Performance of TCP over Multi-Hop ATM Networks: A Comparative Study of ATM-Layer Congestion Control Schemes

Lampros Kalampoukas
Anujan Varma

UCSC-CRL-95-13
February 16, 1995

Baskin Center for
Computer Engineering & Information Sciences
University of California, Santa Cruz
Santa Cruz, CA 95064 USA

ABSTRACT

We study the performance of TCP/IP on a multi-hop ATM network by simulation in an effort to evaluate the effectiveness of various congestion-control schemes at the ATM layer. The congestion control schemes considered are the ATM-Early Packet Discard (ATM-EPD) and link-level flow control. The performance of these schemes are compared against that of an ATM layer with no congestion control, as well as TCP/IP over a datagram network. We compare the schemes in terms of the throughputs obtained by long and short connections, number of retransmissions, end-to-end delays, and fairness in bandwidth allocation between connections with unequal round-trip delays. Our results show that significant unfairness in the amount of bandwidth allocated to connections may result in a large ATM network if no congestion control policies are used at the ATM layer. The ATM-EPD scheme is able to remove some of the unfairness, providing performance close to that of a datagram network, but can still provide unacceptable performance if the buffer sizes in the switches are small. Link-level flow control provided the best performance among the schemes studied.

Keywords: TCP congestion control, TCP over ATM

This research is supported by the Advanced Research Projects Agency (ARPA) under Contract No. F19628-93-C-0175 and by the NSF Young Investigator Award No. MIP-9257103.

1 Introduction

Since its introduction in the ARPANET in the 1970s, the Transmission Control Protocol (TCP) has become the most widely used transport protocol today, due largely to the explosive growth of the TCP/IP Internet in recent years. TCP implementations have been shown to perform well on data networks over a wide range of speeds — from low-speed dial-up links to HIPPI networks operating at a link bandwidth of 800 Mbits/second.

An important component of TCP is the collection of algorithms used to perform congestion control and recovery. Since TCP is designed to run over a connectionless network layer, congestion control is implemented in TCP between the endpoints of each connection. An important characteristic of TCP congestion control algorithms is that they assume no support from the underlying network and lower layers to indicate or control congestion, but instead use implicit signals such as acknowledgements, timeouts, and duplicate acknowledgements to infer the state of the network. These are used as feedback signals to control the amount of traffic injected into the network by modifying the window-size used by the sender. The algorithms attempt to utilize the available bandwidth of the network fully, without, at the same time, introducing congestion. In addition, congestion control policies could be implemented in the IP gateways to effectively complement the TCP end-to-end algorithms such that some degree of fairness can be maintained among the connections sharing resources in the network. Many such gateway congestion-control policies are surveyed in [12].

The congestion control mechanisms used in current TCP implementations are based on a number of ideas proposed by Jacobson [5], some of which were later fine-tuned and refined. Key components of the congestion control algorithm are the *slow-start* algorithm, a congestion-avoidance mechanism, and an algorithm to estimate round-trip delays. The slow-start algorithm is used to perform congestion recovery by decreasing the window-size to one segment and doubling it once every round-trip time. The function of the congestion avoidance mechanism is to probe for additional available bandwidth in the network by gradually increasing the window. The delay-estimation algorithm attempts to maintain a good estimate of the round-trip delay which is used as a basis to set the retransmission timers. The TCP Reno Version, introduced in 1990, added the *fast retransmit and fast recovery* algorithm to avoid performing slow-start when the level of congestion in the network is not severe to warrant a drastic reduction in the window size [16].

The behavior of TCP congestion control algorithms on datagram networks has been studied extensively [18, 19, 3]. Based on the insights gained, further improvements of the algorithms continue to be made [2]. Modifications of the congestion control algorithms for use in high-speed networks have also been proposed [6].

With the recent popularity of Asynchronous Transfer Mode (ATM), interest has risen on studying the behavior of TCP over ATM-based networks and internetworks. In these networks, all communication at the ATM layer is in terms of fixed-size packets, called "cells" in ATM parlance. An ATM cell consists of 48 bytes of payload and 5 bytes for the ATM-layer header. Routing of cells is accomplished through packet switches over virtual circuits set up between endpoints. By the use of proper scheduling algorithms in the packet switches, ATM is capable of handling multiple classes of traffic ranging from real-time video to data requiring no quality-of-service guarantees. Traffic with no real-time deadlines is intended to be transported in the "best-effort" mode, which provides no guarantees on the available bandwidth, delay, or cell loss rate. This is similar in nature to the service provided by the current Internet, except that connections still need to be set up at the ATM layer over virtual circuits between endpoints, either explicitly or implicitly. Because of its installed base, TCP will likely be used as the transport protocol to support these applications over ATM. Therefore, it is important to study how the TCP congestion control mechanisms behave on ATM networks, especially when some form of congestion control is introduced at the ATM layer.

The objective of this paper is to study the dynamic behavior of TCP over ATM networks and internetworks consisting of multiple hops of ATM switches, in an effort to identify the influence of various ATM-layer congestion control approaches on TCP performance. Of particular interest is to investigate the interaction between connections with different round-trip delays to determine how the fairness characteristics vary for the different schemes. To achieve these objectives, we simulated a network configuration with 8 nodes, 4 switches, and a total of 10 TCP sessions sharing network resources. In addition to the baseline case with no congestion control, we considered two distinct congestion control approaches at the ATM layer — the *ATM Early Packet Discard* (ATM-EPD) proposed by Romanow and Floyd [15] and *link-level flow control* proposed by Kung and Chapman [10] and Varghese, et al. [17]. To compare with the performance of TCP over a datagram network, we also simulated a network configuration in which the ATM switches were replaced by IP gateways.

The rest of this paper is organized as follows: In Section 2, we outline the ATM-layer congestion control approaches simulated. In Section 3 we describe the network model and tools used in the simulations. We discuss the simulation results in Section 4 and compare the congestion-control approaches in terms of their effective throughput, fairness in bandwidth usage, number of retransmissions, and delay characteristics. In Section 5 we discuss the problem of setting the threshold in the ATM Early Packet Discard scheme. We conclude the paper in Section 6 with a discussion of the lessons learned and directions for future research. Appendix A provides details of computing the buffer sizes for the link-level flow control scheme we simulated.

2 Congestion Control Schemes at ATM Layer

In this section, we first outline the reasons why the performance of TCP in an ATM network can be inferior to its performance in a conventional datagram network, and then discuss how ATM-layer congestion control schemes can improve TCP performance. We also provide a brief description of the ATM-layer congestion control schemes studied in this paper.

Although TCP is designed to work in networks with no congestion control mechanisms below the transport layer, there are several factors that justify the use of a congestion-control scheme at the ATM layer. Without ATM-layer congestion control, the performance of TCP over ATM can be considerably worse as compared to its performance in datagram networks in certain network configurations, as observed by Romanow and Floyd [15]. Since each TCP segment is fragmented into a large number of cells in the ATM network, the loss of a single cell triggers the retransmission of an entire TCP segment. Even worse, when a switch discards a cell belonging to a TCP segment, the remaining cells of the segment continue traveling towards their destination, wasting network resources such as buffer space and link bandwidth. Although this problem is not fundamentally different from the fragmentation that occurs at the IP layer in a datagram network, the effect could be considerably more pronounced in an ATM network because of the small size of the ATM cell.

A second reason why support from ATM layer may be needed to control congestion is that the gateway congestion control policies currently employed in datagram networks may be difficult to apply in ATM switches. The small cell-size rules out software implementations of congestion control policies in ATM switches in most cases, limiting the complexity of the policy to what is implementable in hardware. Thus, TCP may no longer be able to rely on the congestion control policies in IP gateways to provide fairness in usage of network resources among competing connections. TCP congestion control algorithms are already known to favor connections with shorter round-trip delays over those with longer delays [3], and ATM may exacerbate the problem.

A third difficulty in controlling congestion is caused by the connection-oriented nature of ATM. The ATM layer confines all the traffic in a TCP connection to a fixed path, making it difficult to cure long-term congestion by routing traffic away from congested spots.

Finally, because of the small cell-size in ATM, designers of ATM switches may be tempted to use smaller amount of buffering as compared to IP gateways, resulting in unacceptable performance when running TCP/IP over the ATM network.

Despite these problems introduced by ATM networks, ATM raises new opportunities for implementing congestion control at a lower layer, providing the potential for improving the performance of TCP even above that obtained in today's datagram networks. The connection-oriented nature of ATM allows the use of link-by-link flow control strategies, as well as control of resources within the network on a per-connection basis, which would be infeasible in today's TCP/IP networks owing to the datagram-based routing and the lack of a common data-link layer protocol. In addition, the fixed size of ATM cells simplifies the implementation of flow control algorithms in many cases.

The effectiveness of various congestion control approaches in ATM networks is still being debated in standards forums and there is no consensus yet on the mechanisms to be used. The approaches being discussed include rate control based on explicit congestion notification (forward or backward) [13], packet-discarding algorithms [15], and flow-controlled virtual channels [10, 11, 17]. With forward congestion notification, an ATM switch experiencing congestion sets the "congestion-experienced" bit in the header of ATM cells passing through the congested buffer(s). The destination of the connection may then signal congestion back to the source via one of the virtual channels in the reverse direction. With backward congestion notification [13], the switch notifies the source directly upon congestion by sending a special cell in the backward direction.

Explicit congestion notification schemes are based on feedback, which sometimes make them too slow in reacting to congestion or changes in the available bandwidth, resulting in unacceptably high cell-loss rates [7]. An alternate approach is to use link-level (or, *hop-by-hop*) flow control for each virtual channel. This approach can completely eliminate packet losses due to congestion. Kung and Chapman proposed a number of schemes for *flow-controlled virtual channels* (FCVC) based on this approach. In these schemes, a sender is allowed to transmit cells on a link only when sufficient "credits" have been provided by the receiver at the end of the link. The most promising of their schemes, referred to as the "$N23$ scheme," divides the buffering needed by each VC at each hop into two regions — the operating region $N3$ and a small underflow region $N2$. The size of $N3$ region is determined by the product of the round-trip delay to the next hop and the peak bandwidth allocated to the virtual circuit. The $N2$ region is used by the receiver to aggregate cell credits transmitted to the upstream neighbor; its size determines the frequency of transmission of credit cells upstream.

