Copyright Warning & Restrictions

The copyright law of the United States (Title 17, United States Code) governs the making of photocopies or other reproductions of copyrighted material.

Under certain conditions specified in the law, libraries and archives are authorized to furnish a photocopy or other reproduction. One of these specified conditions is that the photocopy or reproduction is not to be "used for any purpose other than private study, scholarship, or research." If a, user makes a request for, or later uses, a photocopy or reproduction for purposes in excess of "fair use" that user may be liable for copyright infringement,

This institution reserves the right to refuse to accept a copying order if, in its judgment, fulfillment of the order would involve violation of copyright law.

Please Note: The author retains the copyright while the New Jersey Institute of Technology reserves the right to distribute this thesis or dissertation

Printing note: If you do not wish to print this page, then select "Pages from: first page # to: last page #" on the print dialog screen

ABSTRACT

TCP/IP TRAFFIC OVER ATM NETWORK WITH ABR FLOW AND CONGESTION CONTROL

by
Liping An

Most traffics over the existing ATM network are generated by applications running over TCP/IP protocol stack. In the near future, the success of ATM technology will depend largely on how well it supports the huge legacy of existing TCP/IP applications.

In this thesis, we study and compare the performance of TCP/IP traffic running on different rate based ABR flow control algorithms such as EFCI, ERICA and FMMRA by extensive simulations. Infinite source-end traffic behavior is chosen to represent FTP application running on TCP/IP. Background VBR traffic with different ON-OFF frequency is introduced to produce transient network states as well as congestion. The simulations produce many insights on issues such as: ABR queue length in congested ATM switch, source-end ACR (Allowed Cell Rate), link utilization at congestion point, efficient end to end TCP throughput, the TCP congestion control window size, and the TCP round trip time. Based on the simulation results, zero cell loss switch buffer requirement of the three algorithms are compared, and the fairness of ABR bandwidth allocation among TCP connections are analyzed. The interaction between the TCP layer and the ATM layer flow and congestion control mechanism is analyzed. Our simulation results show that in order to get a good TCP throughput and affordable switch buffer requirement, some kind of switch queue length monitoring and control mechanism is necessary in the ABR congestion algorithm.

TCP/IP TRAFFIC OVER ATM NETWORK WITH ABR FLOW AND CONGESTION CONTROL

by
Liping An

A Thesis
Submitted to the Faculty of
New Jersey Institute of Technology
in Partial Fulfillment of the Requirements for the Degree of
Master of Science in Electrical Engineering

Department of Electrical and Computer Engineering

January 1997

APPROVAL PAGE

TCP/IP TRAFFIC OVER ATM NETWORK WITH ABR FLOW AND CONGESTION CONTROL

Liping An

Dr. Nirwan Ansari, Thesis Advisor — Date
Associate Professor of Electrical and Computer Engineering, NJIT

Dr. Ali Akansu, Committee Member — Date
Associate Professor of Electrical and Computer Engineering, NJIT

Dr. John Carpinelli, Committee Member — Date
Associate Professor of Electrical and Computer Engineering, NJIT

BIOGRAPHICAL SKETCH

Author: Liping An

Degree: Master of Science in Electrical Engineering

Date: January 1997

Undergraduate and Graduate Education:

- Master of Science in Electrical Engineering,
  New Jersey Institute of Technology, Newark, NJ, 1997
- Bachelor of Science in Electrical Engineering,
  Tsinghua University, Beijing, P. R. China, 1990

Major: Electrical Engineering

TABLE OF CONTENTS

LIST OF FIGURES

Figure Page

CHAPTER 1

INTRODUCTION

ATM networks provide four classes of service: Constant bit rate (CBR), Variable bit rate (VBR), Available bit rate (ABR), and Unspecified bit rate (UBR). Link bandwidth is first allocated to CBR and VBR classes. The remaining bandwidth, if any, is given to ABR and UBR traffic. ABR service can rapidly allocate the remaining bandwidth among active ABR sources, hence enhancing the overall link utilization and efficiency. The primary goal of ABR service is intended to economically support data applications which do not have explicit throughput and transmission delay requirement, such as file transfer (FTP) and remote login (TELNET). Most of the data applications cannot predict their own traffic parameters and have bursty nature. Although data application can tolerate transmission delay and delay jitters, it is very sensitive to data lost. Because of the well known TCP packet segmentation problem, even a single cell is dropped by a congested switch, the whole packet is discarded at the end system that causes retransmission of the entire packet by the TCP layer. The simulation study in [10] shows that the performance of TCP/IP over ATM networks without ATM level congestion control is quiet poor when the switch is congested and begins dropping cells.

In order to achieve high bandwidth utilization and in the mean while avoid the cell loss due to network congestion when many applications compete for network resources, some kind of ATM level flow and congestion control mechanism is necessary for an acceptable performance of upper layer protocols running over ABR service. After a considerable debate, a rate-based flow control framework for the ABR service has been specified by the ATM Forum [1]. Without making commitment to any particular switch algorithm, the framework provides several types of feedback to control the source rate in response to changes of network conditions by specifying

the source/destination end-system behavior and switch behavior. A network switch is responsible for allocating the fair share of the bandwidth among all connections that compete at this switch point, and send this information back to the source end-system periodically using Resource Management (RM) cells. Since this allocation policy is implementation specific, it has been the focus of switch design and implementation for the last few years. This issue has been becoming one of the important differentiating factors for the next generation of commercially available ATM switches.

Many rate-based ABR flow and congestion control algorithms have been proposed and extensive simulations have been done on the ABR level. A survey of nine proposed rate-based ABR algorithms is presented in [2]. Many of these algorithms exhibit trade-off between performance and implementation complexity. However, few research literature has been found on performance comparison of TCP/IP traffic running on these algorithms. In this thesis, we select three representative ABR algorithms: EFCI, ERICA and FMMRA to simulate the performance of TCP/IP running over them. EFCI is chosen because it is a simple binary scheme which is implemented in most of today's ATM switches. Its implementation cost is low, but it does not ensure fair rate allocation, and has the well known beat-down problem. ERICA is a typical Explicit Rate (ER) algorithm using congestion avoidance scheme, and has a reasonable implementation complexity. FMMRA is a max-min rate based algorithm which has several advantages over other algorithms, but it has relatively higher implementation complexity. In this thesis, we compare simulation results of different algorithms, and from which we gain some insights of the interaction between TCP layer congestion control and the ATM layer congestion control mechanism.

CHAPTER 2

TCP TRAFFIC OVER ATM NETWORKS

2.1 TCP Congestion Control Mechanisms

TCP is one of the few transport protocols that has its own congestion control and recovery mechanisms. Since TCP is designed to run over a connectionless network layer, congestion control is implemented in the TCP layer between the end-points of each connection. An important characteristic of a TCP congestion control algorithm is that it assumes no support from the underlying network and lower layers to indicate or control congestion, but instead it uses implicit signals such as acknowledgments, time-outs, and duplicate acknowledgments to infer the state of the network. These feedback signals are used to control the amount of traffic injected into the network by modifying the window-size used by the sender. The congestion control algorithm attempts to utilize the available bandwidth of the network as much as possible, without, at the same time, introducing congestion. In addition, congestion control policies can be implemented in IP gateways to effectively complement the TCP end-to-end algorithms such that some degree of fairness can be maintained among the connections sharing resources in the network. Many such gateway congestion-control policies are surveyed in [9].

The congestion control mechanisms used in TCP are based on a number of ideas proposed by Jacobson [3]. Many improvements have been proposed over the years for making TCP more suitable for high speed large delay networks [4] but have not been widely implemented yet. Most of today's TCP implementations are based on or derived from either 4.3 BSD UNIX Tahoe or Reno version. The TCP Reno Version, introduced in 1990, was enhanced with the fast retransmit and fast recovery algorithm which tries to avoid performing slow-start when the level of congestion in the network is not so severe that requires a drastic reduction in the congestion control window size.

The TCP congestion control mechanism consist of three parts: slow start, congestion avoidance, retransmission and exponential backoff. The slow-start algorithm is used to perform congestion recovery by decreasing the window-size to one segment, and doubling it once every round-trip time. The function of the congestion avoidance mechanism is to probe for additional available bandwidth in the network by gradually increasing the window size. The retransmission and exponential backoff mechanism retransmits the packet after the packet loss is detected, and attempts to maintain a good estimate of the round-trip delay which is used as a basis to set the retransmission timers.

The TCP source is allowed to transmit up to the size of the transmission window (*wnd*) unacknowledged packets into the network. The value of *wnd* is chosen to be the minimum of two variables: congestion control window (*cwnd*) and maximum window size (*max_wnd*). *max_wnd* is a variable specified by a receiver at the connection setup time to reflect the maximum amount of buffering that it has available for the connection. *cwnd* is used to track network conditions: it is incremented whenever new TCP packet is acknowledged and is decreased when a packet loss is detected. During the slow start phase, the TCP source starts with a transmission window size of one packet. Slow start threshold (*ssthresh*) is set to half of the *max_wnd*. Every time when an acknowledgment is received, the sender increases its *cwnd* by one packet. This results in doubling the transmission window size in each round-trip time. This phase ends when *cwnd* reaches *ssthresh* or a packet loss is detected by a time-out.

The term slow-start may be a misnomer because the start is not really slow under ideal conditions. Actually the window size is exponentially increased during the slow-start stage. It takes only $log_2 N$ round trips before TCP can send N segments. In our simulations, if the distance between the sender and receiver is 1000 meters, and the ATM switch runs at 155 Mbit rate, the average round trip time

is approximately 15 milliseconds. The IP Maximum Segment Size is 9180 byte and the average packet process time is 0.2 milliseconds. After four round trip times (0.06 second), the TCP can transmit up to 16 segments.

$$\text{The amount of data} = 16 * 9180 * 8 = 1.175 Mbit \tag{2.1}$$

If there are multiple TCP connections connected to the same switch, the traffic load could be increased very quickly. Slow start allows the TCP source to quickly attain maximum transmission rates when the network bandwidth is available. To avoid increasing the window size too quickly and causing additional congestion, TCP introduce one additional restriction. Once the congestion window reaches the slow start threshold, TCP enters a congestion avoidance phase, and slows down the rate of increment. During this phase, TCP sender increases its *cwnd* by $1/cwnd$ packet each time an acknowledgment is received. Congestion avoidance forces TCP sender to slow down the rate increment near the congestion point while allowing them to increase windows to take advantage of new bandwidth that may become available.

