Adaptive Buffer Threshold Updating for an ATM Switch

E. Yaprak, Member, IEEE,
A. T. Chronopoulos, Senior Member, IEEE,
Wayne State University
email: yaprak@et 1 .eng.wayne.edu
chronos@cs.wayne.edu

K. Psarris, Member, IEEE
Univ. of Texas at San Antonio
email: psarris @cs.utsa.edu
and
Yi Xiao, Microsoft

Abstract

Efficient and fair use of buffer space in ATM switch is essential to gain high throughput and low cell-loss performance from the network. A shared buffer architecture associated with threshold based virtual partition among output ports is proposed. Thresholds are updated based on traffic characteristics on each outgoing link, so as to adapt to traffic loads. System behavior under varying traffic patterns is investigated via simulation. Cell loss rate is the quality of service(QoS) measure that is used in this paper The study shows that the threshold based adaptive buffer allocation scheme ensures fair share of the buffer space even under bursty loading conditions.

Key Words:
ATM switch, adaptive threshold updating, cell-loss, quality of service.

1 Introduction

The recent emergence of Asynchronous Transfer Mode (ATM) technology provides a unique opportunity to become a driving technology for high speed communication platforms, with line speeds that will scale to interfaces in the gigabit range. Since an ATM based switch has the ability to intermix all classes of traffic (voice, video, data), with different latency and delay constraints each, on the same transmission and switching fabric, with a guaranteed QoS, it provides an ideal platform for supporting multimedia based services.

To achieve this QoS, a *traffic contract* is secured between the host and the network in which the host 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 in each direction of the requested connection upon an initial set-up [5]. The network then takes these traffic parameters and available resources into account in allocating its resources to satisfy the host's requests without adversely affecting existing connections [2],[3],[4],[8],[9]. At this point, even with traffic shaping, ATM networks do not automatically meet the performance requirements set forth in the traffic contract. A potential problem in the ATM networks may be the contention in the switch caused by either a long term or a short term congestion. The first is caused by more incoming traffic than the network can handle, while the second is caused by burstiness in the traffic. In this context, congestion control makes a great contribution for a stable and efficient operation of an ATM network. There are several different mechanisms to avoid congestion: admission control, rate-based control, and resource reservation. In this context, a *congestion control through adequate buffering*, is becoming particularly significant to minimize the probability of cell-loss and cell delay. 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. However, congestion still happens when multiple cells blasting away at the peak rate simultaneously through different incoming links attempt to reach the same outgoing link during the same cell slot time. 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 the 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 em 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. In this paper, *the design of a adaptive buffer allocation through virtual partitioning* , is proposed. Comparisons to the static buffer allocation strategy, via simulations, show significant improvement in QoS.

0-8186-8538-7/98 $10.00 © 1998 IEEE

2 Design Objectives of Adaptive Buffer Allocation

Under bursty traffic, statistical multiplexing of ATM cells will lead to severe cell-loss. Due to high cost of large buffers, which causes excessive cell delay and switch cost, an appropriate number of buffers are allocated in a switch and we seek an optimal utilization of them. The goal is to accommodate more incoming traffic cells from various sources and smooth out the burst arrival rate while limiting the overhead of the switch within a predetermined size. To achieve this, a feasible solution is to use a adaptive buffering strategy which applies a *threshold based sharing policy* to the buffers and a *priority based cell pushed-outpolicy* to either buffered cells or incoming cells.

Choice of a Threshold: There are a number of buffer sharing schemes that have been investigated, namely, (1) **complete sharing** where all cells are accommodated if the storage space is not full, (2) **complete partitioning** where the entire space is divided permanently to all output ports (this is equivalent to output queuing), (3) **partial sharing** or **partial partitioning** where a number of buffers are always reserved while the rest are partitioned among the output ports [1]. Given the scenario of a bursty traffic where the aggregated cell arrival rate exceeds the link transmission rate, the shared buffers in the switch are quickly filled making cell-loss unavoidable. This raises the following question: how to choose the cells to be dropped, while keeping fairness of cell-loss among all output ports when switch buffer is full'? Under *symmetric traffic* (equal traffic on all ports), it has been proved that the optimal sharing policy in shared memory switches with two output ports is the "push-out" type with threshold type (POT) [1]. In other words, whenever the buffer is not full, an incoming cell should be accepted, and once it is full, a cell of type l(or a cell of type 2) is allowed in the switch and a cell of type 2(or a cell of type 1) is pushed-out, if the number of type l(or type 2) cells is below some threshold, say T1, (or T2 for type 2), where T1+ T2 = buffer size. Under *asymmetric traffic*, however, for a N-ported system, the optimal policy is still an open question. With this in mind, we are motivated to impose a threshold on buffer allocation to logical output queues, thus preventing one queue from draining out all of the spaces while having very low throughput due to smaller buffers on other outgoing links [4].

