Early Selective Packet Discard for Alternating Resource Access of TCP over ATM-UBR[*]

Kangsik Cheon[†] and Shivendra S. Panwar
Center for Advanced Technology in Telecommunications
Polytechnic University
5 Metrotech Center, Brooklyn, New York 11201
Email: {kcheon, panwar}@kanchi.poly.edu

Abstract

We investigate packet discarding schemes for TCP over ATM with UBR service. In doing so, we tested the effective throughput of two existing schemes, Partial Packet Discard(PPD) and Early Packet Discard(EPD), as compared to the Random Cell Discard(RCD) scheme which discards any incoming cells after buffer overflow. We observed that PPD alleviates the effect of packet fragmentation so that it gets effective throughput enhancement over RCD, and EPD provides further enhancement over PPD. After closer investigation, we found that there is a sustained congestion problem other than packet fragmentation that causes the effective throughput to be degraded. We noted that sustained congestion resulted in the synchronization of TCP window expansion and shrinkage. To provide a solution for this problem, we propose the Early Selective Packet Discard(ESPD) policy, a strategy which makes sessions take turns in accessing network capacity by discarding packets from selected sessions rather than randomly. Our results shows that ESPD achieves throughput and fairness enhancement over EPD with only a modest increase in implementation complexity.

1 Introduction

For applications like non-real-time data traffic, the ATM Forum has defined two different services, available bit rate(ABR) service and unspecified bit rate(UBR) service [1]. ABR service has attracted much attention as a research topic in recent years and is expected to deliver better quality of service than UBR. But, considering the fact that the implementation complexity and cost of ABR is significantly higher than UBR, and that other low-cost technologies for high speed networks like gigabit Ethernet are developing rapidly, there is interest in investigating the possibility of using UBR services as an interim low-cost alternative to ABR service [2-7,11].

The most commonly proposed congestion control schemes for ATM-UBR are the partial packet discard(PPD)[9] and early packet discard(EPD) scheme[6]. A variation of EPD with multiple thresholds designed to improve performance is discussed in [12].

PPD discards cells after buffer overflow and keeps discarding the following cells of the same packet. This behavior of PPD leads to discarding of the tail part of each packet. With the intention of discarding an entire packet instead of a partial packet, EPD was introduced[7]. EPD discards BOM(Begin of Message) cells and the following cells of the same packet after the buffer occupancy exceeds a threshold. A detailed description of the operation of PPD and EPD is given in section 3.

EPD and PPD have been shown to lead to significant performance improvement for TCP connections [2, 6]. The improvement is mainly due to the fact that EPD and PPD discard cells selectively. But, when it comes to packet discard, EPD discards packets *randomly* so that it spreads packet losses over many sessions. This randomness triggers timeouts across many TCP sessions and causes TCP synchronization, resulting in underloaded network resources.

Based on this observation, we propose a new packet discarding scheme, the *Early Selective Packet Discard(ESPD)*, which has an added packet selection mechanism compared to EPD. This allows ESPD to force selected sources to shrink their window size via the implicit feedback control of TCP resulting in TCP desynchronization.

[*]This work is supported in part by Sumitomo Electric Industries, Inc., and the New York State Center for Advanced Technology in Telecommunications Polytechnic University, Brooklyn, New York.

[†]Kangsik Cheon is also a member of technical staff at Korea Telecom, Seoul, Korea.

In an earlier effort to avoid TCP synchronization, RED(Random Early Detection) gateway was proposed[11]. RED gateway drops or marks each arriving packet with a certain probability when the average queue length exceeds a preset threshold. The probability is a function of the average queue length.

Section 2 describes the simulation network model and the parameter values that we used. Section 3 discusses how each packet discarding scheme works. In section 4, we present the simulation results, address the problems of existing packet discarding schemes and how the new scheme achieves a throughput enhancement. Finally in the conclusion in section 5, we propose future work.

2 Simulation Network and Parameters

In this section, we describe the simulation environment used in our simulation of TCP over ATM-UBR. Our simulation tool is based on OPNET(OPtimized Network Engineering Tools) which is provided by MIL 3, Inc. OPNET is a network simulator capable of simulating large communications networks with detailed protocol modeling and performance analysis.

We simulated two different parameter sets with the network configuration shown in Fig. 1. One set is for an congestion situation where all the TCP connections have the same packet roundtrip times, and the other set is for a congestion situation with asymmetric packet roundtrip times for TCP connections. We also performed, with an augmented network, another simulations of which results are discussed in Appendix.

Fig. 1 shows a network configuration consisting of 3 ATM switches, 10 source nodes and 10 destination nodes. In the figure, *src* refers to the source node whose application layer corresponds to a TCP packet source and *dest* refers to the destination node whose application layer corresponds to a traffic sink. The source node i identified by *src_i* communicates with the destination node i identified by *dest_i*. All application layers use TCP to communicate with their peer layers at the destination node. Each source node sends out as many packets as permitted by the sliding window control of TCP. Our TCP implementation does not use coarse-grain timers.

The packet inter-arrival time of active sources onto the ATM adaptation layer is dominated by the IP service rate which is determined by the packet size at the application layer. We set it to 10,500 packets/sec for a packet size of 500 bytes and to 4,000 packets/sec for packet size of 1,500 bytes. As a result, the total possible aggregated traffic coming to the most congested output port of an ATM switch is limited to 3.5 times of the SONET STS-3c speed. The IP buffer size is set to infinity to ensure that there is no packet loss at the IP layer.

