Buffer Management and Scheduling Schemes for TCP/IP over ATM-GFR*

Dapeng Wu† H. Jonathan Chao‡

Abstract

Today ATM technology is facing challenges from Integrated Service IP, IP switching, Gigabit IP router and Gigabit Ethernet. Although ATM is approved by ITU-T as the standard technology in B-ISDN, its survivability is still in question. Since ATM-UBR (Unspecified Bit Rate) provides no service guarantee and ATM-ABR (Available Bit Rate) is still unattainable for most users, many existing users have little or no incentives to migrate to ATM technology. The Guaranteed Frame Rate (GFR) service is introduced to deal with this dilemma. The GFR can guarantee the Minimum Cell Rate (MCR) with fair access to excess bandwidth. This paper studies various schemes to support the GFR. We have studied different discarding and scheduling schemes, and compared their throughput and fairness when TCP/IP traffic is carried. Through simulations, it is shown that only per-VC queueing with Weighted Round Robin (WRR) can guarantee Minimum Cell Rate. Among all the schemes that have been explored, we recommend DT-EPD (Dynamic Threshold - Early Packet Discard) integrated with MCR+ (a WRR variant) to support the GFR service.

Key Words: TCP/IP, ATM-GFR, MCR, Virtual Queueing, WRR, EPD, Pushout

1 Introduction

Broadband Integrated Service Digital Network (B-ISDN) is based on ATM technology, which has been approved by ITU-T (International Telecommunication Union - Technical Section). One advantage of ATM technology is that significant performance and efficiency benefits can be achieved if and when applications/users are able to exploit the full range of ATM

*The work described in this paper is supported by Sumitomo Electric Industries, NYNEX, and NY State Science and Technology Foundation.

†Carnegie Mellon University, Dept. of Electrical & Computer Engineering, 5000 Forbes Avenue, Pittsburgh, PA 15213, USA. Email: dpwu@cs.cmu.edu.

‡Please direct all correspondence to H. Jonathan Chao, Polytechnic University, Dept. of Electrical Engineering, Six Metrotech Center, Brooklyn, NY 11201, USA. Tel. (718) 260-3302, Fax (718) 260-3074, Email: chao@poly.edu.

traffic parameters and service classes. However, this advantage is still unattainable for many users today. These users are either not able to specify the range of traffic parameters, i.e., PCR (Peak Cell Rate), SCR (Sustainable Cell Rate), MBS (Maximum Burst Size), which are needed to request most ATM services, or are not equipped with devices capable of effectively interacting with an ATM network, i.e., enforcing ABR source mechanism. The only access these users have to ATM networks is through UBR connections, which provide no service guarantees. Schemes such as EPD (Early Packet Discard) or PPD (Partial Packet Discard) [7], though improving the goodput of packet traffic over UBR, cannot guarantee a minimum packet rate. As a result, many existing users have little or no incentive to migrate to ATM technology.

To deal with this situation, Guerin and Heinanen proposed a new service which was originally called "UBR+" [2] but is now called GFR [9]. And ATM Forum is currently discussing the need for GFR service. The objective of the GFR service is to bring as yet unavailable benefits of ATM performance and service guarantees to users. Thus, the GFR service requires minimal interactions between users and ATM networks, but provides users with a certain level of service guarantees. The essence of GFR is to guarantee Minimum Cell Rate (MCR) with fair access to excess bandwidth.

The GFR service is intended to support non-real-time applications. The GFR service requires that the user data cells are organized in the form of frames that can be delineated at the ATM layer. The GFR service provides the user with a minimum service rate guarantee under the assumption of a given maximum frame size. Informally, this says that if the user sends frames (not cells) of size less than the specified maximum size and at a rate less than the specified value, then the user should expect to see all of its frames delivered across the network with minimum losses. The mapping of the packet level guarantees to cell level guarantee is done by the network.

In addition, the service also allows the user to send in excess of its guaranteed service rate, but only guarantees that such frames will be delivered within the limits of available resources, e. g., as best effort. Furthermore, the service also specifies that the excess traffic of each user should have access to a fair share of available resources. The definition of fair share is implementation-specific.

The GFR service does not give the users explicit feedback regarding the current level of network congestion. Currently, the GFR service only applies to VCCs (Virtual Channel Connection), because frame delineation is not visible at the VP (Virtual Path) layer. The GFR service allows a user to expect a minimum level of throughput when the network is congested, while being able to send at a higher rate when additional resources are available. The service requires little or no involvement by the user, as most, if not all of the service

parameters can be negotiated off-line.[1]

To efficiently support the GFR service, some requirements need to be imposed on the network. This paper will answer the following questions:

What mechanisms in ATM switches are required to support the GFR service?

Are these mechanisms simple enough to be implemented?

In this paper we investigate TCP performance over several GFR implementations and present simulation results. The focus of our research is on whether the implementations discussed at least guarantee the MCR and reasonable fairness.

