Increasing Throughput in ATM/B-ISDN Using Simple Buffer Management and Selective Discarding

Samir Chatterjee*
Computer Information Systems Department
Georgia State University
Atlanta, GA 30302.
{schatter@gsu.edu}

Wen Xiao
Lucent Technologies
Alpharetta, GA 30202
{wxiao@lucent.com}

Abstract

If the reliability provided by the network is lower than that requested by the application, the error and flow control systems in the end-nodes must make up for the difference. While most applications use retransmission based schemes to recover lost data, ARQ based closed loop techniques may not be suitable for many new applications. Time sensitive information requiring no-loss of data would constitute a major segment of the emerging traffic (interactive web, videophones, CSCW) that typically employ Forward Error Correction (FEC) codes. Such codes send redundant information which help to recover lost data without retransmission. This paper shows that just plain FEC is not enough when one considers the fragmentation effects that takes place when legacy networks interwork with ATM. Simple yet powerful buffer management techniques can yield higher throughput thereby reducing block loss rates. A combination of a gatekeeping operation along with janitor routine that cleans up the buffer to preserve frames on an end-to-end basis is proposed. A simulation study demonstrates positive results. The proposed schemes even point towards efficient sizing of ATM switch buffers which guarantee no-loss transport of real-time information.

Keywords: ATM networks, internetworking, forward error correction, congestion control, buffer management.

1.0 Introduction

Two significant changes are taking place in the computer communication arena. The first is the rapid standardization and subsequent deployment of Asynchronous Transfer Mode (ATM) networks which has been internationally chosen as the switching architecture for Broadband-ISDN [1]. The other is the changing role of the Internet which is gearing up to provide multimedia services in which real-time conferencing (voice, video, imaging) seems to be viable over IP along with a suitable light-weight adaptive transport layer protocol [12]. Our focus in this paper is on the intricate interworking of these two important technologies and more specifically, we consider mechanisms to control end-to-end packet-loss rates of real-time applications over this mix of IP, legacy LANs and ATM at the wide area.

ATM is a switching and multiplexing mechanism operating over a fiber based physical network such as SONET. It uses a cell (53 byte packet with 5 byte header) as its basic switching element and all information types (voice, data and video) are transported inside the cell. The biggest advantage of ATM is in its ability to do statistical multiplexing (rather than synchronous TDM) and thus can effectively handle bursty, VBR and CBR traffic types. It is primarily a connection-oriented technology using a combination of virtual circuits and virtual paths to establish an end-to-end connection. Wide area ATM standards are coming from CCITT (now called ITU). The ATM forum has been established to speed up the deployment of ATM as a high-speed LAN technology. The two major contributions of this forum has been the LAN emulation service and rate-based congestion control for best-effort traffic also called available-bit-rate (ABR) in the ATM forum [13].

* This work was partly supported by a research grant from the College of Business Administration at Georgia State University.

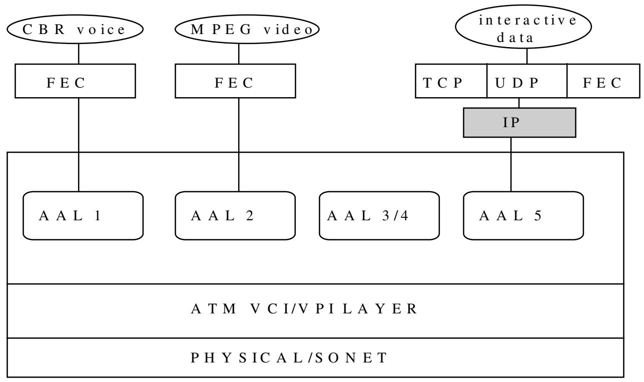


Figure 1: ATM/B-ISDN Protocol Reference Model

The near future will very likely have a communication infrastructure in which all existing LANs (also called legacy LANs) will interoperate over ATM networks at the backbone which provides high-speed wide area services [3]. Since businesses and users have heavily invested in LANs (Ethernet, Token-ring, FDDI), there is no doubt that we will witness a long period of co-existence of legacy LANs with ATM. These diverse networks will then connect to each other with routers and each host and the router will need to run IP protocol to achieve internetworking. The details of IP over ATM are currently being worked on within the IETF group [14].

While the small cell size of ATM provides numerous advantages like easy switching at gigabit speeds, reduced serialization delay, deterministic delay bounds for real-time traffic, this very same small size poses serious limitations in IP over ATM. All legacy frames have to be segmented into tiny cells and then again reassembled back into the higher layer protocol PDU. While this operation of segmenting and reassembling is not a problem (actually done in ATM AAL 5), but during congestion, a loss of one cell may render the entire higher layer AAL5 packet useless. Lost packets are usually retransmitted. TCP at the transport layer uses retransmission with time-out while UDP does not. In fact it has been observed that very low throughput is achieved due to wasted bandwidth as the congested link transmits cells from "corrupted" packets, i.e., packets in which at least one cell is dropped by the switch.

If the reliability provided by the network is lower than the reliability requested by an application, the error and flow control system in the end-nodes must make up for the difference. Two basic mechanisms available to improve reliability are *Automatic Repeat reQuest (ARQ)* and *Forward Error Correction (FEC).* ARQ is a closed-loop technique based on retransmission of data that were not correctly received by the receiver. Such schemes incur overhead in maintaining status information and each retransmission causes a round-trip delay. *ABR* class of traffic (best-effort traffic such as TCP/IP) use ARQs and mechanisms to improve throughput in internetworks (IP over ATM) have been demonstrated by Romanow and Floyd's classic *PPD* and *EPD* work [11].

ARQ may not be applicable for transmitting data from applications with low latency requirements. Multimedia applications (voice, video and animation), remote sensing systems, "push-technology" based web applications need light-weight adaptive protocols which should guarantee the time-critical nature and strict delivery requirements of the application. Time-based information must be presented at specific instants to convey its meaning, i.e., the information has a time component. ATM networks have two classes of service called *CBR* and *VBR* (specifically *real-time*) that requires information delivery for immediate consumption. For time-based and real-time applications, retransmission is just too costly and not possible for several valid reasons (since preserving timing and synchronization is a nightmare). Under such situations, the ATM network must provide guarantees to preserve throughput as much as possible. But as these networks experience congestion, packet preservation without retransmission is extremely challenging. With the future clearly pointing towards interactivity, a new class of traffic that will become dominant is real-time data which requires low delay and extremely low loss-rates (interactive web, videophones, CSCW). FEC is an alternative to ARQ that is well suited for operation in high bandwidth-delay product networks and the above mentioned new class of traffic [6, 7, 9]. FEC involves the transmission of redundant information along with the original data so that if some of the original data is lost, it can be reconstructed using the redundant information. In this paper, we propose the use of FEC within a transport layer protocol (see BISDN protocol model in Fig. 1) and present a novel selective discard and simple buffer management technique to increase end-to-end packet throughput (dynamically and in real-time) in an ATM internetwork [10].

In Section 2, we state the basic nature of the problem that we address here. Section 3 reviews some congestion control techniques and traffic management issues in ATM networks whose modest understanding is essential for the theme of this paper. Section 4 presents our buffer management algorithms. We build a simulation model in section 5 to investigate the throughput behavior of such internetworking systems. We also present a theoretical analysis of our algorithm. In Section 6, we discuss some implementational issues from vendor standpoint. Finally we conclude this paper in Section 7 with pointers to future work.