The function of the ATM adaptation layer of our simulation is very simple. It adds a 14 byte overhead to each IP datagram, and adjusts the size to multiples of 48 bytes. It then does segmentation, adds a 5 byte overhead to each cell, sets an EOM(End of Message) bit at the last cell corresponding to each packet and forwards the cells down to the ATM layer. When an end station receives cells from a peer node and needs to forward them to the upper layer, it performs the reverse functions. Each of the links is full duplex with a bandwidth capacity of 155.52 Mbps, corresponding to the capacity of a SONET STS-3c link. The ATM switch of our model network is a simple output buffered switch that just reads VPI/VCI information of arriving cells and forwards them to the corresponding output port.

Since the sample TCP model of OPNET version 2.4 was based on RFC 793, we added the key components of the congestion control algorithm of TCP, such as the slow-start algorithm, congestion-avoidance mechanism, fast retransmit and fast recovery algorithm and a delayed acknowledgement scheme[8]. In the TCP implementation of OPNET version 2.4, each packet being transmitted from the source node has its own RTO timer so that once a timeout occurs, many subsequent timeouts generally follow. We change this timer scheme so that the resulting TCP sessions have one RTO timer each, making it consistent with common implementations. Because we intended to simulate the *ftp* application, we also devised our own application layer which can generate and inject packets down to the TCP layer as soon as TCP moves the sliding window forward. Each TCP connection has its own corresponding ATM virtual circuit. All the other parameters which are not mentioned above are as follows:

- Parameter values for symmetric roundtrip simulations
  - Minimum TCP RTO(Retransmission Timeout): 50 msec
  - Maximum waiting time for a delayed ACK: 12.5 msec
  - Delay of the five links connecting *src_6* to *src_10*: 2 msec
  - Delay of all the other links: 1 msec
  - Simulation time: 3 sec
  - We ran 10 simulations with different seeds.

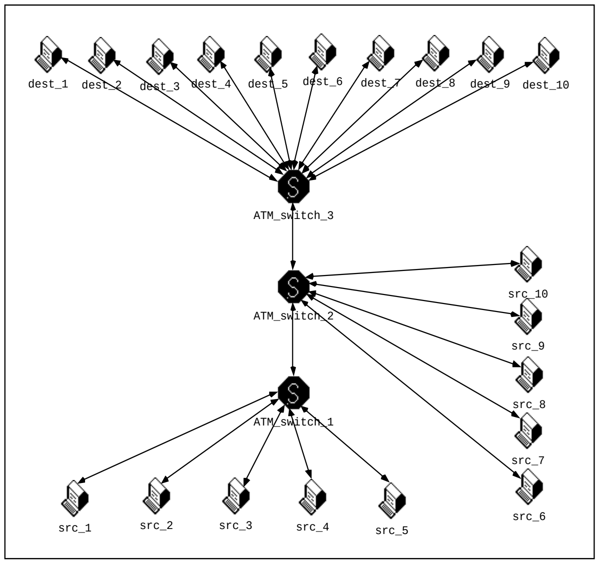

Figure 1: Simulation Network

- Parameter values for asymmetric roundtrip simulations
  - Minimum TCP RTO(Retransmission Timeout): 150 msec
  - Maximum waiting time for delayed ACK: 37.5 msec
  - Delay of the five links connecting *src_6* to *src_10*: 3 msec
  - Delay of the link connecting *ATM_switch_1* and *ATM_switch_2*: 47 msec
  - Delay of all the other links: 1 msec
  - Simulation time: 10 sec
  - We ran 6 simulations with different seeds.
- Common parameter values for both simulations
  - Maximum window size of TCP: 64 Kbytes
  - TCP header size: 20 bytes
  - IP header size: 20 bytes
  - Packet length at application layer: 500 bytes and 1,500 bytes
  - Zero TCP processing time
  - Buffer size of ATM switch: 2,000 cells/port

3 Packet Discard Schemes

3.1 Partial Packet Discard(PPD)

In order to transmit a packet to the designated destination node through the ATM network, a packet is segmented into smaller 53-byte long ATM cells. All the cells must then travel through the ATM network without even one cell being lost for the integrity of the entire packet. If the network drops cells randomly when we have congestion, then the dropped cells may be spread over different packets. This in turn leads to useless cells in the buffer in the sense that any group of cells which cannot be reconstructed back to a packet at the destination node will have to be thrown away. The problem does not end here. The resulting useless cells are still forwarded towards their destination, hence they waste network bandwidth as well as buffer capacity, and sometimes contribute to congestion at the following nodes. This phenomenon is referred to as the fragmentation problem[6].

To alleviate the negative effect of packet fragmentation, the PPD(Partial Packet Discard) scheme was introduced by G. Armitage and K. Adams [9]. The following is the pseudocode of PPD. The 'drop-list' in the following is a set of virtual circuits for which any cell of a packet has been dropped already.

```
When a cell arrives at an ATM buffer:
if the cell's VPI/VCI belongs to drop-list
  discard the cell
  if the cell is an EOM cell
    remove the VPI/VCI from the drop-list
else
  if the buffer is full
    discard the cell
    capture the VPI/VCI into drop-list
  else
    accept the cell into the buffer.
```

As we can see, PPD discards an incoming cell when a buffer is full and also discards all the subsequent cells of the packet. By doing so, we discard either the tail end of a packet or an entire packet. This means we can reduce the amount of useless cells to a certain degree. This results in better throughput than random cell discard(RCD).