The remainder of the paper is organized as follows. Section 2 discusses an EPD (Early Packet Discard) variant using dynamic threshold [1]. Section 3 describes three pushout schemes and a quasi-pushout scheme [6] to implement the GFR service. Through simulations, we compare the performance of the schemes described above. Furthermore, our observation leads to the introduction of an improved scheme, called MCR+. Finally, we conclude our investigation on various discarding and scheduling schemes and give our recommendation on the implementation of the GFR service.

2 Dynamic Threshold EPD

EPD is proposed by Romonov and Floyd [7]. To implement ATM-GFR, Siu proposed an EPD variant using Virtual Queueing (VQ) technique [8]. Here we apply Choudhury and Hahne's scheme in EPD scenario by replacing the static threshold with Dynamic Threshold (DT) [1].

Dynamic Threshold is proposed by Choudhury and Hahne [1] and is used in shared memory ATM switches.

One conventional scheme, the Static Threshold scheme, is used to fairly regulate the sharing of memory among different output port queues in a shared memory ATM switch. In this scheme, an arriving cell is admitted only if the queue length at its destination output port is smaller than a given threshold. The Static Threshold strategy is very simple to implement but it is not adaptive. When only one output queue of the memory is very active, that queue is needlessly denied access to much of the buffer space. This effect can lead to under-utilization of the switch. At other times, when many queues are active, the buffer can fill up completely even though all queues are obeying their threshold constraints. Some queues become starved for space, and this effect can also lead to under-utilization.

The Dynamic Threshold scheme combines the simplicity of Static Threshold and the

[1]By off-line we mean that these parameters can be established via network management, configuration means, or inferred from existing parameters in the SETUP message.

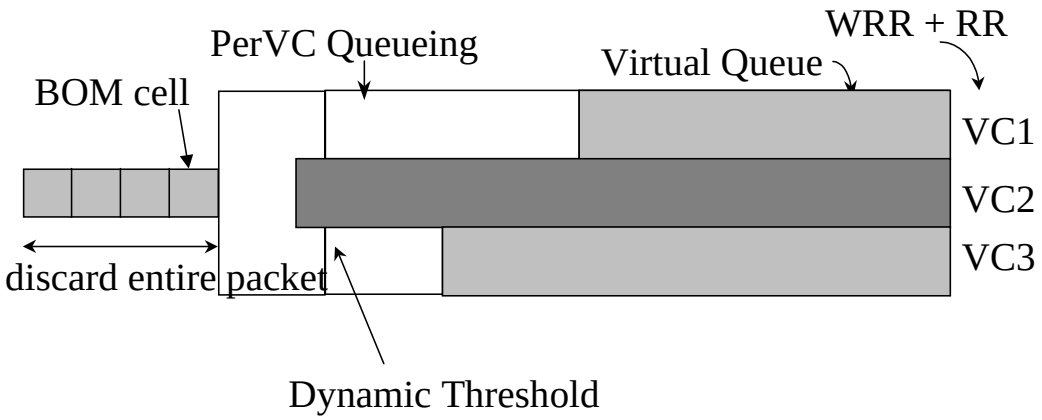

Figure 1: Dynamic Threshold EPD.

adaptivity of Pushout. The key idea is that the maximum permissible length for any individual queue at any instant of time is proportional to the unused buffering in the switch. A queue whose length equals or exceeds the current threshold value may accept no more new cells. The Dynamic Threshold scheme can improve fairness and switch efficiency by guaranteeing access to the buffer space for all output queues. The Dynamic Threshold scheme deliberately wastes a small amount of buffer space, but distributes the remaining buffer space equally among the active output queues.

We now describe the Dynamic Threshold scheme in more detail. When there is only one switch output port that is very active, the Dynamic Threshold scheme allows this output port to have access to as much of the shared buffer memory as possible. When there are many contending queues, however, the Dynamic Threshold scheme divides the memory fairly among them. All queues with sufficient traffic to warrant threshold should obtain the same amount of space, called the *control threshold*. The control threshold value is determined by monitoring the total amount of unused buffer space.

Each output queue attempts to limit its length to some function f of the unused buffer space; output queues with less demand than this can have all the space they wish. At time t, let T(t) be the control threshold and let $Q^i(t)$ be the length of queue . Let Q(t) be the sum of all the queue lengths, e.g., the total occupancy of the shared memory. Then, if B is the total buffer space,

$$T(t) = f(B - Q(t)) = f(B - \sum_i Q^i(t)) \tag{1}$$

An arriving cell for queue will be blocked at time t if $Q^i(t) \geq T(t)$. All cells going to this queue will be blocked until the queue length drops below the control threshold.

The simplest scheme is to set the control threshold to a multiple of the unused buffer space.

$$T(t) = \alpha \cdot (B - Q(t)) = \alpha \cdot (B - \sum_i Q^i(t)) \tag{2}$$

If α is a power of 2 (either positive or negative), then the threshold computation is extremely easy to implement: only a shifter is required.