The $N23$ scheme allows no sharing of buffers between virtual circuits; hence, the amount of buffering needed can be prohibitive in a wide-area network where hundreds or thousands of virtual circuits may share the same link. For example, in a gigabit link with a round-trip delay of 1 ms, more than 125 Kbytes of buffering would be needed per virtual circuit if the peak bandwidth of each virtual circuit is set to the link bandwidth. This problem is easily solved by allowing the virtual circuits to share the available buffers. Two such schemes are described by Kung, et al. [11] and Varghese, et al. [17]. Kung, et al. use statistical multiplexing to reduce the amount of buffering by combining the buffer spaces of the virtual circuits and providing only a fraction of the amount of buffering required to operate all the VCs at peak bandwidth [11]; this introduces a small probability of cell loss when many VCs are congested simultaneously. The scheme by Varghese, et al. avoids cell losses by limiting the bandwidth of individual VCs under congested conditions [17].

An alternate approach, applicable when the IP protocol is used over ATM, is to discard cells corresponding to IP packets selectively upon congestion. Romanow and Floyd [15] proposed such an algorithm — the ATM Early Packet Discard (ATM-EPD) — that drops entire IP packets at the onset of congestion [15]. When the buffer occupancy increases over a set threshold, the algorithm selects the next new IP packet arriving into the buffer and discards the entire block of constituent cells of that packet. This is similar in nature to the congestion control policies proposed for IP gateways [12, 4], except that the process of selection of the packet to be discarded is simplified. While some form of randomness is employed in most gateway congestion control policies to select the packet to be discarded, ATM-EPD chooses the first complete packet to arrive after the instantaneous queue size crosses the set threshold.

The ATM-EPD algorithm effectively prevents fragments of packets from consuming network resources and contributing to further congestion. Romanow and Floyd showed that the performance of TCP over a single ATM switch employing the ATM-EPD scheme is comparable to that of packet

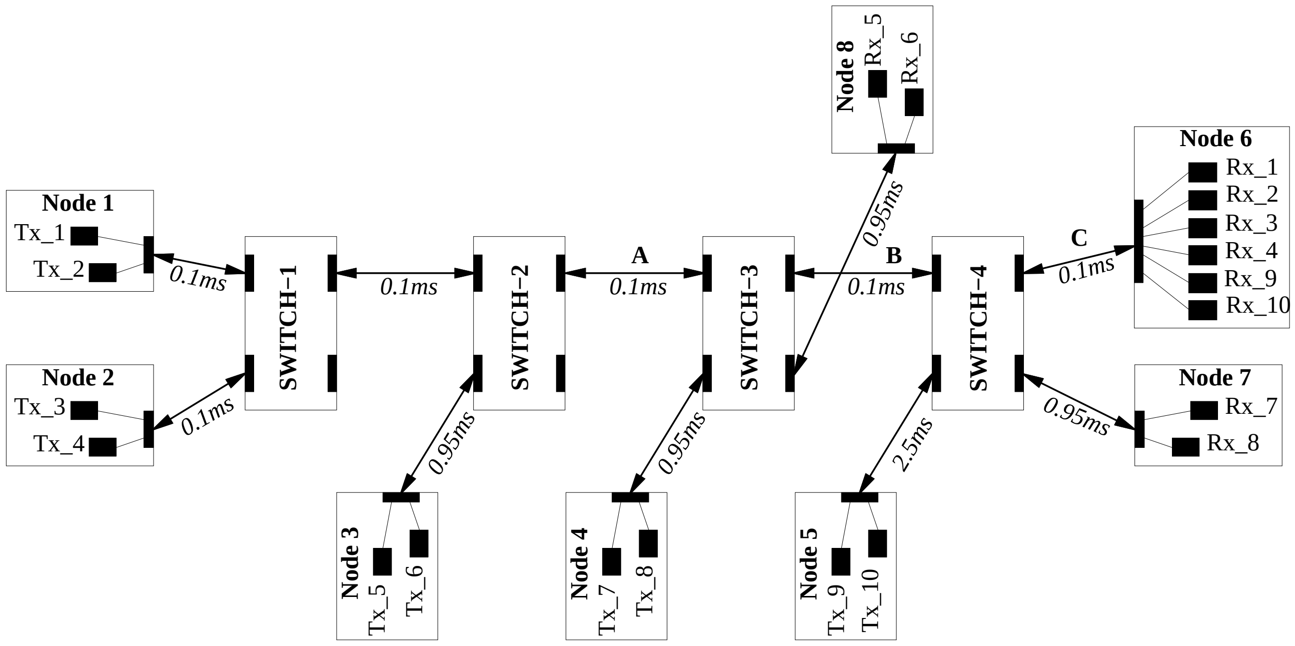


Figure 3.1: Network configuration used in the simulations.

TCP [15]. However, the behavior of the scheme in large multi-hop networks remains to be studied, where the lack of randomization in the selection of packets to discard could lead to unfairness in bandwidth usage among TCP connections sharing network resources.

3 Network Simulation Model

In this section we describe the network configuration and parameters used in our simulation of TCP over ATM and datagram networks. Figure 3.1 shows the network configuration used in our simulations, consisting of 4 switches, 8 end-nodes, and 10 TCP sessions. Each of the links is full duplex with a bandwidth capacity of 149.76 Mbits/second, corresponding to the effective capacity of a SONET STS-3c link for carrying ATM traffic. The specific configuration was chosen for several reasons. First, it allows investigation of the behavior of TCP connections when they are going through multiple hops. Second, it enables us to simulate two types of connections — *end-to-end* connections that go through all the switches, and *cross-traffic* connections that share only one link with end-to-end connections. This allows us to study how the two types of connections interact with each other.

TCP sessions were set up in this configuration such that congestion occurs in all the switches. A total of 10 sessions were assigned to the nodes as shown in Figure 3.1 such that the desired level of congestion was reached (in the figure, Tx refers to the transmitter and Rx refers to the receiver of a session). In every node there are two TCP applications running, except in node 6, where there are six. Each application operates either as a sending application or as receiving one. The applications that operate in nodes 1 through 5 are sending applications; those in the remaining nodes are receiving ones. All applications use TCP to communicate with their peer applications at the receiving node.

The applications with the same index indicate that they belong to the same TCP session. Having two active application in each node makes the models more realistic; the applications have to share resources such as the IP layer and the physical link that connects the specific node to the next switch. Since our objective is to study the network behavior under congestion, we assume that each sending application has infinite supply of data. Note that the most severe congestion occurs in the network at links A, B, and C, as each of them carries six TCP sessions.

Sustaining all the TCP sessions turned out to be a difficult problem; with our initial choice of link delays, some of the TCP sessions went into exponential backoff and were never able to recover from packet losses, making it difficult to create persistent congestion. To eliminate this problem, we used a combination of link delays that allowed all connections to maintain some sustained level of throughput. The delays used in our simulations are shown in Figure 3.1. The larger link delays assigned to the cross-traffic connections have the effect of reducing their aggressiveness, thus allowing the end-to-end connections to be active. Note that, when the switch buffers are close to full, the queueing delays in the switches still dominate the round-trip delays in this configuration. For example, with a buffer size of 100 Kbytes per output link in each switch, the maximum queueing delay in each switch is approximately 5 ms.

The simulation tool we used is the OPNET modeler tool. OPNET allows the definition and modeling of a communication network in a hierarchical manner. At the highest level, the network topology and connectivity are defined, along with several network parameters (node coordinates, link direction, link capacity, propagation delays, link error probabilities, etc.). At the next level, the protocols being used by each node as well as the way they communicate with each other are defined. Finally, at the lowest level, the behavior of all the modules used in the network can be described using a state-machine representation. Each state in the state machine is described using C-language statements. The OPNET simulation kernel is event-scheduled.

We used the TCP-Reno version for our simulations. However, since the TCP model supported by the OPNET tool is based on RFC 793, we made extensive modifications to support the congestion control mechanisms described by Jacobson [5], exponential back-off, enhanced round-trip-time (RTT) estimation based on both the mean and the variance of the measured RTT, and the *fast retransmit and fast recovery* mechanisms. However, some adjustments had to be made to the TCP timers. Since the RTT values in our simulation configuration is of the order of just a few milliseconds, the coarse-grain timers used in Unix TCP implementations (typically, a granularity of 500 ms) would make comparison of the schemes difficult. To avoid the anomalies due to coarse-grain timers, we used double-precision floating-point arithmetic in the RTT estimation algorithm. Several TCP parameters were initialized to values more appropriate to our network configuration: the initial RTT value was set to 10 ms, the maximum timeout value to 50 ms, and the initial mean deviation to 1 ms. This led to a faster convergence of TCP parameters to their actual values. We also added a small random component to the RTT of each segment in order to avoid phase effects in our simulations [3, 18]. The maximum size of the TCP congestion window was set to 64 Kbytes and was not a simulation parameter. The effect of using smaller values for the congestion window would be to reduce the aggressiveness of all the connections and leave the links idle for long periods.

To compare the effectiveness of ATM-layer congestion control techniques on TCP performance, we simulated four different cases in this network configuration:

1. TCP over ATM without any flow control at the ATM layer.
2. TCP over ATM with the ATM-Early Packet Discard algorithm implemented in each of the switches.
3. TCP over ATM with link-level flow control at the ATM layer. The actual flow control scheme implemented was the $N23$ scheme described by Kung and Chapman [10].
4. TCP over a datagram network obtained by replacing each of the ATM switches in Figure 3.1 with IP gateways. No gateway congestion control policies were implemented in the IP gateways.

The IP layer was simulated with a single queue per output port. The service rate of the IP layer was set to be equal to the bandwidth capacity of the physical links. The packet size at the IP layer is a simulation parameter. To study the behavior of the schemes for both small and large packets, we used three IP packet sizes — 1500, 4352, and 9180 bytes. To avoid fragmentation at the IP layer, the TCP segment size was always set so as to fit within a single IP packet.

Each simulation was run for 2.5 seconds of simulation time. This allowed the aggregate transfer of more than 25 Mbytes of data in the network and was sufficient to bring the network into a stable state and perform the necessary measurements. The simulation results do not include the first 0.5 second of simulation time to avoid the effects caused by the simultaneous opening of all the TCP connections.