3.2 Early Packet Discard(EPD)

EPD was proposed [7] with the aim of discarding an entire packet prior to buffer overflow, so that corrupted packets will not be transmitted by the switches.

The EPD mechanism sets a certain buffer threshold considering system parameters, and as soon as the queue length exceeds the threshold, the ATM switch is *ready to discard* incoming cells. The meaning of being *'ready to discard'* is to wait for incoming BOM(Begin of Message) cell instead of immediately discarding any incoming cell. After that, whenever the switch sees a BOM cell, and as long as the queue length is above the threshold, it drops the cell. All the subsequent cells of the packet following the dropped BOM cell are discarded. By doing so, and by setting the proper threshold, we can emulate packet discarding almost perfectly. The following is the pseudocode of EPD, which we implemented by using the OPNET simulator.

```
When a cell arrives at an ATM buffer:
if the cell's VPI/VCI belongs to drop-list
   if the cell is an EOM cell
      if queue_length < buffer_size
         insert the cell into buffer
      else
         discard the cell
      remove the VPI/VCI from the drop-list
   else
      discard the cell
else
   if queue_length < Threshold
      insert the cell into buffer
   else if (BOM cell or (the buffer is full))
      discard the cell
      capture the VPI/VCI into drop-list
   else
      insert the cell into buffer
```

As can be seen from the EPD pseudocode, EOM(End of Message) cells enter the buffer whenever the space is available, because the popular AAL5 protocols only use EOM marking to delimit a packet. For our simulation, we set the threshold for packet discarding to about three packets less than the full buffer size[6].

3.3 New Scheme: Early Selective Packet Discard(ESPD)

When congestion occurs, EPD starts to drop the incoming packet irrespective of which session it belongs to. It does not care which session is highly active, in other words, which session consumes more resources than others, or which session is relatively inactive. One of our motivations for a new scheme, ESPD, is trying to drop the packets from the highly active sessions only.

ESPD has a drop timer and three thresholds which are: buffer threshold 1, buffer threshold 2 and drop-list threshold. Buffer threshold 1 is designed to capture VPI/VCI's into the drop-list and buffer threshold 2 is to release VPI/VCI's from the drop-list. The objective of drop-list threshold is to place a limitation on the number of VPI/VCIs in the drop-list. The drop timer is for avoiding unfairness. In this simulation, buffer threshold 1 is set to about six packets less than buffer size and buffer threshold 2 is set to half of buffer size. The drop-list threshold is set to 20 % of total number of sessions, which is 2 in this case of 10 sessions. The drop timer value we used is 25 msec for symmetric roundtrip time simulations and 75 msec for asymmetric roundtrip time simulations.

If the buffer occupancy exceeds the buffer threshold 1, we add the VPI/VCI to a drop-list in the order of BOM cell arrivals. All the subsequent cells except EOM(End of Message) cell are discarded, even though we have available buffer. We add the VPI/VCI to the drop-list based on this *first-come-first-catch* policy because it is likely to find the BOM cells of high activity sessions first during a congestion epoch. Once the number of VPI/VCIs in the drop-list equals the drop-list threshold, we don't capture VPI/VCI any more as long as there is a buffer space available to admit an incoming cell. But by discarding the cells coming from a limited set of sessions, it sometimes may not be sufficient to control an increasing queue length, hence the queue length may continue to grow past the threshold, and finally fill the buffer. When this happens, we discard incoming cells, capture the VPI/VCI into the drop-list, and keep discarding all subsequent cells belong to the VPI/VCI as do PPD and EPD. By doing so, the queue length decreases. Once the queue length becomes less than buffer threshold 2 and EOM cell arrives, then we release the corresponding VPI/VCI from the drop list.

The drop timer is activated as soon as the first VPI/VCI is captured into the drop-list and deactivated when VPI/VCI release from drop list starts according to queue length. ESPD releases all the captured VPI/VCIs simultaneously in case that a drop timer expires prior to the deactivation.

This timer is designed to ensure fairness between sessions. Because ESPD is trying to concentrate packet discarding on a few sessions only, there is a probability that the queue length does not go below buffer threshold 2 for an excessively long period so that the targeted sessions lose fair access to network resource. For this reason, we remove the targeted sessions from the drop-list if the drop timer expires. The following is the pseudocode for ESPD. For better understanding, we also present the corresponding flow chart in Figure 12.

```
When a cell arrives at an ATM buffer:
if the drop timer has expired
   remove all the entries of drop-list
if the cell's VPI/VCI belongs to drop-list
   if the cell is an EOM cell
      if queue_length < buffer_size
           insert the cell into buffer
      else
           discard the cell
      if queue_length < buffer threshold 2
           remove the VPI/VCI from the drop-list
           deactivate the drop timer if it is active
   else
      discard the cell
else
   if queue_length ≤ buffer threshold 1
      insert the cell into buffer
   else if ((# of entries in drop-list < drop-list threshold
             and BOM cell) or the buffer is full)
      discard the cell
      capture the VPI/VCI into drop-list
      if the first VPI/VCI in drop-list
         activate a drop timer
   else
      insert the cell into buffer
```