2.0 Problem statement

We consider the typical internetworking environment as shown in Fig. 2. Existing LANs (Ethernet, Token-ring or FDDI) connect to an ATM backbone with Interworking Units (IWU) that run the necessary IP over ATM routing software. The hosts that are involved in the communication are attached directly to their respective LANs. They transmit interactive (or multimedia) data inside legacy packets. Upon entering the IWU[1], each IP datagram is encapsulated in a AAL5 packet that is fragmented into 53-byte cells and routed to the ATM switch at OC-3 speeds to be transmitted over the backbone. The IWU at the destination side (and its corresponding AAL5 layer) will have to receive all cells to reassemble them into proper higher layer packets required at the destination LAN site. The following observations on this setup desire further probe:

- First notice that the source and destination sites deal with large packets while ATM backbone deal with small cells. While segmentation and reassembly functions can be handled easily by proper adaptation layer protocols, losing one or more cells of a packet may render the higher layer packet useless.
- The backbone ATM network is typically a public WAN in which congestion is very likely to happen. Congestion leads to buffer overflow and hence reduces throughput. As cell loss starts to happen, more end-to-end packets are rendered useless. ARQ based retransmission tend to worsen the situation. Mechanisms to prevent congestion typically employ strict rate-based controls but they have been shown to have oscillatory effects during which the buffer builds up beyond a certain chosen threshold which signals some discard scheme to be operational on each arriving cells until the congestion subsides. A policing scheme along with selective discard is desirable along with rate enforcement.
- When retransmission is not cost effective or simply just impossible, real-time transport protocols employ Forward Error Correction codes. This groups a number of packets into blocks and appends a redundant packet to every block. Any lost data packet can be recovered on the fly at the destination site using FEC coded packets. The amount of redundant information is typically kept small, so that FEC is efficient. When FEC is used, there are two antagonistic effects at work: the redundant information due to FEC helps to recover part of the losses; while the additional data due to FEC increase the overall load, which makes the loss-rate worse. FEC is only effective if recovery prevails. A necessary condition for FEC to be effective is for packet deletions to be dispersed over several blocks.

We show in this paper for homogeneous traffic that, even with FEC recovery schemes, random dropping within the ATM backbone due to congestion can affect the end-to-end throughput. We propose a simple flag based scheme that keeps track of the cell-loss in a packet and can work in conjunction with the output buffer manager of an ATM switch to significantly increase the end-to-end throughput. Note that this is an optional service that a connection may elect to use. The combination of buffer management and FEC at the higher layer may actually lead to a no-loss scheme for real-time traffic.

3.0 Congestion control in ATM: An overview

Congestion control in ATM networks is very challenging because of the diverse nature of QOS to be guaranteed for voice, data, and video traffic [2, 4, 5]. It is widely held in the research community that congestion control will ultimately decide the viability of ATM technology. Due to the wide variety of traffic classes and high speed of the network, no single mechanism can achieve full control in such networks. Thus a combination of control policies must be used. The ATM Forum have collectively called these mechanisms as traffic management functions. The popular techniques that have been adopted include Call Admission Control (CAC), Traffic Shaping, Usage Parameter Control (UPC), Scheduling and Queuing, Priority Control (using the CLP bit) and finally Selective Discarding. Our paper deals with selective discarding but it works heavily in conjunction with UPC techniques. For ABR traffic, a rate-based closed loop flow control technique have been recently adopted [16].

---

[1] It is assumed that the IWU has sufficient buffer space to hold all arriving data packets.

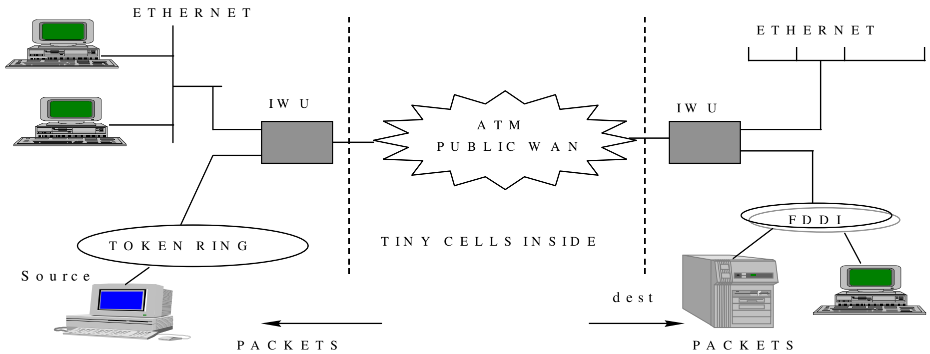


Figure 2: The internetworking environment of legacy packet transmissions over ATM backbones

UPC monitors and controls traffic at network entrance. Each switch is equipped with monitoring modules which has two primary functions: cell marking and cell discarding. The cell marking is done by the marker module and is achieved by setting the cell loss priority bit inside the cell ({CLP=1}). A marked CLP bit signifies that this cell may be dropped in events of congestion. CLP bit marking is usually implemented at the user-network interface (UNI) of an ATM network. Marking may happen at one node but the cell may be allowed to proceed to a subsequent node where it might be dropped if congestion exists. Each marked cell is fed to a selective threshold discarder module. Inside this module a decision is made whether to transmit this marked cell or drop it. The dropping decision is based on a chosen measure of congestion (MOC) value. The MOC is usually chosen to be some factor of the buffer contents at a given time while the comparing threshold is a fixed percentage of the buffer capacity.

Fiber-optic links provide very low-loss and error properties over long distances. The major form of data error in such transmission facilities mainly arises due to cell loss from overflowing buffers as switching and processing delays are encountered. When a packet is lost, the easiest way to recover them is by retransmission based on a time-out strategy. This is typically done in many transport protocols (such as TCP, SPX, DECNET). We consider retransmission to be very expensive in case of real-time communications. Interactive data sessions (games played over a network), a WWW client-server session (in a stock-exchange market) demand fast response time. So retransmission can affect throughput and performance.

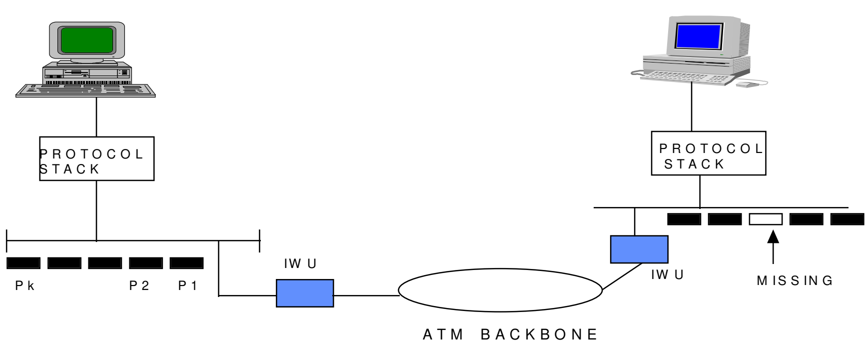


Figure 3: Packet streams through a backbone ATM network.

Forward Error Correction Coding: FEC schemes allow receivers to recover lost data packets without retransmission. We shall assume that data packets are sent out in a sequence with numbers, which are already required in many protocols[2]. Since packets are sent out in ascending order, the recipient can identify missing packets by gaps in the arriving sequence. Thus a missing packet can be considered as a sequence of bit-erasures, whose exact bit location is known. In the example of Fig. 3, the sender transmits a block of K packets. After passing through the ATM backbone WAN, one packet is lost (since some cells within this packet are dropped) and at the receiver side we see an erasure for packet 3, since this number in the sequence is missing. FEC codes can recover such erasures efficiently.

The technique is to group packets into fixed size blocks. Assume that the sender chooses a block size of K packets. To each block, a parity packet is added. If one data packet is lost at the destination side, it can be recovered back from the (K-1) data packets and the parity packet. Let us assume that each packet is *m* bits long. To each block of K data packets, the source adds an m-bit parity packet, whose i^{th} bit is given by:

$$c_{K+1,i} = (\Sigma^{K}_{j=1} c_{j,i}) \; mod\, (2), \; i = 1, 2, ..., m$$

The parity packet can be easily generated using an encoder of m exclusive-or (XOR) gates. The recipient of the packet sequence employs a similar decoder structure and can recover a single erasure. Note however that to decode successfully, it is necessary that erasures be dispersed over different blocks, since multiple erasures over the same block cannot be recovered. To recover more than one missing packet in a block, more FEC coded packets must be sent from the senders side but the complexity of the decoder and encoder circuitry also increases substantially. It is our assumption here that at any point in the system, a single level FEC coding is used [10]. For increased reliability, we also assume in our simulations that a cell level FEC is working within the ATM backbone. Note that FEC parity packets are an overhead which may add to congestion. Since the fiber links and ATM networks promise unlimited potential bandwidth, this small overhead congestion factor is of minor significance[3].

4.0 Buffer management

We present our proposed buffer management techniques to preserve the throughput in an end-to-end internetwork system in two parts: the scheme and then the actual algorithm. The following definitions and notations will be used throughout the rest of the paper while describing the algorithm.

