

Simulation Of Improved ATM Switch using Dynamic Buffer Sharing And Multiprocessing

P.K.Suri, Prof., Dept. of Computer Sc.
& Applications, Kurukshetra
University, Kurukshetra-126119.
Email: pksurikuk@rediffmail.com

K.D.Sharma, Chairman,
Solidcore, New Delhi, Formally
with IIT Delhi, Delhi University,
Email: kds@solidcore.com

Brijesh Kumar,
Asstt.Prof., Lingaya's Institute
of Mgmt & Tech, Faridabad,
Email: brijesh10@hotmail.com

Abstract

In an ATM network, the network performance depends on many parameters, including the architecture of the network and the performance of switching elements. The current paper discusses, the performance improvement by modifying the switch fabric and packet forwarding, using simulation concepts. A switch consists of some input buffers and an algorithm, to select the cells waiting in the queue, to be forwarded, to the output buffer. In this simulator, two processes are involved in picking and forwarding of the cells from input buffer, to the output buffer in ATM Switch. A substantial change in performance and improvement in throughput has been observed, using the proposed simulator, in comparison with the existing model [3] and the same has been depicted diagrammatically.

Keyword: ATM, Network Simulation, Switch, performance, dynamic buffer sharing

Notations:

ATM	Asynchronous Transfer Mode
CBR	Continuous Bit Rate
VBR	Variable Bit Rate
RT/NRT	Real Time/Non Real Time
ABR	Absolute Bit Rate
UBR	Unspecified Bit Rate
PCR	Peak Cell Rate
CDVT	Cell Delay Variation Tolerance
SCR	Sustainable Cell Rate
BR	Burst Tolerance
MCR	Minimum Cell Rate
QoS	Quality of Service
IBM	Input Buffer Module
OBM	Output Buffer Module
SE	Switching Element
FIFO	First In First Out
AvgWT	Average Weighting Time(of Cell)
AvgQL	Average Queue Length(of IBM)
AvgIDT	Average Ideal Time(of processes)

Introduction

Telecommunication networks, based on the Broadband Integrated Service Digital Network (B-ISDN) can provide user with a guaranteed Quality of Service (QoS). In order to achieve this, the user must inform the network, at the time of connection setup, of both the expected nature of the traffic that will be sent along the connection, and of the type of quality of service that the connection requires in each direction. The former is described by a set of traffic description parameters, while the latter is specified by a set of desired QoS parameters for each direction.

ATM networks offer a specific set of service classes, and at the time of connection set-up, the user must request a specific service class from the network for the connection. Service classes are used by ATM networks to differentiate between specific types of connections, each with a particular mix of traffic and QoS parameters, since such traffic may need to be differentiated within the network, for instance, by using priorities to allow for the requested behavior. The current set of QoS Classes, defined by the ATM forum for UNI 4.0, is as follows.

1. Continuous Bit Rate (CBR),
2. Variable Bit Rate-Real Time VBR (RT),
3. Variable Bit Rate-Non Real Time (VBR (NRT))
4. Available Bit Rate (ABR) and
5. Unspecified Bit Rate (UBR)

Traffic sent along connections of any type is defined by a set of traffic descriptive parameters

1. Peak Cell Rate (PCR)
2. Cell Delay Variation Tolerance (CDVT)
3. Sustainable Cell Rate (SCR)
4. Burst Tolerance (BT)
5. Minimum Cell Rate (MCR) for ABR only

These parameters define an envelope around a traffic stream, but not all parameters are valid for all the service classes.

1. For CBR connections, only the PCR and CDVT are defined,
2. For VBR, the SCR and BT are defined,
3. For ABR, the PCR and MCR are defined are defined.
4. For UBR, only PCR is defined

The current set of QoS parameters consists of three delay parameters and one dependability parameter, as follows

- o Peak-to-peak Cell Delay Variation (CDV)
- o Maximum Cell Transfer Delay (Max CTD)
- o Mean Cell transfer Delay (Mean CDV) and
- o Cell Loss Ratio (CLR): dependability parameter.

Using the above information a Switch can decide, the way to handle the cell upon arrival. High-speed cell switches use buffering to queue cells, whose service has been delayed due to contention for resources within the switch. The arrangement of these buffers and the buffer management policy affects the switch performance[12]. Some buffers are present at the input ports to handle/store the input cells temporarily and some at the output port to handle the output traffic/cells. It has been concluded that input queuing and completely shared buffering achieve optimal throughput delay performance. However, there is a fairness problem in input queued shared buffer switches that a single input port can take over most of the buffers, preventing cells destined for less utilized ports from gaining access under overload condition[10]. In order to overcome this problem an output buffer is added to the switch module corresponding to each outgoing port, in addition to the input buffer[1].

