Efficient Support of TCP Traffic in ATM Networks

by

Yuan Wu

B.S., Electrical Engineering
Polytechnic University (Brooklyn), 1996

Submitted to the Department of Electrical Engineering and Computer Science
in Partial Fulfillment of the Requirements for the Degree of

Master of Science in Electrical Engineering and Computer Science

at the

Massachusetts Institute of Technology

January 1999

Signature of Author ....
Department of Electrical Engineering and Computer Science
January 5, 1999

Certified by ....
Kai-Yeung Siu
Assistant Professor
Thesis Supervisor

Accepted by ....
Arthur C. Smith
Chairman, Department Committee on Graduate Students

Efficient Support of TCP Traffic in ATM Networks

by

Yuan Wu

Submitted to the Department of Electrical Engineering and Computer Science
on January 5, 1999 in partial fulfillment of the
requirements for the degree of
Master of Science in Electrical Engineering and Computer Science

Abstract

The ATM Forum has recently defined a new data service called Guaranteed Frame Rate (GFR) Service. This new service aims to support minimum bandwidth guarantee for a VC and allows any excess bandwidth in the network to be shared among the contending VCs in a fair manner.

It was proposed that the service requirement of GFR may be supported using the Generic Cell Rate Algorithm (GCRA), or equivalently, a leaky-bucket algorithm. Such an approach has significant implementation and cost advantage because most commercial ATM switches already have hardware for GCRA, which is used for example in VBR service. However, previous ATM Forum contributions showed that a straightforward application of GCRA may not always guarantee both minimum bandwidth and fairness of excess bandwidth, particularly when bursty TCP traffic is considered. In this paper, we present an enhanced leaky-bucket buffer admission scheme in switches with FIFO queuing for supporting the GFR service requirements. Simulation results are presented to show that our scheme performs well under various TCP and UDP traffic environment.

Thesis Supervisor: Kai-Yeung Siu
Title: Assistant Professor

Contents

Chapter 1

Introduction

While ATM was originally conceived as a technology for supporting integrated services networks, most traffic carried over the Internet today is still data. In ATM networks, the unspecified bit rate (UBR) and the available bit rate (ABR) are service classes that have been specifically developed to support data traffic.

The UBR service is designed for those data applications that want to use any available bandwidth and are not sensitive to cell loss or delay. Such connections are not rejected on the basis of bandwidth shortage and not policed for their usage behavior. During congestion, the cells may be lost but the sources are not expected to reduce their cell rates. Instead, these applications may have their own higher-level loss recovery and retransmission mechanisms, such as the window flow control employed by TCP. The advantage of using UBR service for data transport is its simplicity and minimal interaction required between users and the ATM network.

The ABR service has been developed with the goal of minimizing switch buffer requirement and cell loss in transporting data and allowing users to have fair access to the available bandwidth. To achieve such service requirements, the ABR service uses congestion control at the ATM layer. It requires network switches to constantly monitor the traffic load and feed the information back to the sources. The sources are expected to adjust their input to the network dynamically based on the congestion status of the network. It should be pointed out that the advantage of ABR over UBR does come at the expense of increased complexity in switches and end systems. Moreover, ABR also requires a certain level of interoperability among network switches.

In fact, there has been a continuing debate in the networking community about the need for ABR service. A major argument against ABR is that while ABR assumes that the end systems comply with the ABR source behavior, most current applications are only connected to ATM via legacy networks such as Ethernet. Therefore, ABR may only push congestion to the network edges and cannot provide flow control on an end-to-end basis. Simulation results reported in recent ATM Forum contributions [13, 3] seem to support this argument.

Furthermore, it has been argued in [5] that most users today are typically either not able to specify the range of traffic parameters needed to request most ATM services, or are not equipped to comply with the (source) behavior rules required by existing ATM services. As a result, there are many existing users for whom the benefits of ATM service guarantees remain unattainable. Those users access ATM networks mostly through UBR connections, which provide no service guarantees. In view of this, the ATM Forum has recently defined a new service called guaranteed frame rate (GFR), which will provide users with some level of service guarantees yet require minimal interaction between users and ATM networks.

The GFR service specifies that a user should be provided with a minimum service rate guarantee[1] and with fair access to any excess available bandwidth. In other words, the GFR service will guarantee a user with a minimum throughput when the network is congested, while allowing a user to send at a higher rate when additional bandwidth is available.

Our main contribution is to present a new efficient leaky-bucket buffer admission scheme for supporting the GFR service. The key advantage of our scheme is that it employs only FIFO queuing (instead of per-VC queuing) and makes use of the GCRA hardware that already exists in ATM switches.

This report is organized as follows: as our scheme is based on the leaky-bucket imple-

[1] To be more precise, the minimum service rate is guaranteed under the assumption of a given maximum packet size (SDU).

mentation of the GCRA, we first present the general idea of this standard algorithm in Chapter 2. We then briefly discuss in Chapter 3 the role of early packet discard (EPD) algorithm and the use of round-robin scheduling in supporting GFR service. The algorithms and the simulation results are presented in detail in Chapters 4 and 5. Analytical results are given in Chapter 6 to show that GCRA with round-robin scheduling indeed provides GFR service. Chapter 7 shows the causes of the short term beatdowns observed in our simulations. Possible improvements to the schemes are discussed in Chapter 8. Concluding remarks are given in Chapter 9.

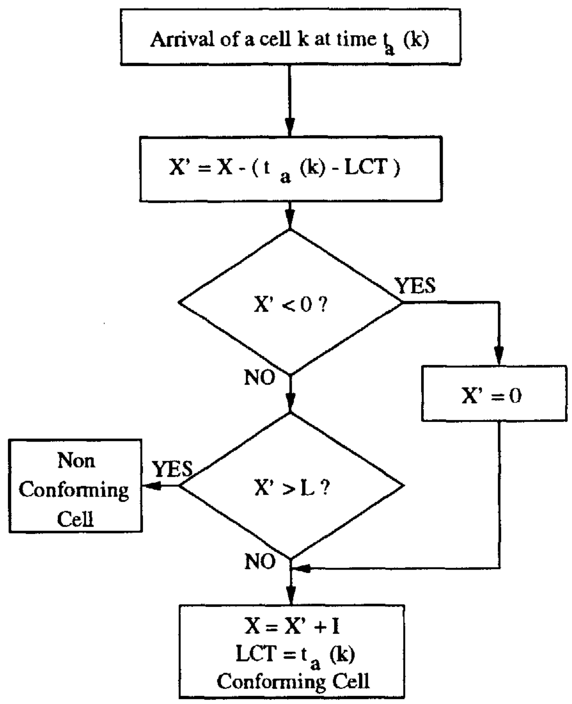
Chapter 2

The Generic Cell Rate Algorithm (GCRA)