For the simulations of TCP over ATM networks, we used the ATM Adaptation Layer Type 5 (AAL 5) [1]. AAL 5 performs segmentation and re-assembly between IP packets and ATM cells. Each IP packet is extended by eight bytes at the AAL layer to accommodate the AAL header. Thus, the number of ATM cells produced by the original IP datagram is given by:

$$\text{No. of cells} = \left\lceil \frac{\text{IP packet size} + 8}{48} \right\rceil$$

In the simulation configuration of Figure 3.1, the characteristics of the switches vary for the different congestion-control schemes simulated. For simulations of TCP over datagram networks, the switches perform the role of IP routers. For the remaining schemes, the switches operate at the ATM layer. For simulations with no ATM-layer congestion control, the switches are nonblocking, output-buffered crossbars. There is one queue per output port and the scheduling policy is FIFO. Each queue is shared by the virtual circuits (VCs) destined to the specific output link of the switch. The size of the queue is a simulation parameter.

In the simulations of ATM with the FCVC scheme, the switch is again a nonblocking, output-buffered crossbar, but now the output buffer is divided into separate queues for each VC sharing the specific output link. The scheduling policy within each queue is FIFO and the queues of the active

VCs (non-empty queues) sharing each output port are serviced in round-robin fashion. The flow-control model is based on the $N23$ scheme described by Kung and Chapman [10]. The buffer size for each virtual circuit was chosen to allow a peak throughput equal to the link capacity. The buffer-size calculations for the scheme are given in Appendix A. Although not the most efficient implementation in terms of its memory requirements in comparison with later schemes [11, 17], the FCVC scheme is straightforward to implement and avoids the effects of buffer sharing from influencing our results.

Finally, the switch architecture used in the ATM-EPD scheme is a nonblocking, output-buffered crossbar. There is a single queue per output port that is being shared by all the active VCs passing through the port and the scheduling scheme is FIFO. An important decision to be made is the choice of the threshold for discarding packets. If the threshold is set too low, a large portion of the buffer space is wasted. On the other hand, if it is set too high, the scheme may become ineffective. We chose the threshold according to the following formula:

$$\text{threshold} = \max\left\{0.8, \frac{\text{buffer size} - 3 \times \text{segment size}}{\text{buffer size}}\right\}.$$

That is, the threshold generally should be set equal to the buffer size less three times the TCP segment size. The role of the first term 0.8 in the above formula is to avoid setting the threshold to a very low value. This could occur when the buffer size is small and the TCP segments large. For example, with a buffer size of 50 Kbytes and a segment size of 9180 bytes, leaving space for three segments is equivalent to setting the threshold to a value equal to 0.45, which could lead to poor performance. Later in Section 5, we will present some results showing how the threshold value may affect the performance of this scheme.

Our simulations used detailed models for the TCP, IP, AAL 5, and the ATM layer. All the schemes used the same models for the TCP and the IP layers. Similarly, the same AAL 5 layer was used in all simulations of ATM networks. We assumed that the physical links are perfectly reliable, so that packet losses occur only due to congestion in the switches.

In the rest of this paper, we will refer to the TCP over datagram network simply as the TCP scheme, ATM without any flow control as the plain ATM scheme, ATM with link-by-link flow control on every VC as the ATM-FCVC scheme, and ATM with early packet discard as the ATM-EPD scheme.

4 Simulation Results

In this section we present the simulation results for the four congestion-control schemes discussed in the previous section. These results will focus primarily on the throughput obtained by the individual connections, number of retransmissions in the network, and fairness in bandwidth allocation among the competing connections.

We divide the 10 active TCP connections in our network configuration into two categories: (i) *end-to-end* connections that go through all the switches in the network, and (ii) *cross-traffic* connections that share exactly a single link in the network with end-to-end connections. Our results will focus more on the end-to-end connections since the congestion control scheme used has a more pronounced influence on their performance as compared to that of cross-traffic connections. Although the total propagation delay of the links for the end-to-end connections in Figure 1 is smaller than those of the cross-traffic connections, the queueing delays in switches can lead to significantly higher round-trip delays for the end-to-end TCP connections compared to cross-traffic ones.

4.1 Effective Throughput

Figure 4.1 presents the effective throughput of the end-to-end connections as a function of buffer size for the four congestion-control scheme and three TCP segment sizes. The buffer sizes in these plots refer to the available buffer size per output port of each of the switches. The effective throughput in these plots is defined as the fraction of the link bandwidth used to transfer traffic belonging to the end-to-end connections. That is,

$$\text{Effective Throughput} = \frac{\text{end-to-end throughput}}{149.76 \text{ Mbits/second}}.$$

where the end-to-end throughput is the actual total throughput of all the end-to-end connections in Mbits/second. For simulations over ATM networks, the end-to-end throughput was scaled by $48/53 = 90.56\%$ to account for the ATM-layer overhead. The loss of throughput due to partially-filled ATM cells was ignored in the throughput computations.

In all the simulations except those under the ATM-FCVC scheme, the buffers at each output port of a switch are shared by the TCP connections passing through the switch. Since each output buffer is shared by packets arriving at two different input ports (end-to-end and cross-traffic connections), we can expect the end-to-end connections to receive 50% of the link bandwidth, if the throughput distribution is perfectly fair. This is the reason why the effective throughput in the results presented in Figure 4.1 converge to a maximum value of approximately 50%. In the case of ATM-FCVC, however, the switch allocates bandwidth on a per-VC basis. Since each output buffer is being shared by four end-to-end connections and two cross-traffic connections, we expect the end-to-end connections to receive approximately 66.67% of the link bandwidth and the cross-traffic connections the remaining 33.33%.

The plots in Figure 4.1 suggest that increasing the buffer size, in general, increases the effective throughput of the end-to-end connections. This is especially true for medium buffer sizes. For large buffer sizes, all the connections have reached their equilibrium and are occupying a fair share of the link bandwidth; thus, further increases in the buffer size do not increase the throughput. Most of the increase in throughput occurs over the range of buffer sizes from 150 to 250 Kbytes. This behavior can be explained by examining how the cross-traffic connections interact with the end-to-end connections. With small buffer sizes, the link delays dominate the round-trip delay. As the buffer size is increased, however, the queueing delays in switches start dominating and contribute the most to the RTT. This causes the RTT estimates for the end-to-end connections to be larger than those of the cross-traffic connections, making them less aggressive in increasing the TCP congestion window during slow-start. This effect is confirmed by the throughput plots of the cross-traffic connections, shown in Figure 4.2, where most of the increase in throughput takes place over the range of buffer sizes from 50 to 125 Kbytes/second. This behavior is also influenced by the TCP segment size since the congestion window increases in multiples of the segment size during the slow-start phase. Thus, the effect is more noticeable at small segment sizes. The end-to-end connections increase their throughputs rapidly when the buffer size is increased from 150 Kbytes.

The throughput of end-to-end connections exhibits the same behavior under the ATM-EPD scheme. However, the throughput in this case is slightly inferior to that of TCP in a datagram network, especially in the range of buffer size 150–250 Kbytes. This is because, for a given buffer size, the effective buffer space available under ATM-EPD is less than that in a datagram network. Two factors contributes to this reduction in available buffering: First, approximately 10% of the buffer space is used to store headers of the queued ATM cells. Second, the threshold for packet discarding is set less than the buffer size. Hence, the ATM-EPD scheme behaves like the TCP scheme but with a smaller buffer size. This explains why the throughput plot for ATM-EPD in Figure 4.1 resembles a shifted version of that for TCP over datagram network.

It is interesting to note that the ATM-EPD scheme provided slightly higher throughput than the TCP scheme for *end-to-end* connections with a TCP segment size of 1500 bytes, over the range of buffer sizes from 75 to 150 Kbytes. This is because of the cell-discard policy favoring the end-to-end traffic under certain conditions. In an ATM network, each source transmits a TCP segment as a burst of ATM cells. As the cells travel through the switches, the cells are interleaved with the cells belonging to packets of other connections. Hence the cells belonging to a packet exhibit a tendency to move away from each other as they go through the switches. Thus, the cells that belong to packets of cross-traffic connections are likely to be closer together as compared to cells belonging to end-to-end connections. Therefore, under ATM-EPD, when the queue size exceeds the threshold value, there is a higher probability for the switch to find the first cell of a packet from a cross-traffic connection than that from an end-to-end connection. This reduces the aggressiveness of cross-traffic connections under some cases, allowing the end-to-end connections to obtain a larger share of the throughput than possible under datagram TCP.

On comparing the plots for end-to-end throughput under ATM-EPD and ATM with no congestion control, the benefit of discarding whole IP packets on congestion is clearly visible. In a plain ATM network, the cells belonging to corrupted TCP segments continue to travel downstream, wasting network resources. The effect is likely to be more pronounced with large TCP segment sizes, because the number of ATM cells belonging to corrupted TCP segments that consume network resources also increases. The plots in Figure 4.1 confirm this behavior, where the worst level of degradation in the plain ATM scheme occurs with the combination of small buffer sizes and 9180-byte TCP segments. The cross-traffic connections, on the other hand, benefit from the reduced bandwidth utilization of the end-to-end connections, allowing them to increase their throughput (Figure 4.2). This effect can