Now we apply the above scheme in EPD scenario with the static threshold replaced by dynamic threshold. Per-VC queueing is utilized in Dynamic Threshold EPD (DT-EPD) (see Figure 3). In this case Weighted Round Robin and Round Robin is real bandwidth allocation compared with the Virtual Queueing scheme. The dynamic threshold (DT) is determined as follows.

$$DT = \alpha \cdot (BufferSize - TotalQueueLength) \tag{3}$$

$$(Note : BufferSize - TotalQueueLength = FreeSpace)$$

If $\alpha = 2$, for example, DT-EPD tries to regulate each queue to be twice the free buffer space. So a single queue with no competition is allowed to take 2/3 of the entire shared memory, and 1/3 of the memory is held back. When two long queues are active, each queue gets 2B/5, and B/5 is unallocated. When there are three long queues, each queue gets 2B/7, with B/7 unallocated. If the number of very active VCs (which are greedy at this time) then increases from 3 to 10, the long queues (which are longer than DT) will drain and the newly active queues will grow until all ten stabilize at 2B/21, with B/21 left unallocated.

DT-EPD deliberately wastes a small amount of buffer space. This "waste" actually serves two useful functions. The first advantage of maintaining some spare space at all times is that it provides a cushion during transient periods when a VC queue first becomes active. This reduces cell loss during such transient periods. Secondly, when a VC queue has such a load increase and begins taking over some of the spare buffer space, this action signals the allocation mechanism that the load conditions have changed and that a threshold adjustment is now required.

3 Pushout Discarding Schemes

3.1 Pushout Schemes

Pushout discarding schemes also treat TCP data stream in packet units. In Pushout schemes, arriving cells are allowed to enter the buffer as long as there is space, and when the buffer

fills up, an incoming cell is allowed to enter by discarding cells in the longest queue. While the incoming cell usurps the physical space of the discarded cell, the incoming cell does not take over the discarded cell's position in its logical queue. Indeed, the pushing and pushed cells may belong to different logical queues. Rather, the arriving cell joins its own logical queue. The Pushout approach has many performance benefits. Pushout is fair: it allows smaller queues to increase in length at the expense of longer queues. Pushout is efficient: no output queue is ever starved for space, and no space is ever held idle while some queue desires more; thus overall system throughput should be high. Pushout is naturally adaptive: when lots of queues are active, their rivalry keeps their queue lengths short; when only one queue is active, it is allowed to become long. The drawback of Pushout is the difficulty of implementing it for high-speed switches. When the shared memory is full, writing a cell into a queue involves the extra step of pushing out cells in the longest queue first. In addition,

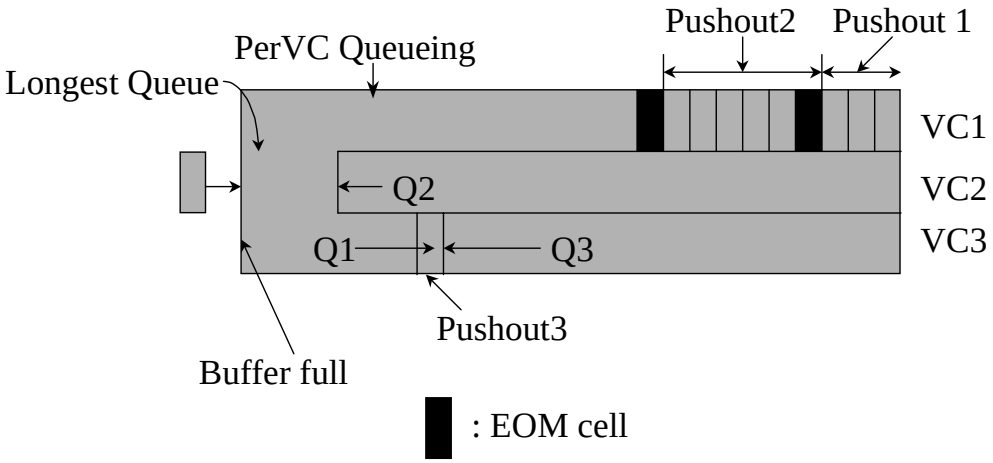

Figure 2: Pushout schemes.

There are three Pushout schemes. Pushout-1 drops the HOL (Head Of Line) partial packet from the longest queue, which is delimited by the HOL cell and the first EOM (End Of Message) cell. Pushout-2 searches for an entire packet from the head of the longest queue and drops it. Pushout-3 drops the partial packet from the tail of the longest queue, which is delimited by the tail cell and the last EOM cell. Based on per-VC queueing, Pushout schemes utilize Weighted Round Robin and Round Robin as the scheduling policies. The three Pushout schemes are shown in Figure 4.