4 Simulation Results and Analysis

4.1 RCD, PPD and EPD

4.1.1 Sustained Congestion Periods

The upper graphs of Figures 2, 3 and 4 show the queue length tracking at the most congested port of ATM switch with RCD, PPD, EPD respectively. The lower graphs of Figures 2, 3 and 4 indicate which sessions suffered from packet discard during a congestion epoch. The session ID, 1 to 10, means the corresponding session is caught in the drop-list. RCD does not maintain a drop-list, since it just discards cells whenever the buffer is full. Therefore, the session ID in the lower graph of Fig. 2 means simply that cells belonging to the corresponding session ID are dropped.

We notice that the congestion is sustained for some period. To illustrate this point, see Fig. 5, which is a magnified version of the first congestion period of Fig. 4. During that period, the queue length goes above and below the threshold frequently[1]. Hence VPI/VCIs are added to the drop-list in the order of packet arrivals and then released from it as frequently as the queue length fluctuates. This means that RCD, PPD and EPD discard packets randomly, and spread out packet discards over many sessions. This results in a large group of sessions going into an idle period because the corresponding source nodes are stuck due to lost packets, and have to wait till the timeout. Another group of sessions activate the fast retransmission and fast recovery algorithm and get into a relatively inactive period because the window size goes down to half of the current window size. This behavior of sessions leads to frequent underload periods.

In most of the simulation experiments we performed, we found that the duration of congestion is about one roundtrip time. The reason is as follows. Sliding window control can send out as much data as the current window size without getting an acknowledgement. This means that even though a packet is lost in the network due to congestion, the source node does not know it and keeps sending packets as long as it gets the acknowledgement for the packets transmitted before the lost packet. Considering a relatively short inter-packet time, we can say that the source nodes continues to transmit the packet during one roundtrip time. In other words, about one roundtrip time later, the source nodes get an acknowledgement for the packet which was sent just before the lost packet, open new windows, and send the permitted amount of packets. Then the congestion is resolved because the source nodes stop sending new packets.

4.1.2 Synchronized TCP Window Expansion and Shrinkage

This problem is a side effect of the sustained congestion problem. We plot the window-size of the 10 sessions as a function of time for the four discarding scheme we are using in Fig. 8 to 11. As we can see from Fig. 8, for RCD all the sessions have synchronized window expansion and shrinkage. That is, they all shrink window-size at the same time when network congestion occurs, go through a low activity period, expand window-sizes together resulting in another congestion involving all sessions. Fig. 9 and Fig. 10 correspond to the PPD and EPD scheme respectively. As we can see from both figures, the degree of window synchronization of PPD is looser than that of RCD, and that of EPD is looser than that of PPD. But window synchronization is still apparent.

[1]See Fig. 5. In this figure, the buffer threshold is set to 1,960 cells considering the packet size of 500 bytes at application layer.

In Figures 8 to 11, it at first appears that there is no slow start phase where the window size is supposed to increase at an exponential rate. Whenever the network congestion occurs, the fast retransmission algorithm is activated a few times, and finally timeout occurs when fast retransmission cannot fully recover the lost packets. A few fast retransmissions followed by a timeout make the slow start period very short, of the order of a few round-trip times. That is why in those figures there appears to be no slow start phase. During the congestion avoidance phase, we increased TCP window size as[8].

$$cwnd \leftarrow cwnd + \frac{segsize \times segsize}{cwnd} + \frac{segsize}{8}$$

In the above equation, *'cwnd'* denotes the current window size and *'segsize'* is the segment size at TCP layer.

4.2 Improvements of ESPD over RCD, PPD and EPD

For the performance evaluation of each packet discarding scheme, our main concern was to improve the end-to-end goodput given the same network resources. A low cell loss ratio does not guarantee low packet loss ratio[2]. Instead, concentrating cell losses on the same packet is more important, as in PPD and EPD. The same reasoning can be extended further. Low packet loss ratio and low packet retransmission ratio, by themselves, do not guarantee higher end-to-end goodput. Instead, concentrating packet losses on the same session during a congestion period is more important due to the nature of TCP window evolution.

4.2.1 Triggering TCP Timeout Selectively

Before we go further, we have to keep in mind that the only way TCP is aware of the network congestion is a packet drop which causes retransmission timeout or fast retransmission. Our new scheme, ESPD, discards packets selectively in order to target highly active sessions for packet loss. Packet loss feedback to selected source nodes leads to a reduction in TCP window size, following TCP timeout.

4.2.2 Breaking TCP Synchronization

The random packet discard of EPD produces the sustained congestion period problem and the synchronized window expansion-shrinkage problem. These two problems reflect the fact that all the sessions share low activity periods and high activity periods together. *'Low activity'* means the aggregated traffic coming from all the sessions are lower than output link capacity. If we can somehow eliminate these underload periods, we can use more of the capacity of the output link resulting in better throughput.