F	the MTU (in bytes) of a physical frame
B_i	ith. block of packets
K	number of packets in a block (block size)
L	number of ATM cells from one packet
B	buffer size in cells
C	channel capacity of ATM link (in Mb/s)

4.1 The scheme

Before we can present our algorithm, we need to carefully examine the protocol conversion that takes place inside the IWU. The FEC recovery schemes and the protocol conversions are shown in Fig. 4. The source sends blocks of K packets along with an FEC coded parity packet. Blocks B_i and B_{i-1} are only shown. At the IWU, each packet gets fragmented into several 53-byte cells as shown. Ethernet frames typically have an MTU size of 1500 bytes which would generate about 32 ATM cells. As cells enter the ATM network, due to UPC and policing techniques, some cells would be marked and dropped at random depending on the MOC at the switch. The cells with an “X” in Fig. 4 represent marked cells and may get dropped. Losing a single cell from a block of 32 ATM cells can be recovered if we assume ATM level FEC[4] to be working. But if more than one cell is lost, then the higher layer packet is rendered useless. If dropping is random to the extent that more than one higher layer packet in the same FEC block is lost, then the whole block becomes useless as such losses cannot be recovered by single level packet FEC coding.

---

[2] Sequencing in ATM is done using an SAR protocol which requires only 3 bytes overhead per cell and can be implemented in hardware [8].

[3] However in the simulation experiments, we do consider the effect of this overhead in congestion and throughput measures.

[4] We assume a single level FEC to be working both at block level and cell level.

The key idea that we employ here is to first distribute the losses over multiple packets in different blocks and this is achieved by localizing cell dropping to one packet as much as possible so that other packets and blocks can remain uncorrupted. As shown in Fig. 4, packet P_1 corresponding to block B_i generates 32 ATM cells (c_1 , c_2 , ..., c_{32}). If more than one cell is dropped in this sequence, then we lose the entire packet P_1. Now if cells corresponding to packet P_k , also of block B_i , are lost, then the packet P_k is also rendered useless. Neither P_1 nor P_k can be recovered in this case resulting in the loss of an entire block. Thus when dropping decisions are to be made, we should try to localize cell dropping confining them to one packet per block as much as possible. We must search for eligible candidates that can be dropped without affecting throughput loss. Cells residing in the buffer from a packet that has suffered too many losses or cells from uncorrupted packets (i.e., those which have not lost even a single cell) are eligible candidates[5]. Also, packets belonging to corrupted blocks (i.e., lost more than one packet) are good candidates. Thus we add intelligence to the buffer manager which now deletes candidate cells and preserves cells from uncorrupted packets and blocks. The next section discusses how to keep track of the correlation and the actual algorithm.

4.2 A gatekeeper plus janitor (GPJ) management algorithm

To keep track of the losses within a frame and the block, we use two flags. The Frame Cycle Flag (FCF) denotes whether a frame (or packet) has suffered any loss yet. It is initialized to 1 signifying that every frame can lose one cell since that can be recovered by ATM level FEC within the WAN. The other flag Block Loss Flag (BLF) keeps track of whether the block has suffered any packet loss. It is also initialized to 1. Both FCF and BLF flags are reinitialized when either the end-of-packet bit[6] is set in a cell (signifying the next arrival is a new packet) or the parity packet ends (signifying the start of a new block). The re-initialization can be done dynamically at run-time.

The flowchart of the algorithm is shown in Fig. 5. Our proposed scheme is both adaptive and dynamic. The buffer manager operates in two states: a *gatekeeper* state and a *janitor* state. The gatekeeper operates between a range of load values that we call the lower and upper threshold. Whenever the load in the buffer exceeds the upper threshold[7], the gatekeeper becomes active when it filters any droppable cell preventing them from entering the buffer. Its main job is to ensure that only non-droppable cells can get past this stage and these must be preserved. At the same time, a janitor routine will sweep all the eligible cells in the buffer to make room for all "*good cells*" that are passed by the gatekeeper.

When a cell arrives, the CLP bit is checked to see if this cell has been marked (see Fig. 5). If CLP bit is not set, it is admitted into the buffer. Otherwise, the current load is measured against the chosen threshold. If the load is below, then the arriving cell is admitted. If the load is greater than threshold, then the gatekeeper becomes active. It tries to filter every droppable cell by checking if the FCF > 0. If it is, then the cell is discarded and FCF is decremented. If not, then the gatekeeper checks for the Boolean condition (FCF = 0 && BLF =1). When this condition is true, it means that the block has not suffered any loss yet and hence the cell (and all the following cells which belong to the same packet) can be discarded and BLF is properly updated to reflect the fact that this block has suffered one frame loss and any future packet loss must be prevented. When the above condition is false, then another Boolean condition (FCF = 0 && BLF = 0) is checked. If this condition is false which means that this cell belongs to an uncorrupted packet, then the arriving cell can be discarded. But when this condition is true, it means that this is a critical cell for the frame and the block and it must be preserved.

As shown in the flowchart, the janitor does a quick salvaging by searching for a cell that has CLP = 1 and FCF > 0. If no such cell can be found, then a cell with CLP = 0 and FCF > 0 is searched and dropped if found. If any one of such cells can be found, it is removed and the new cell is admitted into the buffer. At this time, the janitor routine becomes active which performs a cleanup operation by serially removing all (CLP=1 & FCF >0) cells, (CLP=0 & FCF>0) and finally sequentially searching for cells with BLF > 0. An entire packet from every VC is deleted. Note that the janitor is an expensive procedure as it involves CPU processing time and use of system resources. It is performed only once until the MOC goes below the upper threshold. Note that after the janitor has cleaned up the buffer, the gatekeeper prevents any droppable cell from entering the system. Hence the janitor does not have to do any further work. We switch off the gatekeeper when the load

[5] Using the EPD scheme [11], such candidate cells do not arise as they are always prevented from entering the switch buffer. We feel that this is too much restrictive on the network part since at times when there is no congestion, every cell should be admitted.

[6] This is set in the last cell of an AAL5 packet when transporting IP datagrams over ATM.

[7] The values of upper and lower threshold chosen in simulations were 80% and 40% of buffer capacity.

becomes less than lower threshold. This range of upper and lower thresholds can be used to control the operation of the janitor and gatekeeper and it can be dynamically changed to adapt according to the current load value in the IWU.

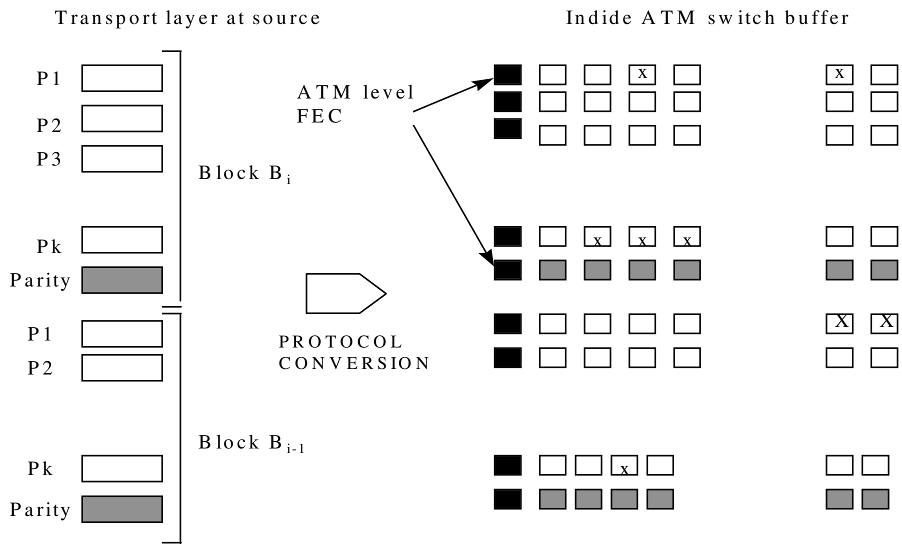


Figure 4: Conversion of packets to cells inside ATM node

5.0 Simulation Model

5.1 Setup: This section describes the simulation environment. We have built a discrete-event simulator to study the behavior of our proposed buffer management schemes with FEC. The simulation model is shown in Fig. 6. For simplicity we consider 10 sources that employ FEC coding. The IWU is modeled by an input packet queue where arriving packets temporarily reside before they are fragmented and cells are transmitted to an output ATM buffer with inter-cell spacing equally distributed over the duration of the packet. This cell buffer is managed and controlled by the buffer manager as shown. Cells are multiplexed over the bottleneck outgoing link. Our proposed scheme is appropriate only within an "edge-switch" as shown in the model.