Through simulation [5], Lakshman, et. al., showed that, for the same buffer size, drop from front results in considerably higher TCP throughput than tail drop, and for all but very small buffers even higher than tail drop combined with partial frame drop. Thus, we expect

PO1 should be better than PO3, and PO2 should be better than PO1. Our simulation results substantiate our expectation.

3.2 Quasi Pushout Scheme

The Pushout (PO) discarding scheme has been shown to offer optimum cell loss performance [10]. However, the Pushout scheme is very difficult to implement because it requires O(N) queue length comparisons to find out the longest queue, where N is the number of output queues. When N is large, these comparisons may become the speed bottleneck.

Y.S. Lin, et. al. [6] proposed the Quasi Pushout (QPO) cell discarding scheme, which features a much reduced hardware complexity than PO. A register Longest Queue (LQ) for the quasi-longest queue is maintained and updated during cell arrival or departure events. At th
make

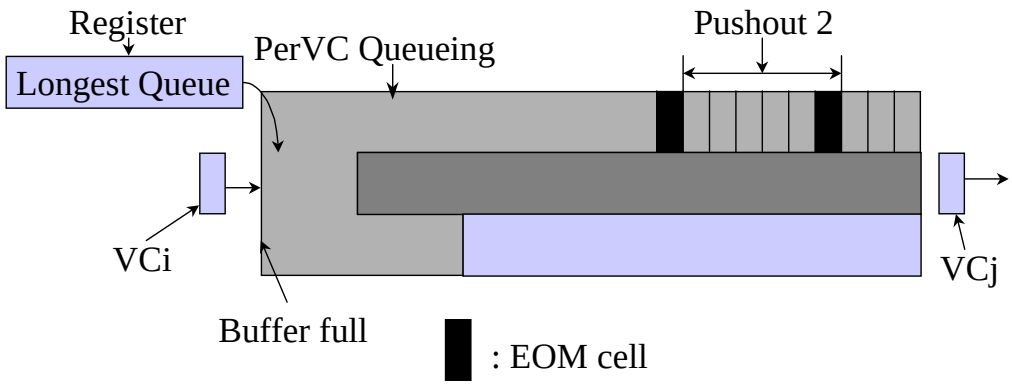

Figure 3: Quasi Pushout scheme based on Pushout2.

The following pseudocode describes the QPO discarding scheme based on PO2:

When a cell in VCi reaches a switch:

if (buffer full)

search for an entire packet from the head of quasi longest queue and drop it

QL[LQ] = QL[LQ] – x; / x is the number of cells discarded */*

accept the incoming cell into the VCi queue

QL[i] = QL[i]+1; / buffering input cell */*

if (QL[LQ] < QL[i]

LQ = i; / input-comparison */*

When a cell in VCj queue is transmitted

QL[j] = QL[j] – 1; / delivering output cell */*

if (QL[LQ] < QL[j]

LQ = j; / output-comparison */*

If the buffer is full, an entire packet from the head of the quasi-longest queue LQ will be discarded to make space for the input cell. In contrast to PO, which needs N comparisons to determine the real longest queue for every discarded cell, QPO tracks the quasi-longest queue by using two comparisons only. One is on the arrival of an input cell: the queue length of the VCi is increased and compared with that of queue LQ. The other is on the departure of an output cell: the queue length of the VCj is decreased and compared with that of queue LQ. If the new length of VCi or VCj is longer than that of queue LQ, the register LQ is redirected to the new quasi-longest queue.

In the Quasi Pushout scheme (see Figure 5), the discarded queue may not be the longest, while Pushout schemes always find the longest queue from all VCs. But the mistrack of the longest queue in QPO scheme will be corrected when the real longest queue (VCk) is served, either through accepting a cell from VCk or transmitting a cell from VCk. Therefore the proposed algorithm may produce sub-optimum results due to mistracking the longest queue occasionally. However, our simulation results show that the performance of QPO is very close to that of the optimum PO scheme. In all our simulations, Quasi Pushout is based on the PO2 scheme.

4 Simulation Experiment

4.1 Simulation Model

Our simulation tool is based on the NIST ATM Network Simulator. This simulator is an event-driven simulator composed of various components that send messages to one another. We have changed some ATM and TCP related components in this simulator to meet our own needs.

In our simulations, we have a fine TCP timer granularity, 10 usec, instead of 300 to 500 msec as in most TCP releases. This corresponds to a high-speed, low-propagation-delay ATM LAN environment. Since the TCP retransmission timer is set as a function of round trip time, a coarse timer granularity will lead to poor TCP performance in ATM LANs [7]. For a discussion on the TCP round trip time (RTT) estimation algorithm, please refer to [3].

