International Journal of Modelling and Simulation, Vol. 20, No. 2, 2000

BUFFER MANAGEMENT SIMULATION IN ATM NETWORKS

E. Yaprak,* Y. Xiao,* A. Chronopoulos,** E. Chow,*** and L. Anneberg****

Abstract

Computer networking communications are now in the era of dedicated, high-speed switched networks. Asynchronous Transfer Mode (ATM) networks are increasing in importance because ATM technology has the ability to deliver a high quality of service, while simultaneously supporting multiple classes of traffic on the same transmission pathway. Traffic management is increasingly becoming an important focus of study in ATM networks. In this context, congestion control through adequate buffering is becoming particularly significant to minimize the probability of cell loss and cell delay when multiple large traffic bursts are received concurrently at a switch [1,2]. This paper presents a simulation of a new dynamic buffer allocation management scheme in ATM networks. To achieve this objective, an algorithm that detects congestion and updates the dynamic buffer allocation scheme was developed for the OPNET simulation package via the creation of a new ATM module. This dynamic buffer allocation scheme utilizes four identified attributes: in-use bandwidth, in-use bandwidth of distinct Quality of Service (QoS) of an incoming traffic, available number of buffers on the corresponding logical queue, and the latest short-term cell arrival rate destined to the corresponding output port. The behavior of the network under bursty traffic conditions was examined.

Key Words

ATM Switch, buffer management, congestion control simulation

1. Introduction

Current technological advances and significant developments in the computer networking industry are positively influencing the global community daily and helping industry migrate to next generation technology. Among the many factors that influence these changes, the emergence of high performance computing platforms for graphics, color, voice, video, and multimedia applications and the exponential growth in data traffic rates appear to be particularly important. This is partly because these changes are creating an interesting problem for traditional data networks. In shared-medium LAN technologies such as Ethernet, Token Ring, and Fiber Distributed Data Interface (FDDI), for example, only a portion of the total network bandwidth is provided for each user. Such bandwidths are becoming insufficient to facilitate communication and the presentation of complex data mainly because of the bandwidth requirements of images and video.

The recent emergence of Asynchronous Transfer Mode technology is becoming a driving technology for high performance computing platforms. Unlike shared-medium LAN technologies in which users compete for bandwidth, the ATM network is capable of dynamically allocating bandwidth as a function of priority. The primary benefits offered by ATM technology over competing technologies are the availability of bandwidth-on-demand for a wide variety of applications (such as data, voice, video) with different latency and delay requirements, and the ability to support all this traffic with a guaranteed QoS.

ATM technology is recognized as the switching and multiplexing solution for Broadband Integrated Services Networks (B-ISDN) and is expected to be the basis for all future high-speed networks. ATM's efficiency can be attributed mainly to two factors: they make use of small fixed-size cells consisting of 5 bytes header and 48 bytes information field (payload), and they deal with the virtual path/virtual channel connection. The absence of variable-length packets in an ATM network eliminates excessive delay variation because the small fixed-size cell enables extremely high switching speeds. Establishing virtual connection by using the address fields in the 5-byte ATM cell header, ATM separates the traffic from the physical channel and allows many users to share links. Over a single virtual path, many individual application streams can be multiplexed together, by allowing the full use of its available bandwidth. Fig. 1 gives the scenarios of VPCs/VCCs multiplexing and VP/VC switching.

ATM technology is able to carry different kinds of traffic over the network with a guaranteed QoS. To achieve this QoS, a traffic contract is secured between the application and the network in which the application informs the network of its anticipated traffic characteristics such as a peak and an average data rate, a maximum delay, and a cell loss probability of each direction of the requested connection upon an initial set-up. The network then takes these traffic parameters and available resources into account in allocating its resources to satisfy the application's

* Wayne State University, Division of Engineering Technology, Detroit MI 48202, USA; e-mail: yaprak@eng.wayne.edu

** The University of Texas at San Antonio, Division of Computer Science, San Antonio, TX 78249, USA; e-mail: atc@ringer.cs.utsa.edu

*** Jet Propulsion Laboratory, Pasadena, CA 91109, USA

**** Lawrence Technological University, Electrical Engineering Department, Southfield, MI 48075, USA; e-mail: anneberg@ltu.edu