Two kinds of buffer sharing schemes have been proposed in shared buffer switches. The first one named as the threshold scheme puts threshold on buffers that should be available to any individual input port. In this method, a threshold is assigned to each input port. An arriving cell is accepted, only if the current queue length is smaller than the threshold. Only comparators and counters are needed in the threshold scheme. Thus it is really simple to implement the threshold scheme[2][12][20].

The second kind of sharing restriction is push out scheme. An arriving cell is allowed to enter the buffer as long as the shared buffer is not full. When the buffer is full, an incoming cell is allowed to enter the buffer by taking the place of a cell that is already in the buffer. For the fairness reason, the cell located at the head of the longest queue among the input ports, is selected, to be discarded first. It should be noticed that the pushing and pushed cells might belong to different input ports. The performance of the pushout scheme is optimal because no input queue is ever starved for space and no space is ever held idle, while some queue desires more[12][1][2].

Chaudhary[1] has proposed a dynamic threshold scheme. The key idea in the dynamic threshold scheme is that the maximum permissible length is proportional to the unused buffering for any individual queue at any instance of time. In fact, the dynamic threshold scheme uses partly threshold scheme and partly pushout scheme.

There has been a considerable amount of work in this area[11][8][19][5][17].

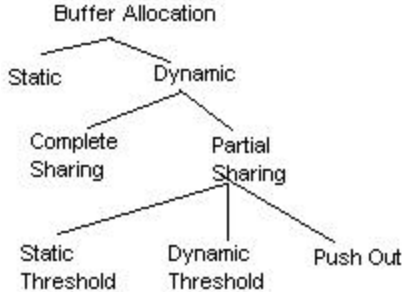


Figure 1

In the current paper, a scheme is proposed where performance is further improved by including more than one processes to transfer the cells from input buffer to the output buffer. Using the combination of dynamic buffer sharing and more than one processes for handling more than one cells in the input buffer, we achieve a superior throughput and a reduced number of lost cells at peak load conditions.

Dynamic Buffer Handling.

An ATM switch is a network element that provides cell transport, connection control and network management. Therefore an ATM switch is supposed to provide the same functions of routing and buffering as traditional switches, but at a much higher speed. Studies of ATM switch architectures, in recent years, have considered several buffering options[16].

Under burst conditions, statistical multiplexing of ATM cells lead to severe cell loss. Large buffers are costly, create excessive cell delay and increase switch cost. So it is highly desired to use optimal number of buffers and higher utilization of the buffers. The goal is to accommodate more incoming traffic cells from various sources and smooth out of burst arrival rate while limiting the overheads of the switch to a pre determined size. In addition, it reduces the waiting time of a cell in the buffer space. One plausible solution is the dynamic buffering strategy, which applies a threshold-based sharing policy to the buffers and a priority-based cell push out policy to either buffered or incoming cells[2][3][15].

Partial buffer sharing scheme with pushout policy outperforms the other schemes. The performance further enhances when the input cell is made to join the other buffer spaces with least queue length, upon arrival, if the parent buffer is full. If the current buffer also becomes full, the new cell from the current buffer is made to join its buffer by pushing out the last entered cell (belonging to other queues)[1].

Under asymmetric traffic, however, for a system with N ports, for optimal utilization, threshold is imposed on buffer allocation to logical input queues, thus preventing one queue from draining out all the space while throughput is very low due to smaller buffers on other outgoing links [2][13][9][14].

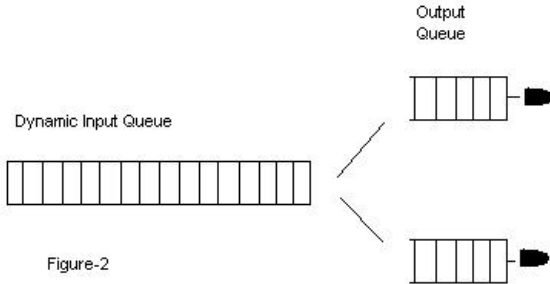


Figure-2

The ATM network considered consists of n identical stages, each having a^{n-1} switching elements (SE). Each SE is an $a \times a$ crossbar, with input buffer modules (dynamic length) to store conflicting cells and output buffer (length two) modules to store unacknowledged cells. Fig-2 shows a sample model of an $a \times a$ proposed shared buffer switch structure. As shown in figure a switching element is composed of an input and output ports. At each input port, a logical input buffer module (IBM) of length B is provided to hold forwarded cells from previous stages. Each output port includes an output buffer module (OBM) of length equal to two[7].