The schemes we present are based on ATM switches which support the GCRA. Two equivalent GCRAs are described in [14]: the Virtual Scheduling Algorithm and the Continuous-State Leaky Bucket Algorithm. The later will be used in our scheme. Fig. 2-1 illustrates the Leaky Bucket Algorithm. The algorithm maintains two state variables which are the leaky bucket counter X, and the variable LCT (last conformance time) which stores the arrival time of the last conforming cell. The GCRA is defined with two parameters: I (increment) and L (limit). I is the ideal cell interarrival time (i.e. the inverse of the ideal cell rate. L is the maximum bucket level determined by the allowed burst tolerance. The notation "GCRA (I,L)" means GCRA with increment I and limit L. When the first cell arrives at time $t_a(1)$, X is set to zero and LCT is set to $t_a(1)$. Upon the arrival time of the k^{th} cell, $t_a(k)$, the bucket is temporarily updated to the value X', which equals the content of the bucket X, last updated after the arrival of the previous conforming cell, minus the amount the bucket has drained since that arrival. The content of the bucket is constrained to be non-negative. If X' is less than or equal to the bucket limit L, then the cell is conforming, and the bucket variables X and LCT are updated. Otherwise, the cell is non-conforming and the values of X and LCT are not changed. We can observe from Fig. 2-1 that for each conforming cell, the bucket counter is updated by:

$X_{new} = X_{old} - (t_a(k) - LCT) + I.$

Thus, the leaky bucket counter increases if the actual cell interarrival time $(t_a(k) - LCT)$ is less than the ideal cell interarrival time I (i.e., if the cell arrives too early), and decreases otherwise. If the cells arrive at the ideal rate, the the bucket counter will stay constant. This also illustrates the fact that the leaky bucket can be viewed as being drained out at the ideal rate (this is true independent of the input traffic) [14]. Note that if X is decreased by I, then one extra cell, beyond the ideal rate, will be admitted into the switch. A proof of this will be given in Chapter 6. The second scheme we present is based on this fact.



CONTINUOUS-STATE LEAKY BUCKET ALGORITHM

X	Value of the Leaky Bucket Counter
X'	auxiliary variable
LCT	Last Compliance Time
I	Increment (set to the reciprocal of cell rate)
L	Limit

Figure 2-1: A flow chart description of the leaky-bucket algorithm

Chapter 3

Leaky Bucket Admission Schemes for Supporting GFR

This chapter gives a high-level description of various GCRA based schemes. The implementation details are presented the the next chapter. In these schemes, a separate set of variables X_i, LCT_i, and constants I_i, L_i are used for each VC. The ideal rate which determines the increment I_i is set to the requested MCR for the VC.

3.1 Leaky Bucket with Early Packet Discard (EPD)

The first scheme we present simply incorporates the Early Packet Discard (EPD) algorithm to ensure a high level of effective throughput. It has been observed in [9] that the effective throughput of TCP over ATM can be quite low in a congested ATM switch. This low throughput is due to wasted bandwidth as the switch transmits cells from "corrupted" packets(i.e. packets in which at least one cell is dropped by the switch). A frame discarding strategy named Early Packet Discard is introduced in [9] to improve the effective throughput. EPD prevents fragmentation of packets by having the switch drop whole packets prior to buffer overflow, whenever the queue size exceeds a congestion threshold.

In our first scheme, we combine the EPD algorithm and GCRA by accepting all packets when the queue level is below the congestion threshold just like with EPD, and passing all packets to the GCRA when the queue level is above the congestion threshold. Scheme one is easy to implement. However, it provides no means of guaranteeing fair excess bandwidth allocation.

Simulation results show that GCRA with EPD can already guarantee MCR for the average throughput. (The instantaneous throughput is largely determined by TCP, as we will show in later chapters.) However, the excess bandwidth is not equally shared among all VCs, and a strong bias exists in favor of VCs with lower requested MCR as shown in Section 4.2 when we present the simulation results.

3.2 Leaky Bucket with Round Robin Scheduling

The second scheme we present involves only a slight modification to the first scheme but improves the fairness performance dramatically. In this scheme, when the queue level is below the congestion threshold, instead of accepting every valid incoming cell (i.e. a cell which does not belong to a corrupted packet) as in EPD, the switch "allocates a credit" of one cell by decrementing the bucket counter X_i by I_i in a round-robin fashion among all the VCs for each valid incoming cell. Also unlike in scheme one, the GCRA continues to apply when the queue level is below the congestion threshold. This way, every cell discard decision is made by the GCRA which leads to improved fairness. High link utilization is maintained since an extra cell is allowed into the switch (although it may not necessarily be the current incoming cell) for every valid incoming cell with the round-robin draining of the leaky buckets when the queue level is below the congestion threshold.

Simulation results show nearly ideal performance with the second scheme, including for configurations which contain connections with different round trip delays, traversing different number of congested links, or with the presence of misbehaving users. This scheme does not require any additional per-VC variable and is easy to implement. It also maintains high throughput as with scheme one.

Chapter 4

Leaky Bucket with Early Packet Discarding

4.1 Implementation

Switches with GCRA are normally used to support non-data services such as VBR. For example, the ATM Forum defines the VBR parameters in relation to two instances of GCRA. In the notation GCRA (I,L), one instance is GCRA (1/PCR, CDVT), which defines the CDVT (cell delay variation tolerance) in relation to PCR (peak cell rate). The second instance GCRA (1/SCR, BT+CDVT) defines the sum of BT (burst tolerance) and CDVT in relation to SCR (sustainable cell rate). PCR specifies an upper bound of the rate at which traffic can be submitted on an ATM connection. SCR is an upper bound of the average rate of the conforming cells of an ATM connection, over time scales which are long relative to those for which the PCR is defined. CDVT and BT specify the upper bound on the "clumping" of cells resulting from the variation in queuing delay, processing delay, etc.[14] We will refer to the GCRAs associated with PCR and SCR as GCRA(1) and GCRA(2), respectively.

When providing best effort data services, to minimize complexity, we will use the same set of parameters as for VBR service. Since our schemes use only one instance of GCRA, the other instance is disabled by choosing the appropriate parameters. If we disable GCRA(1), then we could set the parameters as follows:

- PCR: infinity (i.e. a large number).
- CDVT: 0.
- SCR: MCR.
- MBS(maximum burst size): see description below.

PCR is set to a number large enough so that every incoming cell conforms to GCRA(1). Thus, GCRA(1) has no effects on incoming cells and is disabled. GCRA(2) is used to guarantee the MCR. MBS is the parameter used to determine BT. MBS is specified in number of cells while BT is in units of time. For best effort data applications, to achieve the best fairness performance, MBS should be large enough so that the buckets never become empty. For bursty sources, this would require an arbitrarily large value of MBS. Although a large value of MBS is desirable for achieving good fairness performance, the switch buffer fluctuation also increases with MBS which is undesirable, and a compromise should be made. Since only persistent sources are used in our simulations, a sufficiently large value of MBS would be able to prevent the buckets from becoming empty.

