queueing scheme is simple and natural [3]. It is more difficult, however, to design a MSM switch whose corresponding queueing system is equivalent to the optimal work conserving system, due to the complications addressed in section 2.2. It is clear though that in this case, queues formed for output ports have to be logical ones, since packets destined for the same output port may be stored in different modules that are physically apart, in order to ensure that memory sharing takes place. In the next subsection, we shall study this case in detail.

3.2 Optimal MSM Switch Scheme

The study in Section 2.2 shows that input/output contentions in a MSM switch, if handled improperly, can cause extra packet delay. This is to say that the underlying queueing system may not be work conserving. Eliminate such extra delays involves designing a control scheme such that the underlying queueing system is optimal (or work conserving). Notice that this is equivalent to designing a switch that has the best delay-throughput performance.

Having established the connection between our objectives and the mathematical model with which we will work to achieve our goals, we are now ready to present conditions for a switch scheme to be optimal. Define a function $\varphi(t,s,k)$ be such that

$$\varphi(t,s,k) = \begin{cases} 1 & \text{if module } k \text{ is assigned to a packet at} \\ & \text{minislot } s \text{ of slot } t. \\ 0 & \text{otherwise} \end{cases}$$

Theorem 1 *Suppose during a slot T, a set of packets $Z_T = \{P_1, P_2, \ldots, P_L\}$ $(L \leq N)$ arrive at a MSM switch with a set of modules $\mathcal{K}$, $|\mathcal{K}| = K$. A switch scheme is optimal if and only if*

1. *For any packet $P_j \in Z_T$, there exists a three-tuple $(T + q_{p(P_j)} + 1, s_j, k_j)$ such that $\varphi(T + q_{p(P_j)} + 1, s_j, k_j) = 0$, where $1 \leq s_j \leq R$ and $k_j \in \mathcal{K}$.*
2. *A three-tuple is assigned to each P_j such that, for any $k \in \mathcal{K}$ satisfying condition 1, $\sum_{s=1}^{R} \varphi(T,s,k) \leq R$.*

Proof: Let us prove the *if* part first. If the first condition is satisfied, each incoming packet P_i is always scheduled to be sent out next to the last packet in the queue $Q_{p(P_i)}$, therefore each output line i will always be busy as long as there are packets in the switch destined for it, indicating a work conservation discipline. If Condition 2 is satisfied, it guarantees that no more than R packets are assigned to a module during a slot. With both conditions being satisfied, each packet is guaranteed to be accepted by one of the modules upon arriving and to be read out from this module with the shortest delay.

It is straight forward to show the *only if* part. If Condition 1 is violated, unnecessary delay is introduced. If Condition 2 is violated, then there is at least one packet that cannot be accepted by the switch upon arrival and has to be delayed at least one slot. Therefore the switch can not have the best delay-throughput performance.

Q.E.D.

Notice that an optimal switch scheme solves the input contention, output contention and out-of-sequence problems simultaneously. The first condition guarantees that during a slot time, no attempt is made to send more than one packets out of one output port which avoids HOL blocking at output ports. The second condition guarantees that all of the incoming packets can be accepted within one transmission time upon arriving which avoids HOL blocking at the input ports. Since each of the incoming packets is placed at the end of its output port queue and served in a FIFO fashion, packet sequencing for any virtual circuit is preserved.

Also notice that with an optimal switch scheme, a MSM switch with N input/output ports has the same delay-throughput performance as a $N \times N$ single shared memory switch, due to the fact that both systems can be viewed as N output port work conserving queues with a shared common buffer space. In the next subsection, we shall find out the minimum number of modules required to realize such an optimal MSM switch scheme.

3.3 The Minimum Number of Modules for Optimal MSM Switch Scheme

For an N-input, N-output MSM switch, there has to be at least $\frac{N}{R}$ modules in order to meet the line speed requirement. Since for a particular slot, only N out of KR possible three tuples are used and $KR \geq N$, condition 1 of Theorem 1 can always be satisfied. Condition 2 of Theorem 1, however, may not always be satisfiable. This can be explained in the following example.