Intuitively, the threshold itself should be dynamically changing, unlike static partitioning, for the sake of adapting to different mixes of incoming traffic. As described previously, 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 types the switch can accommodate, the better the quality of service will be. However, the statistical nature of the significant part of the traffic, its burstiness and variability combine to pose difficulties in distributing 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 a preventive congestion control. Several attributes on each outgoing link are taken into account in our algorithm that detects the congestion and updates the corresponding threshold. The threshold updating processes obtain and use the following *attributes* :

In-use bandwidth (i.e. the effective bandwidth of the arriving cells),

In-use *bandwidth ofdistinct QoS (e.g.* real-time traffic) ,

Available buffer space on the corresponding logical queue,

The latest updated cell arrival rate destined to the corresponding output port.

The first two attributes show the anticipated volume as well as the QoS requirement of the incoming traffic on each outgoing link in general, while the last two reveal the real-time evolution state of the switch. This information, gives a heuristic measure of the traffic in the near future. Threshold updating decision will be based on this prediction. The updating processes for thresholds are efficient in the sense that the four attributes can all be easily retrieved, and it doesn't involve complicated calculation. Therefore it causes little latency over the network.

3 Adaptive Buffer Threshold Updating

Using OPNET package by Mil 3 Inc.[6], a simulation model has been built to investigate the proposed method. OPNET provides a very sophisticated software package capable of supporting simulation and performance evaluation of communication networks. An example of a typical source mode model in ATM is shown in Figure 1. We next present our proposed adaptive buffer threshold algorithm and then show simulation results with OPNET. The study focuses on the ATM switch process residing in the ATM Switch module, the goal is to explore a *heuristic buffer management control scheme* to ensure the maximum throughput within a limited buffer capacity. Without generality loss we restrict our study to a buffer shared by real-time and non-real-time traffic. Our solution is to accommodate more cells of the traffic consists of dynamically sharing the limited buffer space among all output ports while maintaining logically separated output queues in the switch. The OPNET standard ATM model component, i.e. the ATM switch process, which is implemented by a static buffer allocating strategy, is obviously not suitable for our use from this point of view.

The general policies of the threshold adaptation strategy

are:
(i) First In First Out (FIFO) policy is enforced at all times.
(ii) Real-time and nonreal-time cells have equal chances to be transmitted over an outgoing link from the corresponding output port.
(iii) Thresholds of the logical queues for all output ports are equally assigned in the initialization phase.
(iv) Every A4 time slots, thresholds are updated.
(v) Any excess of buffer space from any logical queue goes to the shared buffer pool.
(vi) A Cell-Accommodation-Rule (CAR) is applied whenever a cell arrives at the switch.
In addition to the adapting thresholds, a CAR is needed to make use of the thresholds. When the buffer pool is not full, any incoming cell should be accepted else a selective cell-admission/cell-drop mechanism is enforced. With the assumptions (iv) and (v), this algorithm is able to dynamically adapt the traffic and keep fair cell loss among all output ports. This algorithm differs from the static algorithm, in that cell buffering depends on thresholds of output ports and not on the fixed length of each output queue; any buffer space in the buffer pool is potentially going to be occupied by cells destined to various outgoing links. Therefore, there is no dedicated use of buffer space any more.