TCP senders maintain a timer to keep track of round-trip times for successful transmission. Packet losses are detected by the failure of acknowledgments to arrive within a time-out period based on this timer. After a packet loss is found, TCP assumes there is a congestion occurred in the network, and it enters the retransmission and exponential backoff stage. In this stage, the TCP congestion window (*cwnd*) is reduced to one packet, slow start threshold (*ssthres*) is set to half of the size at the time of the loss, and the lost packet is retransmitted. The time-out value is doubled each time when a retransmitted packet fails to be acknowledged, which is called exponential backoff. As a result, the traffic injected by TCP source into the network is sharply decreased in periods of sustained congestion. The TCP sender enters slow start stage again if a successful retransmit acknowledgment is received.

Since TCP reduces the slow start threshold by half for every loss, it decreases the window exponentially if loss persists. In other words, if congestion is likely, TCP reduces the volume of traffic exponentially and the rate of retransmission exponentially. If loss continues, TCP eventually limits transmission to a single datagram and continues to double time-out values before retransmitting. The idea is to provide quick and significant traffic reduction to allow IP routers enough time to clear the datagrams already in their queues.

2.2 Performance Problems of TCP over ATM Network

Each TCP packet is fragmented into many short 53-byte ATM cells. The longer the TCP packet, the more ATM cells are fragmented into. All those ATM cells are transmitted by TCP sources and multiplexed by the switches on the way to their destinations. If one of the switches is congested, it begins to drop ATM cells. Even if only one cell of the TCP packet is dropped by the switch, the whole packet becomes useless, and needs to be retransmitted. Furthermore, the corrupted packet can not be recognized by the network, and is still delivered to its destination and then discarded, thus resulting in a waste of the precious bandwidth and causing further congestion. This is the well known TCP fragmentation problem over ATM network. Owing to this phenomenon, the ATM layer cell loss ratio due to congestion does not indicate the TCP throughput loss at all. One percent cell loss can cause 10% or even 50% amplified throughput loss. Normally, the longer the TCP packet, the worse the performance of TCP due to congestion. However, in high speed networks like ATM networks, we would like to choose large TCP packet size to improve the transmission efficiency and reduce overheads. Also, a bigger TCP packet size will substantially increase the aggressiveness of the TCP's window increase algorithm during the slow start stage.

After the TCP source detects a packet loss using the time out mechanism, it retransmits the lost packet and enters the slow start stage by setting the congestion window to one and the slow start threshold to half of the current value. Although the slow start mechanism can successfully reduce the amount of traffics injected into the already congested network and avoid further congestion collapse, it wastes a considerable time and bandwidth. Most of today's TCP implementations use 0.5 second timer granularity. Compared to the Round Trip Time (RTT) of high speed, low delay ATM network, this timer granularity is too coarse. While TCP sources are waiting for the time out period, a considerable amount of time and bandwidth is wasted because of this idle time. Romanow and Floyd's paper [10] presents a simulation result, and shows that the TCP effective throughput over plain ATM network without any congestion control can be as low as 34% of the maximum possible.

TCP can achieve its maximum throughput only when there is no cells loss due to congestion. The TCP packet length and window size, Round Trip Time (RTT) of the network, the switch buffer size, and the congestion algorithm are factors that contribute to the cell loss ratio. Although reducing the TCP packet length and window size, increasing the switch buffer size can reduce congestion, the congestion caused by high ON/OFF frequency background VBR traffic and long RTT time cannot be eliminated. Also, simply reducing the TCP packet length and window size results in low transmission efficiency and link utilization. In order to achieve an acceptable TCP throughput performance, some kind of ATM layer congestion control algorithm implemented in the ATM switches is necessary.

2.3 Analysis of TCP Behavior over ABR Service

Since TCP is designed to run on different kind of networks such as X.25, Frame relay, ATM, etc., it must have its own transport layer flow and congestion control

mechanism. It assumes no support from other lower layers such as the network or data link layer. The only information it uses to detect a network congestion is packet loss through time out measurement. Although some data link protocols such as ATM ABR service do provide some kind of network congestion information, today's TCP implementations cannot utilize them.

When TCP traffics are transmitted over the ATM ABR service, both data link and transport layer congestion control mechanism affect the TCP behavior and performance. TCP running on different ABR algorithms have different performance and switch buffer requirements. Different TCP and ABR parameter settings and combination also make the TCP behavior different. The complexity of the interaction between TCP and ATM ABR service makes the mathematical analysis impossible. System simulations are the only means to gain insights of the problem.

ATM Forum has standardized the rate-based ABR congestion control framework. It is basically a data link layer algorithm. Based on the traffic load, ATM switches place feedback information in RM cells. RM cells returning to the source carry the congestion status of the network. When the source receives the RM cell from the network, it adjusts its rate according to the feedback. When there is a steady flow of RM cells in the forward and reverse directions, there is a steady flow of feedback from the network to the traffic source.

Congestion control of TCP over ATM network consists of two stages: open loop control stage and close loop control stage. After the source transmits the first ATM cell belonging to a TCP packet, and before it receives the first returned RM cell from the network, this period is known as the open loop control stage. In this period, the source does not have any information about the network status. The source rates are primarily controlled by the ABR Source End System (SES) rules which are specified by ATM Forum Traffic Management Specification, Version 4.0 [1]. Among the SES rules, Rule 6 specifies the Transient Buffer Exposure (TBE) parameter which is used

to limit the number of cells the ABR source can inject into the network before it receives the first returned RM cell. The standard requires that sources reduce its ACR if TBE cells have been transmitted and a feedback from the network has not been received. This mechanism attempts to avoid the cell loss during the open-loop phase. The number of TCP connections and the switch buffer size should be taken into consideration when the TBE parameter is set.

After a round trip time (RTT), the first returned RM cell reaches the source and brings back the network information. The open-loop control is then replaced by the closed-loop control. In this stage, the ABR rate control loop has been established and the source rates are primarily controlled by the network feedback. This network feedback is effective after a time delay. The time delay required for the new feedback to take effect is the sum of the time taken for a RM cell to reach the source from the switch and the time for a cell to reach the switch from the source. We call this time the "feedback delay".

At the beginning of a TCP transmission, the TCP congestion control mechanism sets its congestion window to one. The congestion window size is doubled every round trip time. This procedure is called "slow start" procedure. As shown in Figure 2.1, at the ATM layer, the TCP traffic is considered bursty. Initially, there is a short active period (the first packet is sent) followed by a long idle period (nearly one round-trip time, waiting for an ACK). The length of the active period doubles every round trip time and the idle period reduces correspondingly. Finally, the active period occupies the entire round trip time, and the idle period reduces to zero. After this point, the TCP traffic appears as an infinite traffic stream at the ATM layer.

When only light traffic load is experienced at ATM switches, the switch algorithm typically allocates high rates to the sources. This is likely to be the case when a new TCP connection starts sending data. The data transmission rate is bottlenecked by the TCP congestion window size but not by the ABR source rate.

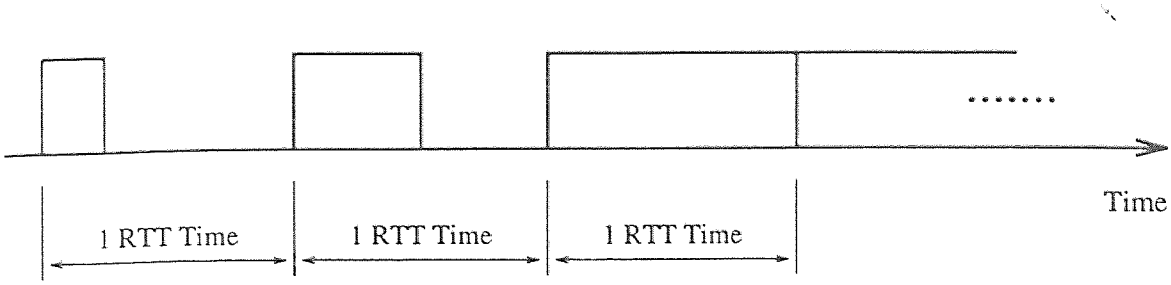

Figure 2.1 The bursty nature of TCP traffic over ATM network

Although the ABR source sends data at the rate indicated by the RM cell feedback from the network switch, the actual data rate is less than the rate specified in the ER field. At this state, the TCP source is said to be window-limited, implying that the source's data rate is bounded by the TCP congestion control window size instead of the Allowed Cell Rate in the ACR field of the RM cell.

The TCP active period doubles every round trip time, and the traffic eventually loades the switches and appears as an infinite traffic at the ATM layer. Since the network capacity is limited, switches which experience congestion begin to request sources to reduce their rates by sending them RM cells which contain the ACR value. The TCP congestion window is now large, and is increasing. Hence, it will send data at a rate greater than the source's allowed sending rate. The source transmission rate is bounded by the ABR source rate, and not by the TCP congestion window size. In this state, the TCP sources are said to be rate-limited.

The ABR queue length at switches starts increasing when TCP idle times are not long enough to clear the queues built up during the TCP active times. Queue length may increase until the ABR source rates converge to optimum values. Once TCP sources are rate-limited and converge to optimum rates, length of ABR queues at switches will start to decrease.

Cell loss will occur in the network if ATM switches do not have sufficient buffers to accommodate the queue build-up. Simulation results in [7] show that TCP achieves maximum throughput over ABR when there is no cell loss. When

cell loss does occur, the cell loss ratio is not an accurate indicator of the TCP throughput degradation. This is due to both the TCP fragmentation and TCP slow start mechanism. When the slow start is trigged, TCP loses precious time and bandwidth to wait for the retransmision time out. Finer timer granularity can improve the TCP throughput in this situation.