Because ESPD is trying to concentrate the packet discard on a few sessions selected on first-come-first-catch basis and makes these sessions shrink their window size, the remaining intact sessions can contribute to make the output link busy resulting in better throughput. Some time later, these sessions increase window sizes, and may become the major contributors to the next congestion epoch. This means those sessions are more likely to be caught in the drop-list by the first-come-first-catch policy. Therefore, sessions tend to use the shared link capacity alternately under the ESPD policy.

Fig. 11 shows the size of each session window as a function of time. The figure shows that ESPD makes the window expansion and shrinkage of sessions asynchronous.

4.2.3 Increased Effective Throughput

Tables 1 and 3 the summarize the results of the symmetric roundtrip time simulations and Tables 2 and 4 correspond to the asymmetric roundtrip time simulations.

The effective throughput is defined as the total aggregated number of packets which arrive at the 10 destinations in the Fig 1. We counted duplicate packets that might be a result of packet retransmission as one packet. We ran 10 simulations for symmetric roundtrip times and 6 simulations for asymmetric roundtrip times for each packet discarding scheme. The simulation time was 3 seconds for symmetric roundtrip time simulations and 10 seconds for asymmetric roundtrip time simulations, but we collected result after 0.5 *sec*, in order to disregard the initial transient period. The effective throughputs in Tables 1 and 3 are the averages over 10 simulations and the effective throughputs in Tables 2 and 4 are the averages over 6 simulations. The maximum effective throughput indicated in those tables is the maximum number of packets that can be carried by an OC-3 link.

From Table 1, we can see that PPD improves the effective throughput by 28.3 % over random cell discard(RCD) and EPD gets another effective throughput enhancement of 21.4 % compared to PPD. The new scheme, ESPD, leads to a further improvement of 11.4% over EPD.

Table 3 shows that PPD improved the throughput by 27.0 % over RCD and EPD enhanced it by 27.2

% beyond PPD. ESPD added 5.0% improvement over EPD. Thus, ESPD provides only a modest improvement in this case. To show that our improvement is statistically significant, we calculated an estimate of the standard deviation for each case and listed those values in the Tables.

From Tables 2 and 4, we can notice that ESPD delivers more significant improvement over EPD. For the long roundtrip time sessions, ESPD improves the effective throughput by 87.7% and 54.1% over EPD respectively, while it produces 25.6% and 18.4% improvement for the short roundtrip time sessions, respectively. This fact tells us that ESPD provides better overall fairness, since it favors sessions with lower throughputs. This better fairness results from the fact that ESPD tends to drop the packets from high activity sessions with short roundtrip times so that the low activity sessions with long roundtrip times survive the congestion periods with higher probability. The more significant improvement of ESPD in Tables 2 and 4 is mainly due to the asymmetric TCP environment. In the asymmetric TCP environment, the activity difference between connections is more apparent so that it is easier for ESPD to identify the high activity sessions.

If we see the fairness index in Tables 1 to 4, we note that ESPD does not sacrifice fairness for increased throughput. On the contrary, it offers improved fairness over EPD. The fairness index is calculated as follows [5].

$$Fairness = \frac{(\sum_{i=1}^{n} x_i)^2}{n \sum_{i=1}^{n} x_i^2}$$

In the above equation, x_i is the effective throughput of *i-th* session and n is 10 in our simulation.

We can see that the relative effective throughput compared to maximum value in Tables 2 and 4 is worse than that in Tables 1 and 3 except for ESPD. This throughput degradation of RCD, PPD and EPD is due to the increased RTO minimum value. But because ESPD makes sessions take turns to access the network resources, it can maintain better throughput performance over different RTO minimum values.

5 Conclusions and Future Work

In this paper, we presented the performance of various packet discarding schemes for TCP traffic over ATM. Upon closer investigation of EPD, we found that even after applying the packet discarding schemes, there is a sustained period of congestion and dropped packets are spread over almost all sessions. This results in the TCP congestion control mechanism causing many sessions to be idle together.

Instead of sharing the idle period during the same time frame, taking turns should produce better thoughput. Based on this motivation, we devised the *Early Selective Packet Discard(ESPD),* a strategy in which we concentrate packet discards on the selected sessions. We verified by tracing the TCP window that the ESPD makes sessions take turns to access network resources. In other words, it provides a kind of media access control of TCP/IP traffic over ATM-UBR. This property of ESPD provides effective throughput and fairness enhancement over EPD.

We tried to catch VPI/VCI based on the first come-first-catch policy, since this method is very simple to implement and produces more throughput. But with first-come-first-catch policy we cannot guarantee that we can catch the high activity sessions only. Currently, we are working on other possible policies based on monitoring the queue length of each session.

APPENDIX: an additional simulation

In order to get further throughput comparison between EPD and ESPD, we simulated a larger network. This network has the same "parking lot" configuration as Fig. 1, but it has one more ATM switch to which 5 source nodes are connected. We also added 5 more destination nodes to the final stage ATM switch to make 15 TCP sessions. Every simulation parameter is, unless otherwise mentioned, exactly the same with the case of symmetric roundtrip simulation described before. All the link delays are set to 1 msec and the buffer threshold of EPD is set to 95 % of buffer which produced the best results of EPD. The buffer threshold 1 and 2 of ESPD is set to 70 % and 50 % respectively. The packet length is 500 bytes at the application layer.

The summary of simulation results is shown in Table 5. With this network configuration and parameters, ESPD improved overall throughput over EPD by 12.4% and improved the fairness index from 0.837 to 0.867. We also can verify that ESPD improves the throughput of disadvantaged session groups more than that of the advantaged session group.