With these policies, our algorithm is able to dynamically adapt the traffic and maintain fair cell-loss among all output ports. To measure and predict the incoming traffic the four aforementioned attributes are identified on any outgoing link i (we assume that the total number of links is n).
1. m_{ai} = in-use bandwidth of the non-real-time traffic and $A_i = \frac{m_{ai}}{\sum_{j=1}^{n} m_{aj}}$, where n is the number of links.
2. m_{ui} = in-use bandwidth of the real-time traffic and $U_i = \frac{m_{ui}}{\sum_{j=1}^{n} m_{uj}}$, where n is the number of links.
3. m_{bi} = the reciprocal of the available buffer space of the corresponding logical and $B_i = m_{bi}/\sum_{j=1}^{n} m_{bi}$. (Note, $m_{bi} = 0$ if available buffer space is less than/equal to 0).
4. m_{li} = the updated short-term cell-arrival rate destined to corresponding output port and $L_i = m_{li}/\sum_{j=1}^{n} m_{li}$

For $i = 1, .., n$, the weights of each traffic attribute are $W_{i1}, \ldots, W_{i4}$, where $\sum_{j=1}^{4} W_{ij} = 1$. Thus, the new threshold R_i of logical queue *i* for the outgoing link ***i*** is adjusted by the following formula:
$R_i = S_{buff} * (X_i/\sum_{j=1}^{n} X_j)$, where S_{buff} is the size of the shared buffer space, and $X_i = A_i * W_{i1} + U_i * W_{i2} + B_i * W_{i3} + L_i * W_{i4}$.

Exception cases happen when there is no incoming traffic or no call being set up during certain threshold updating interval time at all output ports (or over all outgoing links). The following polices are used to deal with these cases:
1. Let K_i be any one of A, U, B and L, and let k be any one of a, u, b or l. If $K_i = \sum_{j=1}^{n} m_{kj} = 0$, then the corresponding K_i is assigned the value 0.
2. If $\sum_{j=1}^{n} X_j = 0$, then the thresholds of the logical queues for all output ports are equally assigned.
3. If any $X_i = 0$, then the threshold R_i is assigned a default share of S_{buff}, and the rest of the buffer space, $S_{buff} = S_{buff} - R_i$, is distributed by the above formula.

If the length of the logical queue, where the cell is headed to, is within its threshold, we term it conforming queue (Qc) (else nonconforming queue (Qnc)). The Cell-Delay-Variation (CDV) is a parameter showing how much the cels are delayed than the nominal interval [5]. When the entire buffer pool is full a new cell is buffered by dropping one buffered cell from either Qnc or Qc. The selection policy is listed as follows in descending priority:
(i) A cell from either Qnc or Qc whose CVD exceeds the tolerable limit. **(ii)** A nonreal-time cell from a Qnc queue, if its position exceeds the threshold of its queue. **(iii) If** there are nonreal-time cells in the same queue, then the last-in nonreal-time cell is dropped. **(iv)** A real-time cell from a Qnc queue, if its position exceeds the threshold of its queue. (v) If the new arrival is a nonreal-time cell, it gets dropped immediately. **(vi)** If the new arrival is a real-time cell, it either is dropped immediately, if no nonreal-time cell exists in all queues, or replaces the last-in nonreal-time cell in some other Qnc queue.

4 Simulation Study

Fig. 1 shows a typical ATM network source node model. Fig. 2 shows the topmost layer of an ATM network model constructed by integrating one traffic 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 **nodes.** Like the parameter used in [7], the link rates still have been set to 5 1.84 Mbits/sec, which corresponds to *STS- 1*.

TSi= traffic source i, TC = Traffic Characteristics, 10^{-4}

I

TC	TS0	TS1	TS2	TS3	TS4	TS5
MT	0.9	0.1	0.1	0.9	0.12	0.12
CD	0.23	0.2	0.2	0.2	0.156	0.2
CW	0.2	0.415	0.51	0.4	0.5	0.32
TT	NR	R	NR	R	R	NR
D	1	2	3	1	3	2

Table 1. Traffic parameters.

Let SBS = Switch buffer size , RT= Real-time Traffic, NRT= nonreal-time Traffic, OP_ST= Output Port $i=$, 1,2,3 (static buffering) buffer space, SBP-DB = Shared Buffer Pool (adaptive buffering), Table 2 shows the buffer allocation in each strategy.

SBS	RT	NRT
OP_ST	950	724
SBP_DB	5022	5022

Table 2. Static/Adaptive buffer allocation in simulation

The total buffer size for the simulation of the adaptive buffering strategy switch 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 accomodating real-time traffic in each output port, the rest for nonreal-time traffic. Since the packet inter-arrival time is exponentially distributed, by selecting different seed in our simulations, the mixed pattern of a bursty traffic is obtained.