A flow chart of scheme one is shown in Fig. 4-1. Since GCRA(1) is disabled, only GCRA(2) is shown in the figure. The Leaky Bucket Algorithm is used in our implementation although the equivalent Virtual Scheduling Algorithm could also be used. As shown in the figure, when a cell arrives at the switch, the first condition checks if the cell belongs to a corrupted packet, and discards the cell if it does. This ensures that no bandwidth is wasted delivering corrupted packets. The second condition (Q<Th?) checks if the current queue level is below the congestion threshold, and accepts all incoming cells if this is the case. This prevents underutilization, and allows the VCs to randomly contend for the excess bandwidth. The cell is then tested for MCR conformance with GCRA(2). If the cell is conforming, it is accepted. If the cell is non conforming, then another condition checks if the cell is the beginning of a new packet. If so, it is discarded. Otherwise, it is the continuation

of a partially transmitted packet, and is accepted to prevent the packet from becoming corrupted. Note that a slight change is made to the leaky bucket algorithm which is that a branch (marked with a "*" in Fig. 4-1) is made into the GCRA if a non conforming cell must be accepted. This means that when a non conforming cell is accepted, the leaky bucket variables (X and LCT) are updated just like when a conforming cell is accepted. This ensures that when there is congestion (i.e. when the queue level is above the congestion threshold), all cells accepted will be accounted for by the GCRA. The pseudocode for scheme one is included in the appendix.

The main advantage of Leaky Bucket Algorithm based schemes over per-VC accounting or per-VC queuing based schemes is that the leaky buckets can be drained at all time, regardless of the actual per-VC queue levels. The primary limitation of per-VC accounting and per-VC queuing based schemes is that when a source idles and its per-VC queue level reaches zero, the switch has no information on how long the VC has idled. As a result, TCP connections with longer idling time during window growth such as those connections with longer round trip time or smaller packet size, or VCs with bursty sources, will be "beaten down" and have lower throughput, than other VCs. With the Leaky Bucket Algorithm, however, the amount of idling time is reflected by the bucket level. For example, since buckets are drained at all time, a VC which idles longer will have a lower bucket level than other VCs, and will be able to redeem the lost bandwidth when the TCP window size catches up, or when the bursty source starts transmitting again.

4.2 Simulation Parameters and Configurations

Our simulation tool is based on the MIT Network Simulator (NetSim). NetSim is an event-driven simulator composed of various components that send messages to one another. We have built upon the NetSim package various ATM and TCP/UDP related components. These components include an ATM switch component, a SONET OC-x link component, an ATM host component, a greedy/bursty source component, a multiplexing and demultiplexing

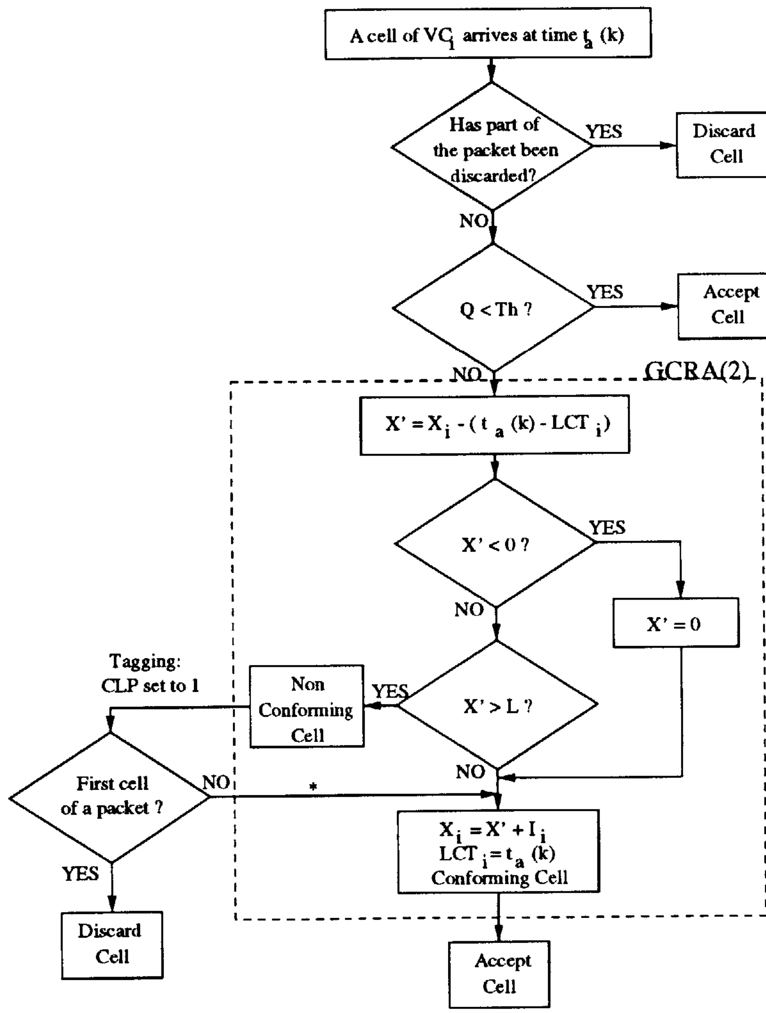


Figure 4-1: GCRA with Early Packet Discard.