The senders are assumed to be bursty sources with peak rate of λ packets/sec. We used bursty sources since LAN interconnection type of traffic on the Internet is bursty in nature. Each source is also characterized by a maximum burst size, interarrival time between bursts and an activity ratio "*a*" which represents the fraction of time the source is active in a ON/Off (periodic) cycle. The wide-area link has a channel capacity of $C=155$ Mb/s which gives a load of $\rho = (\lambda \,.\, a.\, n)/\, C$, where n is the number of bursty sources. Thus to increase load, one can increase either n or λ. The receiver is a simple sink where all statistics are measured.

FEC increases the total amount of traffic in the network by the number of redundant cells generated by the FEC encoder. Therefore, the cell loss rate will increase whenever FEC is used. FEC is only effective, if the FEC decoder recovers enough lost cells to reduce the block loss rate to a level lower than the loss rate when no FEC is used. If h redundant packets are generated by the encoder in each block of K packets, then effective arrival rate changes from λ to $\lambda_{eff} = \lambda(1+ h/K)$.

Several parameters used in the simulation represents real-life values. Frame sizes are chosen to represent LAN MTU values, e.g., 512 for UDP datagrams, 1500 bytes for Ethernet, 2048 bytes for Token ring, 4352 bytes for FDDI and 9180 bytes for IP MTU over ATM. The cell buffer size is chosen to range from 500 to 8000 (in cells) representing small and medium sized physical ATM switch buffers. FEC block sizes of K = 10 packets are chosen and each virtual circuit generates 10 blocks of packets. Considering that each packet gives rise to several cells in the ATM buffer, this represents a steady-state effect (in terms of cell transmission time), so that edge effects can be ignored in the system. Channel capacity C is chosen to be 155 Mb/s, a typical OC-3 rate.

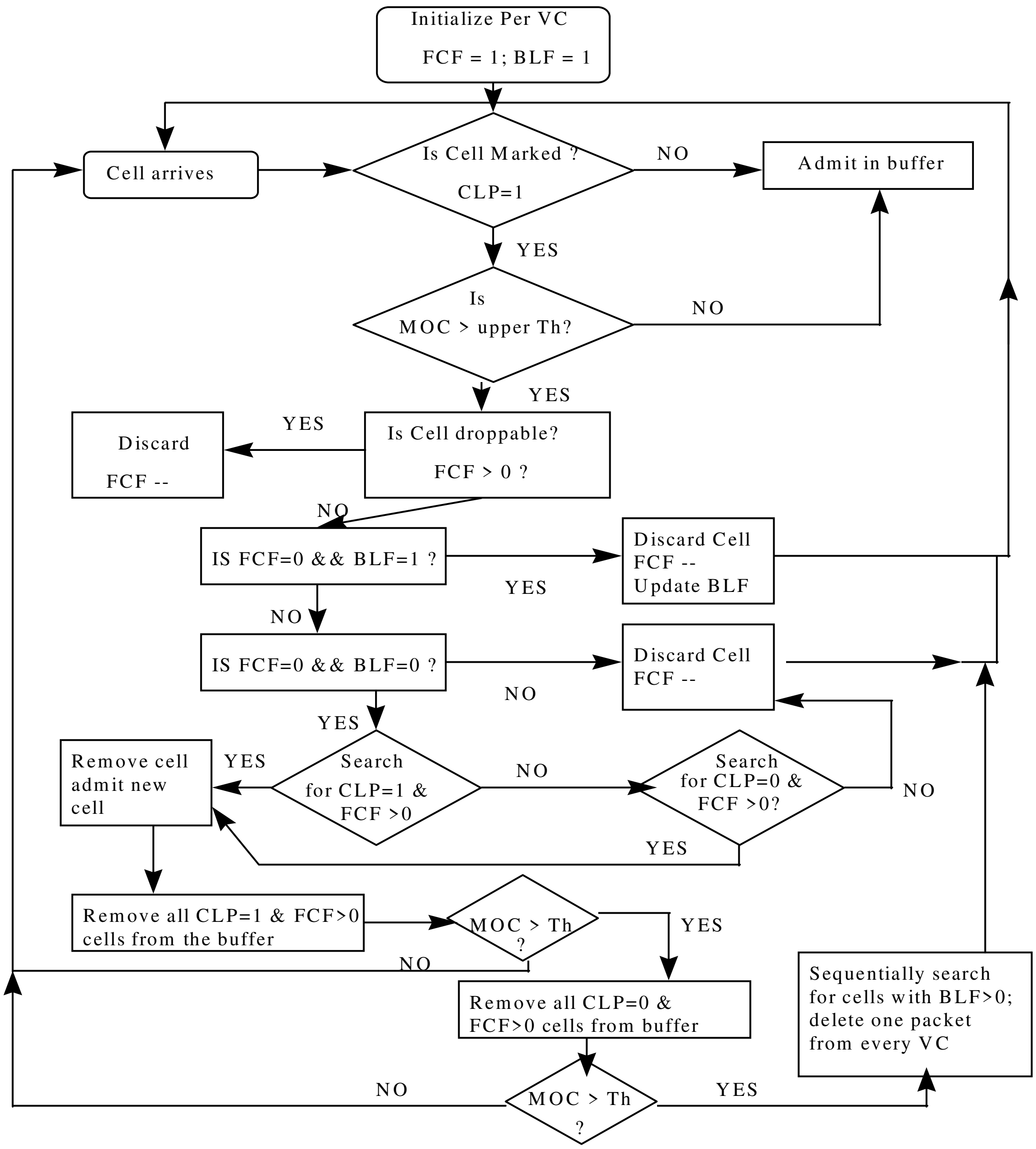


Figure 5: Flowchart showing the gatekeeper and janitor function

In order to investigate the performance of the proposed schemes, we need to distinguish between three models that we have consistently compared against each other. Model 1 represents plain traffic which employs no FEC so that we can find the loss-rates for such plain systems. Model 2 is bursty traffic with FEC coding that represents traffic which employ FEC to overcome potential losses over the internetwork. Finally Model 3 (our GPJ scheme) uses bursty traffic with FEC along with the gatekeeper plus janitor buffer management algorithms. In order to better highlight performance, we define the following gains for the three chosen models:

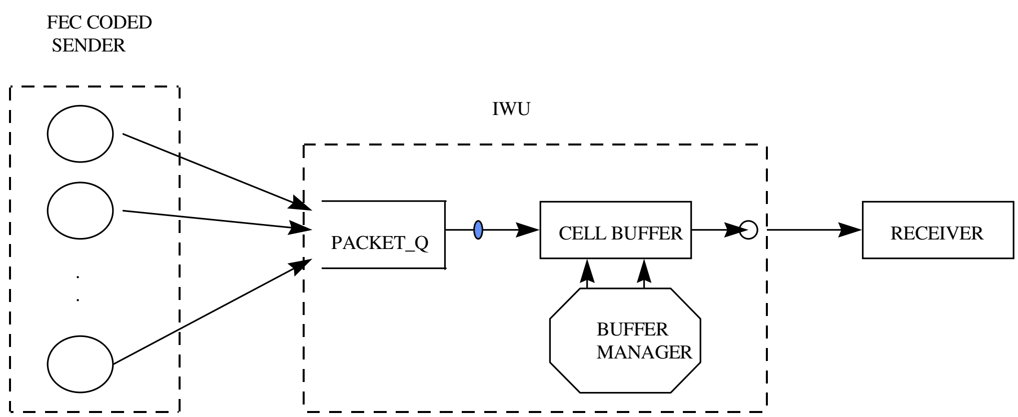
G = overcoding gain (due to FEC over plain traffic) = P_{DEC} *(Model 2)*/P_{loss} *(Model 1)*

D = buffer management gain (due to proposed algorithms) = P_{GPJ} *(Model 3)*/P_{loss} *(Model 1).*

Note that both *G* and *D* should be less than 1 for FEC and buffer management to be effective. We claim and show in this paper that *D* is lower than *G*.

5.2 Results:
In our simulation, we consider packet throughput as our primary performance measurement for all three models. End-to-end hosts transmit packets which are organized into blocks. Although packets travel across ATM networks as cells, we consider the higher layer packet throughput for applications.