2.4 AAL5 Encapsulation

In our simulation model, AAL5 IP datagram encapsulation scheme is used to simulate TCP traffic over ATM network. The following is a brief introduction of AAL5 and the IP datagram encapsulation proposed by Internet Engineering Task Force's (IETF) RFC 1577 [8].

ATM is generally considered to have three layers: the physical layer, the ATM layer, and the ATM adaptation layer (AAL). The physical layer is responsible for transmission of bits over the physical media. The ATM layer consists of connection-oriented flows of fixed-length cells. These cells are made up of a five-byte header and 48 bytes of data. The ATM layer guarantees that cells are delivered to the receiver in the order that they were transmitted by the sender. However, it does not guarantee that bits in the user data field will be unaltered, and cells will not be lost during transmission. The AAL currently consists of four defined protocols: AAL1, AAL2, AAL3/4 , and AAL5. AAL5 is the adaptation layer of the choice for IP over ATM. Figure 2.2 indicates an AAL5 PDU.

The AAL5 PDU can hold up to 65536 bytes of data. The two-byte length field specifies the actual length of the user data field. The pad field ensures that the entire AAL5 PDU exactly fills the 48-byte user data field. The user-user (UU) field transparently conveys a single byte of data between AAL5 users. It is not used in IP over ATM. The common part indicator (CPI) currently must be set to zero. The integrity of an AAL5 PDU is ensured by a 32-bit cyclical redundancy check (CRC).

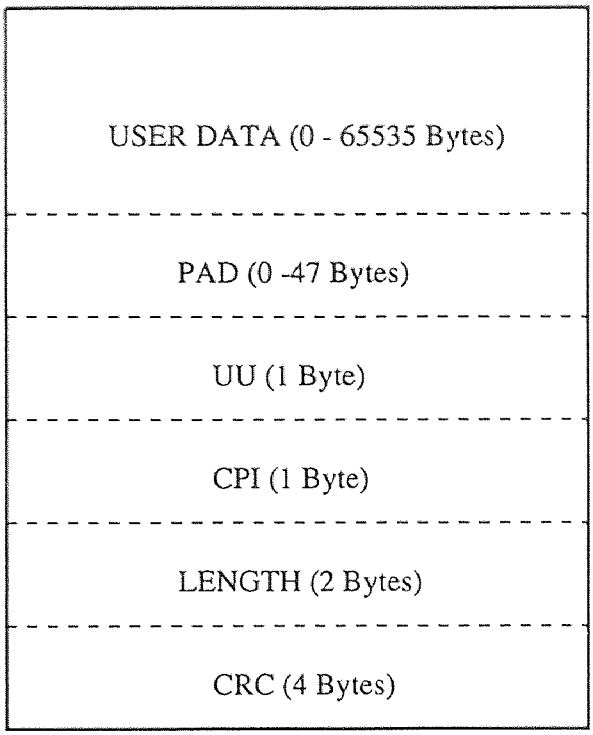

Figure 2.2 Format of AAL5 PDU

The process of transmitting AAL5 over the ATM layer is conceptually simple. The AAL5 PDU is fragmented into 48-byte chunks and transmitted in successive ATM cells. All of the cells, except the last one, have the user indication bit in the ATM cell header PTI set to zero. The last cell has the user indication set to one.

The receiver reassembles the AAL5 PDU cell by cell, using the user indication bit to tell when the last cell has been received. The AAL5 length field specifies where the actual user data end and the pad begins, and the CRC provides assurance that no cells have been lost or altered.

IP over ATM is currently being standardized by the Internet Engineering Task Force's (IETF) IP over ATM working group. There are two draft standards or requests for comments (RFC) that currently address running IP over ATM networks.

RFC 1483, "multiprotocol Encapsulation over ATM Adaptation Layer 5" discusses a number of different protocols that may be either routed or bridged over

ATM networks. RFC 1577, "Classical IP and ARP over ATM" addresses IP over ATM specifically [8]. It coins the term "classical model" to describe ATM integrated with IP as a standard subnet protocol. It applies the concept of a logical IP subnet to an ATM network configuration, and defines how address resolution is accomplished over ATM.

CHAPTER 3

ABR RATE-BASED FLOW CONTROL MECHANISM

ATM Forum has defined a family of four service categories: Constant Bit Rate (CBR), Variable Bit Rate (VBR), Unspecified Bit Rate (UBR), and Available Bit Rate (ABR). The first two service categories were intended to address applications like circuit emulation or real-time video, with precisely defined requirements for throughputs and delays. The total link bandwidth is first allocated to the VBR and CBR classes. The remaining bandwidth is then given to ABR and UBR traffic. UBR was intended for applications with minimal service requirements. It does not specify any traffic control mechanism and does not guarantee any cell loss ratio. The UBR service allows users to dump their data into the network, and it transports the data if sufficient bandwidth and buffers are available. If the required capacity is not available, cells are dropped.

The primary goal of the ABR service is to economically support data applications. Most of the data applications cannot predict their bandwidth requirements, and are sensitive to cell loss, but can tolerate certain delay. ABR is designed in such a way that these applications can dynamically and efficiently grab any unused network resources such as: bandwidth and buffer space. Although ABR can improve total link utilization by statistical resource sharing, it brings the risk of potential network congestion when many applications compete for network resources. Proper congestion control must be in place to ensure that the network resource can be shared in a fair manner and that performance objectives such as cell loss ratio can be maintained. The bursty nature of data applications makes the ABR congestion control more challenging and complex.

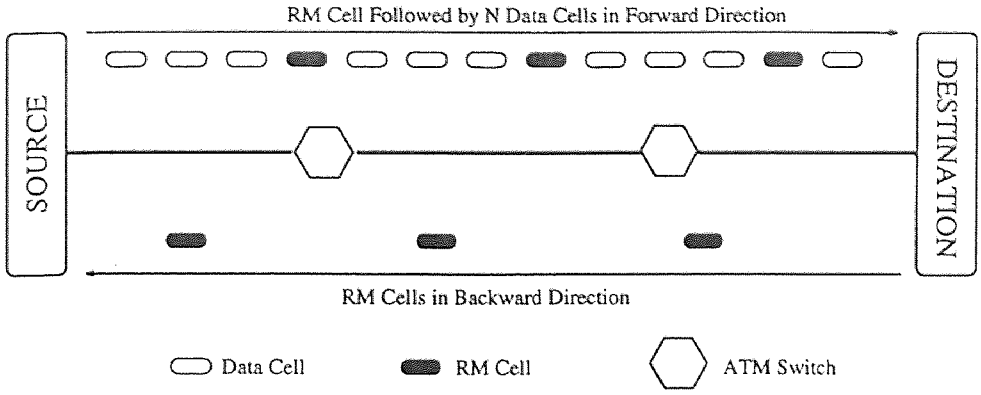

Figure 3.1 Rate-Based End-to-End Congestion Control Scheme

3.1 ATM Forum Congestion Control Framework for ABR Service

The ATM Forum has received many congestion control schemes over the past few years. After a considerable debate, the ATM Forum has adopted a rate-based congestion control approach, without making commitments to any particular switch algorithm. The endorsement for the rate-based approach came about because of the architectural flexibility it provided, including a wide range of possible switch implementations with varying degrees of cost and complexity.

In the ABR service, the source adapts its rate to changing network conditions. Information about the state of the network, such as bandwidth availability, state of congestion, and impending congestion, is conveyed to the source through special probe cells called Resource Management Cells (RM-cells). The scheme is based on a closed-loop, "positive feedback" rate control principle. Here, the source only increases its sending rate for a connection when given an explicit positive indication to do so, and in absence of such a positive indication, continually decreases its sending rate.

The source generates RM cells in proportion to its current data cell rate. The destination will turn around and send back the RM cell to the source in the backward direction. RM cells which can be examined and modified by switches in both forward and backward directions carry the feedback information of the state

of congestion and the fair rate allocation. The typical operation of the rate-based control is illustrated in Figure 3.1. Details of the RM cell format can be found in [1]. A switch shall implement at least one of the following methods to control congestion at queuing point: (1) Explicit Forward Congestion Indication (EFCI) marking in which the switch may set the EFCI state in the data cell headers, and most of the first generation switches had implemented this mechanism before the RM cell was fully defined; (2) Relative rate marking in which the switch may set the congestion indication (CI) bit or the no increase (NI) bit in forward and/or backward RM cells; (3) Explicit rate marking in which the switch may reduce the explicit rate (ER) field in forward and/or backward RM cells. Switches that implement options (1) and (2) are known as binary switches which can reduce implementation complexity but may result in unfairness, congestion oscillation, and slow congestion response. Switches that implement option (3) are generally called ER switches which require sophisticated mechanisms in place at switches for the computation of a fair share of the bandwidth.

The standard-defined source and destination behaviors allow the inter-operation of the above three options. Once the source has received permission, it begins scheduling cells for transmission at the rate of Allowed Cell Rate (ACR). The ACR is initially set to the Initial Cell Rate (ICR) and is always bounded between the Minimum Cell Rate (MCR) and the Peak Cell Rate (PCR) which are specified by the application at the call set up. Transmission of data cells is preceded by sending a RM cell. The source will continue to send RM cells, typically after every N_{RM} data cells. The source places the ACR value in the Current Cell Rate (CCR) field of the RM cell, and the rate at which it wishes to transmit cells (usually the PCR value) in the ER field. The RM cells traverse forward through the network and the destination turns around the RM cells in the backward direction. Intermediate switches on the path notify the source of congestion by marking the EFCI bit in

the data cells, CI or NI bit in the RM cell, and/or reducing the ER value in the RM cells. Upon receiving the returned RM cell, the source should adapt its ACR to the information carried in the RM cell. If CI is not set, then the source may linearly increase its ACR by a fixed increment (RIF * PCR), where Rate Increase Factor (RIF) is determined at the call setup. This increase can reach up to the ER value in the RM cell, but should never exceed the PCR. If CI is set, then the source must exponentially decrease its ACR by an amount greater than or equal to a proportion of its current ACR, (RDF * ACR), where Rate Decrease Factor (RDF) is also determined at the call setup. The factors RIF and RDF control the rate at which the source increases and decrease its rate, respectively. If ACR is still greater than the returned ER, the source must further decrease its ACR to the returned ER value, although it should never be below the MCR. If NI is set, the source should observe the CI and ER fields in the RM cell, but is not allowed to increase the ACR above its current value. Cell flow regulation allows for end-to-end, segment-by-segment, or link-by-link. Additional operational details are beyond the scope of this thesis and can be found in [1].