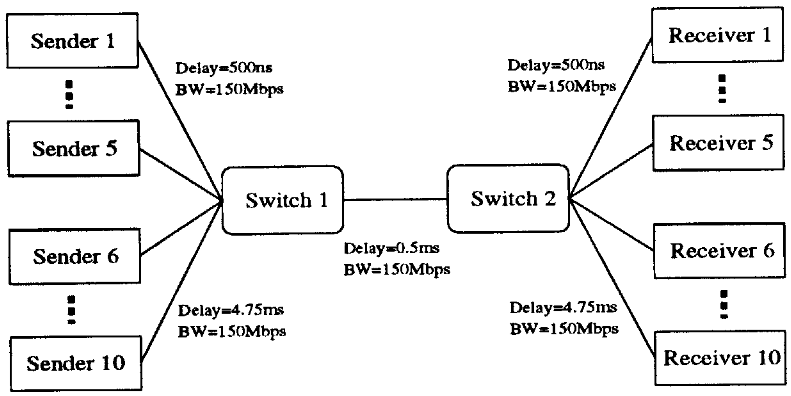


Figure 4-2: A 10-VC peer-to-peer configuration

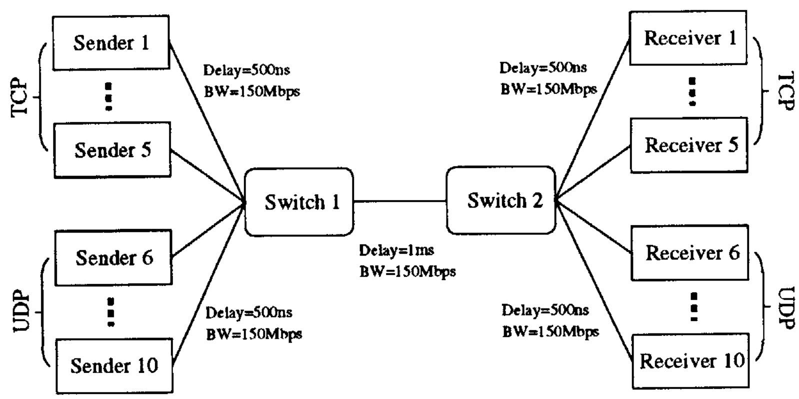


Figure 4-3: A 10-VC peer-to-peer configuration

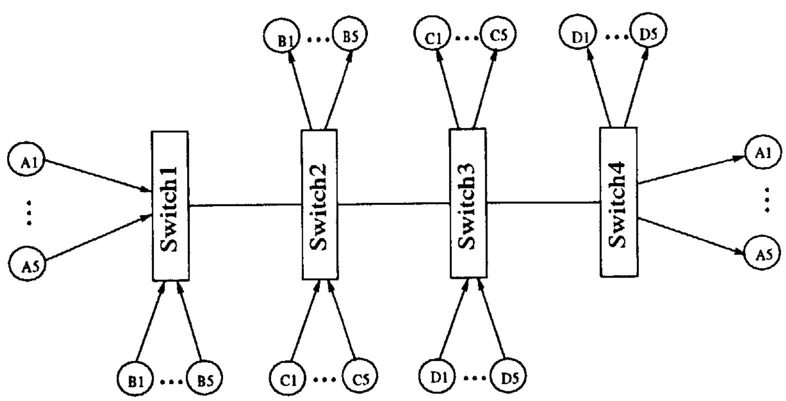


Figure 4-4: A 10-VC peer-to-peer configuration

component, a UDP component, and a TCP Reno component. We use a fine timer granularity of 10 μs in our simulations.

We tested each scheme in three different network scenarios. The first network, illustrated in Fig. 4-2, consists of peer-to-peer TCP connections with different round trip delays. The round trip link delay is 1 ms for VC1-VC5 and is 20 ms for VC6-VC10. The second network, shown in Fig. 4-3, contains five TCP and five UDP sources. The third network, shown in Fig. 4-4, has a "chain configuration". The network contains 4 groups of VCs (labeled A, B, C, and D), with 5 TCP connections in each group. VCs A1-A5 traverse three congested links while all other VCs traverse a single congested link through the network. The link delay is 500 ns between each host and switch, and is 2 ms between switches. All link bandwidths are 150Mbps. In all three network configurations, the senders are greedy sources which means for TCP, data is output as fast as allowed by the TCP sliding window mechanism and for UDP, data is sent at the link rate.

Each sender and receiver shown in the figures consists of a user component, a TCP or UDP component, and an ATM host component. On the sending side, the user components generate data for the TCP/UDP components, which form TCP/UDP packets. The packets are then segmented into cells during the AAL5 processing in the host adapters. The ATM switches perform cell switching between their input and output ports. Each ATM switch in our simulations uses a single FIFO queue in each output port. On the receiving side, cells are reassembled and passed to the TCP/UDP components and then the user components. By running multiple concurrent connections, we create congested links between the ATM switches.

The other parameters are summarized below:

TCP:

Mean Packet Processing Delay = 100 μsec,

Packet Processing Delay Variation = 10 μsec,

Packet Size = 1024 Bytes for peer-to-peer configurations, 8192 Bytes for chain configuration,

Maximum Receiver Window Size = 520 KB,

	1		2		3		4		5	
Round Trip Delay	1ms	20ms	1ms	20ms	1ms	20ms	1ms	20ms	1ms	20ms
MCR (Mbps)	3	3	6	6	9	9	12	12	15	15
Actual BW(Mbps)	7.0	5.0	11.6	9.0	15.9	13.9	20.7	18.6	25.5	20.5
Difference(Mbps)	4.0	2.0	5.6	3.0	6.9	4.9	8.7	6.6	10.5	5.5

Table 4.1: Throughput for TCP traffic under Scheme 1.

Default Timeout = 500 *m*sec.

UDP:

Packet Size = 1024 Bytes (used only in network 2). Host:

Packet Processing Delay = 500 *n*sec,

Maximum Cell Transmission Rate = 150Mbps.

PCR = infinity (1e7),

SCR = requested MCR. See Tables 4.1, 4.2, and 4.3,

CDVT = 0,

MBS = 10 MB.

Switch:

Packet Processing Delay = 4 μsec,

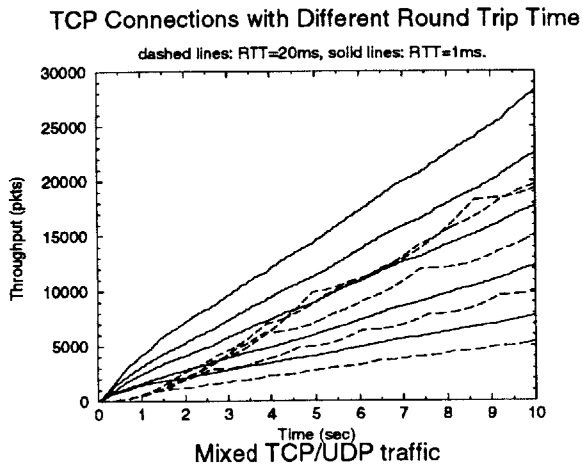
Congestion Threshold (Th) = 1000 cells.

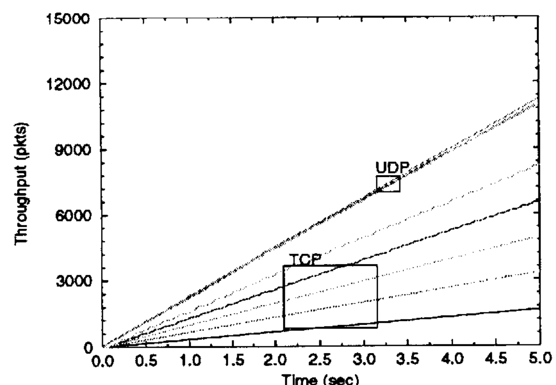
Buffer Size (Q_{max}) = infinity.