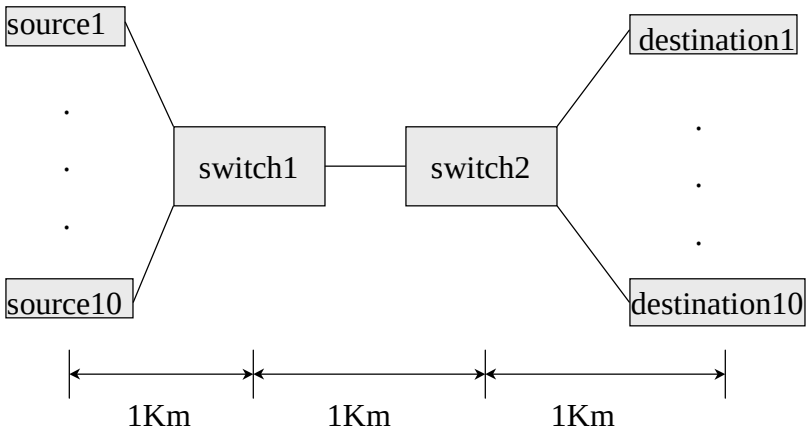

Figure 4: Simulation configuration.

The source and destination ATM host components perform AAL5 for data services including segmentation and reassembly (SAR) of TCP/IP packets. The ATM switch component models a UBR switch with different discarding and scheduling policies. The ATM switch is an output-buffered switch.

Figure 6 illustrates the simulation model of a network with ten peer-to-peer connections. On the sending side, the sources (source 1 to source 10) generate data for TCP/IP components, which form TCP/IP packets, and which in turn are passed on for AAL5 processing. The two ATM switches perform cell switching between their input and output ports. On the receiving side, cells are reassembled and passed to the TCP/IP components. By running ten concurrent connections, we create a congested link between the two ATM switches.

The following parameters are employed in our simulations:

TCP:

Mean Packet Processing Delay = 300 usec
Packet Procesing Delay Variation = 10 usec
Packet Size = 2K Bytes
Maximum Receiver Window Size = 64K Bytes
Default Timeout = 500 ms
Timer Granularity = 10usec
TCP Reno version
Unidirectional traffic
Greedy sources

Link:

Speed = 155.52 Mbps

Delay = 5 us (1Km) in LAN or 5ms (1000Km) in WAN

UBR-End System:

Packet Processing Delay = 500 usec

Buffer Size = infinity

Cell Transmission Rate = 155.52Mbps

UBR-Switch:

Non-blocking Output-buffered Switch

Packet Processing Delay = 4 usec

Buffer Size (Qmax) = 3000 cells (for LAN) and 36000 cells (for WAN)

EPD Threshold (Th) = 1000 cells

T= 1 msec

Simulation Time = *2 seconds (for LAN) and 5 seconds (for WAN)*

The TCP packet processing delay is the time that it takes the TCP source to handle the transmission of a data packet or the receipt of an acknowledgment packet. It also represents the time that it takes the TCP destination to handle the receipt of a data packet or the transmission of an acknowledgment packet. Since TCP processing time may vary from packet to packet, we simulate it by using the mean packet processing delay plus or minus a random time, which is in the range of packet processing delay variation.

Note that the TCP default timeout will be used only when the first packet of a connection is lost, since no round trip time can be applied to calculate the first retransmission timeout. However, in all our simulations, no first packet was lost due to sufficiently large buffer, and the default timeout was never triggered.

4.2 Performance Metrics

The performance of TCP over GFR is measured by efficiency, fairness [4] and difference, which are defined as follows:

$$Efficiency = (Sum\ of\ TCP\ throughput)/(Maximum\ possible\ TCP\ throughput) \quad (4)$$

$$FairnessIndex = \frac{(\sum(Throughput_i - MCR_i))^2}{n \times \sum(Throughput_i - MCR_i)^2} \quad (5)$$

$$Difference = (MaxTh - MinTh)/AverageTh \quad (6)$$

Note: *MaxTh = Maximum Throughput of the TCP connection among all*
MinTh = Minimum Throughput of the TCP connection among all
AverageTh = (Sum of TCP throughput)/ (Number of TCP connections)

n = the number of TCP connections

The TCP throughputs are measured at the destination TCP layers. Throughput is defined as the total number of bytes delivered to the destination application divided by the total simulation time. The results are reported in Mbps. The maximum possible TCP throughput is the throughput attainable by the TCP layer running over UBR on a 155.52 Mbps link. For 2048 bytes of data (TCP maximum segment size), the ATM layer receives 2048 bytes of data + 20 bytes of TCP header + 20 bytes of IP header + 8 bytes of LLC header + 8 bytes of AAL5 trailer. These are padded to produce 44 ATM cells. Thus, each TCP segment results in 2332 bytes at the ATM layer. From this, the maximum possible throughput = 2048/2332 = 87.8% = 136.6 Mbps approximately on a 155.52 Mbps link.