Suppose there are 4 input lines that need to be switched to 4 output lines and suppose we use 4 switch modules with each one having $R = 1$. Figure 3 shows a scheduling diagram in which rows correspond to the output port queues, columns correspond to the time slots during which packets in the queues are to be sent out and the numbers are the modules in which the packets are stored.

Suppose at the beginning of slot T the packet at the end of queue 4 (Q_4) is scheduled to be sent out at slot T_4 and the only remaining tuples for slots T_4+1, T_4+2, T_4+3 and T_4+4 are

$$(T_4+1,1,4),\ (T_4+2,1,4),\ (T_4+3,1,4),\ (T_4+4,1,4).$$

(In Figure 3, since modules 1, 2 and 3 already hold packets

to be sent out during these time slots, only module 4 is available to assign the packets to be sent out during time slots T_4+1 through T_4+4.)

Consider a situation where during slot T four packets arrive and all of them are destined for output port 4. By condition 1 in Theorem 1, each packet must be assigned a tuple from the above four tuples. On the other hand, if we accept all four packets during slot T by condition 2, then we have

$$\sum_{s=1}^{R}\varphi(T,s,k)=\begin{cases}0 & \text{if } k=1\\ 0 & \text{if } k=2\\ 0 & \text{if } k=3\\ 4 & \text{if } k=4\end{cases}$$

Since $R=1$, and $\sum_{s=1}^{R}\varphi(T,s,4)=4>R$, condition 2 in Theorem 1 is violated and input contention occurs. Three of the four incoming packets can not be accepted into the switch, and have to either be dropped or wait until the next slot time, resulting in at least one slot extra delay.

It is obvious that if we increase the number of modules in the switch, then $\sum_{s=1}^{R}\varphi(T,s,k)$'s will decrease because packets can be spread out among different modules. The more modules we have, the less possible that condition 2 in Theorem 1 will be violated.

Theorem 2 *The minimum number of modules to realize an optimal switch scheme for a MSM switch with N input/output ports is $(2\frac{N}{R}-1)$.*

Proof: When there are $(2\frac{N}{R}-1)$ modules in a MSM switch, the maximum number of three tuples available during each slot is,

$$R(2\frac{N}{R}-1)=2N-R$$

Consider a packet P_j which arrives during slot T. For the switch scheme to be optimal, this packet should be sent out during the slot $(T+q_{p(P_j)}+1)$ from the earlier discussion. Note that during slot $(T+q_{p(P_j)}+1)$, there are at most $(N-1)$ three tuples that have been assigned to some packets, each corresponding to a different output port, therefore packet P_j has at least

$$2N-R-(N-1)=N-R+1\geq 1$$

three tuples to choose from, since $N\geq R$. Thus, the first condition in Theorem 1 can always be satisfied.

We can show by contradiction that this also implies that packet P_j has at least $\frac{N}{R}$ possible modules to choose to be stored upon arriving during slot T. Suppose it has only $(\frac{N}{R}-1)$ possible modules to choose to enter. Then the maximum number of unassigned three tuples corresponding to these unoccupied $(\frac{N}{R}-1)$ modules during slot $(T+q_{p(P_j)}+1)$ is

$$R(\frac{N}{R}-1)=N-R<N-R+1.$$

This contradicts the conclusion that it has at least $(N-R+1)$ possible three tuples to choose from, if there are $(2\frac{N}{R}-1)$ modules in the switch. Therefore when a packet is scheduled to be sent out at a particular slot, it has at least $\frac{N}{R}$ modules to choose to enter upon arriving.

The worst case happens when all N arriving packets (during slot T) have the same set of modules to choose from. That is, among $(2\frac{N}{R}-1)$ modules, only $\frac{N}{R}$ modules, denoted by $k_1,k_2,\ldots,k_{\frac{N}{R}}$, are chosen to accept these arriving packets by condition 1 of Theorem 1. Since

$$R\frac{N}{R}=N,$$

by assigning R packets to each module $k_1,k_2,\ldots,k_{\frac{N}{R}}$, we have $\sum_{s=1}^{R}\varphi(T,s,k_i)=R$, for $i=1,2,...,\frac{N}{R}$, which satisfy condition 2 of Theorem 1.