4.3 Simulation Results

The throughput results of the three simulations are plotted in Fig. 4-5. The numerical values of the throughput are shown in Tables 4.1-reftab:thr3. Corrupted packets are not counted toward the throughput. Repeated deliveries of a packet are counted only once.

Some degree of unfairness are seen in all three cases, especially in the case with the mixed TCP/UDP traffic. There are two main causes for the beat-downs observed in our





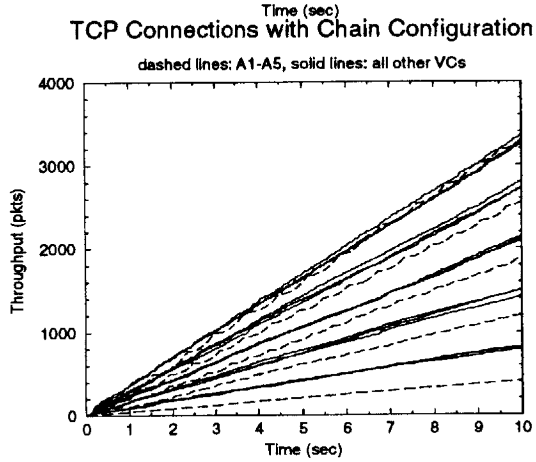


Figure 4-5: Throughput performance for Scheme One

	1		2		3		4		5	
Source	TCP	UDP	TCP	UDP	TCP	UDP	TCP	UDP	TCP	UDP
MCR (Mbps)	3	3	6	6	9	9	12	12	15	15
Actual BW(Mbps)	3.0	20.6	6.2	20.5	9.2	20.5	12.2	20.5	15.3	21.0
Difference (Mbps)	0	17.6	0.2	14.5	0.2	11.5	0.2	8.5	0.3	6.0

Table 4.2: Throughput for mixed TCP and UDP traffic under Scheme 1.

	1				2			
VC	A1	B1	C1	D1	A2	B2	C2	D2
MCR (Mbps)	2	2	2	2	6	6	6	6
Actual BW(Mbps)	3.0	5.9	5.9	5.7	8.7	10.4	10.9	10.9
Difference (Mbps)	1.0	3.9	3.9	3.7	2.7	4.4	4.9	4.9

3				4				5			
A3	B3	C3	D3	A4	B4	C4	D4	A5	B5	C5	D5
10	10	10	10	14	14	14	14	18	18	18	18
13.6	15.2	15.3	15.6	18.5	19.7	19.8	20.3	23.9	23.8	24.4	23.6
3.6	5.2	5.3	5.6	4.5	5.7	5.8	6.3	5.9	5.8	6.4	5.6

Table 4.3: Throughput for TCP Connections with Chain Configuration Under Scheme One.

simulations:

- The leaky bucket counters are not updated when the queue level is below the congestion threshold. As a result, the bucket counters will not be able to reflect the amount of bandwidths used by the VCs during this period of time. The more robust VCs, such as the ones with UDP sources or TCP connections with shorter round trip time, are able to use more bandwidths during this period of time than the less robust VCs. Without updating the bucket counters, the GCRA will not be able to restore fairness after this period of time.

- When the queue level is below the congestion threshold, all packets are accepted, regardless of the leaky bucket levels. The leaky buckets for the more robust VCs may have overflown, but these VCs will still be able to use more bandwidths than the less robust VCs during this period of time.

Both problems will be addressed in scheme two. Even with these problems, however, scheme one is still able to guarantee the MCR, as observed in our simulations. The reason MCR is guaranteed with scheme one is that the leaky buckets are drained by GCRA at the rate of MCR at all time. As a result, if the queue level stays above the congestion threshold, each VC would be able to get an average throughput of exactly MCR. If the queue level ever fall below the congestion threshold (which should happen since the total requested MCR is less than the average output link rate), all packets are accepted and the throughput values could only get higher.

Chapter 5

Leaky Bucket with Round Robin Scheduling

5.1 Implementation

Our second scheme is illustrated in Fig. 5-1. The only change made is that when the queue level is below the congestion threshold, instead of accepting every valid cell, the leaky bucket counters are decremented in a round robin fashion for each valid incoming cell, and the incoming cells are then passed on to the GCRA. The change in the pseudocode is shown in the Appendix.

This simple change solves both problems discussed earlier with scheme one. By passing every cell to the GCRA, we ensure that all cells are accounted for by the GCRA, and every cell discard decision is made based on the GCRA. Besides eliminating the problems with scheme one, we also have to make sure that one more cell would be accepted into the queue for every valid incoming cell when the queue is below the congestion threshold. In scheme one, this is done simply by accepting the incoming cell, which may be unfair. In scheme two, this is done by decrementing the buckets by one cell in a round robin fashion, which ensures that equal amount of excess bandwidth is allocated to each VC.

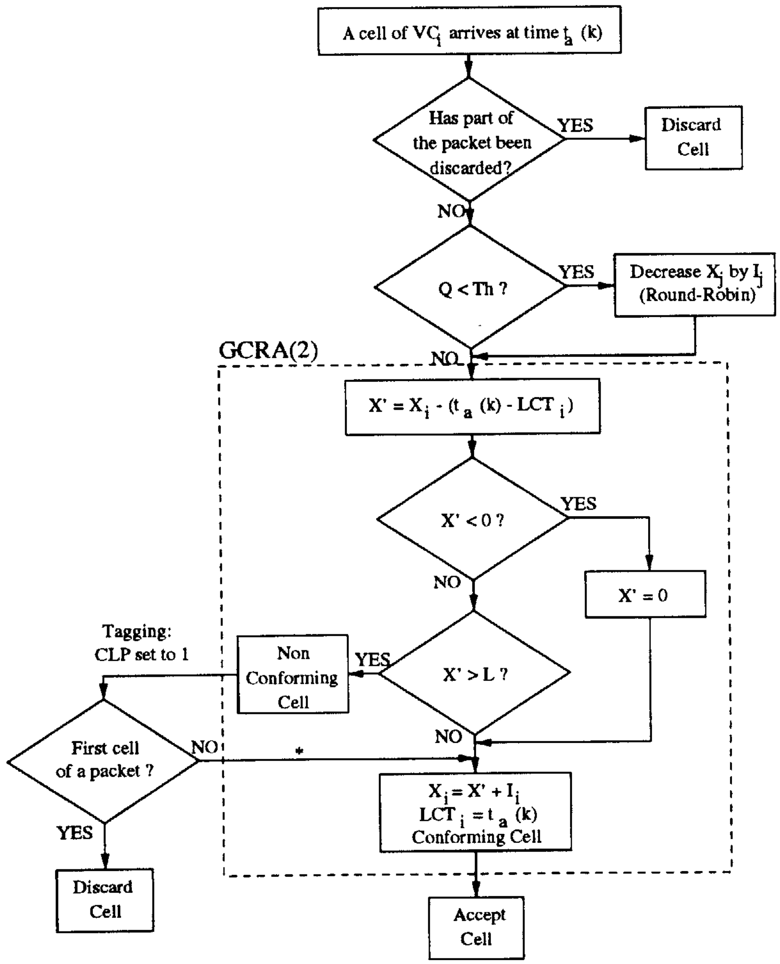


Figure 5-1: GCRA with Round-Robin Scheduling.