Based on the ATM Forum's rate-based congestion control framework, many ABR algorithms with different performance and implementation complexity have been proposed in the past few years. Among these algorithms, the following three representative algorithms are studied for TCP over ATM in this thesis: Explicit Forward Congestion Indication (EFCI), Explicit Rate Indication for Congestion Avoidance (ERICA), Fast Max-Min Rate Allocation (FMMRA) algorithm.

3.2 Explicit Forward Congestion Indication Algorithm

In an EFCI-based switch, if congestion is experienced in an intermediate switch during the connection, the EFCI bit in the data cell will be set to 1 to indicate congestion. The CI field in the RM cell is set by the destination if the last received

data cell has the EFCI field set and is returned back to the source. If the source receives an RM cell with no congestion indication, the source is allowed to increase its rate. If the congestion indication bit is set, the source should decrease its rate. The parameters RIF and RDF control the rate by which the source increases or decreases its rate.

The EFCI-based switches suffer from a phenomenon called the beat-down problem. In a network using only EFCI-based switches , where a congested switch marks the EFCI bit of the data cell, sources traveling more hops have a higher probability of getting their cells marked than those traveling fewer hops. As a result, it is unlikely that these long-hop connections are able to increase their rates and consequently are beaten down by these short-hop connections.

There are several alternatives that can alleviate the unfairness problem in binary feedback schemes. The first alternative is to provide each connection or a group of connections with its own separate queue. Such a per-connection queuing mechanism makes it possible to determine the congestion status of a connection in isolation from the rest of connections. Because only those connections that cause the congestion of their associated queues are subject to congestion marking, the beat-down problem in the basic binary scheme can be eliminated. Thus, this isolation ensures fair access to buffer space and bandwidth among all competing connections. The second feasible alternative is to provide selective or intelligent marking on a common FIFO queuing mechanism when congestion takes place. In this alternative, a switch computes a fair share and a connection is asked to decrease its rate by marking the EFCI bit only if its current cell rate is above the computed fair share. Although this approach can provide a good solution for the beat-down problem, it is generally slow in response to congestion as it may take several round trip times to alleviate congestion or utilize any unused bandwidth.

3.3 Explicit Rate Indication for Congestion Avoidance Algorithm

The ERICA algorithm proposed in [6] is an approximation fair rate computation and congestion avoidance algorithm. The switch periodically monitors the load on each link and determines a load factor, (Z), the available capacity, and the number of currently active VCs (N). The load factor is calculated as the ratio of the measured input rate at the port to the target capacity of the output link.

$$Z = \frac{\text{ABR Input Rate}}{\text{ABR Capacity}}, \tag{3.1}$$

where

$$\text{ABR capacity} = \text{Target Utilization (U)} * \text{Link Bandwidth} \tag{3.2}$$

The Input Rate is measured over an interval called the switch averaging interval. The above steps are executed at the end of the switch averaging interval. Target utilization (U) is a parameter which is set to a fraction of the available capacity. Typical values of target utilization are 0.9 and 0.95.

The load factor, Z, is an indicator of the congestion level of the link. High overload values are undesirable because they indicate excessive congestion, so are low overload values which indicate link underutilization. The optimal operating point is at an overload value equal to one. The goal of the switch is to maintain the network at unit overload. The fair share of each VC, $FairShare$, is also computed as follows:

$$FairShare = \frac{\text{ABR Capacity}}{\text{Number of Active Sources}} \tag{3.3}$$

The switch allows each source sending at a rate below the $FairShare$ to rise to $FairShare$ every time it sends a feedback to the source. If the source does not use all of its $FairShare$, then the switch fairly allocates the remaining capacity to sources which can use it. For this purpose, the switch calculates the quantity:

$$VCShare = \frac{CCR}{Z} \tag{3.4}$$

If all VCs change their rate to their VCShare values, then in the next cycle, the switch will experience unit overload (Z equals one). Hence, VCShare aims at bringing the system to an efficient operating point, which may not necessarily be fair, and $FairShare$ allocation aims at ensuring fairness, possibly leading to overload (inefficient operation). A combination of these two quantities is used to rapidly reach optimal operation as follows:

$$\text{ER Calculated} = Max(FairShare, VCShare) \tag{3.5}$$

Sources are allowed to send at a rate of at least FairShare within the first round-trip. This ensures minimum fairness between sources. If the VCShare value is greater than the FairShare value, the source is allowed to send at VCShare so that the link is not underutilized. This step also allows an unconstrained source to proceed towards its Max-Min rate. The previous step is one of the key innovations of the ERICA scheme because it improves fairness at every step, even under overload conditions.

Since every output port is a queuing point through which a VC passes, every source ought to send at no more than the ER calculated at its bottleneck queuing point. To ensure that the bottleneck ER reaches the source, each switch computes the minimum of the ER it has calculated as above and the ER value in the RM cell. This value is inserted in the ER field of the RM cell.

$$\text{ER in RM Cell} = Min(\text{ER in RM cell}, \text{ER Calculated}) \tag{3.6}$$

The ERICA algorithm can operate in a congestion-avoidance state, is insensitive to parameter variations, and proves to be very robust. The rate converges

very quickly with little oscillation. This algorithm, however, has some fundamental limitations in achieving desired fairness for all the connections and buffer requirements. In some cases, a connection that gets started late although it gets its equal link share it does not get the max-min rate. Furthermore, during transient periods, and if the desired target utilization is set close to the full link rate, the queue grows rapidly and results in heavy cell loss. Although not discussed here, there exist many extensions and modifications of this algorithm. These extensions, with added complexity, try to eliminate some of the problems found in the basic ERICA algorithm. For complete details, the reader is referred to [5], [6].

3.4 Fast Max-Min Rate Allocation Algorithm

Fast Max-Min Rate Allocation (FMMRA) [2] algorithm is based on measurement of available capacity and exact calculation of fair rates. An additional important feature of this algorithm is that it is not sensitive to inaccuracies in CCR values.

Each ABR queue in the switch computes a rate that it can support. This rate is referred to as the advertised rate, (r). The advertised rate along with the ER field in the RM cell are used to determine if the connection is bottlenecked elsewhere. If a connection cannot use the advertised rate, it is marked as a bottlenecked elsewhere and its bottleneck bandwidth is recorded.

The ER field in the RM cell is read and marked in both directions to speed up the rate allocation process. The bi-directional ER marking in this algorithm makes it possible for downstream switches to learn bottleneck bandwidth information or upstream switches, and the upstream switches to learn bottleneck bandwidth information of the downstream switches. Many of the proposed algorithms mark the ER field only in the backward direction. Because of the uni-directional ER marking, switches closer to the source get more accurate ER information than those closer to the destination. As a result, this may result in slower response to congestion. This

bi-directional updating of ER in the RM cell plays a significant role in drastically reducing the convergence time of max-min fair rate allocation process.

This algorithm uses the load factor and the ER to compute an exponential running average of the maximum value of ER, denoted as ER_{max}. This operation is shown below:

$$ER_{max} = (1 - \alpha)ER_{max} + \alpha * Max(ER, \frac{ER_{max}}{LF}), \tag{3.7}$$

where α is an averaging factor and can be set to 1/8. The computation of ER_{max} is done in the backward direction, and it reflects the advertised rate after taking into consideration of the load. The load factor reflects how well the ABR bandwidth is utilized. Based on the level of congestion, which is determined as a function of the queue length and the load factor, the ER field in the RM cell (both forward and backward) is updated according to

$$ER = Min(ER, max(r, (1 - \beta) * ER_{max})), \tag{3.8}$$

where β is a single bit value indicating if the connection is bottlenecked elsewhere. The use of the advertised rate along with the ER_{max} helps the fair rate to converge faster, and to adjust the fair share according to load variations in the network, in order to achieve full link utilization.

Selective measures are taken in updating the ER field in order to control the queue growth to prevent potential cell loss. If the queue length reaches a low threshold QT and the load factor $LF > 1$, only the advertised rate is used in marking the ER field in the RM cell. Under severe congestion, the ER_{max} is set to the advertised rate. This ensures that whenever a potential for congestion is detected, even if some connections are idle, non-idle connections are not given to any extra bandwidth, allowing the queue to drain. Furthermore, if the queue length exceeds a high threshold DQT (indicating heavy congestion), the target utilization

factor is reduced by a target rate reduction factor, until the queue length drops below the low threshold QT. This technique allows the queue to drain and operate at a desired queue length level, whenever the switch is heavily congested, for example, when many new connections are established.

CHAPTER 4

SIMULATIONS AND OBSERVATIONS

4.1 Simulation Model

In this section we describe the network configuration and parameters used in our simulations of TCP traffic over ATM networks. Figure 4.1 shows the network configuration used in our simulations consisting of two switches, N TCP sources and destinations, and one background VBR traffic. All links run at 155 Mbps. This configuration has a single bottleneck link in the "Backbone" shared by N ABR sources and one VBR source. All traffic is unidirectional. A large infinite file transfer application runs on top of TCP for sources. Values of $N = 2$ and 5 are presented here.