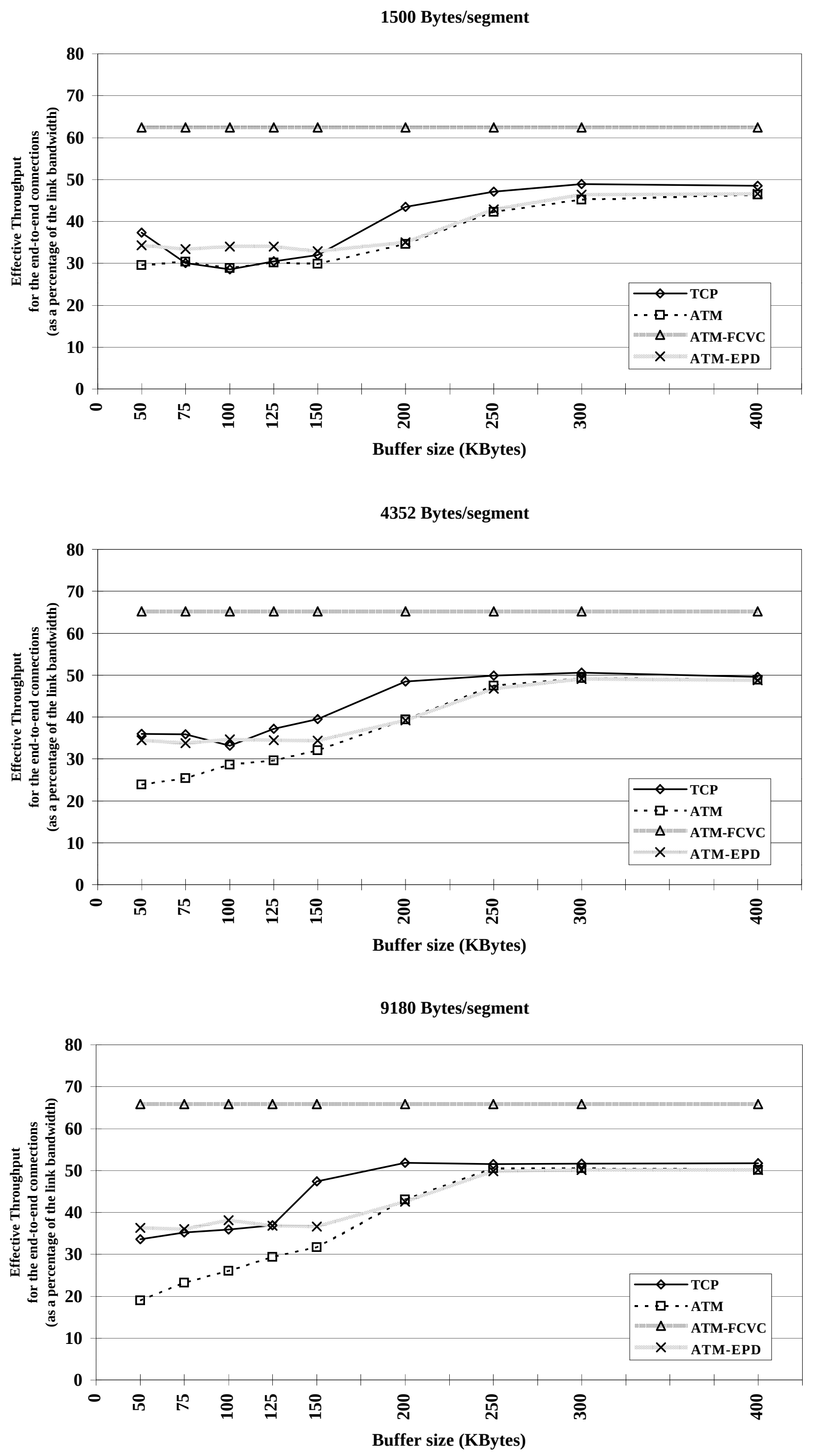


Figure 4.1: Effective throughput of the end-to-end connections as a function of the buffer size.

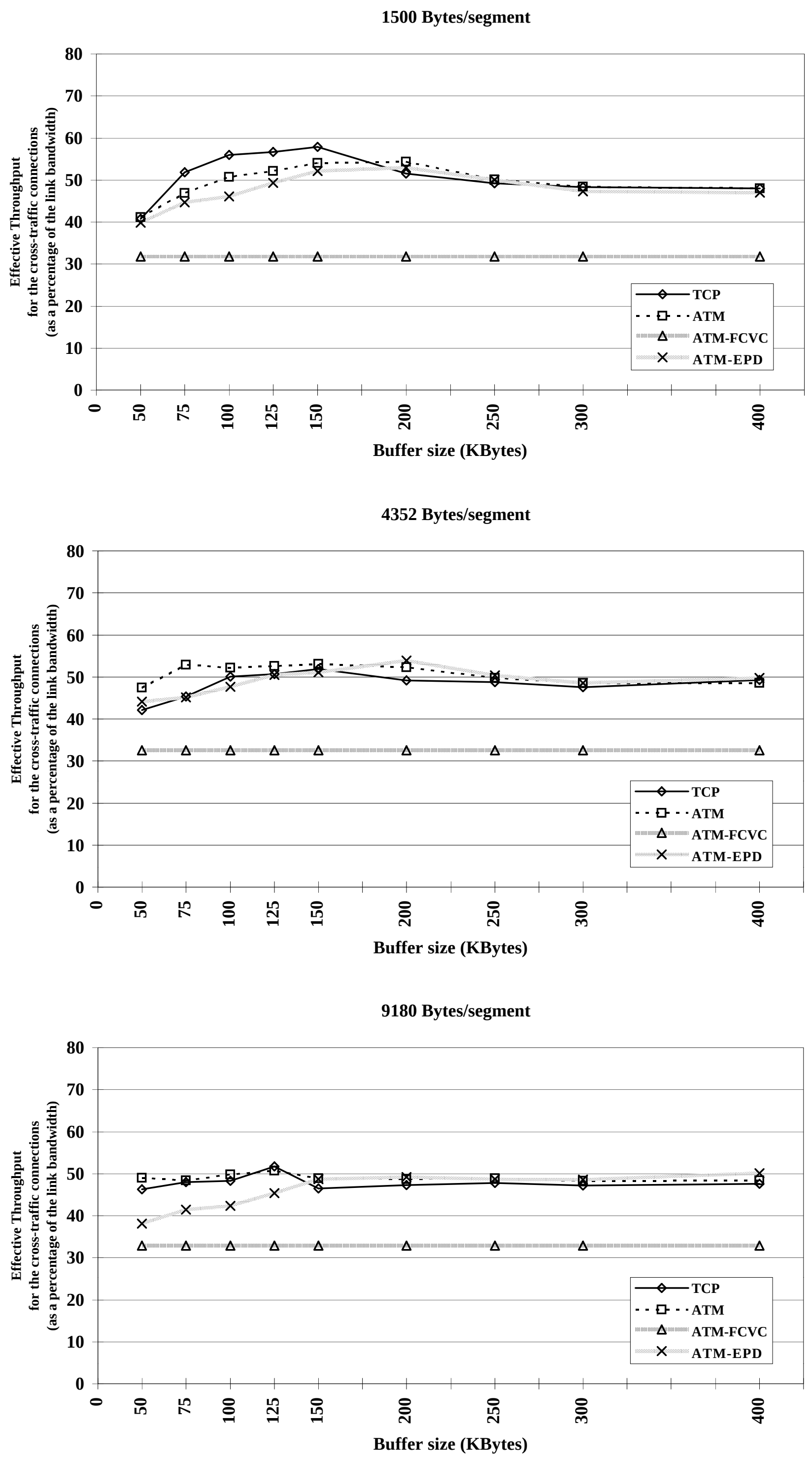


Figure 4.2: Effective Throughput for the cross-traffic connections as a function of the buffer size.

potentially cause severe unfairness in large ATM networks if no congestion control schemes are used at the ATM layer.

The behavior of the ATM-FCVC scheme is considerably more predictable. Since the buffer requirements of this scheme are predetermined and fixed, the plots for this scheme in Figures 4.1 and 4.2 are not a function of buffer size. In all the cases, the connections make almost perfect utilization of their allocated bandwidth. Some small differences among the three simulated TCP segment sizes arise from two sources: (i) the wasted bandwidth in the partially-filled last cell of IP packets (which decreases as the segment size is increased), and (ii) the retransmissions that occur during the simultaneous opening of the connections when the RTT estimates have not converged.

Figure 4.2 plots the effective throughput for the cross traffic connections. On comparing with the plots in Figure 4.1, it is easy to observe that whenever the effective throughput of the end-to-end connections in a specific scheme is worse than that with datagram TCP, the throughput of the cross-traffic connections is correspondingly higher. Excluding ATM-FCVC, all the schemes favored cross-traffic connections over end-to-end connections when the buffer size was less than 200 Kbytes; beyond 200 Kbytes, the throughputs of the two types of connections are comparable.

4.2 Retransmissions

We now consider the retransmissions of TCP segments that take place in the network. Retransmissions are defined as the ratio of the total number of retransmissions that take place in the network over the total number of packet transmissions. That is,

$$\text{Retransmissions} = \frac{\text{number of retransmissions}}{\text{number of total transmissions}},$$

where the number of total transmissions is the sum of the number of retransmissions and the number of transmissions of original packets.

Figures 4.3 and 4.4 show the total number of retransmissions in the network and the retransmissions caused by actual packet losses (or cell losses in the case of ATM networks), respectively, for the three simulated TCP segment sizes for all the four schemes simulated. It is easy to infer from this figure how the buffer size affects the number of retransmissions in the network; any increase in the buffer size over the range 50–250 Kbytes results in a corresponding decrease in the number of retransmissions.

The number of retransmissions under datagram TCP is lower than those under both the ATM and ATM-EPD schemes, especially with large TCP segments. The plain ATM scheme is expected to incur more retransmissions since the probability of loss of a single cell is fairly high as can be verified by Figure 4.4, and this is reflected in the retransmissions of TCP segments. In the case of the ATM-EPD, since the transmissions of cells that belong to corrupted packets in not very likely, the difference comes from the fact that ATM-EPD behaves as the TCP scheme but with less buffer space available.