5.2 Simulation Results

The simulations are run with the same network configurations used for scheme one. The throughput results are plotted in Fig. 5-2 and the numerical values are shown in Tables 5.1-5.3. The results show nearly ideal fairness performance in terms of the average throughput. The short term behavior is largely determined by the dynamics of TCP which will be discussed in Chapter 7. The slight beatdown seen in the mixed TCP/UDP results is largely due to duplicated packets received by the receiving TCP, which only occur with TCP but

	1		2		3		4		5	
Round Trip Delay	1ms	20ms	1ms	20ms	1ms	20ms	1ms	20ms	1ms	20ms
MCR (Mbps)	3	3	6	6	9	9	12	12	15	15
Actual BW(Mbps)	9.0	8.9	11.8	11.5	14.9	15.0	17.8	17.4	20.7	20.3
Difference (Mbps)	6.0	5.9	5.8	5.5	5.9	6.0	5.8	5.4	5.7	5.3

Table 5.1: Throughput for TCP Connections with Different Round Trip Time under Scheme Two.

	1		2		3		4		5	
Source	TCP	UDP	TCP	UDP	TCP	UDP	TCP	UDP	TCP	UDP
MCR (Mbps)	3	3	6	6	9	9	12	12	15	15
Actual BW(Mbps)	8.7	9.1	11.5	12.1	14.5	15.1	17.3	18.0	20.1	21.0
Difference (Mbps)	5.7	6.1	5.5	6.1	5.5	6.0	5.3	6.0	5.1	6.0

Table 5.2: Throughput for mixed TCP and UDP traffic under Scheme Two.

not with UDP sources. Duplicated packets could result during retransmissions since except during fast retransmission, TCP retransmits as many packets as allowed by the congestion window size, which could include packets that have already been correctly received by the receiving TCP.

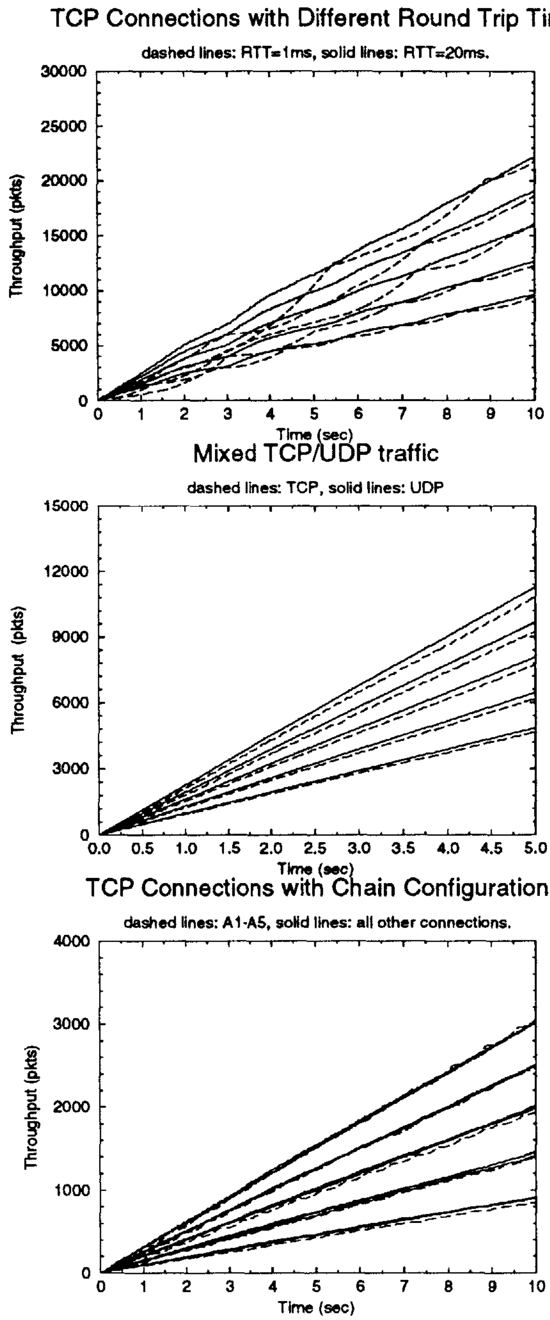


Figure 5-2: Throughput performance for Scheme Two

	1				2			
VC	A1	B1	C1	D1	A2	B2	C2	D2
MCR (Mbps)	2	2	2	2	6	6	6	6
Actual BW(Mbps)	6.2	6.6	6.6	6.6	10.2	10.2	10.6	10.6
Difference (Mbps)	4.2	4.6	4.6	4.6	4.2	4.2	4.6	4.6

3				4				5			
A3	B3	C3	D3	A4	B4	C4	D4	A5	B5	C5	D5
10	10	10	10	14	14	14	14	18	18	18	18
14.0	14.4	14.6	14.4	17.9	18.1	18.1	18.2	22.0	22.0	21.8	21.9
4.0	4.4	4.6	4.4	3.9	4.1	4.1	4.2	4.0	4.0	3.8	3.9

Table 5.3: Throughput for TCP Connections with Chain Configuration Under Scheme Two.

Chapter 6

Analytical Results

In this chapter, we will provide analytical results to show that scheme two is able to provide GFR service for the average throughput. The key to providing fair excess bandwidth allocation with scheme two is the round robin decrementation of the leaky bucket counters. As mentioned earlier, when the leaky bucket counter X_i is decremented by I_i, one extra cell (above the MCR rate) will be accepted into VC_i. This can be shown as follows.