Therefore we have proved that if an MSM switch has $(2\frac{N}{R}-1)$ modules, an optimal switch scheme can always be realized.

We now prove that $(2\frac{N}{R}-1)$ is the minimum number of modules required to realize an optimal switch scheme. Let us assume that there are only $(2\frac{N}{R}-2)$ modules in the switch. When a packet arrives, it has at least

$$R(2\frac{N}{R}-2)-(N-1)=N-2R+1$$

three tuples to choose from. However, the guaranteed number of modules this packet can choose to go to is

$$\lceil\frac{N-2R+1}{R}\rceil=\frac{N}{R}-2+\lceil\frac{1}{R}\rceil=\frac{N}{R}-1$$

Again in the worst case, all N incoming packets have the same set of $(\frac{N}{R}-1)$ modules to choose from. This implies that we have to distribute N packets among $(\frac{N}{R}-1)$ modules; therefore, there must exist a module $i\in\{1,2,...,(\frac{N}{R}-1)\}$ such that

$$\sum_{s=1}^{R}\varphi(T,s,k_i)>R$$

Thus the second condition in Theorem 1 is violated and the optimal switch scheme can not be realized.

Q.E.D.

For example, if $R=N$ in a MSM switch, then the minimum number of modules needed for this switch to have the best delay-throughput performance is $2\frac{N}{R}-1=1$, which is the case of a single shared memory switch. If $R=1$, then the number is $2\frac{N}{R}-1=2N-1$. That is, if the module speed is equal to the line speed, $2N-1$ modules are needed to realize the optimal switch scheme for a N-input, N-output port MSM switch.

With a complete analogy, we can extend the above results to the more general case where the number of inputs is not equal to the number of outputs.

Corollary 1 *The minimum number of modules to realize an optimal switch scheme for a MSM switch with M input ports and N output ports is $(\frac{M+N}{R}-1)$, where R is the ratio of module speed to the line speed.*

4 A Control Scheme for an Optimal MSM Switch

In this section, we propose an algorithm which yields the optimal switch scheme discussed earlier for an MSM switch with $(2\frac{N}{R} - 1)$ modules. Before we proceed, we make the following definitions.

b_i – the mini-slot at which a packet is written into module i, $i = 1, 2, \ldots, 2\frac{N}{R} - 1$.

A – the set of all module numbers, $A = \{1, 2, \ldots, 2\frac{N}{R} - 1\}$.

B – the set of available module numbers.

$X_{T+q_p(P_i)+1}$ – the set of possible modules packet P_i scheduled to be sent out at slot $T+q_p(P_i)+1$ can be assigned to.

Z_T – the set of all packets arriving in slot T.

During a slot T, the algorithm is described as follows:

Step 0. Initialization.
$b_i = 1$, for $i = 1, 2, \ldots, 2\frac{N}{R} - 1$;
$B = A$;
$Z_T = \{P_1, P_2, \ldots, P_L\}$.

Step 1. Pick an element (packet) P_i, $P_i \in Z_T$. Look at the header of this packet to find out its output port, then obtain logical queue $Q_p(P_i)$ and its length $q_p(P_i)$.

Step 2. Find out the available three tuples for packet P_i and extract the third tuple (module numbers) and add to set $X_{T+q_p(P_i)+1}$.

Step 3. If $(X_{T+q_p(P_i)+1} \cap B \neq \phi)$, then pick one element w_i from the intersection set. This w_i is the module in which packet P_i will be stored.

Step 4. Update

$$Z_T = Z_T - P_i;$$
$$b_{w_i} = b_{w_i} + 1;$$

Step 5. if $b_{w_i} > R$, then $B = B - w_i$.