(paper no. 1997-013)

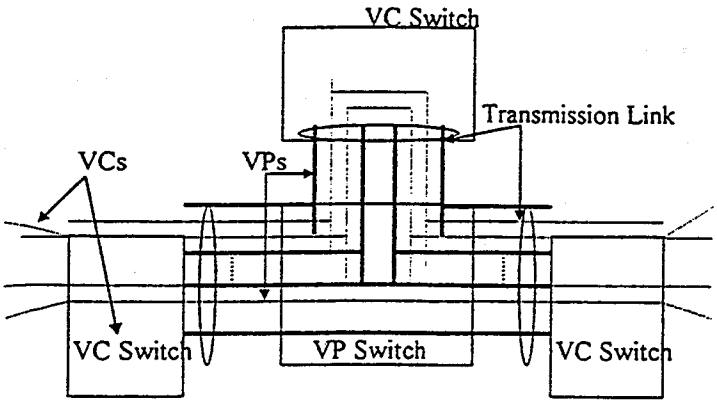


Figure 1. VPC/VCC multiplexing and switching.

requests without adversely affecting existing connections. However, congestion occurs when multiple cells arriving simultaneously through different incoming links attempt to reach the same outgoing link during the same cell slot time. This is because, unlike the deterministic multiplexing where each connection is allocated its peak bandwidth, statistical multiplexing allows several connections which might be very bursty at times, to share the same link based on their traffic characteristics in the hope that statistically they will not all burst at the same time. Thus, the flow rate is within the link bandwidth most of the time. In other words, statistical multiplexing would even allow the sum of the peak bandwidth requirement of all connections on a link to exceed the aggregate available bandwidth of the link under some condition of discipline. In this context, a certain degree of "overbooking" is possible to degrade the QoS, therefore, congestion control is particularly important to providing guaranteed QoS for use in the ATM network.

There are a number of mechanisms (both preventive and reactive) to avoid congestion in an ATM network: for example, Usage Parameter Control (UPC)/Network Parameter Control (NPC), Connection Admission Control (CAC), rate-based control, and so forth. However, congestion control through adequate buffering, is increasingly being used to minimize the probability of cell loss and cell delay. In this case, only one cell is allowed to go through the network, while the others must be stored in buffers. At this time, a switch buffering strategy as well as buffer size become important because buffers are required to secure low cell-loss rate by providing a place to guard against cell loss when the switch is overloaded with bursty traffic. The choice of either of them can have a dramatic effect on the performance of the switch. If cell loss is experienced due to overflowing buffers, this will introduce a degradation in the overall system performance. The goal of this paper is to demonstrate a new design of a threshold based dynamic buffer allocating switch using the Optimized Network Engineering Tools (OPNET) simulation package from Mil.3, Inc. [3].

2. Dynamic Buffer Management

Under bursty traffic, statistical multiplexing of ATM cells will lead to severe cell loss. Huge amount of buffers cause excessive cell delay and switch cost, therefore, an appropriate number of buffers are allocated in a switch and an attempt is made to ensure that they are optimally utilized. The goal is to ensure a fair sharing of the buffer space and to achieve low cell loss even under bursty loading conditions. To achieve this, a feasible solution is to use a dynamic buffering strategy that applies a threshold based sharing policy to the buffers and a priority based cell pushed-out policy to either buffered cells or incoming cells.

There are a number of buffer sharing schemes available through research from the past years, namely: (a) complete sharing where all cells are accommodated if the storage space is not full, (b) complete partitioning where the entire space is divided permanently to all output ports (in fact, it has the same performance as output queuing does), (c) partial sharing/partial partitioning where a number of buffers are always reserved while the rest are partitioned among the output ports and so on [4–7]. Intuitively, the threshold (for a logical output queue) itself should be dynamically changing rather than statically partitioning for the sake of adapting the different mixes of incoming traffic [8–10]. A scheme of complete sharing based on virtual partition will achieve better performance than complete partition [11]. However, the statistical nature of the significant part of the traffic, its burstiness, and variability combine to pose difficulties in deciding on the appropriate partition of the fixed size buffers. Therefore, in a statistical multiplexing environment with heterogeneous traffic, before ending up with an optimal threshold, it is quite important to define a congestion detection criteria for adaptive threshold updating. The complexity and the performance of the switch mechanism play a critical role in such preventive congestion control. Therefore, to detect the congestion and update the corresponding threshold in real-time, several traffic attributes on each outgoing link are taken into account in our algorithm. There are a number of factors that help gauge the traffic volume.