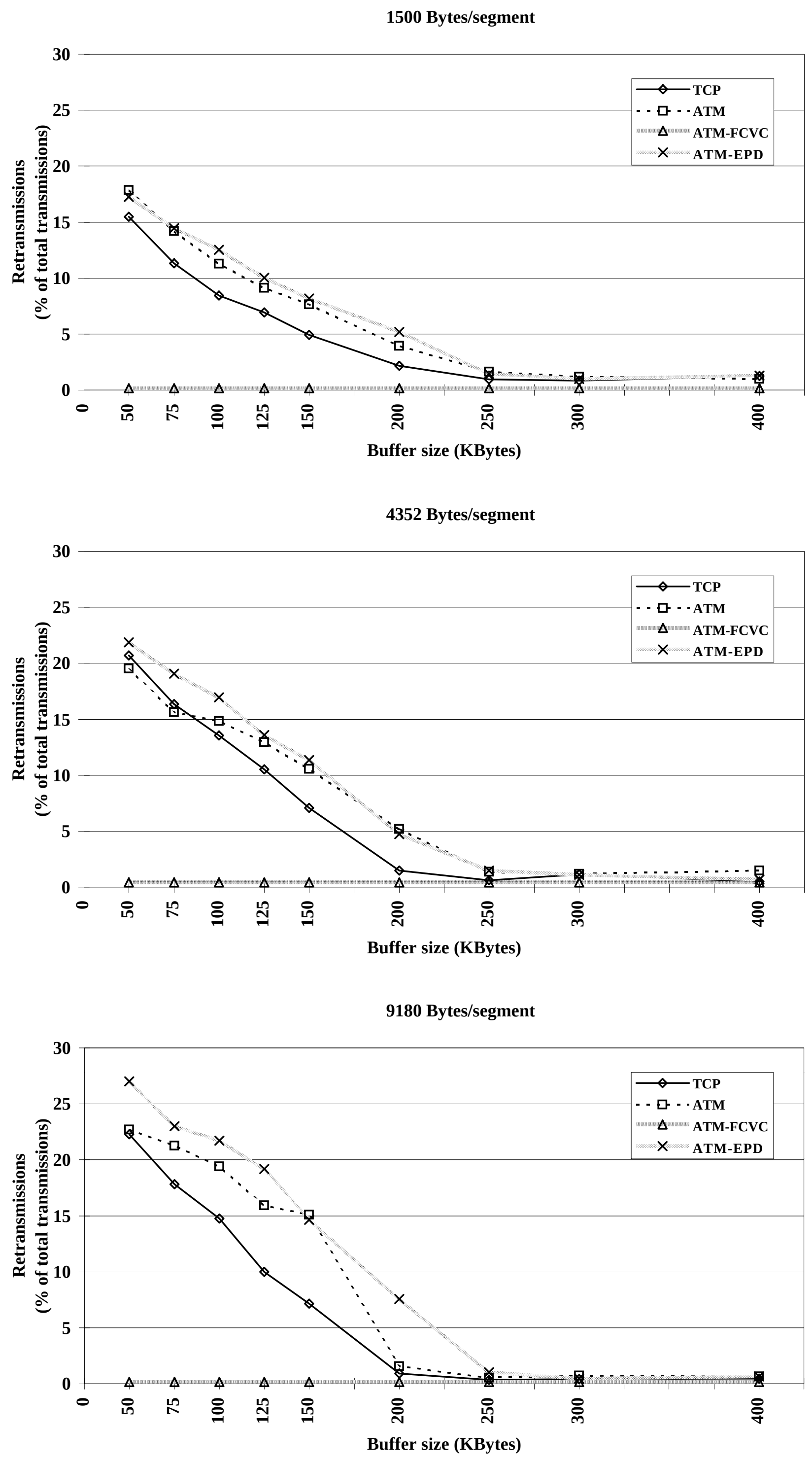


Figure 4.3: Total retransmissions (as a percentage of the total transmissions) in the network as a function of buffer size.

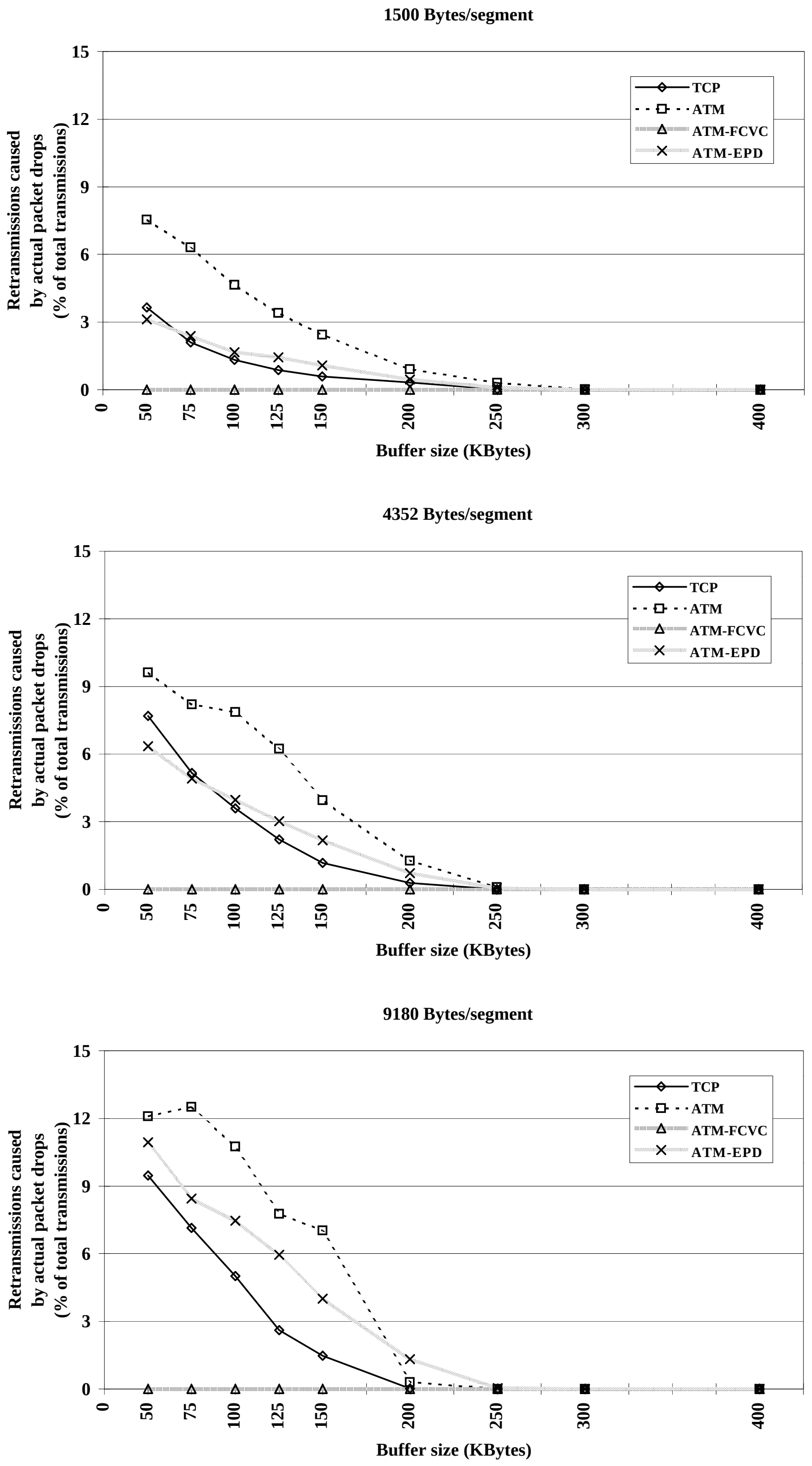


Figure 4.4: Retransmissions (as a percentage of the total transmissions) caused by packets dropped in a switch, as a function of buffer size.

For small buffer sizes the ATM-EPD seems to perform like the TCP scheme (as seen by the end-to-end connections). However, if we examine Figure 4.3, it is easy to see that the number of retransmissions for the ATM-EPD scheme is more than that in datagram TCP. Since the effective throughput of the end-to-end connections for the ATM-EPD scheme is close to (and is some cases slightly higher than) that of the TCP scheme, but the number of retransmissions higher, we conclude that the effective throughput of the cross-traffic connections for ATM-EPD is less than that of the TCP scheme. This is also verified by Figure 4.2.

The ATM-FCVC scheme achieves the smallest number of retransmissions. Since congestion losses cannot occur under this scheme, all the retransmissions are due to early timeouts. These timeouts occurred as a result of the low variance of the measured RTT. When the RTTs remain remain relatively unchanged for a long period of time, the timeout estimates converge to actual RTT values. At this point, a slight increase in RTT may cause the timer to expire and trigger a packet retransmission. These are unlikely to occur in practice, however, because of the coarse-grain TCP timers used in current implementations. Also, some of the retransmissions are due to the effects of the simultaneous opening of the TCP connections.

4.3 Fairness in Bandwidth Allocation

So far we focused on the behavior of the end-to-end connections and cross-traffic connections separately. In this section, we examine the fairness in the bandwidth allocation between the two classes of connections under each of the congestion-control policies.

In the case of datagram TCP, plain ATM, and ATM-EPD, where there is no explicit bandwidth allocation, we define fairness to be the ratio of the bandwidth obtained by the end-to-end connections over that of the cross-traffic connections. In the case of ATM-FCVC scheme, where bandwidth is allocated explicitly, the fairness needs to be computed in a slightly different manner. We must first normalize the measured effective throughput of the two types of connections over the maximum attainable bandwidth that they may take, assuming a perfectly fair allocation. This latter definition applies to all the scheme, since the maximum attainable effective throughput for each type of connections in TCP, ATM, and ATM-EPD schemes, under a perfectly fair allocation, is 50% of the link capacity. Thus, the definition of the fairness we use is given by:

$$\text{fairness} = \frac{\left(\dfrac{\text{end-to-end effective throughput}}{\text{maximum end-to-end effective throughput under fair allocation}}\right)}{\left(\dfrac{\text{cross-traffic effective throughput}}{\text{maximum cross-traffic effective throughput under fair allocation}}\right)}.$$

For the ATM-FCVC scheme, the maximum end-to-end effective throughput for the specific network configuration is 66.67% and the maximum cross-traffic throughput is 33.33%, assuming fair allocation of bandwidth among the virtual circuits that share a common link. For other schemes, both the maximum end-to-end effective throughput and the maximum cross-traffic throughput are set to 50%, since there is no VC-level bandwidth allocation.

Using the above definition, it is desirable for the fairness to attain a value as close to 1 as possible. A value of 1 implies a perfectly fair allocation, but is difficult to achieve in a real network. In practice, a value within the range 1 ± 0.2 can be considered adequate.

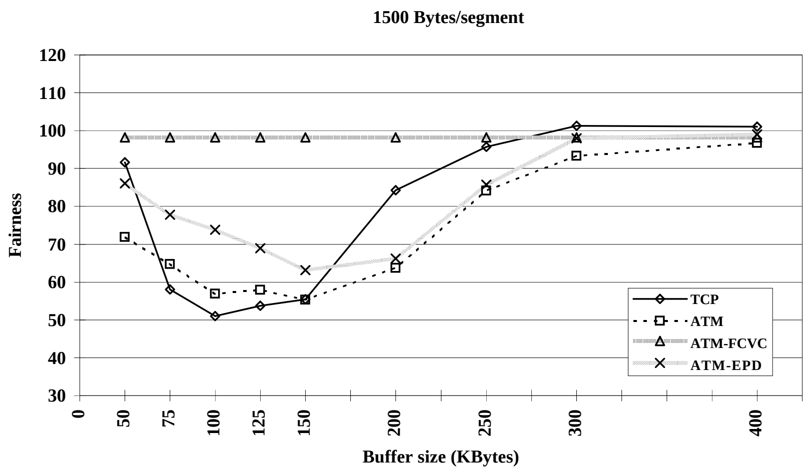
Figure 4.5 presents the results on fairness in bandwidth allocation as a function of the buffer size for all the three TCP segment sizes considered. First, we observe that the ATM-FCVC scheme is almost perfectly fair. This is expected since the scheme is capable of providing explicit bandwidth allocation. The fairness, however, is not 100% because of a small number of early retransmissions (caused by wrong RTT estimations) that occurred even in this scheme.