In Figure 7, we investigate the packet throughput under high congestion for a load of $\rho = 0.9$. As we can see from the diagram, packet throughput is linearly proportional to the size of the buffer until it hits saturation (maximum 100% throughput) for a certain buffer size. For small buffer sizes (< 3500), all models suffer losses. As buffer size becomes larger, packet throughput can reach as much as 100%. Figure 7 shows that in any buffer size, model 3 outperforms model 2. Also, it is interesting to note that using model 3, we can get 100% throughput at a buffer size of 3500 while model 2 (plain FEC) gives us the same throughput for a buffer of size 6500. Thus model 3 clearly gives a much better margin for designing buffers in an ATM internetwork edge-switch. For buffer sizes less than 3500, model 3 performs better than model 1 also but only a slight performance gain. If we can increase the number of FEC parity packets in a block (both at packet and cell level), then we might get significant improvements over model 1 in this range. However, as mentioned earlier, there is a tradeoff between FEC and overall performance since more parity packets also increases the traffic load.



	SENDER	SWITCH	RECEIVER
MODEL 1	λ	Ploss (λ)	Ploss (λ)
MODEL 2	λ(1 + h/ K)	Ploss (λ(1 + h/ K))	Pdec((λ(1 + h / K))
MODEL 3	λ(1 + h/ K)	Ploss (λ(1 + h / K))	$P_{FEC + BM}$(λ(1 + h/ K))

Figure 6: Simulation Model

Figure 8 plots the packet throughput performance under varying load condition with a fixed buffer size of 2500 cells. As load increases, loss begins in the switch buffer. But again model 3 gives a better loss range than that compared to model 2. At very high loads, our management schemes out performs both model 1 and model 2. Also, from the diagram we can see that loss happens to model 2 when the load is over 0.6. Model 3 does not suffer losses until the load reaches 0.7. This gives us a 10% improvement in load margin which means model 3 gives us more room to work with under heavy load conditions.

Figure 9 and 10 plots the gains with just FEC and FEC with buffer management. Any gain between 0 and 1 makes FEC efficient, otherwise a gain greater than 1 simply means that FEC causes more traffic congestion that results in greater losses which cannot be recovered by the decoder. Under those situations, plain traffic (model 1) is likely to give better throughput. As link load is increased, Figure 9 shows that gain D (model 3/model 1) is lower than G (model 2/model 1) which illustrates better performance of our schemes. Also note that D is more efficient across the entire range of load values compared to G. In Fig. 10, D performs better than G for a wide range of possible buffer sizes. For example with a buffer size of 3500, D is zero which means that there are no loss with FEC and buffer management, while by employing only FEC, we get G of 2 making it highly inefficient. Large buffers give more leverage to the buffer manager to salvage more blocks, resulting in higher throughput. We later prove this through analytical models.

We have so far demonstrated that model 3 performs better than model 2 in terms of packet throughput and gains. We now consider several associated parameters that affect the behavior of model 3 so that we can fine tune the performance of our proposed schemes.

One important parameter is the block size *K* for employing FEC. Figure 11 plots the packet throughput with single FEC for various block sizes. To eliminate other factors, we keep load fixed at 0.9 and buffer of size 3500. Block size determines both the FEC overhead and the efficiency of our searching mechanism. The smaller the block size, the larger the transmission overhead. For example, overhead is 10% when K=10, and it is 20% when K=5. On the other hand, the smaller the block size, the more blocks and hence candidate cells are found by the buffer manager to drop, thereby improving performance. The larger the block size, the less the FEC overhead, but this also decreases the buffer manager efficiency. From Fig. 11, we note that the difference among the two models become less and less when the block size increases. If the block size is large enough, there will be no difference for all these three models. We chose a value of $K = 10$ in most of the simulation runs, simply because it shows the greatest variation (see Fig. 11) in throughput for the models considered. With $K = 10$, the overhead is just *10%*.

The frame MTU size also has a direct effect on the performance of model 3. In Figure 12, we plot the packet throughput for various MTU chosen from existing legacy LANs like Ethernet, Token ring, FDDI and IP. We can see that the smaller the MTU, the better the performance. This is simply because a smaller MTU results in more number of packets to reside in the buffer thus increasing the search range of the buffer manager. When MTU is very large, for example 9180 bytes, we see lowest throughput since the manager has access to very few packets in the buffer and it does not have many candidate cells to search for. Another reason is the bursty traffic characteristic. VBR sources generate packets and send them in bursts. IWU fragments those packets into cells and send them to the ATM switch. This process takes virtually no time. The larger the MTU, the more number of cells will be generated that might flood the switch buffer in a short period of time. This will overflow the switch buffer and result in data loss.

Since our proposed technique uses some searching and gatekeeping operations, it might imply that it slows down the CPU processing speed which could increase the average time cells spend in the buffer. If model 3 causes more delay than the other two models, then the gain in throughput and block loss rates is negated by additional delay our mechanisms impose. So we plotted the average end-to-end cell delay in the system for all three models as shown in Fig. 13. Clearly model 1 suffers minimum delay for any buffer size range, since it has the least overhead. It is interesting to notice that model 1 and model 3 almost have the same average cell delay when buffer size is small (less than 5000). The reason is that when buffer size is small, congestion happens more frequently and buffer manager works more frequently. Once the gatekeeper works, the only cells allowed in the buffer are good cells and others with overhead including corrupted packets and blocks are filtered out. So the average number of cells in the buffer will be the same or less than that of model 1. Although model 3 seems to cause more delay for higher values of buffer sizes, it helps to reduce the retransmissions, which might affect response time performance in case of model 1. From the diagram, we can also see that model 3 performs better than model 2. This happens because in model 3 when the gatekeeper becomes active, cell filtering takes place and all cells that are admitted are good cells. As many useless cells are prevented to enter the system, the average number of cells in the buffer is less than compared to model 2. This is also intuitive from Little's Result, i.e., $N_{avg} = \lambda. W_{avg.}$ Hence model 3 not only performs better than model 2 in throughput and gain, it does not add any significant delay which might otherwise affect real-time-based applications using FEC.

Figure 14 plots the buffer size required for various link loads to achieve 100% throughput. It helps to analyze the phenomenon in terms of a new performance measure which we call *saturation point*. It is a combination of buffer size and load that achieves 100% throughput in the ATM switch. The motivation is that no matter what control mechanisms are used, congestion will happen if the load is too large and buffer is too small. We can either require retransmission (ARQ) or blocking the incoming traffic, which may not meet the delay requirement for real-time-based applications such as video conferencing or interactive events. At current stage, the network operator cannot always estimate the bandwidth usage and limit the traffic. Hence the two factors that must be considered are buffer size and load. If the buffer size is fixed, traffic into the network will be limited so that the switch does not overflow. If load is too high, probably a larger switch will be needed. The *saturation point* is an index that a network operator can use to measure the tradeoff between buffer size and load. The saturation point for a desired throughput will be a function of both buffer size and load.

Figure 14 shows the improvement our model 3 brings with respect to designing switch buffer sizes. At a load of say 0.9, a buffer size of 6390 would yield 100% throughput in model 2. But at the same load only a buffer size of 3520 is needed for model 3 to yield 100% throughput. To measure the improvement, we define the following improvement measure:

$$I = \frac{B_{FEC} - B_{FEC+BM}}{B_{FEC}}$$

where 0 < I < 1. For load of 0.9, the improvement measure I = 0.44, i.e., the improvement in buffer dimensioning is 44%. Such improvements can be very beneficial to switch vendors building ATM IWUs.

5.3 Analysis of the GPJ scheme

GPJ is based on the premise that the buffer manager can find enough cells from packets belonging to several VC's in order to perform the searching and deleting efficiently. With interleaving from several sources inside the IWU, this will likely happen. The search and deletions are only done when the MOC is greater than the chosen threshold value, i.e., at periods of high congestion in the buffers. We are bound to find a lot of cells from different packets at such times. But the crucial question is whether we will find packets with FCF < 0. Let us consider the case when the MOC is based on the number of cells in the queue. Thus MOC = M which is the number of cells in the buffer that would be enough to trigger the GPJ algorithm. If the number of cells in the buffer is at least M, then the average number of packets at that time is $\frac{M}{L}$.