The first two factors, in-use bandwidth and in-use bandwidth of distinct QoS of an incoming traffic, indicate the anticipated volume as well as the QoS requirement of the incoming traffic on each outgoing link in general. The last two factors, available number of buffer on the corresponding logical queue and the latest short-term cell arrival rate destined to the corresponding output port, reveal the real-time evolution state of the switch. This information gives a heuristic estimation of the traffic in the near future. Therefore, threshold updating decision will be made based on this estimation.

The ATM differentiates various kinds of QoS for the traffic in terms of the bounds on the bandwidth and the cell-loss ratio. There is a serious need to incorporate explicit burstiness modelling in the traffic models used for performance analysis. It has been found that the traffic parameters, such as inter-arrival time and loss probabilities are very sensitive to the assumed source characteristic. To simulate the traffic types in general, we would like to categorize them into two classes: real-time traffic, which

The updating processes for thresholds are efficient in the sense that the four factors can all be easily retrieved, and they do not involve complicated calculations. Therefore, they cause little latency over the network. Fig. 4 shows a portion of a C-code to compute the new threshold.

```
/*compute the new threshold*/
if (X==0)
      {
      /* Handle the exception case 1 of no call being routed
      through the link */
      for (port_index = 0; port_index < port_count;
            port_index++)
      {
      /* Obtain the i-th port descriptor pointer. */
      pdesc_ptr = &pdata_ptr->port_array [port_index];

      /* Check whether the port_index is the last one, if
      notassign the threshold = buffer_pool.max/port_count,
      otherwise threshold = the rest of buffer space, it
      might not be as much as others*/
      if (port_index != port_count - 1)
            port_queue [port_index].threshold =
                  buffer_pool.max_bufsize/port_count;
      else
            port_queue [port_index].threshold =
                  buffer_pool.max_bufsize -
                  (buffer_pool.max_bufsize/port_count)*
                  (port_count-1);

      /* set the other field of port_queue[port_index] */
      port_queue [port_index].latest_arrival = 0;
      }
      } /*End of exception case 1 */
else  {
      bufspace = buffer_pool.max_bufsize;
      for (port_index = 0; port_index < port_count;
            port_index++)
      {
      /* Check whether the port_index is the last one*/
      if (port_index != port_count -1)
            {
      if (x[port_index]== 0)
                  port_queue [port_index].threshold =
                  floor(bufspace*DEFAULT_SHARE);
            else
                  {
                  if (x[port_index]/X == 1)
                        port_queue [port_index].threshold =
                  floor(bufspace*(1-DEFAULT_SHARE));
                  else
                        port_queue [port_index].threshold =
                        floor(bufspace*x[port_index]/X);
                  }

            bufspace -= port_queue [port_index].threshold;
            }
      else
            port_queue [port_index].threshold = bufspace;

      /* set the other field of port_queue[port_index] */
      port_queue [port_index].latest_arrival = 0;
      } /*End for */

      }
```

Figure 4. C-code to compute the new threshold.

In addition to the adapting thresholds, a (CAR) is needed to make use of the thresholds. Basically, when A buffer pool is not full, any incoming cell should be accepted. When the buffer pools are all filled in, however, a selective admission and push-out mechanism should be enforced.

If the length of the logical queue, towards which the cell is destined, is less than its threshold, it is obvious that there are some other queues whose length exceeds their thresholds. We denote these kind of queues as Q_{nc} (non-conforming queue) and the remainder of queues as Q_c (conforming queue). In this case, a new cell is buffered by dropping one buffered cell from either Q_{nc} or Q_c. The selection policy is listed as follows with descending priority:

1. a real-time cell from either Q_{nc} or Q_c whose Cell-Delay-Variation (CDV) exceeds the tolerable limit in any kind of queue;
2. a nonreal-time cell from Q_{nc} queue where its position is outside of the threshold of its corresponding queue;
3. a nonreal-time cell in the same queue where the new arrival is destined; and
4. a real-time cell from Q_{nc} queue where its position is outside of the threshold of its corresponding queue.