Let t_n be the arrival time of the nth conforming cell, and X_n be the value of the bucket counter after the arrival of the nth conforming cell. Assuming that the bucket level does not reach the limit values (i.e. $0 < X_n < L$). As mentioned earlier, after accepting the nth conforming cell, the bucket counter is updated as

$X_n = X_{n-1} - (t_n - t_{n-1}) + I.$

The bucket counter is set to zero at the arrival of the first cell. Thus,

$$
\begin{aligned}
X_1 &= 0 \\
X_2 &= 0 - (t_2 - t_1) + I \\
&= t_1 - t_2 + I \\
X_3 &= X_2 - (t_3 - t_2) + I \\
&= -(t_2 - t_1) - (t_3 - t_2) + 2I \\
&= t_1 - t_3 + 2I \\
X_4 &= X_3 - (t_4 - t_3) + I
\end{aligned}
$$

$= -(t_2 - t_1) - (t_3 - t_2) - (t_4 - t_3) + 3I$

$= t_1 - t_4 + 3I$

$\vdots$

$X_n = (n-1)I + t_1 - t_n. \qquad (1)$

From the Leaky Bucket Algorithm in Fig. 2-1 and Equation 1 above, we see that at the arrival of the nth cell,

$X'_n = (n-2)I + t_1 - t_n.$

The cell is considered conforming if

$L > (n-2)I + t_1 - t_n.$

Thus, $(n-2)I+t_1-L$ is the earliest (smallest) arrival time of the nth cell to be considered conforming. If X is decremented by mI (i.e. $X_n = X_n - mI$), then

$X_n = (n-1-m)I + t_1 - t_n$

and the cell is considered conforming if

$L > (n-2-m)I + t_1 - t_n.$

This way, the earliest conforming arrival time is set as if m less conforming cells have arrived. Thus, m extra cells will be considered conforming and will be accepted into the switch buffer.

Besides the MCR allocation provided by GCRA, the round-robin allocation is the only other mechanism of allocating the bandwidth. Thus, all excess bandwidths are allocated round-robin. Let R_{excess} be the total excess bandwidth, let E_i be the excess bandwidth allocated to VC_i, and let N denote the number of VCs, we have

$E_i = \frac{R_{excess}}{N}.$

The total bandwidth allocated to each VC is thus

$R_i = MCR_i + E_i$

$= MCR_i + \frac{R_{excess}}{N}. \qquad (2)$

The buffer level will be finite is evident from the fact that scheme two behaves exactly as a VBR switch when the buffer level is above the congestion threshold. As long as the total requested MCR is less than the average output link rate, the GCRA algorithm will ensure a finite buffer level. A finite buffer level implies that the total average input rate of all VCs $\sum_{all i} MCR_i + R_{excess}$ equals the total average output rate R_{out}, or

$$R_{excess} = R_{out} - \sum_{all i} MCR_i. \qquad (3)$$

Substituting (3) into (2) gives

$$R_i = MCR_i + \frac{R_{out} - \sum_{all i} MCR_i}{N},$$

which is the ideal bandwidth specified by the GFR service.

Chapter 7

Short Term Behaviors

Although scheme two is able to provide long term fairness, some short term beat-downs still remain, as seen from Fig.5-2. In Fig.5-2, the most sever short term beat-down is observed in the first graph for the connections with 20 ms round trip time.

The beat-downs observed are mainly due to the slow linear window growth during TCP congestion avoidance. Fig.7-1 shows the TCP congestion window size (cwnd) and the slow start threshold (ssthresh) for a connection with 20 ms round trip time and packet size of 1024 Bytes. When the window size is below the slow start threshold, TCP is in the slow start phase. During slow start, the congestion window grows exponentially, doubling after every round trip time. TCP enters the congestion avoidance phase when its window size exceeds

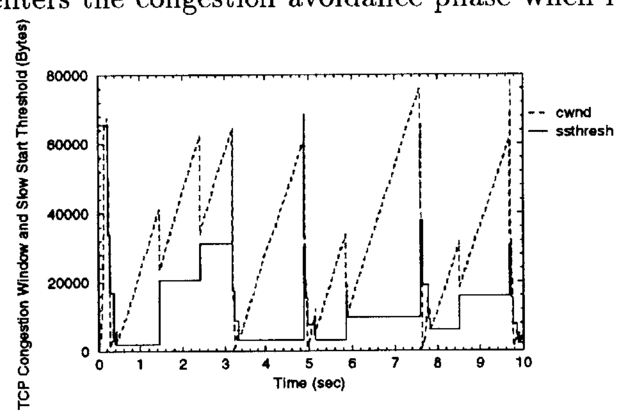


Figure 7-1: TCP congestion window size and slow start threshold of connection with RTT of 20 ms and packet size of 1024 Bytes.

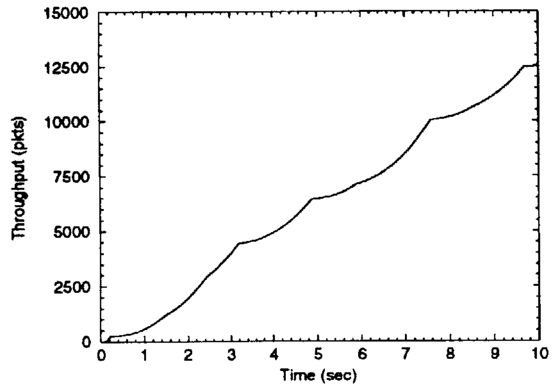


Figure 7-2: Throughput for the TCP connection studied in Fig.7-1

the slow start threshold. During congestion avoidance, the TCP window grows linearly, by at most one packet in each round trip time[2][12]. As seen in Fig.7-1, the window size is above the slow start threshold most of the time, which means that the TCP connection is in congestion avoidance, with linear window growth most of the time. The maximum value of the congestion window is approximately 6200 Bytes. It takes approximately 6200/1024 or 60 round trip times for TCP to reach this congestion window size. Since the round trip link delay is 20 ms, and the queuing delay is approximately 3ms, the time it takes for the linear TCP window growth is 60*23 ms or 1.4 seconds. This is why it takes on the order of seconds for the TCP connection to ramp up to the maximum rate after each slow start. The throughput for the TCP connection is shown in Fig.7-2.

There are less short term beat-downs for connections with shorter round trip time as seen in the first graph in Fig.5-2 for connections with a round trip time of 1ms. This is because shorter round trip time enables faster window growth. The short term behavior also improves with larger packet size, as we can see from the third graph in Fig.5-2 where the packet size is 8192 Bytes. The reason is that with larger packet size, the same bandwidth can be achieved with a smaller number of packets in the window which means fewer round trips and hence a shorter amount of time is needed for the rate to recover after each slow start.

Chapter 8

Future Considerations

In scheme two, when the queue level is below the congestion threshold, the incoming cell is passed to the GCRA and may still be discarded. In some cases, the throughput may be improved by adding a lower discard threshold. When the queue level is below this lower congestion threshold, all valid incoming cells would be accepted to maintain high link utilization. The cells are accepted regardless of the bucket levels which is not fair. However, fairness will be restored when the queue level rises above the lower congestion threshold, as long as the buckets are still drained in a round-robin fashion while the queue level is below the lower congestion threshold. To accept every valid incoming cell and still drain the buckets round-robin, we could increment the bucket counter X_i of the VC which the accepted incoming cell belongs to by I_i to take a "credit" of one cell away from this VC, and then decrement the bucket counters by one cell round-robin to give away this "credit". The problem with this approach, however, is that X_i could rise so high that when the queue level increases above the lower threshold and GCRA is again applied, VC_i could be shut off long enough for its rate to temporarily fall below the requested MCR. To ensure MCR guarantee, we could implement a separate counter Y_i for each VC. When the bucket counter X_i reaches the limit L_i, instead of increasing X_i when a cell of VC_i is accepted while the queue is below the lower congestion threshold, Y_i will be incremented. The implementation of the GCRA remains unchanged and will continue to use only the bucket counter X_i. During the round-robin draining of the leaky buckets, the counter Y_i will be decremented, until it reaches zero, before X_i is decremented. This implementation could improve the throughput while ensuring the MCR guarantee.