We show simulations to compare the adaptive versus the static algorithms in terms of cell-loss. We chose the simulation time = 1.5 seconds and the seed = 5000. Fig. 3 shows the cell-loss in the static buffering algorithm. Cell-loss occured at output ports 0 and 1 and began at time = 0.450014 second. In total 1811 real-time cells at output port 1 were lost. And in total 1914 and 5643 nonreal-time cells were lost at output port 1 and 0 respectively.

Fig. 4 shows the cell-loss in the adaptive buffering strategy switch, with threshold updating frequency M= 500. Cell-loss occured only at output portl. The nonreal-time cell-loss began at time = 0.5244 second. When the simulation ended the total cell-loss was 2212 cells and no real-time cell was dropped.

Thus, this simulation shows that the adaptive buffering algorithm is superior to the static algorithm.

5 Conclusion

For the purpose of getting efficient and fair use of buffer space in ATM switches, we have proposed a threshold based adaptive buffer allocation scheme in the ATM switch. System behavior under varying on-off bursty traffic pattern is investigated via simulation. The results show that the adaptive buffer allocation strategy benefits substantially from the dynamically updated thresholds, therefore it is superior to the static complete partitioning scheme in the sense of low cell-loss. Under a heavy traffic load, the frequency of the threshold updating plays a critical role in gaining higher throughput and in maintaining fair cell-loss among all output ports.

Acknowledgement: Some comments of anonymous referees, which helped improve the presentation of some parts of the paper are gratefully acknowledged. This work was supported in part by NSF Grant ASC-9634775.

References

[1] I. Cidon, L. Georgiadis, R. Gue'rin, and A. Khamisy, "Optimal Buffer Sharing", IEEE Journal on Selected Areas in Communications, Vol. 13. No. 7, September, 1995, pp. 1229-1239.

[2] A. Elwalid, D. Mitra, and R. H. Wentworth, "A New Approach for Allocating Buffers and Bandwidth to Heterogeneous, Regulated Traffic in an ATM Node", IEEE Journal on Selected Areas in Communications, Vol. 13. No. 6, August 1995, pp. 1115-l 127.

[3] R. Jain, "Congestion control and traffic management in ATM networks: Recent advances and a survey", Computer Networks and ISDN Systems, 28, 1996, pp. 1723-1738.

[4] M. Katevenis, I? Vatsolaki, A. Efthymiou, "Pipelined Memory shared Buffer for VLSI Switches", Computer Communication Review. Vol. 25. No. 4. Oct., 1995. Proceedings of ACM SIGCOMM'95, pp. 39-48.

[5] D. E. McDysan and D. L. Spohn, "ATM Theory and Application", McGraw - Hill, 1994.

[6] "OPNET Modeler: Modeling and Simulation kernel", Mil 3, Inc. , 1996.

[7] S. Roy, E. Yaprak, J. Liu, E. Chow, R. Markley, "ATM Switch Modeling and Simulation", The International Association of Science and Technology for Development (IASTED) on Networks, January, 1996, pp. 187-190.

[8] L. Tassiulas, Y. C. Hung, S. S. Panwar, "Optimal Buffer Control During Congestion in an ATM Network Node", IEEE/ACM Transactions on Networking, Vol. 2, No. 4, 1994, pp. 374-386.

[9] G.-L. Wu, J. W. Mark, "A Buffer Allocation Scheme for ATM Networks: Complete Sharing Based on Virtual Partition", IEEE Transaction on Networking. Vol. 3. No. 6. December 1995, pp. 660-669.