If the logical queue towards which the cell is destined has alength equal to or greater than its threshold, then the cell is either accommodated or dropped depending on the following situation:

5. if there is a real-time cell in Q_{nc} or Q_c whose CDV exceeds the tolerable limit, then it gets dropped. Otherwise;
6. if there are nonreal-time cells in the same queue, then the newest nonreal-time cell is dropped. Otherwise;
7. if the new arrival is a nonreal-time cell, it gets dropped immediately. Otherwise;
8. if the new arrival is a real-time cell, it is either dropped immediately, if no nonreal-time cell exists in all queues, or replaces a newest nonreal-time cell in some other queue.

5. Simulation Study

Fig. 5 shows the topmost layer of an ATM network model constructed by integrating one source node (f_0) and three sink nodes (f_1, f_2, f_3). Each sink node has its own unique ATM network address as its node serial number in the network. The traffic source generates either real-time or nonreal-time traffic destined to one sink node. The link rate is set to 51.84 Mbits/sec, which corresponds to STS-1. The Segmentation and Assembly Rate (SAR) has been kept very high so that there is no traffic shaping due to the AAL5 layer. Since packet size is fixed at 320 bits, segmentation is not required.

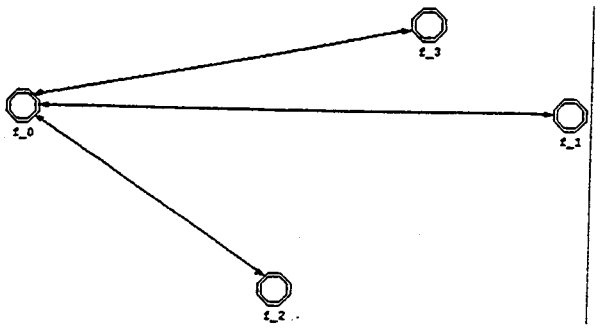


Figure 5. An ATM network model used for this study.

Two distinct models were created by using a static complete partition based on a ATM Switch process in one model, and a dynamic shared buffer based ATM switch process in the other. Switch performance based on static strategy versus dynamic strategy are then evaluated.

Table 1 depicts the typical traffic parameters used in our simulation. Other than packet size, traffic from an individual application is also characterized by call duration time, call wait time, mean packet inter-arrival time,

QoS, and destination. Table 2 shows the buffer allocation in each buffering strategy. The total buffer size for the dynamic buffering strategy switch simulation is 5022 cells. For the static buffering strategy, 5022-cell buffer space is equally assigned to 3 output port queues, and 950-cell space out of 1674-cell space is used for accommodating real-time traffic in each output port. The remainder is utilized for nonreal-time traffic.

Table 1
Traffic Parameter for Investigating the Performance of Static Buffering Strategy Versus Dynamic Buffering Strategy

	traf_src0	traf_src1	traf_src2	traf_src3	traf_src4	traf_src5
mean peak interarrival time	0.00009	0.00001	0.00001	0.000012	0.000012	0.00001
call duration time	0.23	0.2	0.2	0.2	0.156	0.2
call wait time	0.2	0.415	0.51	0.4	0.5	0.32
QoS	nonreal-time	real-time	nonreal-time	real-time	real-time	nonreal-time
destination	1	2	3	1	3	2

Table 2
Buffer Allocation in Two Distinct Simulations

	real-time traffic buffer size	nonreal-time traffic buffer size
output port 0, 1, 2 (static buffering)	950	724
buffer pool (dynamic buffering)	5022	

Since the packet interarrival time is exponentially distributed, by selecting a different seed in our simulations, the mixed pattern of bursty traffic is obtained. Simulation results in terms of cell loss are shown in fig. 6 and fig. 7. Fig. 6 illustrates the cell loss in the dynamic buffing strategy switch, where the threshold updating frequency is $M = 500$, and cell loss happened at output port 1. The nonreal-time cell loss began on the 0.5244 second. When the simulation stopped, the total cell loss was 2212 cells. There were no real-time cells dropped during this period of time. On the other hand, fig. 7 shows the cell loss in dynamic buffering strategy at a frequency $M = 3000$.