	RCD	PPD	EPD	ESPD
Effective Throughput (Maximum Value) (Improvement)	33,999 (76,415)	43,614 (76,415) (28.3 %)	52,956 (76,415) (21.4 %)	58,999 (76,415) (11.4 %)
Standard Deviation of Effective Throughput	816	2,416	1,382	980
Fairness Index (Maximum Value)	0.973 (1.000)	0.877 (1.000)	0.921 (1.000)	0.893 (1.000)

Table 1: Summary of simulation results for symmetric roundtrip time case. Packet Size = 500 bytes at application layer

	RCD	PPD	EPD	ESPD
Effective Throughput of Long Roundtrip Time Sessions : Session 1 to 5 (Improvement %)	6,018	6,648 (10.4 %)	9,699 (45.8 %)	18,214 (87.7 %)
Effective Throughput of Short Roundtrip Time Sessions : Session 6 to 10 (Improvement %)	106,579	132,048 (23.8 %)	163,816 (24.0 %)	205,892 (25.6 %)
Sum of effective throughputs (Maximum Value) (Improvement %)	112,597 (290,377)	138,696 (290,377) (23.1 %)	173,515 (290,377) (25.1 %)	224,106 (290,377) (29.1 %)
Standard Deviation of Effective Throughput	7,623	14,228	17,186	3,713
Fairness Index (Maximum Value)	0.528 (1.000)	0.508 (1.000)	0.540 (1.000)	0.573 (1.000)

Table 2: Summary of simulation results for asymmetric roundtrip time case. Packet Size = 500 bytes at application layer

Figure 2: Queue length and sessions damaged during congestion for RCD. (Packet Length = 500 bytes. Symmetric roundtrip times)

	RCD	PPD	EPD	ESPD
Effective Throughput (Maximum Value) (Improvement)	14,968 (27,787)	19,008 (27,787) (27.0 %)	24,178 (27,787) (27.2 %)	25,394 (27,787) (5.0 %)
Standard Deviation of Effective Throughput	956	886	491	277
Fairness Index (Maximum Value)	0.953 (1.000)	0.946 (1.000)	0.928 (1.000)	0.929 (1.000)

Table 3: Summary of simulation results for symmetric roundtrip time case. Packet Size = 1,500 bytes at application layer

	RCD	PPD	EPD	ESPD
Effective Throughput of Long Roundtrip Time Sessions : Session 1 to 5 (Improvement %)	3,471	4,518 (30.1 %)	6,435 (42.4 %)	9,917 (54.1 %)
Effective Throughput of Short Roundtrip Time Sessions : Session 6 to 10 (Improvement %)	47,784	60,027 (25.6 %)	72,817 (21.3 %)	86,247 (18.4 %)
Sum of effective throughputs (Maximum Value) (Improvement %)	51,255 (105,590)	64,545 (105,590) (25.9 %)	79,252 (105,590) (22.7 %)	96,164 (105,590) (21.3 %)
Standard Deviation of Effective Throughput	4,062	2,279	3,046	1,401
Fairness Index (Maximum Value)	0.561 (1.000)	0.565 (1.000)	0.581 (1.000)	0.603 (1.000)

Table 4: Summary of simulation results for asymmetric roundtrip time case. Packet Size = 1,500 bytes at application layer

Figure 3: Queue length and sessions damaged during congestion for PPD. (Packet Length = 500 bytes. Symmetric roundtrip times)

Figure 4: Queue length and sessions damaged during congestion for EPD. (Packet Length = 500 bytes. Symmetric roundtrip times)

Figure 6: Queue length and sessions damaged during congestion for ESPD. (Packet Length = 500 bytes. Symmetric roundtrip times)

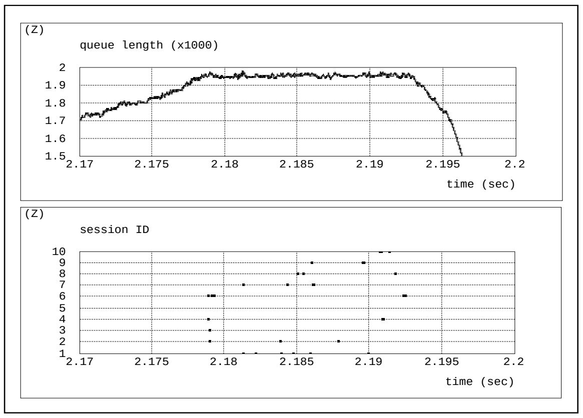

Figure 5: Magnified portion of the first congestion. in Fig. 4. (Packet Length = 500 bytes. Symmetric roundtrip times)

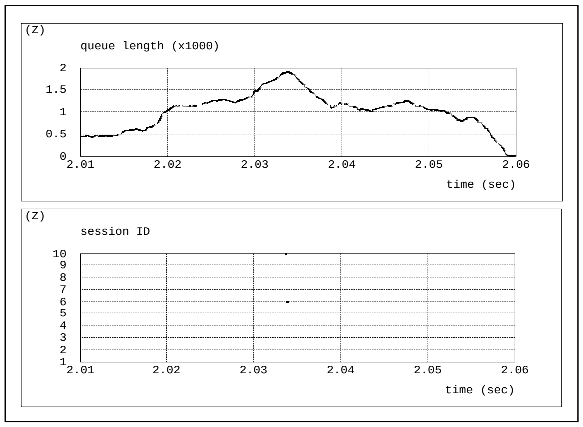

Figure 7: Magnified portion of the first congestion. in Fig. 6. (Packet Length = 500 bytes. Symmetric roundtrip times)