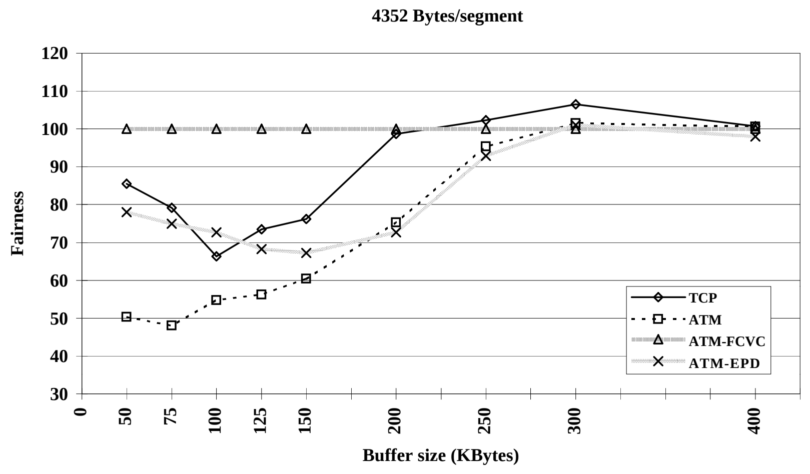
For the remaining schemes, the fairness varies significantly as a function of the buffer size. The behavior can be explained by considering how the buffer size affects the effective throughput of the two types of connections. For small buffer sizes, both the end-to-end and cross-traffic connections face severe packet losses and therefore the bandwidth allocation seems to be fair enough for the TCP and ATM-Early scheme, and for the ATM scheme with 1500-byte segments. As explained in the previous section, the cross-traffic connections attain slightly higher throughput because of their responsiveness. As the buffer size is increased, the cross-traffic connections take most of the additional throughput because of their aggressiveness, making the bandwidth allocation less fair. This behavior extends until the time when the buffer size is large enough to let the end-to-end connections increase their own throughput.

The ATM scheme seems to behave close to datagram TCP for small segment sizes. In the case of large segment sizes, however, the wasted buffer space and link bandwidth due to the handling of cells that belong to corrupted packets significantly impacts the bandwidth allocation, making the bandwidth allocation extremely unfair. The fairness reaches an acceptable region only when the amount of buffering is sufficient to achieve very low packet-loss rates.

Note that fairness assumes a value of greater than 1 in some cases in Figure 4.5. These correspond to cases where the end-to-end connections take slightly more bandwidth than the cross-traffic ones. This is because a few more retransmissions in the cross-traffic connections caused by an early timeout can be harmful enough. The cross-traffic connections will need several RTT delays before they will reach the previous operating point while the end-to-end connections maintain their steady rate during this time interval. It is just a matter of chance which connections go through more early retransmissions. In either case, however, the impact on the fairness due to early retransmissions is small.

The fairness behavior of ATM-EPD resembles a shifted version of that of datagram TCP. This is again due to the ATM-EPD scheme behaving like packet TCP, but with a smaller buffer size. Note also that the ATM-EPD is fairer than datagram TCP in some cases. This is due to the reason outlined in the previous section: Since a packet is transmitted in the ATM network as a burst of individual cells, whenever the buffer-occupancy threshold in a switch is exceeded, there is a higher probability for the cells of a packet that belong to cross-traffic connections to be dropped, thus reducing their aggressiveness slightly.





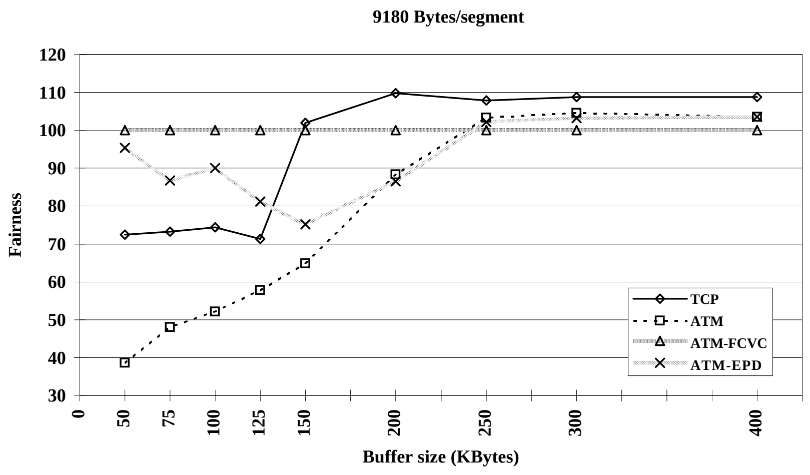


Figure 4.5: Degree of fairness in bandwidth allocation between the end-to-end and the cross-traffic connections.

4.4 Mean Packet Delivery Time

In this section we present results that show how each scheme affects the *mean packet delivery time* and its standard deviation. The mean packet delivery time is defined as the elapsed time from the *first* transmission of a segment to the receipt of an acknowledgement that covers this segment. For convenience, we measured only the packet delivery time for the segments that are timed by TCP. Obviously, this definition includes the delays caused by the retransmissions of the timed segment. Figures 4.6 and 4.7 present the simulation results.

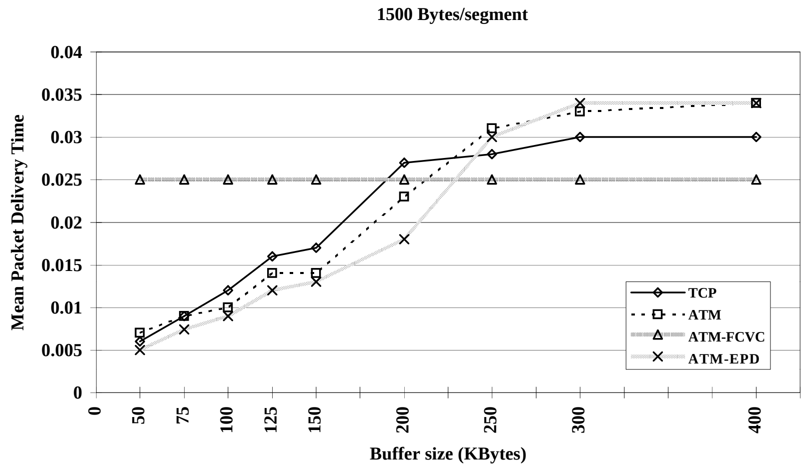
The packet delivery time is affected by two factors. The first one is the retransmissions caused by packet losses and the second is the queueing delays in the switches. Retransmissions caused by wrong RTT estimation are not contributing significantly to the packet delivery time since very soon the acknowledgement that covers the segment which retransmitted early will be received. The queueing delays in our results seem to dominate for relatively large buffer sizes.

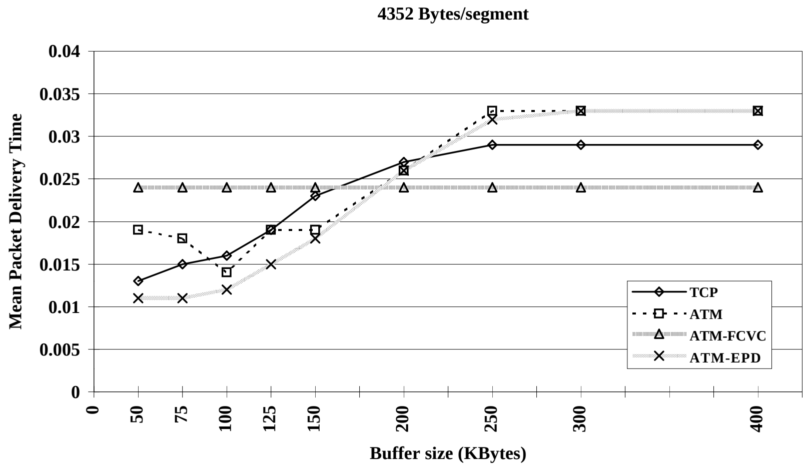
For small packet sizes, where the number of retransmissions due to dropped packets is small, the delays are mainly due to queueing in the switches. We can verify in the plots presented that an increase in the buffer size generally results in an increase in the packet delivery time. For very large buffer sizes we observe that the delays for the ATM and ATM-EPD schemes are higher than that for the TCP scheme. These additional delays are due to the fact that in the ATM-based schemes, the cell headers are also stored.

For larger packet sizes, the delays due to the retransmission caused by actual packet losses dominate when the buffer size is small, leading to fairly high packet delivery times. As the buffer size is increased further, the retransmissions decrease and the queueing delays start dominating.

The packet delivery times in the ATM-FCVC scheme is higher than that of all the other schemes when the buffer sizes in the latter schemes are small. The latter schemes, however, require buffer sizes in the range of 150–200 Kbytes per switch port to attain an acceptable degree of fairness and throughput. At these buffer sizes the queueing delays are significant, causing higher delays in these schemes as compared to ATM-FCVC.

Figure 4.7 presents the standard deviation of packet delivery times. For small packet sizes, where the packet loss rate is also small, the standard deviation increases with increasing buffer size until the buffer size is large enough not to cause any more losses. At that point the standard deviation starts to decrease. For larger packet sizes, however, the packet loss rate is much higher, and the standard deviation has a relatively large value. On increasing the buffer space the packet loss rate decreases, but the large variations in the instantaneous queue sizes in the switches keeps the standard deviation at a relatively large value. For even larger buffer sizes the packet losses are small, and the occupancy of the queues tend to be more constant, resulting in a decrease in the standard deviation.