Step 6. Repeat *Step 1* through *Step 5* until $Z_T = \phi$.

Step 7. Write each packet P_i to its module w_i at mini-slot b_{w_i}.

Step 8. Send out packets which were scheduled to be sent out at slot T.

To prove the correctness of this algorithm, we must show that it satisfies both conditions in Theorem 1. From the proof of the Theorem 1, we know that Condition 1 is always satisfied since there are $(2\frac{N}{R} - 1)$ modules in the MSM switch. Notice that *Step 5* imposes a constraint on the number of times each module can be used during a slot, therefore Condition 2 will be satisfied, as long as the algorithm can be executed at each step, i.e., the condition in *Step 3* is always satisfied.

To show the intersection set in *Step 3* is never empty, we first prove that when processing packet P_i, there are at most $(\frac{N}{R} - 1)$ elements that have been deleted from set B in *Step 5*. Suppose $(\frac{N}{R})$ elements have been deleted from set B, then by the condition of deleting a packet in *Step 5*, $R\frac{N}{R} = N$ packets must have been processed before processing packet P_i during the current slot. This contradicts the fact that there are at most N packets arriving during each slot, therefore at most $(\frac{N}{R} - 1)$ packets would have been deleted.

Since $|A| = 2\frac{N}{R} - 1$, $|B| \geq |A| - (\frac{N}{R} - 1) = \frac{N}{R}$. Recall that for packet P_i, there are at least $\frac{N}{R}$ modules to choose from, i.e., $|X_{T+q_p(P_i)+1}| \geq \frac{N}{R}$. Now if we had $X_{T+q_p(P_i)+1} \cap B = \phi$, then

$$|B| + |X_{T+q_p(P_i)+1}| \geq 2\frac{N}{R}$$

which violates the fact that there are only $2\frac{N}{R} - 1$ modules in the MSM switch. This completes the proof.

5 Conclusions

In this paper, we have investigated a new approach to building large ATM switches based on shared memory switching modules arranged in such a way that the total memory space is shared among all input ports and output ports. The conditions for optimal (best delay-throughput) operation of the proposed architecture, called the Multiple Shared Memory module switch (MSM switch), were given; the minimum number of modules required for optimal operation were derived; a control procedure for the MSM switch that yields the best delay-throughput performance, while preserving the packet sequence and achieving global memory sharing, was also given.

The key issues to be resolved for the MSM approach to become suitable for implementation are: hardware-assisted implementation of the control, either based on the algorithm given in Section 4 or other, possibly decentralized, algorithms; interconnection schemes to connect inputs to the memory modules and the memories to the outputs (either memory-less, nonblocking, electronic circuit switches or optical interconnects). We are currently studying these issues.

References

[1] H. Ahmadi and W. E. Denzel, "A survery of modern high-performance switching techniques," *IEEE Journal on Selected Areas in Communications,* vol 7, No. 7, September 1989, pp. 1091-1103.

[2] F. A. Tobagi, "Fast packet switch architectures for Broadband Integrated Services Digital Networks," *Proceedings of the IEEE,* vol. 78, No. 1, January 1990, pp. 113-167.

[3] H. Kuwahara, N. Endo, M. Ogino and T Kozaki, "A Shared Buffer Memory Switch for an ATM Exchange," *Proc. ICC*, June 1989, pp. 118–122.

[4] M.A. Henrion, et al, "Switching network architecture for ATM based broadband communications," *Proc. 1990 Intl. Switching Symposium*, pp. 1–8.

[5] Y. Sakurai, N. Ido, S. Gohara and N. Endo, "Large-Scale ATM Multistage Switching Network with Shared Buffer Memory Switches," *Communication Mag.*, January 1991, pp. 90–96.

[6] K.Y. Eng and M.J. Karol, "Gigabit-per-second ATM packet switching with the growable switch architecture," *ICC'91*, pp. 1014–1020.