Configurations with the "Backbone" link length of 1km represent typical Local Area Network (LAN) situations. The VBR background traffic is running at 100Mbps rate which is 2/3 of the total bandwidth. It starts at t=300 ms and is an ON-OFF source. We choose two different ON-OFF frequency at 10 ms and 100 ms in our simulations. The purpose is to find out how the background VBR ON-OFF frequency affects the switch buffer requirement for different ABR congestion control algorithms. At the switch, VBR is given higher priority than ABR. If a VBR cell arrives at the switch, it will be scheduled for output before any waiting ABR cells

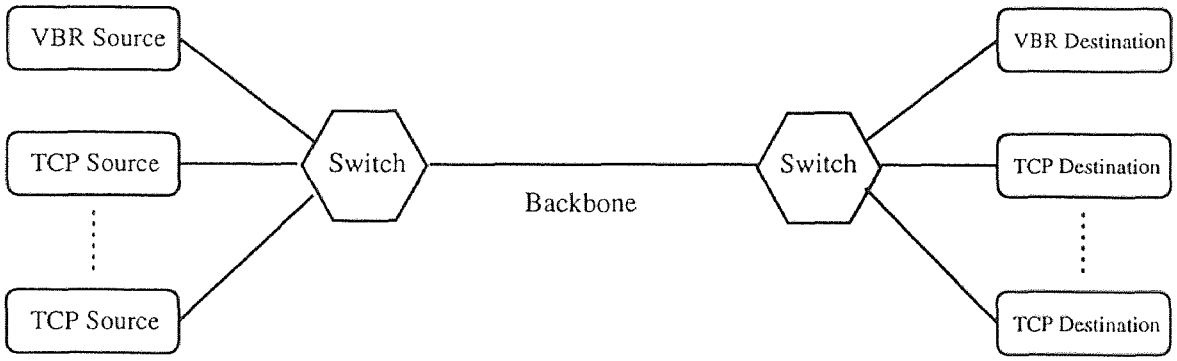

Figure 4.1 Network configuration

are scheduled. Because the link bandwidth is limited, after the VBR is activated at t=300 ms, the switch will experience a congestion. In order to avoid buffer overflow, it then sends RM cells back to the source, and inform the ABR source to reduce their transmission rate. Different ABR congestion control algorithms will result in different buffer requirements and TCP performance.

The simulation tool used here is NIST ATM Network Simulator. The simulator is developed at the National Institute of Standards and Technology (NIST). Its purpose is to provide a flexible testbed for studying and evaluating the performance of ATM networks. It is a tool that gives the user an interactive modeling environment with a graphical user interface. NIST developed this tool using both C language and the X Window System running on a UNIX platform; event handling and some of the user interface routines come from a network simulator developed at MIT.

The ATM Network Simulator allows the user to create different network topologies, set parameters of component operation, and save/load different simulated configurations. While the simulation is running, various instantaneous performance measures can be displayed in graphical/text form on the screen or saved to files for subsequent analysis.

We implement ERICA and FMMRA algorithm in the ATM switch component and fix some bugs in the TCP component that caused erroneous results in TCP over ATM simulations. In the NIST simulator, the TCP model is based on Jacobson's congestion control mechanisms [3], exponential back-off, enhanced round-trip time (RTT) estimation based on both the mean and variance of the measured RTT. Since RTT values in our simulation configuration is in the order of milliseconds, the coarse-grain timer used in Unix TCP implementations (typically, a granularity of 500 ms) would make comparison of the schemes difficult. To avoid the anomalies due to coarse-grain timers, we used double-precision floating-point arithmetic in the RTT estimation algorithm.

4.2 ABR Parameters and TCP Parameters

We use an infinite source mode at the application layer running on top of TCP. This implies that TCP always has a packet to send as long as its window permits it. Our goal is to explore the possible limitations that ATM networks place on the performance of the TCP protocol. The infinite source mode best meet our requirement. The source TCP parameter values are chosen as follow:

- TCP maximum segment size = 9180 bytes
- Mean packet processing time = 200 uSec
- Packet processing time variation = 50 uSec
- Receive window size = 64 K bytes
- Bit rate = 155 Mbit/s
- Delay-ack timer = 0

In our simulations, the "fast retransmit and recovery" mechanism has not been implemented.

The ATM source end system (SEC) parameter of ABR are selected to maximize the Available Cell Rate (ACR). The values of SES parameters are:

- Peak cell rate = 155 Mbits
- N_{rm} = 32 cells
- M_{rm} = 2 cells
- ICR = 10 Mbits
- MCR = 0
- CRM = TBE/N_{rm} = 20 cells

- CDF = 0.5
- TRM = 100 ms
- TCR = 0.00424 Mbits

For the ERICA and FMMRA algorithms, RDF = 1/512, RIF =1; For binary scheme EFCI algorithm, RDF = 1/16, RIF = 0.1.

These SES parameters are defined and explained in reference [2].

4.3 Simulation Results and Observations

In our simulations, the following TCP and ABR performance metrics are evaluated:

- ABR queue length in switch at the congestion point;
- TCP effective throughput;
- Link utilization at the congestion point;
- TCP round trip time (RTT);
- Source Allowed Cell Rate (ACR);

These performance metrics can be used to evaluate and compare the performance of different ABR congestion control algorithms. A good ABR congestion control algorithm should exhibit the following merits:

1. Keep the ABR queue length in switch at the congestion point as small as possible.
2. Fairly allocate the bandwidth among TCP sources, and maximize the individual TCP source's effective throughput.
3. Keep the link utilization as high as possible, the ideal case is 100 percent.

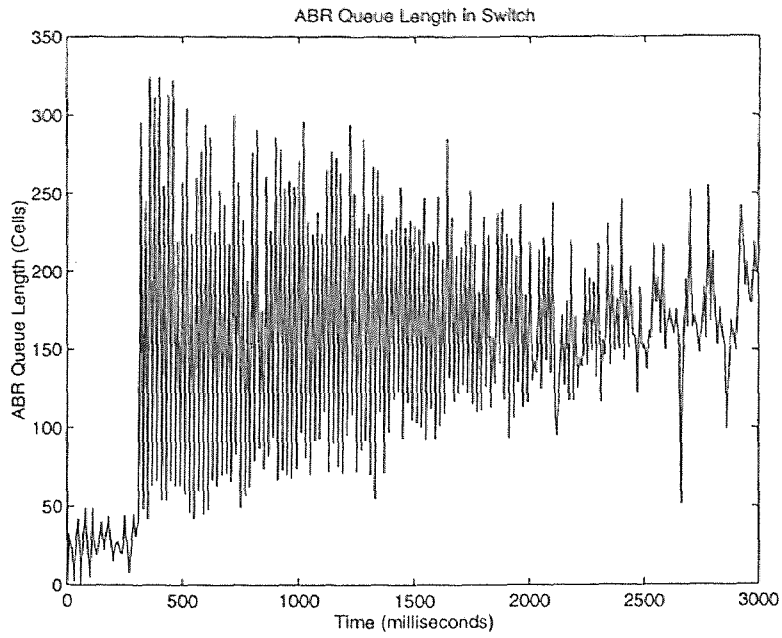

Figure 4.2 ABR queue length vs. time using the FMMRA algorithm, 2 TCP connections

4. Source ACR should reflect the changes of the network condition.

From the simulation results, the following observations are obtained:

1. Among the three ABR congestion control algorithms, FMMRA has the best performance in keeping the ABR queue length smallest under different network configurations. Figure 4.2 to Figure 4.7 show the simulation results of ABR queue length vs. time for all of the three algorithms. Figure 4.2 to Figure 4.4 are for two TCP source and one VBR background traffic. Figure 4.5 to Figure 4.7 are for five TCP source and one VBR background. From the figures we can find that under same network configurations, ATM switches with FMMRA algorithm implemented has the smallest ABR queue length, implying that FMMRA algorithm has the least buffer requirement for zero cell lost.

2. FMMRA has the best performance in fairly allocating available network bandwidth to individual TCP source. Figure 4.8 to Figure 4.10 are the results of

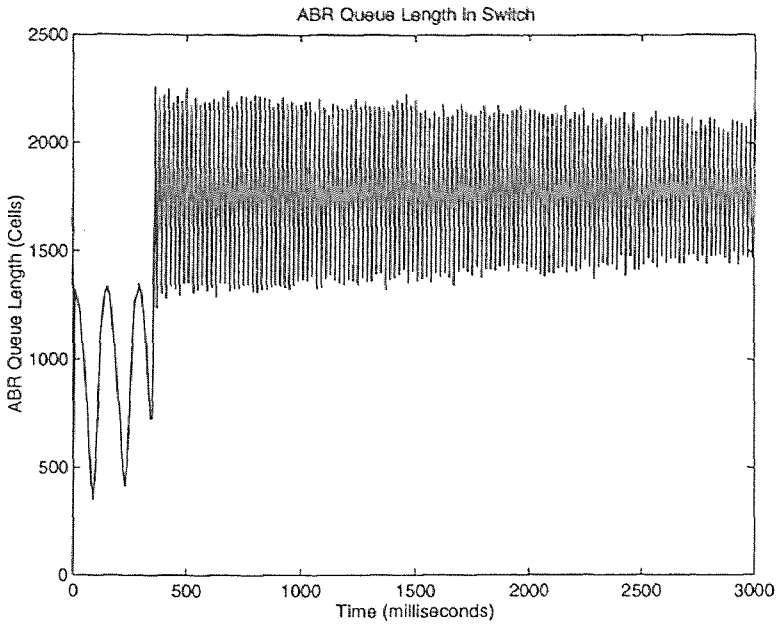

Figure 4.3 ABR queue length vs. time using the EFCI algorithm, 2 TCP connections

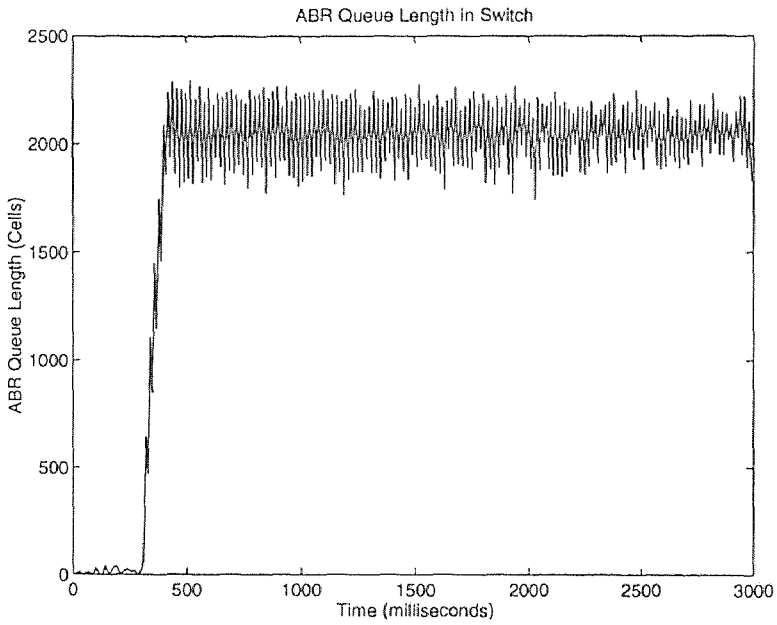

Figure 4.4 ABR queue length vs. time using the ERICA algorithm, 2 TCP connections