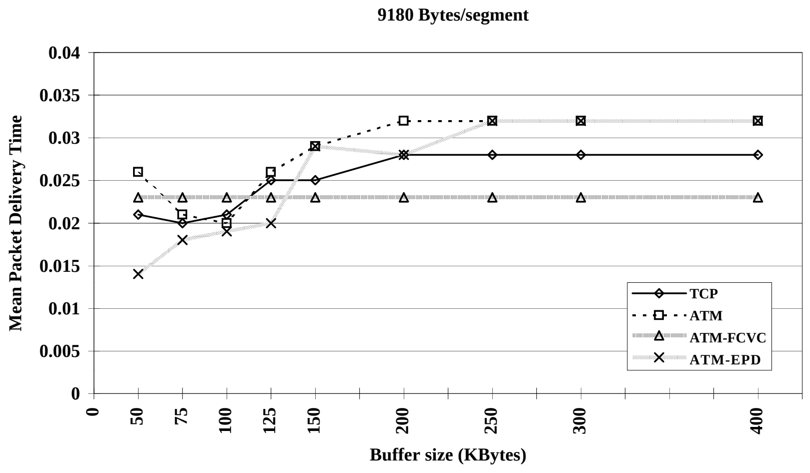
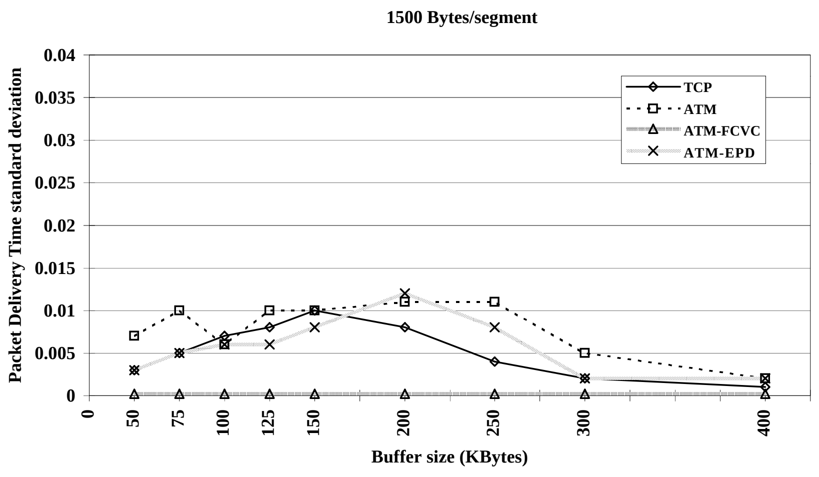
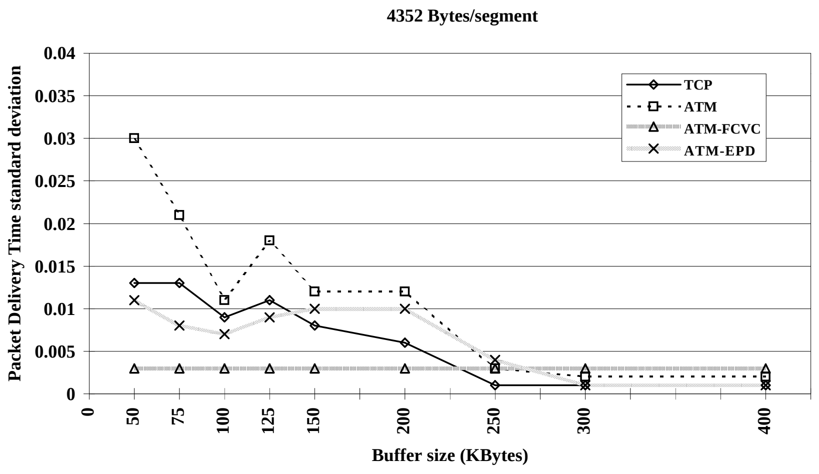


Figure 4.6: Mean packet delivery time for the *end-to-end* connections as a function of the switch buffer size for all the simulated schemes.





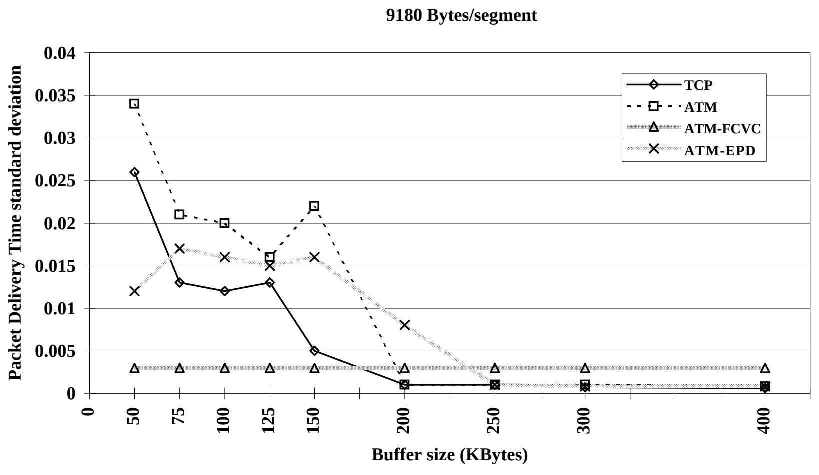


Figure 4.7: Packet delivery time standard deviation for the *end-to-end* connections as a function of the switch buffer size for all the simulated schemes.

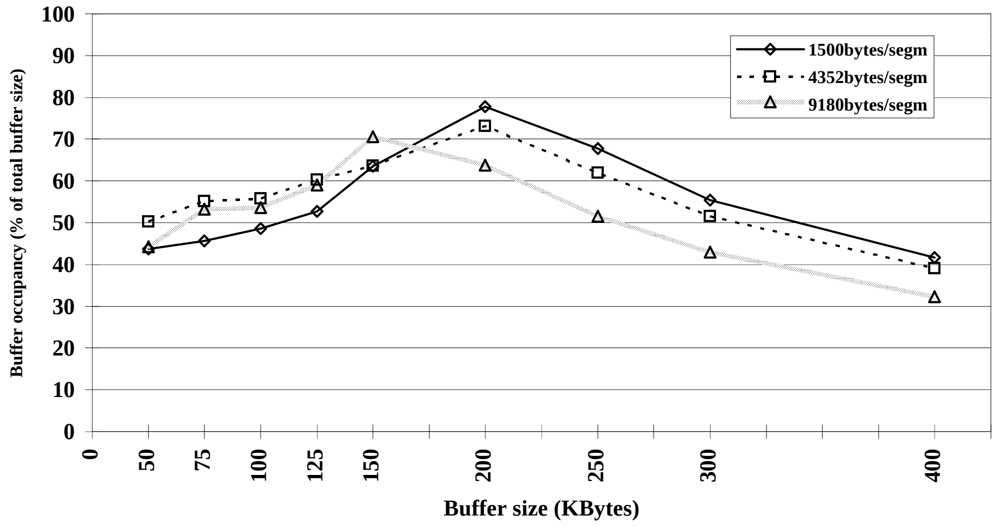


Figure 5.1: Average buffer occupancy as a function of the buffer size for the TCP scheme.

5 Setting the Threshold for ATM-Early Packet Discard

A critical decision in switches employing the ATM-EPD scheme is how to set the threshold value. We now present some results that will help gain insight into how the threshold value affects the scheme's performance.

There are two general approaches in setting the buffer threshold. The first one set the threshold as a percentage of the total buffer size. This approach, however, can lead to a very conservative configuration. If the threshold is set to be too small, then most of the buffer space remains un-utilized and is actually wasted. For example, if the buffer size is 50 Kbytes and the threshold is set to 80%, then only 10 Kbytes of the buffer will be reserved. However, if the buffer size is 300 Kbytes, then the reserved space will be 60 Kbytes, far more than what is necessary for the ATM-Early scheme. Figure 5.1 shows the average buffer occupancy in the switches (the specific measurements are from Switch-2) as a function of the buffer size for the TCP scheme. We chose this scheme because all the buffer space is made available to the connections. We can easily realize from this plot how conservative a threshold value of 50% or 60% can be.

The alternative approach is to set the threshold to a value that will always reserve a certain amount of space (for example, the equivalent space of two packets). This approach also seems to be more reasonable to be used under TCP. The reason is that TCP will control the traffic going through a specific link in order to avoid the congestion. Thus, on the average, TCP will not send more packets than can be serviced by the link and since the same will happen to all the TCP connections we expect that, on the average, only one TCP segment will be queued in the buffer per segment transmission time. However, since the TCP segment is split into ATM cells, it is possible for cells belonging to

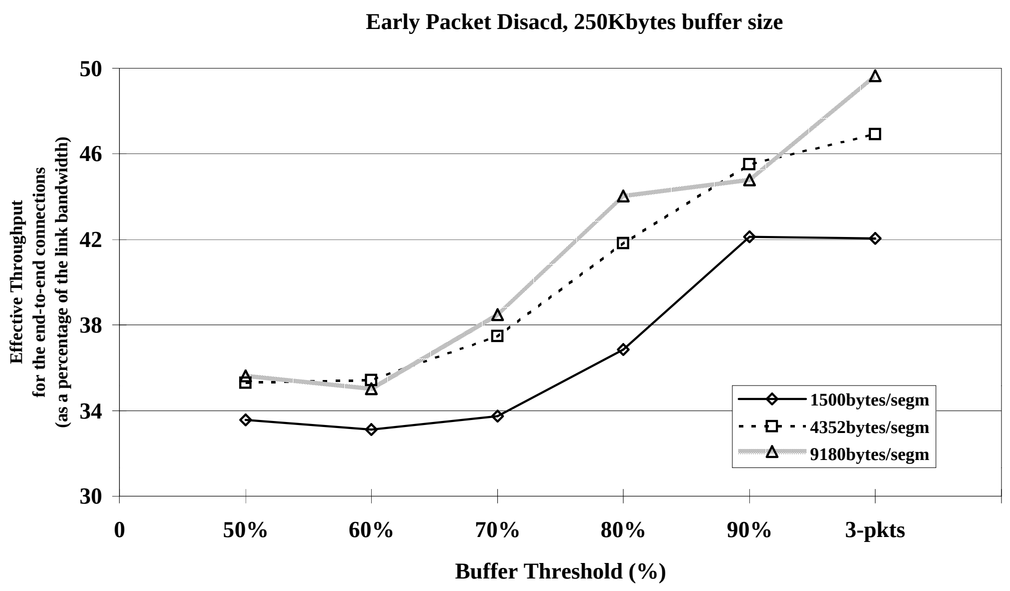


Figure 5.2: Effective throughput of the end-to-end connections under the ATM-EPD scheme as a function of the threshold value (buffer size = 250 Kbytes).

segments from different TCP connections to get interleaved; thus the reserved space should be set to more than one segment size. Our simulations suggest that a value more than two segments is adequate.

The advantage of the latter scheme is that it avoids wasting space when the buffer is large. However, it can be very conservative when the buffer is small (for example, 50 Kbytes) and the packet size large (9180 Kbytes). In this case, more than half of the buffer space is reserved. Therefore, in our simulations, we used a combination of the two approaches to avoid the problems mentioned above.