[7] K.Y. Eng and M.J. Karol, "The Growable switch Architecture: a self-routing implementation for large ATM applications," *To appear in the proceedings of Globecom'91.*

[8] J. Y. Hui and E. Arthurs, "A Broadband Packet Switch for Integrated Transport," *IEEE J. on Selected Area in Commun.*, vol. SAC–5, No. 8, October 1987, pp.1264–1273.

[9] M. G. Hluchyj and M. J. Karol, "Queueing in High-Performance Packet Switching," *IEEE J. on Selected Area in Commun.*, vol. 6, no. 9, Dec. 1988, pp. 1587–97.

[10] Y. S. Yeh, M. G. Hluchyj and A. S. Acampora, "The Knockout Switch: A Simple Architecture for High-Performance Packet Switching," *IEEE J. on Selected Area in Commun.*, vol. SAC–5, No. 8, October 1987, pp.1274–1283.

[11] F. Kamoun and L. Kleinrock, "Analysis of Shared Finite Storage in a Computer Node Environment Under General Traffic Conditions," *IEEE Trans. Commun.* COM–28, 7, July 1980, pp. 992–1003.

[12] G. J. Foschini and B. Gopinath, "Sharing Memory Optimally," *IEEE Trans. Commun.* COM–31, March 1983, pp. 352-360.

[13] S.X. Wei, E.J. Coyle and M-T T. Hsiao, "An Optimal Buffer Management Plicy for High-Performance Packet Switching," to appear in Glbecom'91.

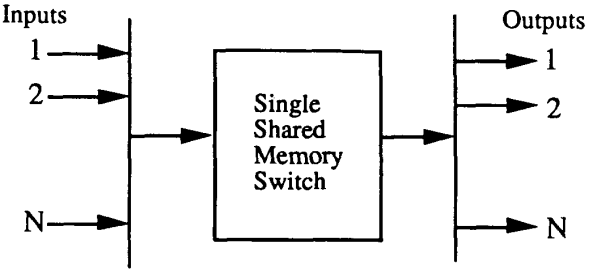

Figure 1: A Shared Memory Based Packet Switch

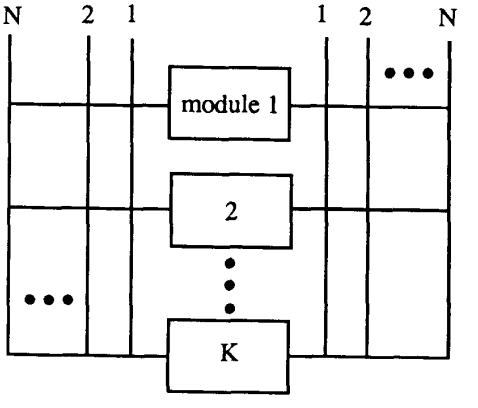

Figure 2: A conceptual view of a MSM switch

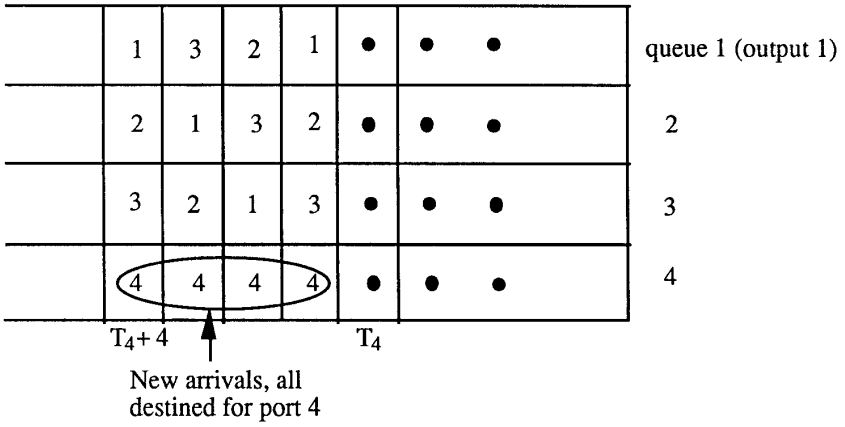

Figure 3: An example illustrating input contention.