4.3 Simulation Results

4.3.1 Performance Comparison with Equal MCRs

In Table 2, the reserved MCR of each VC is shown in the first row. The rates in the table are TCP layer rates, which exclude TCP, IP, LLC, and AAL5 overheads. Since MCR is equal to each VC, each VC should therefore get the same throughput. As seen from Table 2, the fairness index cannot show the difference among the schemes. Thus we introduce the metric of difference in order to see how well each scheme performs.

MCR (Mbps)	1	1	1	1	1	1	1	1	1	1	Total	Fairness	Eff	diff%
DropTail	13.26	11.97	11.42	11.7	10.27	11.83	11.15	11.27	11.64	13.04	117.54	0.9941	0.8606	13
VQ	13.1	13.62	12.62	13.33	14.2	11.38	12.58	13.37	13.07	13.77	131.05	0.9962	0.9595	11
DT-EPD	13.48	13.23	13.15	13.27	13.29	13.01	12.98	12.99	13.28	13.15	131.82	0.9998	0.9651	1.9
PO1	13.75	13.51	13.65	12.66	13.18	13.17	13.06	13.26	13.21	13	132.45	0.9994	0.9697	4.1
PO2	13.69	13.5	13.57	13.33	13.21	13.55	13.35	13.35	13.39	13.34	134.28	0.9999	0.9832	1.8
PO3	12.19	12.34	11.78	11.45	12.83	11.63	11.36	11.89	12.75	13.1	121.32	0.9973	0.8883	7.2
QPO	14.07	13.47	13.36	13.44	13.35	13.56	12.98	13.53	13.41	13.22	134.41	0.9996	0.9841	4.1

*The rates shown in the table are the average TCP rates of the entire simulation.

Table 1: Simulation results for equal MCRs.

Our simulation shows that PO2 has the best performance in terms of fairness (difference is only 1.8%). QPO has the highest throughput because QPO favors some connections. As a result, QPO does not achieve as good a fairness as PO2. Since the performances of PO1 and PO3 are not as good as PO2, we will not discuss PO1 and PO3 in the following sections. DT-EPD is comparable to PO2. Compared with the no-control case (DropTail), VQ can improve throughput and fairness but not as well as QPO.

4.3.2 Performance Comparison with Different MCRs (Case 1)

The reserved MCR of each VC is shown in the first row of Table 3. Since MCR is different for different VC, IdealRate parameters will be more straightforward to see how fair each scheme is. PO2, DT-EPD and QPO are comparable, as shown from Table 3. The performance of VQ is the worst.

$$IdealRate(i) = FairShare + MCRi \tag{7}$$

$$FairShare = ExcessBandwidth/(Number\ of\ VC) \tag{8}$$

$$ExcessBandwidth = MaxTh - (sum\ of\ MCRi) \tag{9}$$

MCR (Mbps)	1	2	3	4	5	6	7	8	9	10	Total	Fairness	Eff
IdealRate	9.16	10.16	11.16	12.16	13.16	14.16	15.16	16.16	17.16	18.16			
VQ	14.1	12.11	13.7	14.14	12.33	13.15	12.69	14.41	12.87	12.32	131.8	0.8556	0.965
DT-EPD	8.874	10.15	10.6	12.09	12.79	13.76	14.63	15.75	16.58	17.25	132.5	0.999	0.97
PO2	9.353	10.1	11.65	12.42	13.22	13.47	14.35	15.38	16.47	17.24	133.7	0.9959	0.979
QPO	9.461	10.78	11.61	11.9	12.98	13.96	14.81	15.31	16.17	17.31	134.3	0.9955	0.983

Table 2: Simulation results for different MCRs (case 1).

4.3.3 Performance Comparison with Different MCRs (Case 2)

As shown from Table 4, PO2, DT-EPD and QPO are comparable, and none of these schemes can guarantee perfect fairness. We also observe that the larger the MCR is, the smaller share in excess bandwidth the VC gets. This is non-linearity. In addition, we can see that VQ can not guarantee MCR.

4.3.4 Performance of Pushout Scheme with Buffer Share

From Table 4, we can see that the fairness index is not as good as we expect. Intuitively we try to improve the fairness through considering the effect of buffer share. We will discard cells from the VC with the longest virtual queue length rather than the actual queue length. Since buffer occupancy is proportional to bandwidth share in the FIFO case, we give the buffer share as follows.

$$BufferShare = MCR * BufferSize/LinkRate \tag{10}$$

MCR (Mbps)	2	4	8	10	10	15	15	18	18	20	Total	Fairness	Eff
ideal rate	3.66	5.66	9.66	11.66	11.66	16.66	16.66	19.66	19.66	21.66			
VQ	13.84	13.9	11.9	13.39	13.24	13.93	13.96	10.77	12.93	13.88	131.74	0.03509	0.9646
DT-EPD	4.505	6.002	9.632	11.17	11.49	15.82	15.82	18.29	18.39	20.43	131.56	0.72969	0.9632
PO2	4.689	6.423	9.732	11.63	11.67	15.91	15.98	18.71	18.6	20.38	133.72	0.77361	0.979
QPO	4.598	6.287	10.13	11.5	11.62	15.98	16.1	18.76	18.51	20.23	133.71	0.76899	0.979

Table 3: Simulation results for different MCRs (case 2).

$$VirtualQueueLength = QueueLength - BufferShare \tag{11}$$

It is shown from Table 5 that considering the effect of buffer share can improve the fairness. But it's not enough.