6. Conclusion

A new module for use in the OPNET simulation package was written to reallocate the buffer space dynamically. The Proc-C code and the simulation results of this study were shown. Advantages of the threshold based dynamic buffering strategy have been examined. For the purpose of obtaining an efficient and fair use of buffer space in ATM switches, we have simulated a threshold based dynamic buffer allocation scheme in the ATM switch, and then the system behavior under varying on-off bursty traffic patterns was investigated via simulation.

Under a heavy traffic load, the frequency of threshold updating plays a critical role in gaining higher throughput and maintaining fair cell loss among all output ports.

Acknowledgements

This work was supported in part by Jet Propulsion Laboratory, California Institute of Technology under Contract No. 960414. Dr. Chronopoulos acknowledges the NSF grant ASC-9634775, which provided partial support for this research.

References

[1] D.E. McDysan & D. Spohn, *ATM theory and application* (McGraw-Hill, Inc., 1994).

[2] W. Stallings, *Data and computer communications* (MacMillan, 1994).

[3] Mil 3, Inc., ATM Model Description, *OPNET example models manual*, Chapter ATM.

[4] L. Tassiulas, Y.-C. Hung, & S.S. Panwar , Optimal buffer control during congestion in an ATM network node, *IEEE/ACM Transactions on Networking*, *2*(4), Aug. 1994, 374–386.

[5] W. Kowalk & R. Lehnert, The policing function to control user access in ATM networks: Definition and implementation, *IEEE ISSLS '88*, Boston, MA, Sept. 1988, 240–245.

[6] I. Cidon, L. Georgiadis, R. Guérin, & A. Khamisy, Optimal buffer sharing, *IEEE J. on Selected Areas in Comm.*, *13*(7), September 1995.

[7] G. Niestegge, The leaky bucket policing method in the ATM network, *Int. J. of Digital and Analog Comm. Systems*, *3*(2), June 1990, 187–197.

[8] M.J. Karol, M.G. Hluchyj, & S.P. Morgan, Input versus output queueing on a space-division packet switch, *IEEE Trans. on Comm., COM-35*(12), December 1987.

[9] A. Elwalid, D. Mitra, & R.H. Wentworth, A new approach for allocating buffers and bandwidth to heterogeneous, regulated traffic in an ATM node, *IEEE J. on Selected Areas in Comm.*, *13*(6), August 1995.

[10] A. Iwata, N. Mori, C. Ikeda, H. Suzuki, & M. Ott, ATM connection and traffic management schemes for multimedia internetworking, *Comm. of the ACM*, *38*(2), February 1995.

[11] G.-L. Wu & J.W. Mark, A buffer allocation scheme for ATM networks: Complete sharing based on virtual partition, *IEEE Trans. on Networking*, *3*(6), December, 1995.

Biographies

Ece Yaprak received her B.Sc. degree in electrical in 1980 from the University of Michigan-Dearborn; her M.Sc. degree in computer engineering in 1984, and her Ph.D. degree in 1989 both from Wayne State University. Her research interests include network traffic modelling, simulation and high-speed networking, and ATM. Prior to joining WSU, she worked for Western Michigan University, General Electric, Ford Motor Company, NASA Lewis Research Center, and Jet Propulsion Laboratory of California Institute of Technology. Dr. Yaprak is a member of IEEE.

Anthony Theodore Chronopoulos received his Ph.D. degree at the University of Illinois in Urbana-Champaign in 1987.