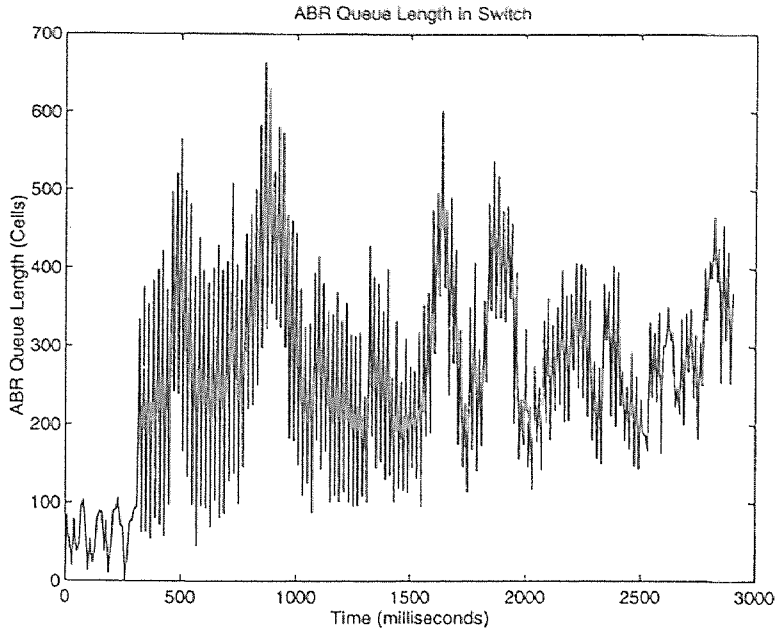

Figure 4.5 ABR queue length vs. time using the FMMRA algorithm, 5 TCP connections

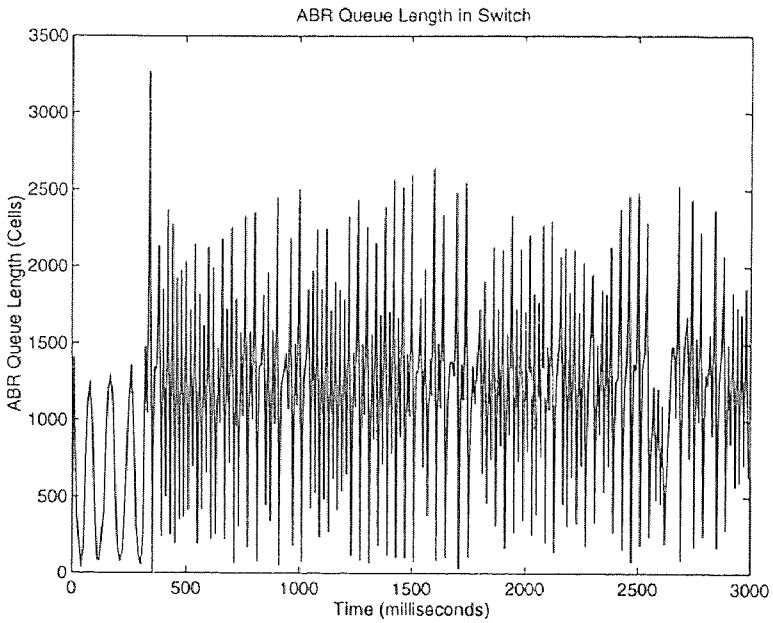

Figure 4.6 ABR queue length vs. time using the EFCI algorithm, 5 TCP connections

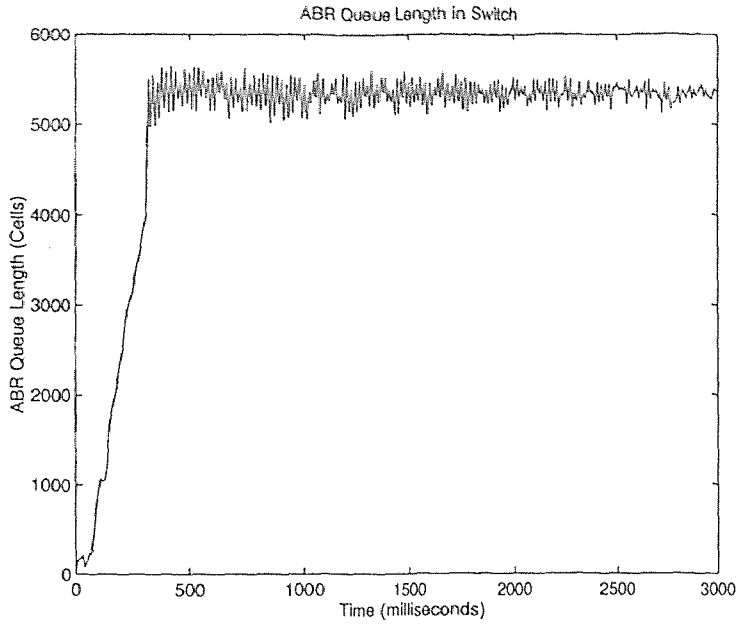

Figure 4.7 ABR queue length vs. time using the ERICA algorithm, 5 TCP connections

effective TCP throughput vs. time for the three algorithms. There are five TCP source and one VBR background traffic with 10 milliseconds ON and 10 millisecond OFF bursty nature. From these figures, we can find that both ERICA and EFCI lead to some unfairness among the individual TCP source during the transient period. Some source get higher throughput than the others. FMMRA algorithm keep every TCP source has the same effective throughput.

3. FMMRA has the fastest and most accurate response to the source ACR rate according to the changes of network traffic. Figure 4.11 to Figure 4.13 show results of ABR source Allowed Cell Rate (ACR) vs. time for the three algorithms. There are two TCP source and one VBR background traffic with 100 milliseconds ON and 100 millisecond OFF bursty nature. From these figures, FMMRA has the fastest and most accurate response to the changes of avaliable network ABR capacity.

4. Link utilization and ABR queue length are two way trade off. High ABR queue length can result in high link utilization. From the simulation results we find

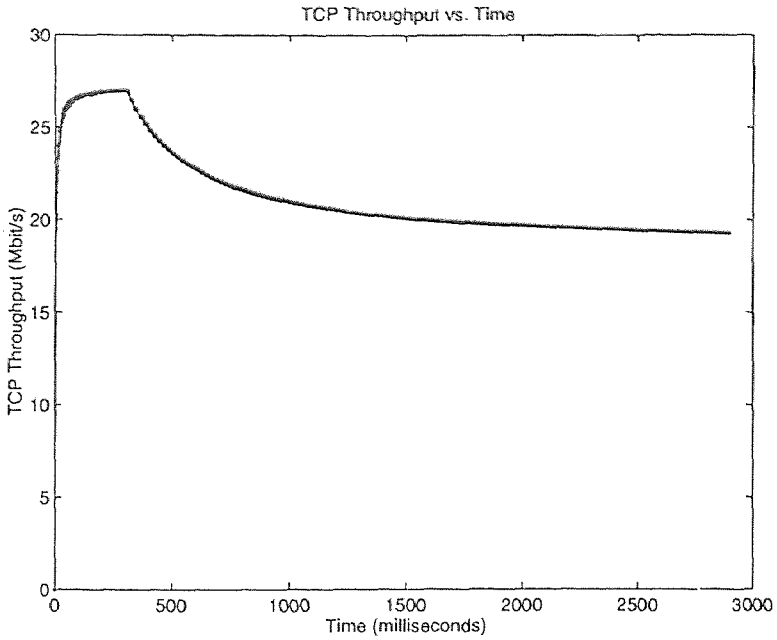

Figure 4.8 TCP effective throughput vs. time using the FMMRA algorithm, 5 TCP connections

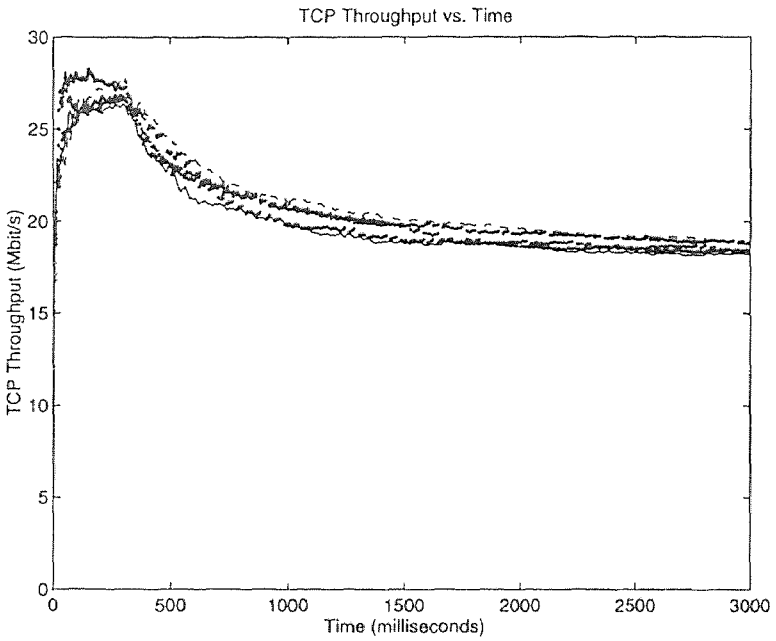

Figure 4.9 TCP effective throughput vs. time using the EFCI algorithm, 5 TCP connections