Let m be the probability that a cell is marked by the discarder. The analysis assumes that m is known (e.g., from experiments conducted within the marker UPC module or if an upper bound is known for acceptable operation).

Prob {a packet contains k marked cells} = $r_k = \binom{L}{k} m^k (1-m)^{L-1}$

Prob {a packet contains at least two marked cells} = $1 - r_0 - r_1$. Thus,

$$r_{\geq 2} = 1 - (1 - m + L)m(1-m)^{L-1}$$

Without invoking the GPJ algorithm, all with two or more marked cells (at time of congestion) will be rendered useless and cannot be recovered by FEC. The average number of these packets in the buffer is

$$H = \frac{M}{L} * r_{\geq 2} = \frac{M}{L} - \frac{M}{L}(1 - m + L)m(1-m)^{L-1}$$

Let the GPJ mechanism save a fraction "*s*" of these packets. This is done by sacrificing a fraction (*1-s*) of these packets and deleting their cells since our algorithm will find FCF < 0 for such packets. Therefore the number of saved packets is *sH*. Each of the saved packets will have their FCF value kept at 0. The number of cells from these packets that will replace cells from other unlucky packets is given by

$$\Pi = b \sum_{k=2}^{L} [r_k * sH * (k-1)]$$

where $b = \frac{1}{1 - r_0 - r_1}$. This number cannot exceed the total number of cells in unlucky packets which is given by

$\Phi = (1\text{-}s)HL.$

The best performance is obtained when $\Pi = \Phi$.

Solving this equation for s we get $s = \dfrac{1}{\frac{b}{L}\sum_{k=2}^{L}[r_k + (k-1)] + 1}$

.

6.0 Implementational issues

In order to implement the proposed buffer management algorithms, we must propose solutions to certain overheads that are present which cannot be neglected in high-speed networks. First overhead is in the form of extra storage required to keep track of the losses occurred. Both flags FCF and BLF can either be two additional columns inside the virtual-circuit routing table of the ATM switch or can be implemented as a separate table within the buffer manager. Note that only applications that use FEC and require such throughput improvement using our buffer management scheme need to have the extra column corresponding to the respective VC entry. Our implementation requires FCF to be an array data structure of size K. While we used K=10 for all simulations, the value of K used by the FEC protocol can be negotiated during VC connection set up and space reserved for it in the manager.

FCF or BLF flag values may not necessarily be initialized to one. They can be adjusted to implement negotiated QOS for multimedia services. As we know, certain real-time-based services, such as human conversation, have a relatively higher tolerance for data loss. The users will hardly notice if we lose, say, 0.5 percent of the whole conversation. For those kind of services, we can set FCF and BLF to different values during the session setup. In [15], it has been shown that for MPEG-2 over ATM transmission, a loss ratio of 1% is still acceptable using a dummy cell technique. FEC can be effectively used to preserve cells over that 1% zone. That means more than one cell could be dropped in case of congestion. This issue is certainly worth more investigation in the future.

The second overhead is incurred in time to search for a droppable candidate by the janitor routine. A hardware cache can be designed which can achieve cell searching in constant time thereby reducing any significant overhead delay. The hardware cache at any time stores the most recent 64 buffer addresses of cells which are likely candidates for removal by the janitor. While the design of such a dynamic hardware cache is by no means trivial, it can be done. The cache can be implemented as a circular list in which cell addresses are placed in the tail and janitor extracts addresses from the head.

7.0 Conclusions and future work

This paper proposed and evaluated a congestion control scheme for ATM-based internetworks. The scheme may be viewed as a combination of frame-level discard with forward error correction. The FEC is applied at two levels – at the packet level and the cell level. Packets are grouped into blocks and a parity packet is appended to each block of packets. Similarly, at the ATM layer, a parity cell is generated for each frame of L cells. These FEC mechanisms allow the recovery of one cell within each frame and of one packet within each block. The discard policy within each edge-switch is designed to minimize actual data losses by attempting to limit the cell losses to one cell within each frame and one lost packet within a block of packets.

The simulation model chosen in this paper has only a single bottleneck. This edge-switch model demonstrated the effectiveness of the GPJ management scheme. The performance gains that were obtained over other models are positive and encouraging. However, we did not investigate any fairness problems of the scheme. As a future work, we will simulate a multi-hop network with multiple bottlenecks in which end-to-end traffic will compete with cross traffic to expose any fairness problems.

Finally it is important to compare our GPJ scheme with some other schemes that have been proposed in the literature. We shall only discuss TCP, the DECbit scheme, Random Early Detection (RED) gateways and the Early Packet Discard (EPD). Each of these other schemes are based on a modification of the ARQ flow control scheme which relies on retransmission and acknowledgments. GPJ using FEC on the other hand is for those applications that cannot withstand the RTT delays of retransmission. TCP's strategy is to control congestion once it happens, as opposed to trying to avoid congestion in the first place. On the other hand the DECbit scheme [17] and the RED gateways [18] try to avoid congestion. Both rely on a feedback to the source. In case of DECbit, the feedback is explicit in the form of a binary congestion notification in the ACK sent in the reverse direction to the source. The source adapts accordingly. The RED scheme uses an implicit signaling by dropping packets at the onset of congestion. A dropped packet results in time-out and the source can adapt its rate for retransmission. All schemes use a method of measuring the queue length, which gives a measure of the congestion load. GPJ like RED uses two thresholds to switch on and off the management mechanism. The gatekeeper function within GPJ is similar in spirit to the Partial Packet Discard (PPD) proposed in [11] in which all subsequent cells from a packet are dropped as soon as one cell has been dropped. This causes less wastage of valuable bandwidth but works only because corrupted packets can be retransmitted. The PPD idea has been combined with RED idea to have an aggressive congestion control mechanism called EPD. With GPJ, the janitor needs to search for likely candidate cells to be selectively discarded so that the end-to-end throughput is maximized without retransmission. FEC based schemes are typically open-loop controls in comparison to the ARQ-based ones, which are closed loop controls.

Thus the GPJ algorithm may be slightly more complex in implementation but the added complexity is required to handle an open-loop control system where real-time packets (voice, video, multimedia) cannot be retransmitted. The improvement measure shown in this paper should be of direct interest to vendors building switches that would employ some form of reliable transport protocol using FEC and buffer management.

Acknowledgment
The authors wish to thank Mostafa Bassiouni for insights into the analysis of the GPJ algorithm.

References

[1] CCITT Draft Recommendation I.150, "B-ISDN ATM aspects", Geneva, Switzerland, Jan. 1990.

[2] J. J. Bae, T. Suda, "Survey of traffic control schemes and protocols in ATM networks", *Proceedings of the IEEE*, Vol. 79, No. 2, Feb. 1991.

[3] D. D. Clark, B. S. Davie, D. J. Farber, I. S. Gopal, B. K. Kadaba, W. D. Sincoskie, J. M. Smith and D. L. Tenenhouse, "An overview of the AURORA gigabit testbed", *Proceedings of IEEE INFOCOM'92*, pp. 569-581, Florence, Italy, May 1992.

[4] J. Turner, "New directions in communications (or which way to the information age)", *IEEE Communications magazine*, pp. 8-15, Vol. 24, 1986.

[5] G. Ramamurthy, R. S. Dighe, "A multidimensional framework for congestion control in B-ISDN", *IEEE Journal on Sel. Areas in Communications*, Vol. 9, No, 9, Dec. 1991.

[6] N. Shachum, P. Mckenney, "Packet recovery in high-speed networks using coding and buffer management", *Proceedings of IEEE INFOCOM*, 1990.

[7] Ramesh Kalathur, "Cell-loss recovery in ATM networks using FEC and buffer management algorithms", *Masters Thesis*, Computer Science Department, University of Central Florida, 1993.

[8] Julio Escobar, C. Partridge, "A segmentation and reassembly (SAR) protocol for use with ATM", *Proceedings of Second IFIP WG6.1/WG6.4 International Workshop on Protocols for High-Speed Networks*, Stanford, CA, Nov. 1990.

[9] Ernst W. Biersack, "Performance evaluation of forward error correction in ATM networks", *Proceedings of SIGCOMM*, pp. 248-257, Baltimore, Aug. 1992.

[10] S. Chatterjee, M. A. Bassiouni, "Increasing multimedia traffic throughput in high-speed WANs using buffer management", *Proceedings of ICC'95*, Seattle, WA, 1995.