MCR (Mbps)	2	4	8	10	10	15	15	18	18	20	Total	Fairness	Eff
ideal rate	3.66	5.66	9.66	11.66	11.66	16.66	16.66	19.66	19.66	21.66			
share	4.42	6.128	10.01	11.64	11.65	16	16.1	18.84	18.65	20.5	133.9	0.8292	0.981
no share	4.69	6.423	9.732	11.63	11.67	15.91	15.98	18.71	18.6	20.38	133.7	0.77361	0.979

* no share: Pushout 2 scheme
* share: Pushout 2 scheme with VirtualQueueLength

Table 4: Simulation results with buffer share.

4.3.5 Performance of MCR+

From Table 4, we observe that VCs with smaller MCR can get larger share in excess bandwidth. One possible solution is to increase MCR non-linearly by favoring VCs with larger MCR. We call this scheme MCR+. In MCR+ scheme $MCR^{'}$ is used for Weighted Round Robin and given as follows.

$$MCR_i^{'} = MCR_i + ReservedBandwidth \times MCR_i/(\sum MCR_i) \tag{12}$$

Through simulation, we find that MCR+ scheme can achieve much better fairness under different MCR combinations when ReservedBandwidth is 10% of the link rate. Reserving

10% of the link bandwidth is feasible since normally ATM switches will reserve at least 10% of whole bandwidth to deal with burstiness. In this simulation, MCR+ is based on PO2 and DT-EPD respectively. Table 6 shows the simulation results in LAN environment and Table 7 shows those in WAN environment.

MCR (Mbps)	2	4	8	10	10	15	15	18	18	20	Total	Fairness	Eff
ideal rate	3.66	5.66	9.66	11.66	11.66	16.66	16.66	19.66	19.66	21.66			
VQ	13.8	13.9	11.9	13.39	13.24	13.93	13.96	10.77	12.93	13.88	131.7	0.03509	0.9646
DT-EPD	4.5	6.002	9.632	11.17	11.49	15.82	15.82	18.29	18.39	20.43	131.6	0.72969	0.9632
PO2	4.69	6.423	9.732	11.63	11.67	15.91	15.98	18.71	18.6	20.38	133.7	0.77361	0.979
QPO	4.6	6.287	10.13	11.5	11.62	15.98	16.1	18.76	18.51	20.23	133.7	0.76899	0.979
MCR+(PO2)	3.45	5.378	9.523	11.46	11.41	16.34	16.33	19.27	19.32	21.17	133.6	0.99491	0.9785
MCR+(DT-EPD)	3.32	5.494	9.102	10.9	10.82	16.31	16.03	19.01	19.05	21.08	131.1	0.96968	0.96

Table 5: Simulation results for MCR+ compared with other four schemes in LAN.

MCR (Mbps)	2	4	8	10	10	15	15	18	18	20	Total	Fairness	Eff
ideal rate	3.66	5.66	9.66	11.66	11.66	16.66	16.66	19.66	19.66	21.66			
VQ	11.7	13.59	13.03	13.38	13.45	13.13	13.09	13.58	13.67	13.74	132.4	0.04788	0.9692
DT-EPD	4.31	5.953	9.67	11.61	11.16	15.81	16	18.53	18.55	20.53	132.1	0.79988	0.9675
PO2	4.58	6.256	9.992	11.5	11.58	16.05	16.07	18.62	18.72	20.55	133.9	0.80967	0.9804
QPO	4.53	6.354	9.952	11.66	11.73	16	16.15	18.59	18.61	20.39	134	0.78897	0.9808
MCR+(PO2)	3.46	5.456	9.382	11.54	11.42	16.38	16.28	19.35	19.23	21.34	133.8	0.99614	0.9798
MCR+(DT-EPD)	2.65	4.782	9.19	11.19	11.24	16.4	16.39	19.42	19.41	21.43	132.1	0.95434	0.9673

Table 6: Simulation results for MCR+ compared with other four schemes in WAN.

Robustness of MCR+

To investigate the robustness of MCR+ under different scenarios, we do simulations for three cases. Table 8 shows that MCR+ can also achieve good fairness even if some VCs have requested MCRs (MCR > 0) but are idle. As comparison, Table 9 shows the results when some VCs do not request MCR (MCR $= 0$) and are idle; Table 10 shows the results when some VCs do not request MCR (MCR $= 0$) and are greedy sources. The results in Table 8, 9 and 10 have demonstrated the robustness of MCR+, which is due to the work conserving nature of MCR+. The reserved bandwidth of idle VCs can be utilized by active connections with fair alloction.