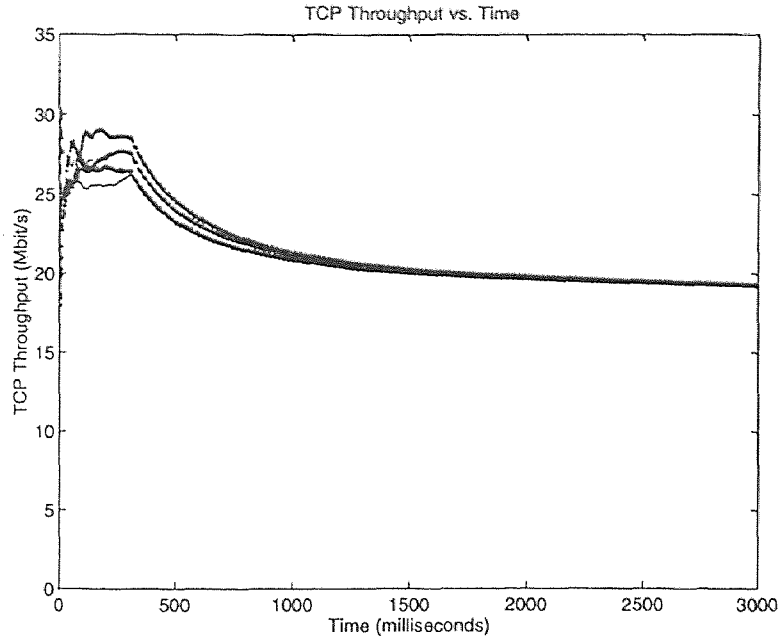

Figure 4.10 TCP effective throughput vs. time using the ERICA algorithm, 5 TCP connections

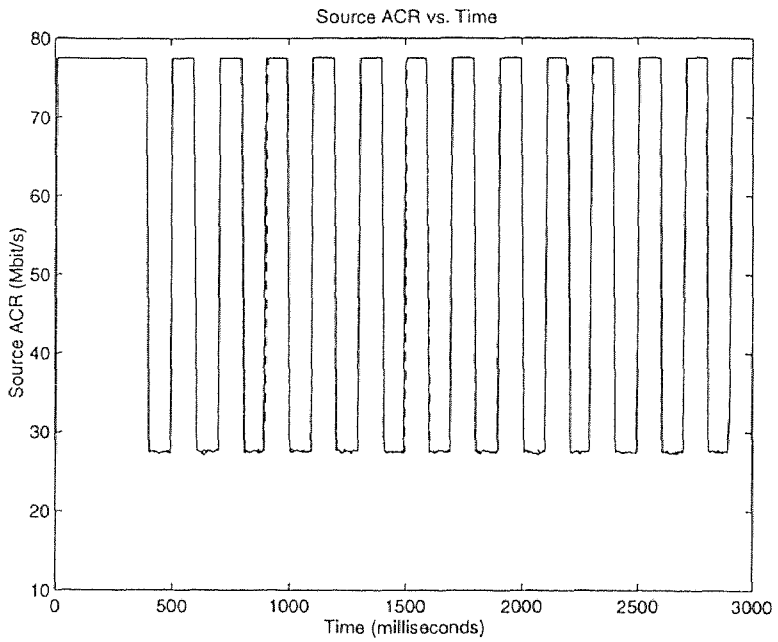

Figure 4.11 TCP source Allowed Cell Rate(ACR) vs. time using the FMMRA algorithm, 2 TCP connections

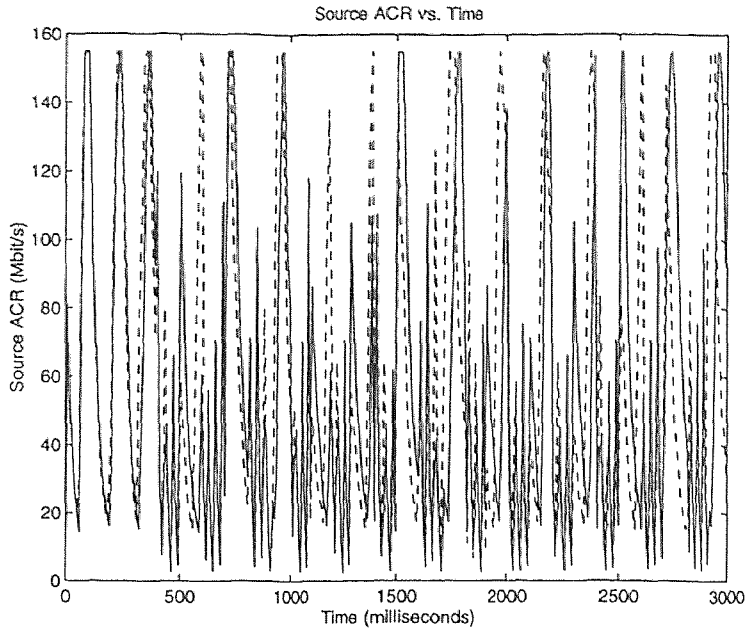

Figure 4.12 TCP source Allowed Cell Rate(ACR) vs. time using the EFCI algorithm, 2 TCP connections

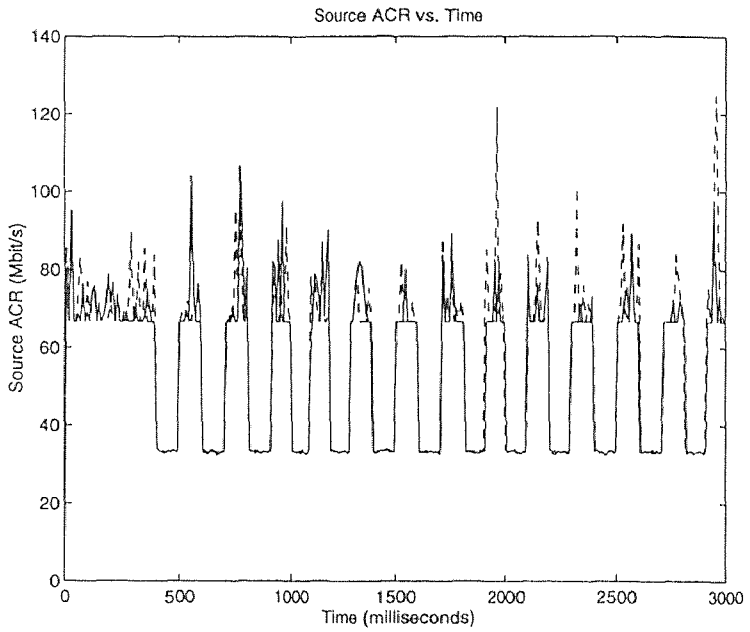

Figure 4.13 TCP source Allowed Cell Rate(ACR) vs. time using the ERICA algorithm, 2 TCP connections

that EFCI and ERICA has relative bigger ABR queue length than FMMRA, so they have higher overall link utilization. However in real world, the buffer space in an ATM switch is limited. Keeping the ABR queue length as small as possible is much more important than getting a high link utilization. Figure 4.14 to Figure 4.16 show the effective TCP throughput of the three algorithm under the limited buffer size condition. From these figures, we can find that with switch buffer size of 1500 cells , FMMRA can achieve zero cell loss, hence achieving the highest throughput. Although we can choose proper High Queue length Threshold (HQT) and RIF, RDF parameter to control the cell lost due to congestion, there is a severe unfairness of bandwidth allocation among TCP connections for the EFCI algorithm. This is because EFCI only provides source with binary feedback instead of calculated fair rate. Once a TCP source experiences a cell loss and enters the slow start stage, other TCP sources take this advantage and increase their rates. When the next congestion occurs, this TCP source compete with others at an unfavorite situation.

5. If there is no queue length monitoring and control mechanism implemented in the ABR congestion control algorithm, the ABR queue length in a switch will increase unboundedly under the following situations: large number of TCP connections, high frequency background VBR traffic, and long round trip delay. Figure 4.17 to Figure 4.18 show the ABR queue length for ERICA and FMMRA algorithm with 10 TCP connections and one high frequency VBR background traffic. While the queue length using FMMRA reaches more than 3000 cells, that using ERICA is above 7000 cells. The large ABR queue length produces long delay and makes the ATM switch expensive and complex. Some kind of queue length control mechanism is necessary.