Figure 5.2 shows the effect of the threshold value on the effective throughput of the end-to-end connections. As can be seen, the performance can be very poor with small threshold values. Also notice that the choice of threshold used in our simulations (indicated in the plots as *3-pkts*) gives best results in all the cases. Figure 5.3 shows the retransmissions that take place in the network as a function of the threshold value. Again, a poor selection of threshold value can lead to excessive number of retransmissions.

6 Conclusions

We presented simulation results in an effort to compare the performance of TCP over multi-hop datagram and ATM networks. For the case of ATM networks we modeled and simulated three schemes: ATM without any congestion control at the ATM layer, ATM with Early Packet Discard strategy and ATM with Flow-Controlled Virtual Channels. The schemes were compared in terms of the effective throughput achieved by specific connections, retransmissions that took place in the network and the fairness in bandwidth allocation between connections with long and short round-trip delays. We also

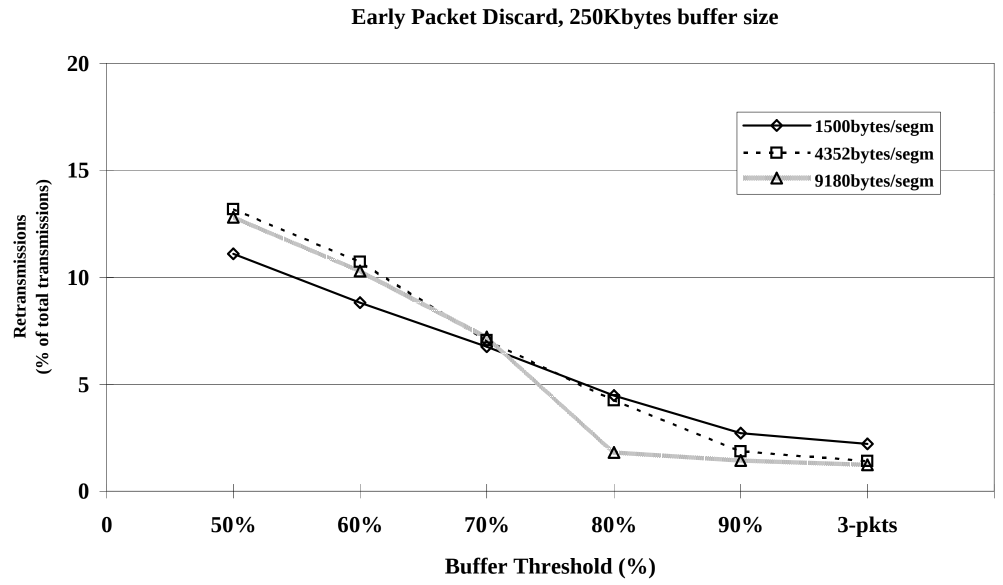


Figure 5.3: Retransmissions that take place in the network under the ATM-EPD scheme as a function of the threshold value for 250-Kbyte buffer.

simulated TCP over a datagram network to provide us a basis for comparison. Performance close to that of datagram TCP can be considered acceptable in most cases.

Our results show that TCP over ATM networks without ATM-layer congestion control may operate very inefficiently, especially when the buffer sizes in the switches are small. Although the total network throughput and the retransmissions are close to those of datagram TCP, its behavior can be very unfair.

TCP over ATM networks with early packet discard seems to provide performance comparable to that of datagram TCP, except when the buffers have a medium size; in this case it operates like the TCP scheme but with less buffer space available. We observed also that in some cases it behaves even more fair as a result of splitting the packet into a large number of cells. Our simulations showed that this scheme eliminated the problem of transmitting cells that belong to corrupted packets. However, since the behavior of this scheme resembles closely that of TCP over a datagram network, its behavior can be expected to be unpredictable and, in some cases, unfair. In addition, the use of the scheme requires the switches to have knowledge of the high-level protocol data units. Also, setting the threshold in a network where the packet sizes vary can be difficult and can lead to conservative results. Finally, the memory requirements under this scheme are determined by the characteristics of TCP, not by those of the ATM layer.

Finally, our simulations verified the superiority of the ATM-FCVC scheme. It achieves perfectly fair bandwidth allocation, almost zero retransmissions and, most importantly, its performance is predictable. Part of the inherently high performance of the ATM-FCVC scheme is due to explicit bandwidth allocation. Also, the memory requirements for this scheme are solely dependent on the network configuration and are independent of the high-level protocols. TCP seems to adjust well and to operate efficiently over this scheme. However, problems may arise if some of the proposed changes are incorporated to TCP to make it work more efficiently over networks with high bandwidth-delay products [6]. The TCP timing algorithms can fall into a periodic behavior which can cause unacceptably large number of early retransmissions. This problem needs to be investigated further.

Acknowledgements

This research is supported by the Advanced Research Projects Agency (ARPA) under Contract No. F19628-93-C-0175 and by the NSF Young Investigator Award No. MIP-9257103. We thank MIL-3 for donating us the OPNET modeling and simulation tools under their university program.

References

[1] CCITT, "Draft Recommendation I.363", CCITT Study Group XVIII, Geneva, January 1993.

[2] L. S. Brakmo, S. W. O'Malley, and L. L. Peterson, "TCP-Vegas: New Techniques for Congestion Detection and Avoidance," *Proceedings of SIGCOMM '94*, September 1994.

[3] S. Floyd and V. Jacobson, "On Traffic Phase Effects in Packet-Switched Gateways," *Internetworking: Research and Experience*, pp. 115-156, September 1992.

[4] S. Floyd and V. Jacobson, "Early Random Detection Gateways for Congestion Avoidance," *IEEE/ACM Transactions on Networking*, Vol. 1, No. 4, August 1993, pp. 397–413.

[5] V. Jacobson, "Congestion Avoidance and Control,", *Proceedings of Sigcomm'88*, pp. 314–329, 1988.

[6] V. Jacobson, R. Braden, and D. Borman, "TCP Extensions for High Performance'," *Request for Comments: 1323*, May 1992.

[7] R. Jain, "Myths about congestion management in high-speed networks," *Internetworking: Research and Experience*, Vol. 3, No. 3, pp. 101-113, September 1992.

[8] P. Karn and C. Partridge, "Improving Round-Trip Time Estimates in Reliable Transport Protocols," *ACM Transactions on Computer Systems*, Vol.9, No 4, pp. 364-373, November 1991.

[9] H. T. Kung and A. Chapman, "The FCVC (Flow-Controlled Virtual Channels) Proposal for ATM Networks," *Proceedings of the 1993 International Conference on Network Protocols*, October 1993, pp. 116–127.

[10] H.T. Kung, R. Morris, T. Charuhas and D. Lin, "Use of Link-by-Link Flow Control in Maximizing ATM Network Performance: Simulation Results," *Proc. IEEE Hot Interconnects Symposium*, 1993

[11] H. T. Kung, T. Blackwell, and A. Chapman, "Credit-Based Flow Control for ATM Networks: Credit Update Protocol, Adaptive Credit Allocation, and Statistical Multiplexing," *Proceedings of SIGCOMM '94*, September 1994.

[12] A. Mankin and K. K. Ramakrishnan, "Gateway Congestion Control Survey," RFC 1254, August 1991.

[13] P. Newman, "Traffic Management for ATM Local Area Networks," *IEEE Communications*, August 1994, pp. 34–50.

[14] K. K. Ramakrishnan and R. Jain, "A Binary Feedback Scheme for Congestion Avoidance in Computer Networks," *ACM Transactions on Computer Systems*, Vol. 8, No. 2, pp. 158-181, May 1990.

[15] A. Romanow and S. Floyd, "Dynamics of TCP traffic over ATM Networks," *Proceedings of SIGCOMM '94*, September 1994.

[16] W. R. Stevens, *TCP/IP Illustrated, Vol. I*, Addison Wesley Publishing Company, 1994.

[17] G. Varghese, C. Ozveren, and R. Simcoe, "Reliable and Efficient Hop-by-Hop Flow Control," *Proceedings of SIGCOMM '94*, September 1994.

[18] L. Zhang and D. D. Clark, "Oscillating Behavior of Network Traffic: A Case Study Simulation,", *Internetworking: Research and Experience*, Vol. 1, pp. 101-112, 1990.

[19] L. Zhang, S. Shenker, and D. D. Clark, "Observations on the Dynamics of a Congestion Control Algorithm: The Effects of Two-Way Traffic," *Proceedings of Sigcomm'91*, pp. 133–147, 1991.

Appendix A: Memory Requirements for the ATM-FCVC Scheme

Figure 4.1 suggests that in order for the ATM-EPD scheme to behave fairly with a small number of retransmissions, the required buffer space per switch port should be in the range of 150–200 Kbytes. Thus, ATM-EPD needs at least 600 Kbytes of memory (150 Kbytes in every switch since in our configuration only one port is congested) before it will become fair and efficient. Notice that in this estimate we did not count the buffering needed in the source and destination nodes.

In the ATM-FCVC scheme, each VC is allocated its own queue at every switch it traverses. In the FCVC scheme by Kung and Chapman [10], each VC queue is divided into two regions: N2 and N3. The N2 region determines the frequency at which credit cells are sent to the upstream node and its size is fixed. The N3 region determines the peak bandwidth that each VC may occupy when the other VCs are inactive. Given the RTT between the current and the downstream node and target peak bandwidth, the size of the N3 region (in cells) can be computed as:

$$N3 = \frac{RTT \times B_{VC}}{\text{cell size}},$$

where RTT is the round-trip-delay between the current and the downstream switch and B_{VC} is the VC's target peak bandwidth.

In our simulations the N2 region was set to 20 cells. Table 6.1 summarizes the sizes of the N3 regions for all the links in the simulation network configuration, as well as the total amount of memory needed in the network. Note that the total memory requirement of 612,574 bytes is comparable to the amount of buffering needed in ATM-EPD to obtain maximal TCP throughput. However, it should be noted that the basic ATM-FCVC allows no sharing of buffers among the VCs sharing a common link; hence the amount of memory needed for ATM-EPD is likely to be much less than that for ATM-FCVC when a large number of VCs share the links. As a final note, the amount of memory needed in ATM-FCVC can be reduced substantially by employing one of the schemes proposed for sharing the memory among VCs [10, 17].

Link delay (ms)	N2 size (cells)	N3 size (cells)	Total VCs	Buffer size (cells)	Buffer size (bytes)
0.1	20	73	26	2,418	128,154
0.95	20	675	8	5,560	294,680
2.5	20	1,770	2	3,580	189,740
			Total:	11,558	612,574

Table 6.1: Total memory requirements for the simulation network configuration with ATM-FCVC.