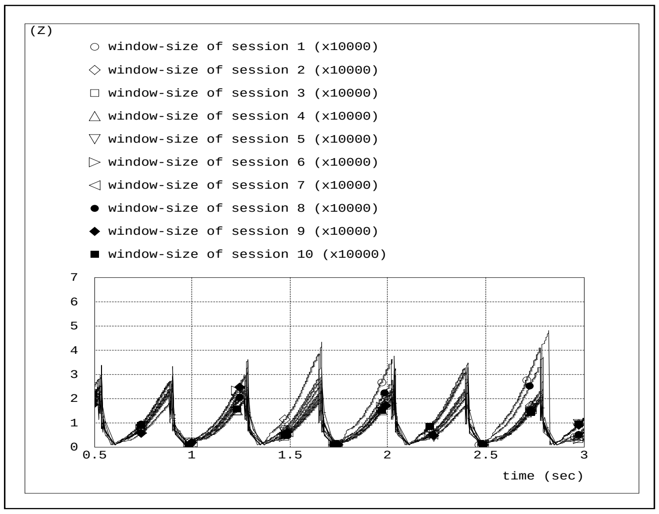

Figure 8: Session window sizes for RCD as a function of time (Packet Length = 500 bytes. Symmetric roundtrip times).

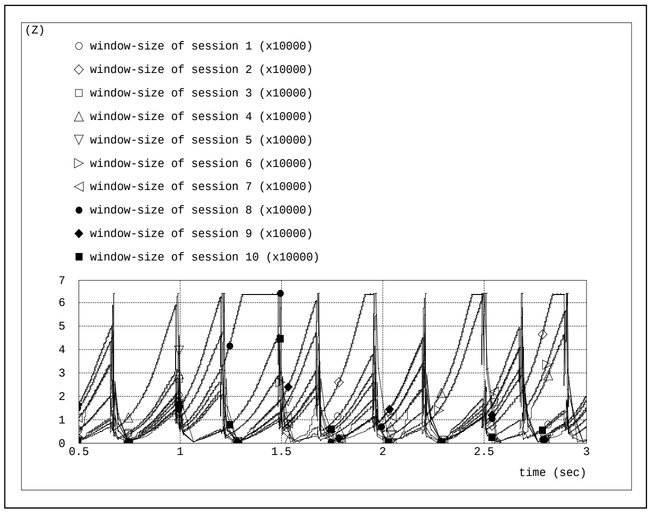

Figure 10: Session window sizes for EPD as a function of time (Packet Length = 500 bytes. Symmetric roundtrip times).

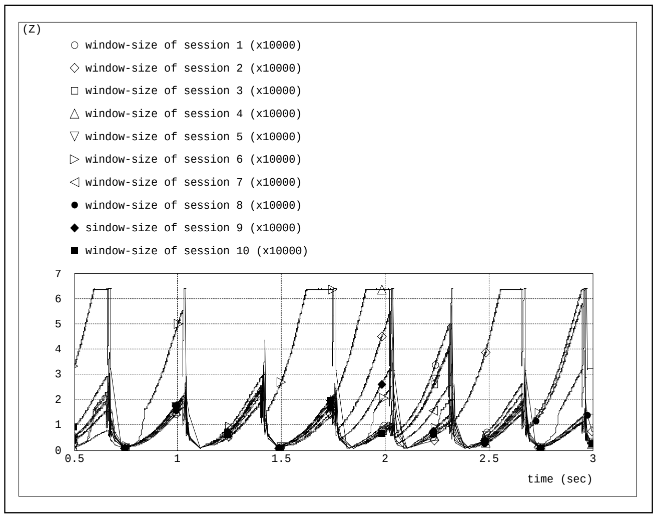

Figure 9: Session window sizes for PPD as a function of time (Packet Length = 500 bytes. Symmetric roundtrip times).

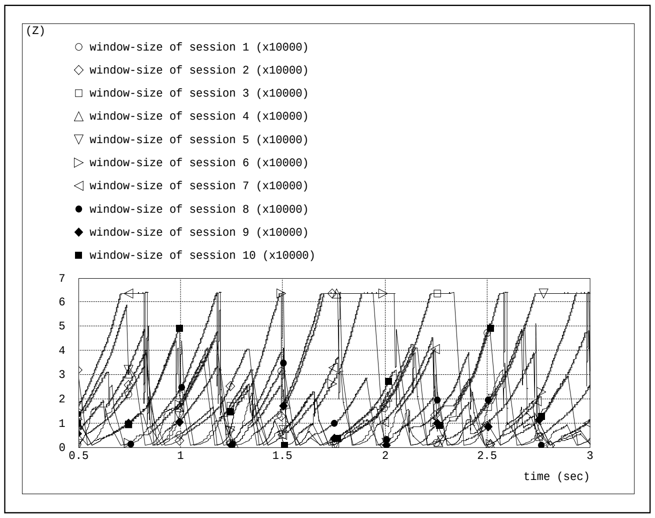

Figure 11: Session window sizes for ESPD as a function of time (Packet Length = 500 bytes. Symmetric roundtrip times).