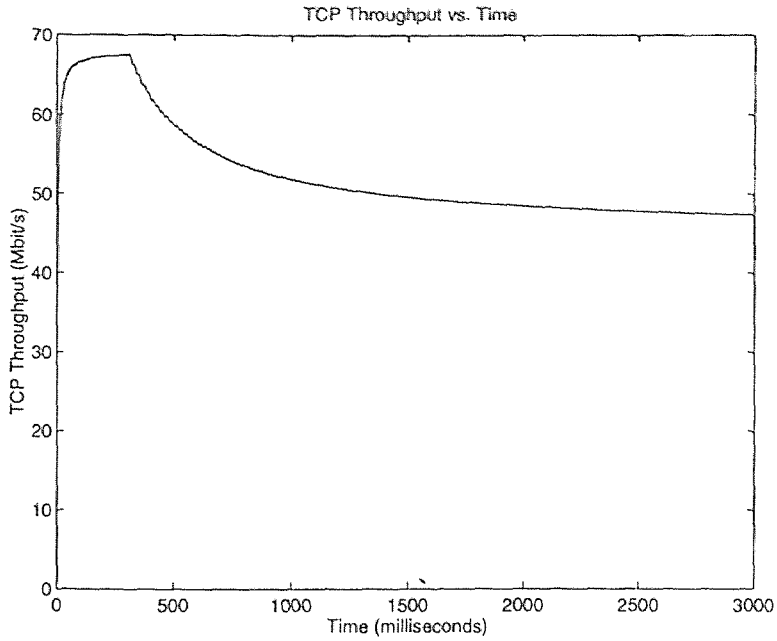

Figure 4.14 Effective TCP throughput vs. time using the FMMRA algorithm with limited buffer size of 1500 cells, 2 TCP connections

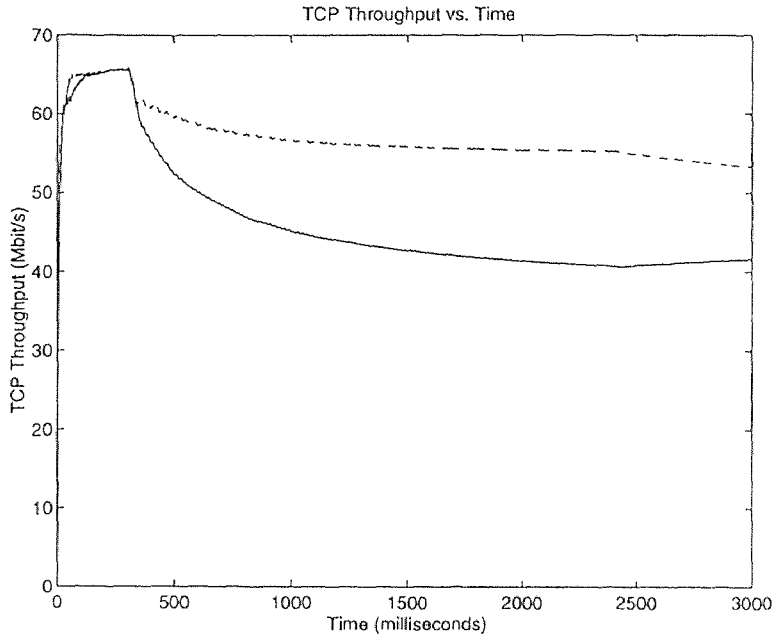

Figure 4.15 Effective TCP throughput vs. time using the EFCI algorithm with limited buffer size of 1500 cells, 2 TCP connections

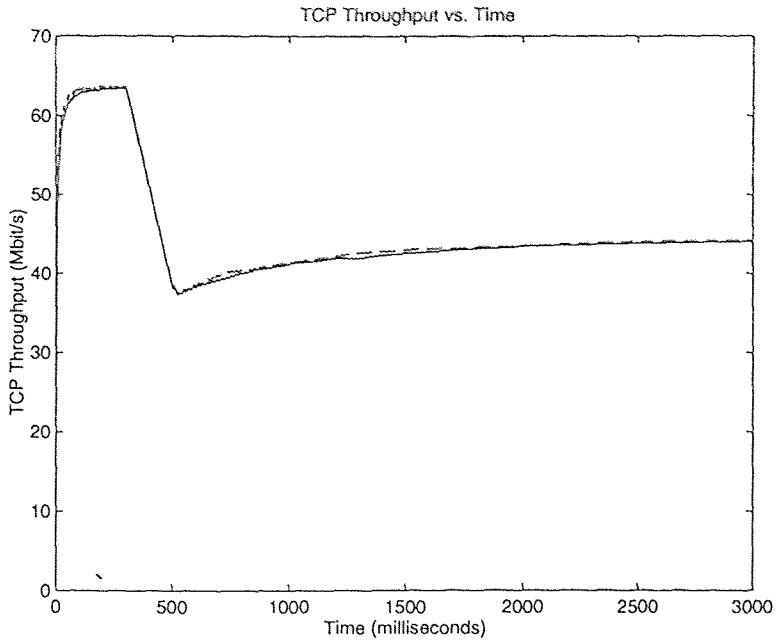

Figure 4.16 Effective TCP throughput vs. time using ERICA algorithm with limited buffer size of 1500 cells, 2 TCP connections

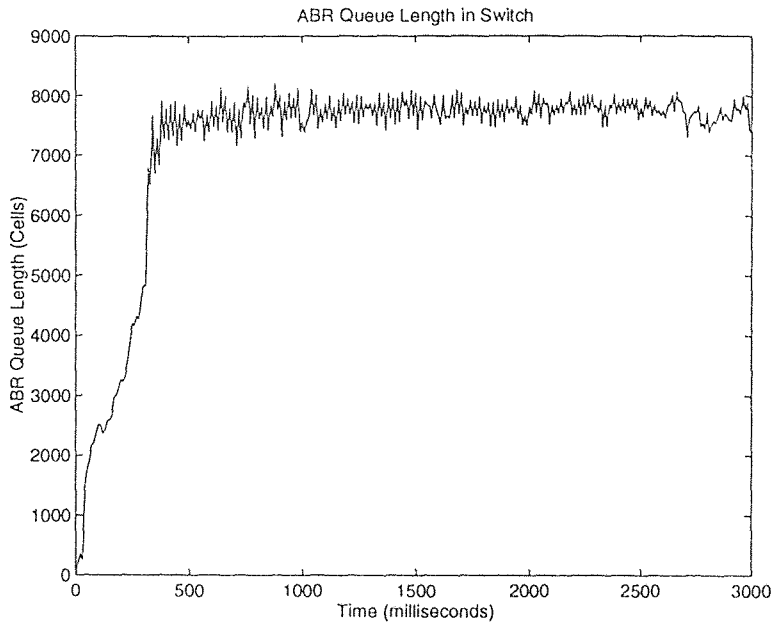

Figure 4.17 ABR queue length vs. time using the ERICA algorithm, 10 TCP connections

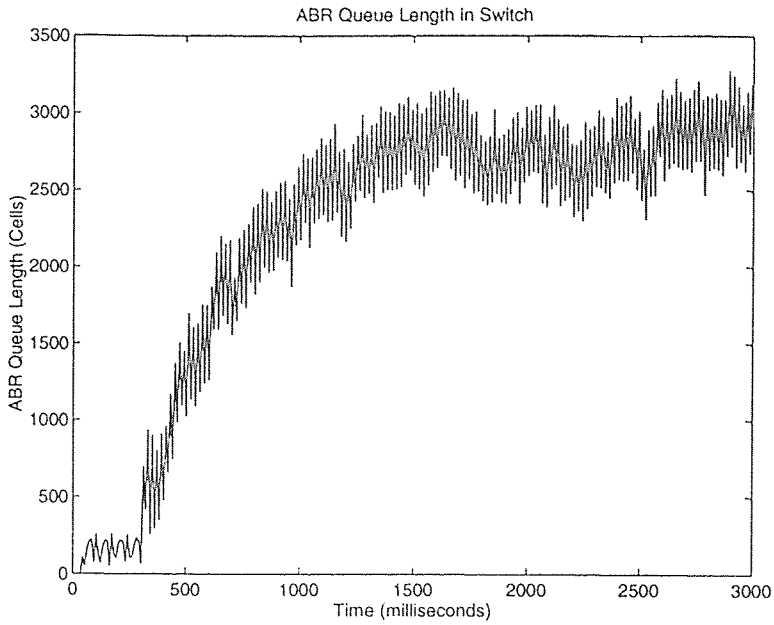

Figure 4.18 ABR queue length vs. time using the FMMRA algorithm, 10 TCP connections

CHAPTER 5

CONCLUSIONS

In this thesis we have presented and analyzed simulation results of TCP/IP traffic running over ATM network with different ABR congestion control schemes. From simulation results and the analysis, among the EFCI, ERICA and FMMRA algorithms, under severe network congestion conditions, FMMRA exhibits the following favorable features compared to the other two algorithms:

- The least buffer requirement for zero cell lost;
- Fairly allocate available network bandwidth to individual TCP connection;
- Adjust source ACR rates fast and accurately according to the changes of network traffic.
- Under limited switch buffer size situation, FMMRA algorithm achieves the best TCP throughput.

Also based on the simulation results, in order to achieve an acceptable TCP performance and affordable switch buffer size requirement for the performance, some kind of queue length monitoring and control mechanism should be implemented in the ABR congestion control algorithm.

REFERENCES

1. "The ATM Forum Traffic Management Specification, Version 4.0," *The ATM Forum Specification*, April 1996.

2. A. Arulambalam, X. Chen, and N. Ansari, "Allocating Fair Rates for Available Bit Rate Service in ATM Networks," *IEEE Communications Magazine*, vol. 34, no. 11, pp. 92–100, Nov. 1996.

3. V. Jacobson, R. Braden, and D. Borman, "Congestion Avoidance and Control,," *Proceedings of Sigcomm'88*, pp. 314–329, 1988.

4. V. Jacobson, R. Braden, and D. Borman, "TCP extensions for high performance," *RFC 1323*, Nov. 1992.

5. R. Jain, S. Kalyanaraman, R. Goyal, S. Fahmy, and R. Viswanthan, "ERICA Switch Algorithm: A Complete Description," *The ATM Forum Contribution 96-1172*, Aug. 1996.

6. R. Jain, S. Kalyanaraman, and R. Viswanthan, "Explicit Rate Indication for Congestion Avoidance," *The ATM Forum Contribution*, Nov. 1995.

7. S. Kalyanaraman, R. Jain, R. Goyal, S. Fahmy, F. Lu, and S. Srinidhi, "Performance of TCP/IP over ABR," *Proceedings of Globecom'96, London*, vol. 1, pp. 468–475, Nov. 1996.

8. M. Laubach, "Classical IP and ARP over ATM," *RFC 1577*, Jan. 1994.

9. A. Mankin and K. K. Ramakrishnan, "Gateway Congestion Control Survey,," *RFC 1254*, 1991.

10. 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.