Figure 1. A typical source node model in an ATM network

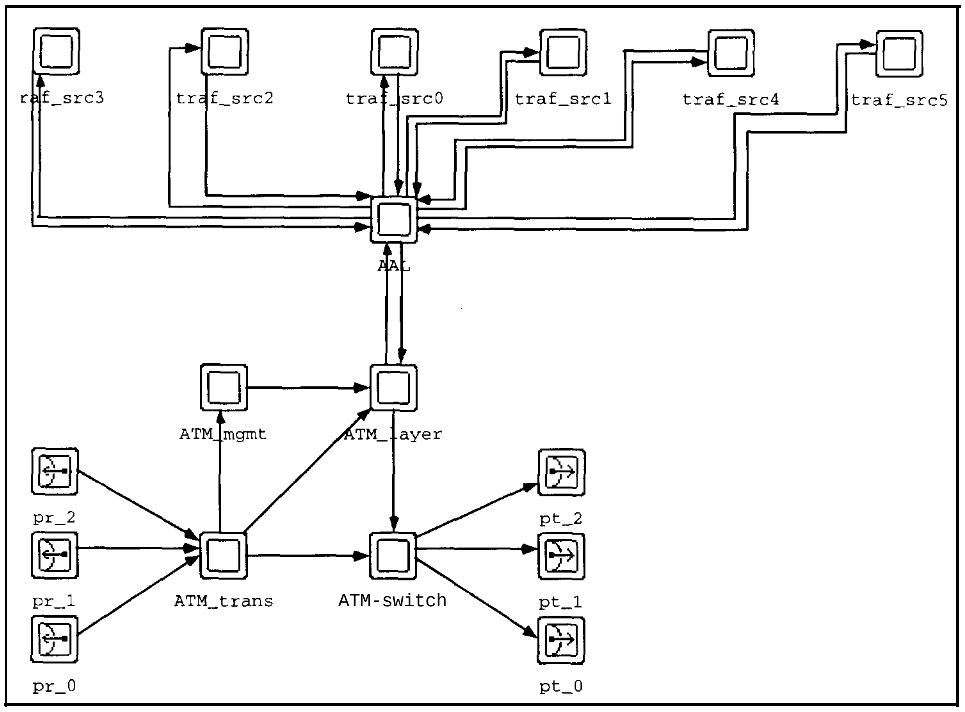


Figure 2. An ATM network model used in the simulation

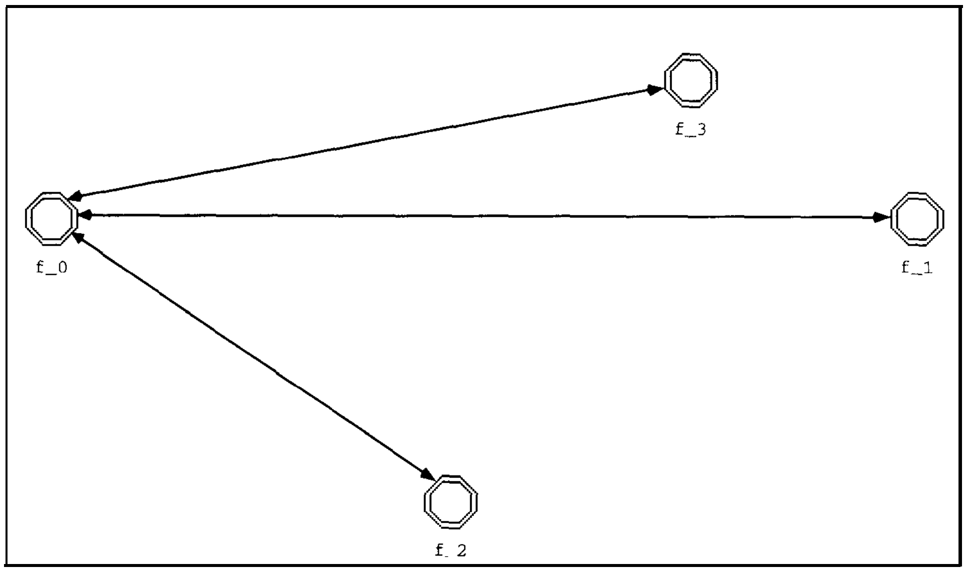