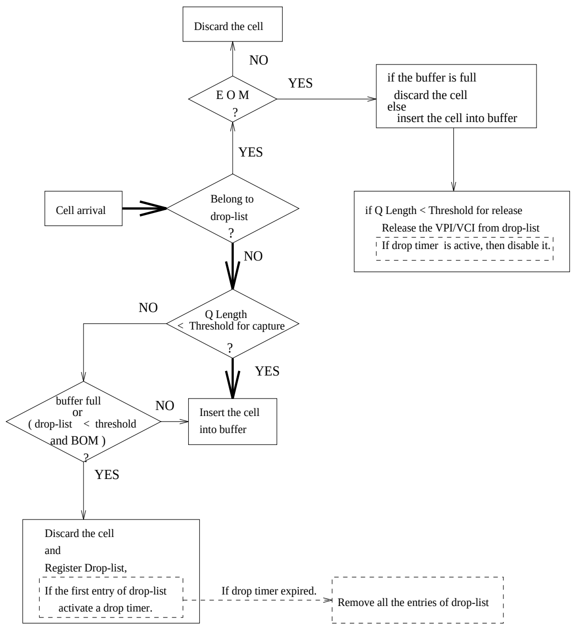
Discard the cell
NO
YES
E O M ?
if the buffer is full discard the cell else insert the cell into buffer
YES
Cell arrival
Belong to drop-list ?
if Q Length < Threshold for release Release the VPI/VCI from drop-list
If drop timer is active, then disable it.
NO
NO
Q Length < Threshold for capture ?
YES
buffer full or (drop-list < threshold and BOM) ?
NO
Insert the cell into buffer
YES
Discard the cell and Register Drop-list,
If the first entry of drop-list activate a drop timer.
If drop timer expired.
Remove all the entries of drop-list

Figure 12: Flow Chart Description of ESPD.

	EPD	ESPD
Effective Throughput of Long Roundtrip Time Sessions : Session 1 to 5 (Improvement %)	12,429	17,930 (44.3%)
Effective Throughput of Medium Roundtrip Time Sessions : Session 6 to 10 (Improvement %)	16,363	17,996 (10.0%)
Effective Throughput of Short Roundtrip Time Sessions : Session 11 to 15 (Improvement %)	30,262	30,475 (0.7%)
Sum of effective throughputs (Maximum Value) (Improvement %)	59,054 (76,415)	66,401 (76,415) (12.4%)
Standard Deviation of Effective Throughput	837	1,399
Fairness Index (Maximum Value)	0.837 (1.000)	0.867 (1.000)

Table 5: Summary of simulation results for bigger networks with asymmetric roundtrip times. Packet Size = 500 bytes at application layer

References

[1] ATM Forum, "Traffic Management Specification Version 4.0," April, 1996. available as *ftp://ftp.atmforum.com/pub/approved-specs/af-tm-0056.000.ps*

[2] Chien Fang, Helen Chen and Jim Hutchins, "A simulation study of TCP performance in ATM networks," *Proc. IEEE GLOBECOM '94,* pp. 1217-1223, Nov. 1994.

[3] Chien Fang and Arthur Lin, "On TCP performance of UBR with EPD and UBR-EPD with a fair buffer allocation scheme," AF-TM 95-1645, Dec. 1995.

[4] H. Li, K.-Y. Siu, H.-Y. Tzeng, C. Ikeda and H. Suzuki, "Performance of TCP over UBR service in ATM networks with per-VC early packet discarding schemes," *Proc. the International Phoenix Conference on Computers and Communications,* pp. 350-357, Mar. 1996.

[5] R. Goyal, G. Jain, S. Fahmy, S. Fahmy and S Kim, "Performance of TCP over UBR+," AF-TM 96-1269, Oct. 1996.

[6] Allyn Romanow and Sally Floyd, "Dynamics of TCP Traffic over ATM Networks," *IEEE Journal of Selected Areas in Communications,* vol. 13, no. 4, May 1995.

[7] A. Romanov and R. Oskouy, "A Performance Enhancement for Packetized ABR and VBR+Data," AF-TM 940295, Mar. 1994.

[8] W. R. Stevens, *TCP/IP Illustrated,* volume 1. Addison-Wesley, 1994.

[9] G. Armitage and K. Adams, "Packet Reassembly during Cell Loss," *IEEE Networks,* vol. 7, no. 5, pp. 26-34, Sept. 1993.

[10] H. Li, K. Siu, H. Tzeng, C. Ikeda and H. Suzuki, "A simulation Study of TCP Performance in ATM Networks with ABR and UBR Services," *Proc. IEEE INFOCOM'96,* pp. 1269-1276, Mar. 1996.

[11] Sally Floyd and Van Jacobson, "Random Early Detection Gateways for Congestion Avoidance," *IEEE/ACM Transactions on Networking,* vol. 1, no. 4, pp. 397-413, Aug. 1993.

[12] Jonathan S. Turner, "Maintaining High Throughput during Overload in ATM Switched," *Proc. IEEE INFOCOM'96,* pp. 287-295, Mar. 1996.