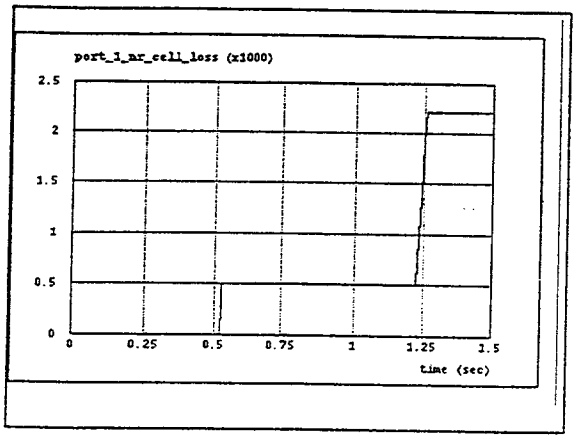
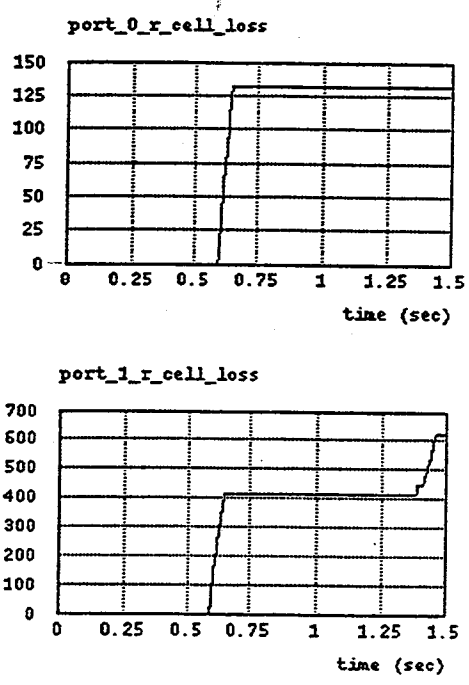


Figure 6. Cell loss in a dynamic buffering strategy switch.

Dr. Chronopoulos has published over 40 refereed papers in journal and conference proceedings in the areas of scientific computation, parallel and distributed computing, and simulations. He is a senior member of IEEE.

L. Anneberg received her B.Sc. degree in industrial and operations engineering in 1979 from the University of Michigan; her M.Sc. degree in computer engineering in 1983 and her Ph.D. degree in 1991 from Wayne State University. Her research interests include visual perception data analysis, electronic heat management, parallel processing, and networking.



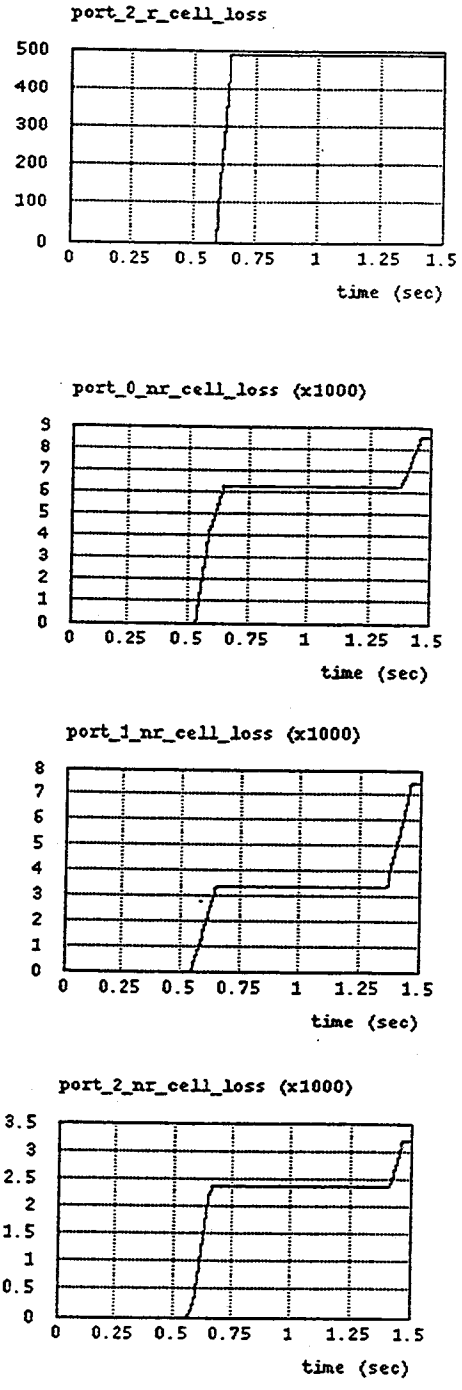


Figure 7. Cell loss under threshold updating frequency $M = 3000$ slots.