Chapter 9

Conclusion

In this report, we presented two new schemes for supporting the GFR service. The schemes are based on the GCRA algorithm and can be implemented with slight modifications to existing VBR switches. The first scheme incorporates GCRA with the EPD algorithm to provide MCR guarantee and random allocation of excess bandwidths. Although scheme one is able to provide MCR guarantee, it does not provide fair excess bandwidth allocation in most situations. The second scheme presented in this report incorporates round-robin excess bandwidth allocation with scheme one. Analytical results and simulations presented in this report show that scheme two is able to provide GFR service for the average bandwidth. We also showed in this report that the short term beatdowns observed in our simulations are mainly due to the limitations of the rate of TCP window growth.

Bibliography

[1] F. Chiussi, Y. Xia, and V. P. Kumar. Virtual Queuing Techniques for ABR Service: Improving ABR/VBR Interaction. To appear in Infocom'97.

[2] V. Jacobson. Congestion Avoidance and Control. *Proc. ACM SIGCOMM'88*, pages 314-329, August 1988.

[3] S. Jagannath and N. Yin. End-to-End TCP Performance in IP/ATM Internetworks. *ATM Forum Contribution 96-1711*, December 1996.

[4] R. Jain *The Art of Computer Systems Performance Analysis: Techniques for Experimental Design, Measurement, Simulation, and Modeling*, Wiley- Interscience, New York, NY, April 1991.

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

[6] J. B. Kenney Satisfying UBR+ Requirements via a New VBR Conformance Definition. *ATM Forum Contribution 97-0185*, February 1997.

[7] H. Li, K.-Y. Siu, H.-Y. Tzeng, C. Ikeda and H. Suzuki. A Simulation Study of TCP Performance in ATM Networks with ABR and UBR Services. *Proceedings of IEEE INFOCOM'96*, March 1996. A journal version of this paper is to appear in the *IEEE Transactions on Communications*, Special Issue on High Speed Local Area Network, May 1996.

[8] 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 Discard Schemes. *Proceedings of the International*

Phoenix Conference on Computers and Communications, March 1996. A journal version of this paper is to appear in *Computer Communications*, Special Issue on Enabling ATM Networks.

[9] A. Romanow and S. Floyd. Dynamics of TCP Traffic over ATM Networks. *IEEE Journal on Selected Areas in Communications.* pp. 633–41, vol. 13, No. 4, May 1995.

[10] Y. Wu, K.-Y. Siu, and W. Ren. Improved Virtual Queueing and EPD Techniques for TCP over ATM. Proceedings. IEEE International Conference on Network Protocols, 1997.

[11] K.-Y. Siu, Y. Wu, and W. Ren. Virtual Queueing Techniques for UBR+ Service in ATM With Fair Access and Minimum Bandwidth Guarantee. , IEEE Globecom, 1997.

[12] R. Stevens. TCP/IP Illustrated, Volume 1. , Addison-Wesley Publishing Company, Massachusetts, 1994.

[13] N. Yin and S. Jagannath. End-to-End Traffic Management in IP/ATM Internetworks *ATM Forum Contribution 96-1406*, October 1996.

[14] ATM Traffic Manegement Specifications 4.0. *ATM Forum*

Appendix

The following is the pseudocode for scheme one. We denote "Q" and "Qmax" as the current queue occupancy and the maximum capacity of the ATM switch buffer, respectively. "Th" is the discard threshold. "Receiving_i" is a flag which indicates whether an incoming cell is the first cell of a packet in VC_i. "Discarding_i" is used to indicate whether some cells of a packet has already been discarded. "BT" is the burst tolerance and is obtained from: BT = (MBS-1)(T_s-T) where T_s is 1/SCR and T is 1/PCR.

Initialization:

    $Receiving_i = 0$
    $Discarding_i = 0$

When a cell is coming to an ATM switch:

    if $Q > Qmax$
        discard the cell
        $Discarding_i = 1$
        $Receiving_i = 1$
    else if $Discarding_i = 1$
        discard the cell
    else if $Q < Th$
        accept the cell
        $Receiving_i = 1$
    else
        /*GCRA(1)*/

$X'_{1i} = X_{1i} - (current\ time - LCT_{1i})$

if $X'_{1i} < 0$

    $X'_{1i} = 0$

else if $X'_{1i} > CDVT_i$

    $conform_1 = 0$

    goto GCRA(2)

$X_{1i} = X'_{1i} + 1/PCR_i$

$LCT_{1i} = current\ time$

$conform_1 = 1$

GCRA(2):

if $CLP = 0$

    $X'_{2i} = X_{2i} - (current\ time - LCT_{2i})$

    if $X'_{2i} < 0$

        $X'_{2i} = 0$

    else if $X'_{2i} > CDVT_i + BT_i$

        $conform_2 = 0$

        goto endGCRA

    $X_{2i} = X'_{2i} + 1/SCR_i$

    $LCT_{2i} = current\ time$

    $conform_2 = 1$

endGCRA:

$accept = 0$

if $CLP = 0$ and $conform_1 = 1$ and $conform_2 = 1$

    $accept = 1$

else if $CLP = 0$ and $conform_1 = 1$ and $conform_2 = 0$

    $CLP = 1$ /* tagging */

if $CLP = 1$ and $conform_1 = 1$ and $Receiving_i = 1$

$X_{2i} = X'_{2i} + 1/SCR_i$

$LCT_{2i} = current\ time$

$accept = 1$

if $accept = 1$

accept the cell

else

discard the cell

$Discarding_i = 1$

$Receiving_i = 1$

if incoming cell is an EOM cell

$Receiving_i = 0$

$Discarding_i = 0$ □

For scheme two, the only change that needs to be made is to replace the following lines in the pseudocode for scheme one

⋮

else if $Q < Th$

accept the cell

$Receiving_i = 1$

else

/*GCRA(1)*/

⋮ □

with the the lines below.

⋮

else if $Q < Th$

$X_{2j} = X_{2j} - 1/SCR_j$

set j to the VC Number of the next VC (round robin)

/*GCRA(1)*/

⋮ □