1. Cells are assumed to be of fixed length and are generated randomly at the beginning of a clock period. They are admitted at the IBM of stage 0 only if there is place in the corresponding IBM.
2. The switch uses input queuing and shared buffer. i.e. each input port of the fabric has a long logical queue, but these queues all share the same fabric memory. Each input port has a logical buffer of size B . An arriving cell destined for input port i joins the input queue i from head to tail sequentially. When the queue length of input queue i reaches B (i.e. queue is full), the control logic will choose a shortest-length input queue among the other $a-1$ input queues, said as input queue j . Now input queue j is used to allocate the new arriving cells destined for the input port i , and the new arriving cells destined for input port i will borrow the storage space from input queue j and join to input queue j from tail to head sequentially. The control logic sets a connection between input queue i and the input queue j . When there is no empty space left in both input queue i and input queue j , an arriving cell destined for input port i will be discarded and an arriving cell destined to input port j will take the place of the last cell stored in the queue and destined for input port i .
3. The time needed to allocated space to an IBM is assumed negligible with respect to the cycle timing.
4. The switching network operated synchronously at the rate of τ , which is equal to the sum of the time taken to select a cell and forward it to the next stage IBM.
5. At the beginning of each cycle, cells waiting to be forwarded in the IBMs are checked and if two cells are destined to the same OBM, they are randomly resolved.
6. Cells generated by the cell sources are uniformly distributed over all SE's IBMs. A source of cells is connected to each port of stage 0 IBMs. The rates at which cells are generated at source nodes determine the traffic load.
7. The output ports of the SEs in the final stage of the switching network are connected to sink nodes, Hence, any cell that arrives at those output ports is accepted by the sink nodes with a probability of one.
8. A cell is transferred from the OBM of stage k to the IBM of stage $k+1$ when the buffer space is available for transmission at stage k OBM. A cell is available at stage k OBM when there are two cells in the OBM, or there is one cell and it is not held up.
9. A cell is accepted by the $k+1^{\text{th}}$ stage IBM in a cycle if it is not full or it is full, and a cell in that IBM is able to move forward to an OBM.
10. The SE are $a \times a$ switches which have the capability of connecting an input to one of the output ports labeled 0 and a , depending upon the state of the corresponding value of the destination address.
11. The operation of loading a cell from the IBM to the OBM, and its transmission to the next stage, takes place in parallel with the same cycle. If that cell is accepted in the next stage IBM, it does not contribute towards the delay experienced by the cell in the network.

Intuitively, in order to adapt to different mixes of incoming traffic, the threshold should be changing dynamically, unlike static partitioning. Some kinds of traffic applications, such as video and audio, are very sensitive to delay and cell loss. Under a bursty condition, the more traffic of such type, the switch can accommodate, the better the QoS will be. Yet, the statistical nature of the significant part of the traffic, its burstiness, and its variability combine to pose difficulties in distributing buffers of fixed size. Therefore, in a statistical multiplexing environment i^{th} heterogeneous traffic, before arriving at an optimal threshold, it is quite important to define congestion detection criteria for dynamic threshold updating. The complexity and the performance of the switch mechanism play a critical role in preventive congestion control. Our algorithm prevents the congestion to happen by faster processing, with the help of multiple processes.

Proposed scheme for buffer handling

Two Control Processes pick up the first two cells near the head, identify their output ports and forward them appropriately.

It has been assumed that both the processes take a nearly constant and same time in the identifying the target output port and forwarding the cells to the corresponding output port.

In the current modal, the control and data information is being transmitted in parallel. The control signals exchanged between consecutive stages are changed, whereby the control logic consists of a three state machine, each one working independent of others, dealing with cell transmission, transmission of acknowledgement and acknowledgement reception. In each cycle a cell forwarded from stage k output buffer SE is accepted by stage $(k+1)$ SE if its Input buffer module is not full or stage $(k+1)$ SE. When a forwarded cell cannot be accepted by stage $(k+1)$, it is to be dropped, which results in cell loss, in the absence of retransmission. So each OBM is of length two. As soon as the cell is transmitted to the next stage, the control logic saves the cell in the output buffer module^[1].