[11] A. Romanow, S. Floyd, "Dynamics of TCP traffic over ATM networks", *IEEE Journal on Sel. Areas in Communications*, Vol. 13, No. 4, 1995.

[12] S. Floyd, V. Jacobson, S. McCanne, L. Zhang, Ching-Gung Liu, "A reliable multicast framework for light-weight sessions and application level framing", Proceedings of SIGCOMM'95, pp. 342-356, Cambridge, MA, Sept. 1995.

[13] Kai-Yeung Siu, Raj Jain, "A brief overview of ATM: Protocol layers, LAN Emulation and Traffic Management", Computer Communication Review, pp. 6-20, Vol. 25, No. 2, April 1995.

[14] J. Crowcroft, Z. Wang, A. Smith, J. Adams, "A rough comparison of the IETF and ATM service models", IEEE Network, Vol. 9, No. 6, Nov./Dec. 1995.

[15] T. Han, Luis Orozco-Barbosa, "Performance requirements for the transport of MPEG video streams over ATM networks", *Proceedings of ICC'95*, Seattle, WA, 1995.

[16]T. M. Chen, S. S. Liu, V. K. Samalan, "The available bit rate service for data in ATM networks", IEEE Communications, May 1996.

[17] K. K. Ramakrishnan, R. Jain, "A binary feedback scheme for congestion avoidance in computer networks with connectionless network layer", ACM Transactions on Computer Systems, 8(2):158-181, May 1990.

[18] S. Floyd, V. Jacobson, "Random early detection gateways for congestion avoidance", IEEE/ACM Transactions on Networking, 1(4):397-413, August 1993.