Figure 3. Cell loss in the static buffering algorithm

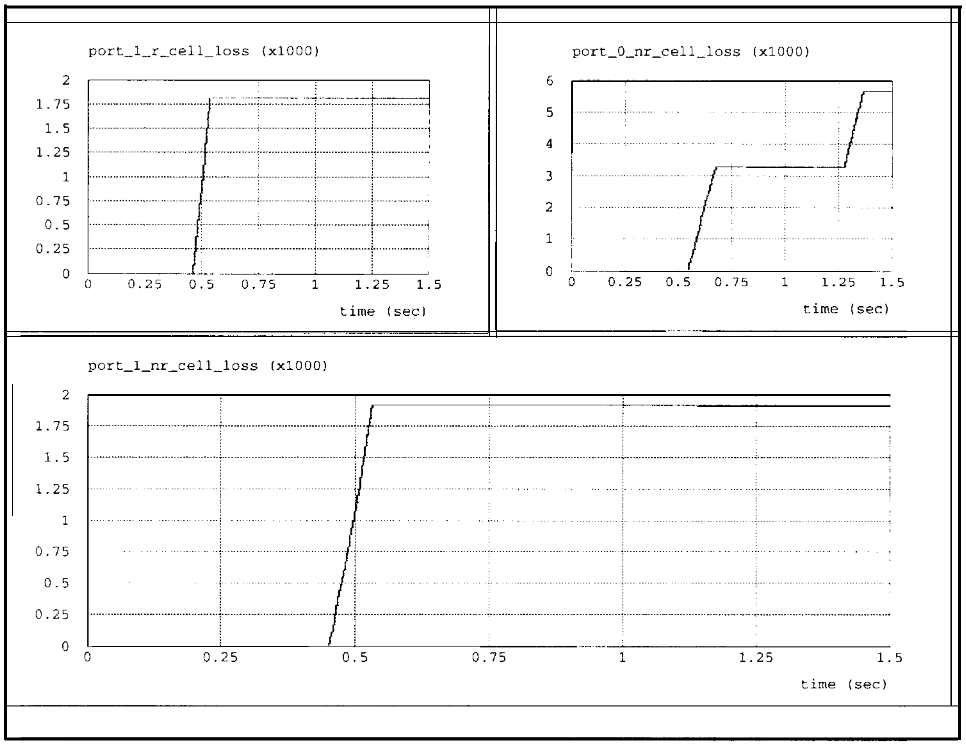
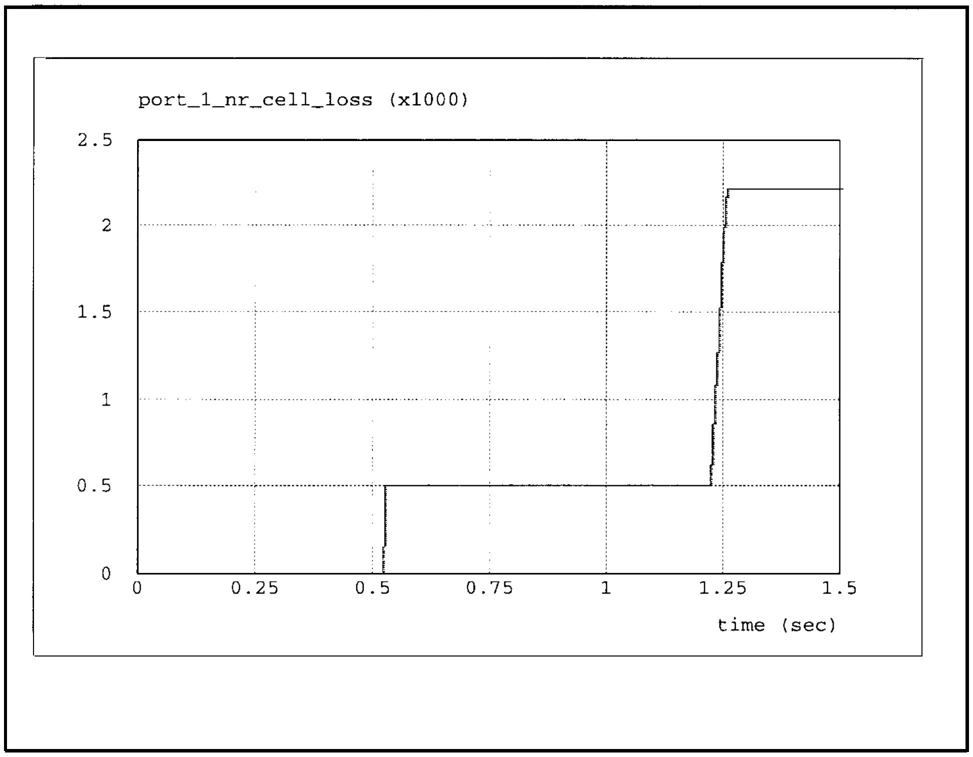


Figure 4. Cell loss in the adaptive buffering algorithm


port_1_nr_cell_loss (x1000)
time (sec)