The corresponding algorithm using single process has been discussed extensively by A.Subash & E. Koklulkayan[1]. In the proposed method two processes, instead of a single processes are being used to enhance the cell transfer rate by the switch fabric.

Simulation Study

A discrete event simulator for the above mentioned approach has been designed in Java. The cells in the simulator consists of the (a) destination address (b) Time admitted into the network (C) Time released from the network (d) class of the cell. The switching element consists of the following attributes (a) IBM's current length; (b) list of cells in each IBM (c) list of cells in each OBM and the next stage input port number linked to each OBM.

Following assumptions were kept mind, while developing the networking simulator

1. The cells are generated as per the Poisson distribution and stored in a file. For each type of simulation(combination of variables, same file is used).
2. The destination of each stage cycle by the processor is set randomly by a random number generator to simulate uniform traffic.
3. If there is a traffic conflict among cells then FIFO approach is used.
4. The throughput and the delay are measured at each output port of the network, and averaged over the network size and simulation time span to get the normalized throughput and the normalized delay of the network.
5. Cell service at each buffer module is done by using FIFO policy.

The simulation starts by initializing the IBM and OBMs for each Switching element to null.

Results and Discussion

The principal goal of this work is to find an ATM switch which improves the network performance. A new model was introduced in an ATM environment. In this model, more than one processes are involved in forwarding the packets from IBM to OBM at each stage. Besides the performance of the single and multiple processes have been compared using simulation technique. The generated data is so huge that it cannot be displayed in tabular format in this length defined paper. Following terms have been used for comparison with the existing models:

Normalized throughput: This is calculated in terms of the average queue length

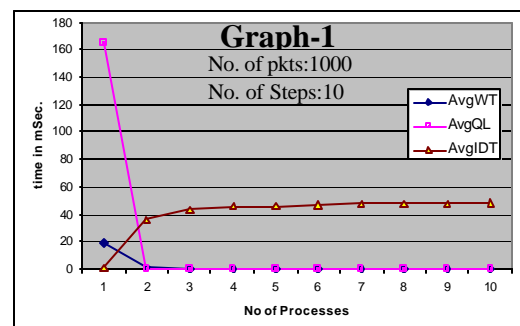
Normalized delay: This is measured in terms of average wait time.

Normalized Ideal Time: This is measured in terms of average time for which a process remains Ideal.

Following graphs display the results.

1. **Graph-I:** Normalized Throughput, Normalized Delay and Normalized Ideal Time for processes Vs Number of Processes in each stage of a switch. In the graph, simulation was done for a traffic load of 1000 packets and 10 steps/elements in the switch.

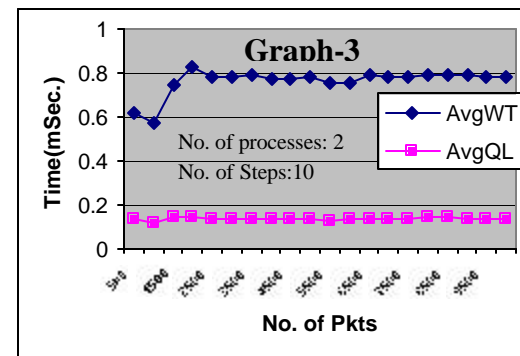
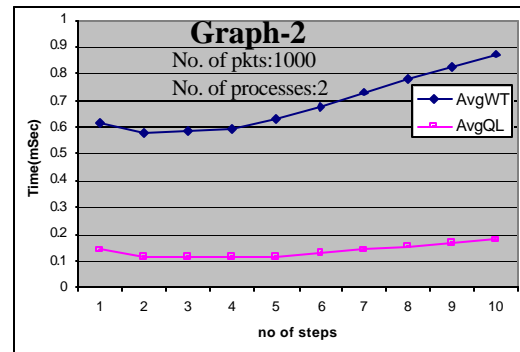
It was found that just two processes are sufficient to forward the cells from IBM to OBM. The AvgWT



and AvgQL was found to be reduced by increasing the processes from one to two, but at the same time AvgIDT increases. No further appreciable change was found by increasing the number of processes, in above mentioned parameters. So in further simulation, only two processes were considered.