Insight of MCR+

MCR (Mbps)	2	4	8	10	10	15	15	18	18	20	Total	Fairness	Eff
MCR+(PO2)	0	0	0	0	0	26.17	26.06	27.23	27.24	27.23	133.9	0.9777	0.981
MCR+(DT-EPD)	0	0	0	0	0	26.05	25.95	27.21	27.23	27.21	133.7	0.9787	0.979

Table 7: Simulation results when some VCs request MCRs (MCR > 0) but are idle.

MCR (Mbps)	0	0	0	0	0	15	15	18	18	20	Total	Fairness	Eff
MCR+(PO2)	0	0	0	0	0	25.56	25.59	27.22	27.19	27.21	132.8	0.9828	0.972
MCR+(DT-EPD)	0	0	0	0	0	25.54	25.99	27.23	27.21	27.23	133.2	0.9812	0.975

Table 8: Simulation results when some VCs do not request MCR (MCR$= 0$) and are idle.

MCR (Mbps)	0	0	0	0	0	15	15	18	18	20	Total	Fairness	Eff
MCR+(PO2)	4.78	4.629	4.747	4.755	4.589	20.12	19.81	23.1	22.91	24.5	134	0.9984	0.981
MCR+(DT-EPD)	4.534	4.7	4.516	4.451	4.253	20.1	19.7	22.43	22.95	24.72	132.4	0.9973	0.969

Table 9: Simulation results when some VCs do not request MCR (MCR$= 0$) and are greedy.

As we know, TCP is window-based flow control rather than rate-based flow control. When there is a packet loss for VCi, slow-start phase with small window size will happen for this VCi connection. (Note: one TCP connection corresponds to one VC connection. There is no aggregation of TCP connections on one VC connection.) So the instant rate of VCi will be smaller than MCRi. At the same time, other VCs will get more share in excess bandwidth. Therefore, the larger the MCR is, the longer it takes for the VC's rate to reach the MCR, and the higher the probability that the VC gets smaller share in excess bandwidth is. This is non-linearity as mentioned before.

Accordingly non-linearly increasing MCR by favoring larger MCR can compensate the fair share loss during the slow-start phase, and thus achieve good fairness.

5 Conclusion

To support the GFR service, we have studied and compared different buffer management schemes: Dynamic Threshold EPD (DT-EPD), Pushout 1, Pushout 2 (PO2), Pushout 3, Quasi Pushout (QPO), and Virtual Queueing. Through simulations, we reach the following conclusions.

• Scheduling policies:

Only per-VC queueing with Weighted Round Robin for allocated MCR and Round Robin

for excess bandwidth can guarantee MCR and achieve fairness.

• Discarding policies:

The performances of PO2, DT-EPD and QPO are comparable. DT-EPD is the simplest to implement.

• Among all the schemes, we recommend DT-EPD combined with MCR+, which increases MCR non-linearly by favoring VCs with larger MCR, to support the GFR service. The simulation results shows that DT-EPD combined with MCR+ scheme performs well in LAN and WAN environment and is robust under different scenarios.

References

[1] A. K. Choudhury and E. L. Hahne, "Dynamic Queue Length Thresholds in a Shared Memory ATM Switch," *Proc. IEEE INFOCOM'96,* pp. 679–687, Mar. 1996.

[2] R. Guerin and J. Heinanen, "UBR+ Service Category Definition," *ATM Forum Contribution 96-1598,* Dec. 1996.

[3] V. Jacobson, "Congestion Avoidance and Control," *Proc. ACM SIGCOMM'88,* pp. 314–329, Aug. 1988.

[4] R. Jain, "Fairness: How to Measure It Quantitatively?" *ATM Forum Contribution 94-0881.*

[5] T. V. Lakshman, et. al., "The Drop from Front Strategy in TCP and in TCP over ATM," *Proc. IEEE INFOCOM'96,* pp. 1242–1250, Mar. 1996.

[6] Y. S. Lin, et. al., "Quasi-Pustout Cell Discarding," *IEEE Communications Letters,* pp. 146–148, Sept. 1997.

[7] A. Romanow and S. Floyd, "Dynamics of TCP Traffic over ATM Networks," *IEEE Journal on Selected Areas in Communications,* vol. 13, no. 4, pp. 633–641, May 1995.

[8] K. Y. Siu, "Virtual Queueing Techniques for UBR+ services in ATM with Fair Access and Minimum Bandwidth Guarantee," preprint.

[9] F. Street, "Traffic Management Working Group Living List," *ATM Forum LTD-TM-01.03,* Chicago, Apr. 1997.

[10] L. Tassiulas, et. al., "Optimal Buffer Control during Congestion in an ATM Network Node," *IEEE/ACM Trans. on Networking,* vol. 2, no. 4, pp. 374–386, Aug. 1994.