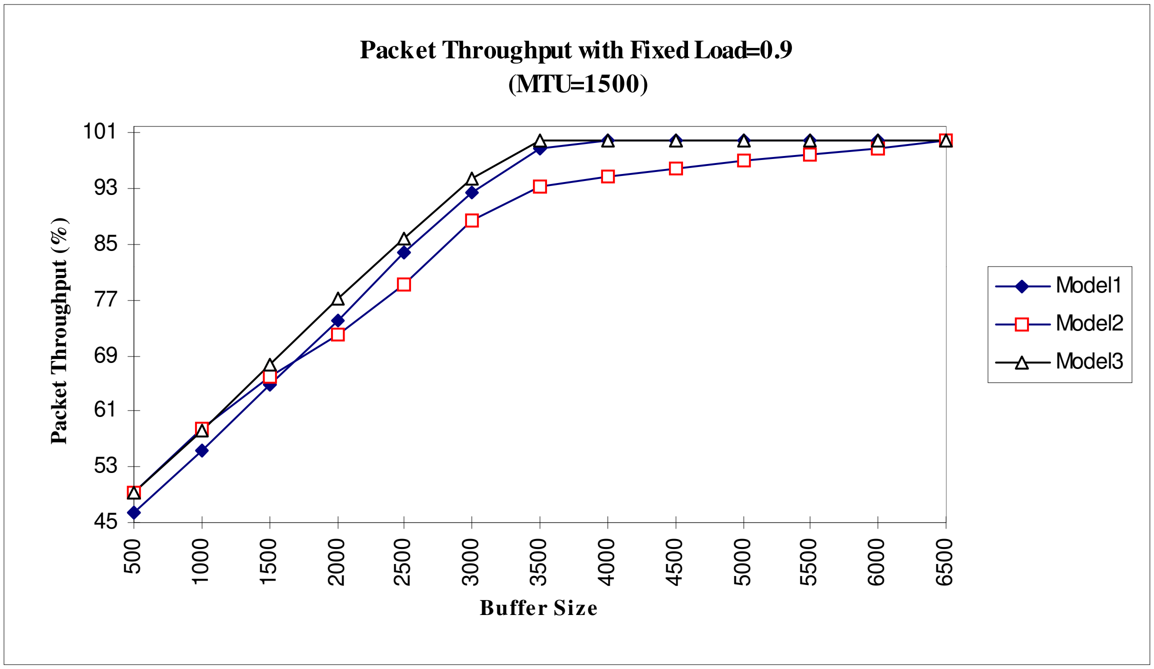


Figure 7: Packet throughput versus buffer sizes for various models

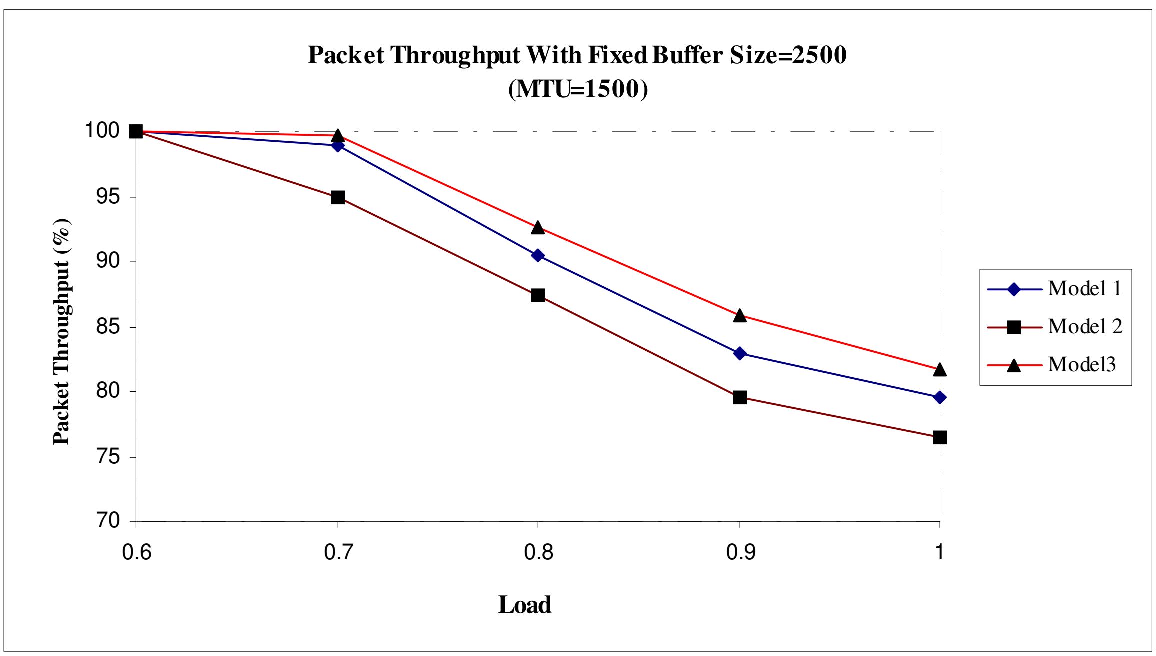


Figure 8: Packet throughput versus load for various models

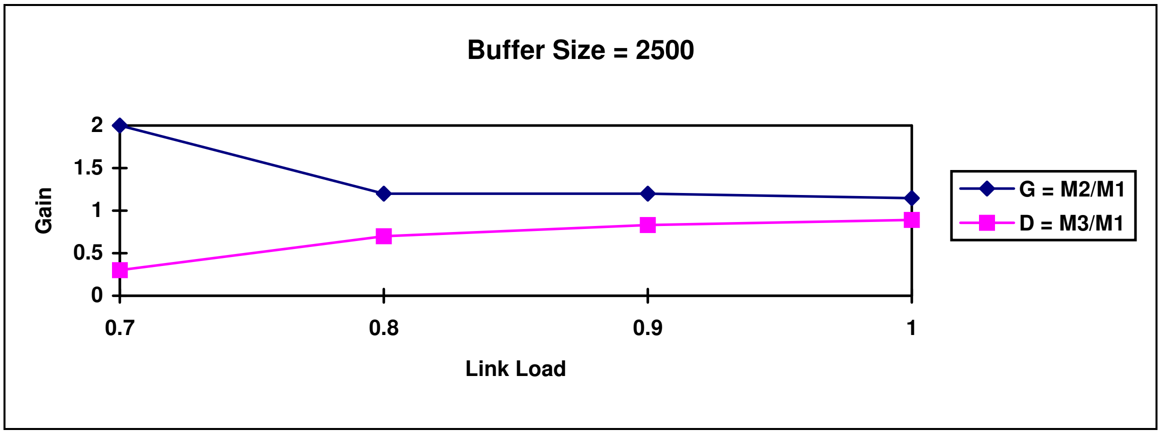


Figure 9: Overcoding and management Gain versus link load

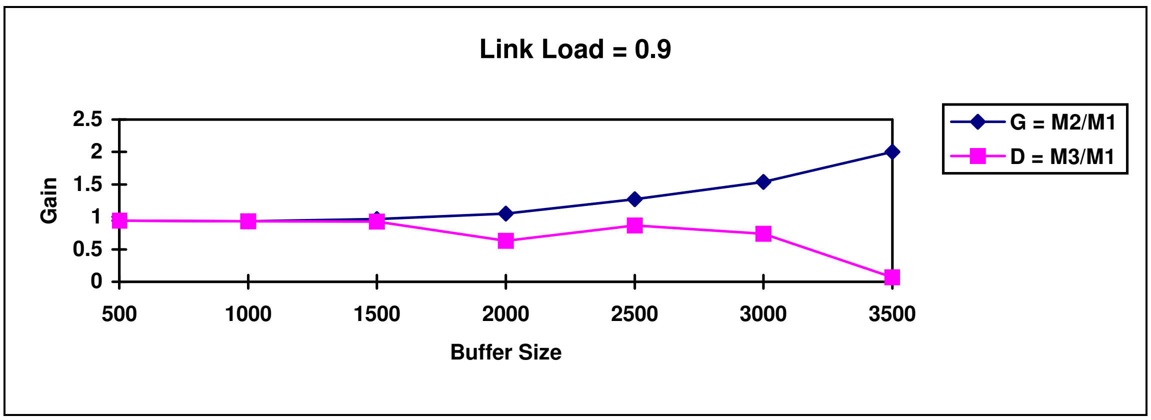


Figure 10: Overcoding and management gain versus different buffer sizes

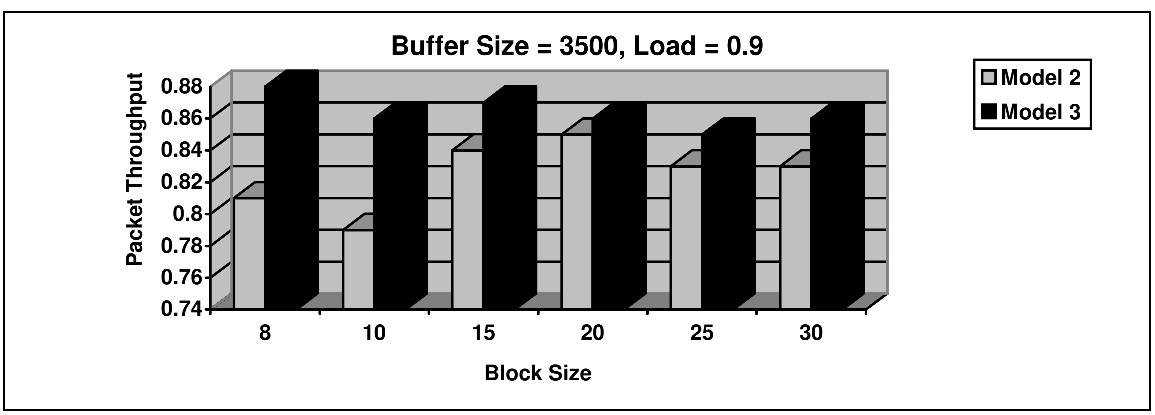


Figure 11: Packet throughput versus different FEC block sizes

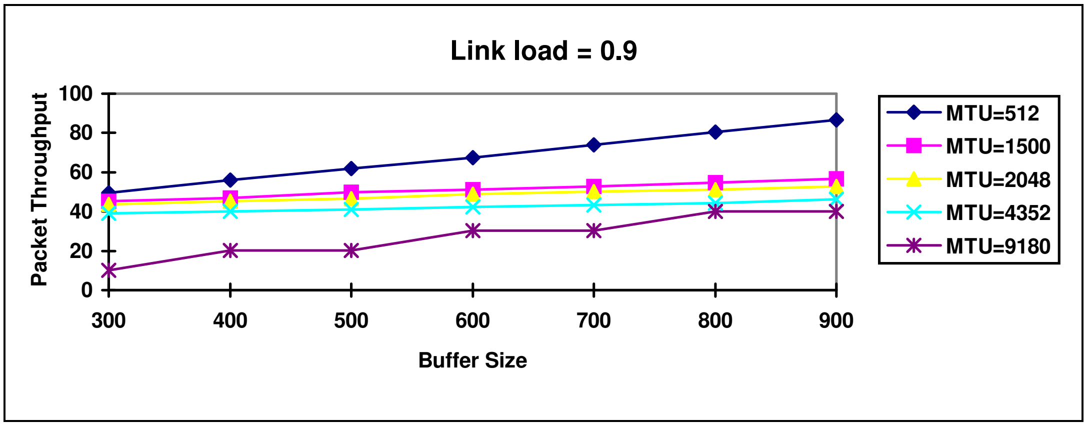


Figure 12: Packet throughput versus MTU for various legacy LANs

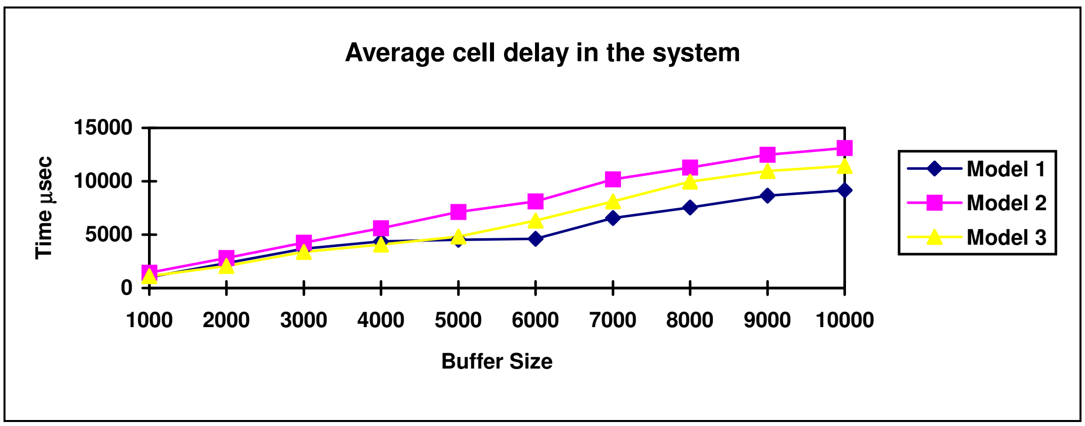


Figure 13: Average end-to-end delay for the three models

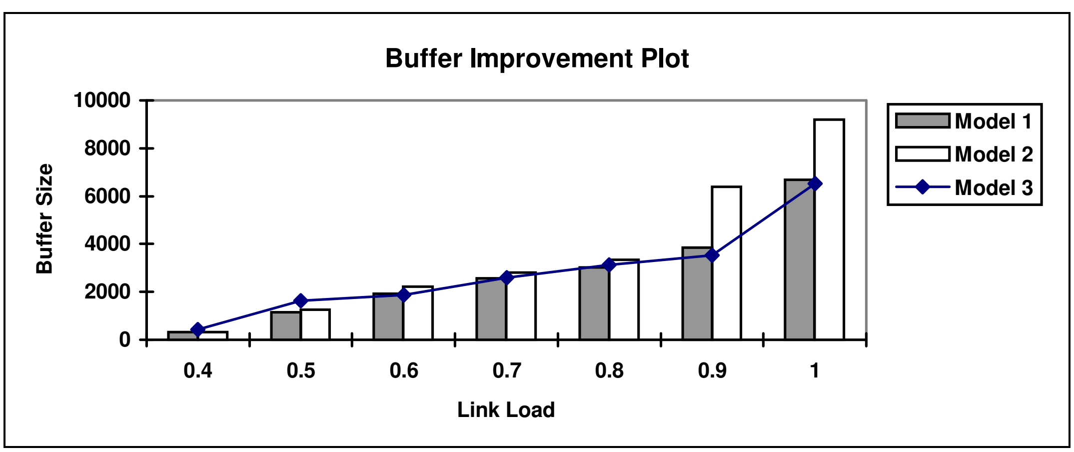


Figure 14: Improvement in buffer sizes to achieve 100% throughput