2. **Graph-2:** Normalized throughput and normalized delay Vs no of switching elements in a switch. In this graph, the simulation was done assuming a fixed number of packets:1000 and two processes. The graph shows an upward trend in the values of AvgWT and AvgQL with increase in number of steps in switching fabric.
3. **Graph-3:** Normalized throughput and normalized delay Vs Traffic load. In this graph the simulation was done assuming two servers and two step/elements in the switch. The graph shows that even in bursty traffic conditions merely two processes are sufficient enough to handle the situation.

From graph-I it can be easily concluded that just two processes are sufficient for optimal utilization of resources. With the help of this conclusion shown in graph-1, the next two graphs have been drawn to prove that there is very negligible effect of the traffic load and no. of stages in the switch. Only two processes are sufficient to forward the packets from IBM to OBM for optimal utilization.



Conclusion

The proposed scheme discusses the shared buffering along with an approach to include more than one process for achieving the shortest queue length and minimum waiting time. The proposed structure will decrease the number of out-of-order and lost packets for high traffic loads. In addition, an increased network throughput and an increased network performance is achieved. Thus by using the proposed model, a superior and faster ATM switch is achieved. This Switch with the improved performance may be used in wide variety of applications of ATM networks.

References:

1. Abdulhamit Subashi, Etem Koklukaya, Performance improvement of dynamic buffered ATM Switch, Computer and Electrical Engineering 2005, V.31, pp152-165.
2. Badran H., Mouftah H., ATM switch architectures with input-output buffering : effect of input traffic correlation contention resolution policies, buffer allocation strategies and delay in back pressure signal. Comput Networks ISDN System 1994, V. 26, pp1187-213
3. Becker M, Beylot AL., Performance comparison criteria for ATM Switch models. Computer Networks 2000, V. 34, pp85-95
4. Bhogavilli SK, Abu-Amara H., Design and analysis of high performance multistage interconnection networks. IEEE Trans. Computer systems1997, V.46(1), pp110-117.
5. Cidon I., Georgiadis L., Guerin R. and Khamisy A., Optimal buffer sharing, IEEE J.Select. Area Commun, SAC-13,September 1995, pp1229-1240.
6. Dhamdhare D.M. , System Programming and operating system, second rev. ed. , Tata Mc-GrawHill, 2000.
7. Diab H, Tabbara H, Mansour N. Simulation of dynamic input buffer space in multistage interconnection networks. Adv Eng. Software 2000, V. 31, pp13-24.
8. Foschini G.J. and Gopinath B., Sharing Memory optimally, IEEE Trans on Comm: Comm31, March' 1983, V.3
9. Ho JD, Sharma NK. Multicast performance in shared memory ATM switches. Performance Evaluation 2000, V. 41, pp23-36.
10. Huang TY. A new shared buffer packet switch in ATM networks, Comput Commun 2001, V. 24, pp445-51.

11. Kamoun F. and Kleinrock L., Analysis of shared Finite storage in a computer Network Node Environment under General Traffic conditions, IEEE Trans on Comm: Com- 28, July 1980, V. 7
 12. Mark J. Karol, Michel J. Hluchy j and Samuel P. Morgan, Input versus output queuing on a space division packet switch, IEEE Transactions on Communications, Dec'1987:V35;N12;pp 1347-1356.
 13. Papadimitiou GI, Pomportsis AS. Learning-automata based scheduling algorithms for input-queued ATM Switches. Neurocomputing 2000, V. 31, pp191-5.
 14. Prabhakar B, McKeown N. On the speedup required for combined input and output-queued switching. Automatica 1999, V. 35, pp1909-20.
 15. Sharma NK. Effect of windowing policies for input buffered ATM switch. Computer Networks ISDN Systems 1998;V. 30, pp1083-1090
 16. Sharma S., Viniotis Y. Optimal buffer management policies for shared buffer ATM switches . IEEE/ACM Transaction Network 1999, V. 7(4), pp 575-87
 17. Tassioulas L., Chung Y. and Panwar S.S., Optimal buffer control during congestion in an ATM network node, IEEE/ACM Trans Networking August 1994, V.2, pp374-386.
 18. Trivedi, K.S., Probability and statistics with reliability –queuing and computer science applications, prentice hall, Englewood cliffs, 1982.
 19. Wei S.X., Coyle E.J. and Hsiao M.T., An optimal buffer management policy for high performance packet switching , IEEE GLOBE Com'91, December 1991, V.2, pp 924-928.
 20. Yaprak E, Chronopoulos AT, Psarris K, Xiao Y. Dynamic buffer allocation in an ATM switch. Computer Networks 1999, V. 31, pp 1927-33
-