ELSEVIER

Computer Networks and ISDN Systems 29 (1998) 2091–2102

COMPUTER NETWORKS and ISDN SYSTEMS

TCP/IP traffic over ATM networks with FMMRA ABR flow and congestion control [1]

Liping An [a,*], Nirwan Ansari [a,2], Ambalavanar Arulambalam [b,3]

[a] *Center for Communications and Signal Processing, Electrical and Computer Engineering Department, New Jersey Institute of Technology, Newark, NJ 07102, USA*

[b] *Bell Laboratories, Lucent Technologies, Murray Hill, NJ 07974, USA*

Abstract

In this paper, 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. The FMMRA algorithm is shown to exhibit the favorable features of least buffer requirement, fair bandwidth allocation to TCP connections, fast and accurate ACR rate adjustment according to the changes of network traffic, and the highest effective TCP throughput. © 1998 Elsevier Science B.V.

Keywords: TCP/IP; Congestion control; ABR; EFCI; ERICA; FMMRA

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. Data applications can tolerate transmission delay and delay jitters, but they are usually very sensitive to data lost. The well known TCP packet segmentation problem—the whole packet is discarded at the end system even if only a single cell is dropped by a congested switch —causes retransmission of the entire packet by the TCP layer. The simulation study in [1] shows that the performance of TCP/IP over ATM networks without ATM level congestion control is quite poor when the switch is congested and begins dropping cells.

In order to achieve high bandwidth utilization and 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 to attain an acceptable performance of upper layer protocols running over ABR service. After a considerable debate, a rate-based

* Corresponding author. E-mail: lipingan@hotmail.com.

[1] This work was supported in part by AT&T/Lucent Technologies Foundation through a special purpose grant.

[2] E-mail: ang@njit.edu.

[3] E-mail: arul@lucent.com.

flow control framework for the ABR service has been specified by the ATM Forum [2]. 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 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 was reported in [3]. 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 paper, 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. From these simulation results, we have gained some insights on the interaction between the TCP and ATM layer congestion control mechanism.

2. TCP traffic over ATM networks

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.

The congestion control mechanisms used in TCP are based on a number of ideas proposed by Jacobson [4]. Many improvements have been proposed over the years to make TCP more suitable for high speed large delay networks [5] 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 recovery algorithm which tries to avoid performing slowstart 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 consists 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 term slow-start may be a misnomer because the actual window size is increased exponentially. It takes only $\log_2 N$ round trips before TCP can send N segments. If there are multiple TCP connections connected to the same ATM 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.

Once the congestion window reaches the slow start threshold, TCP enters the congestion avoidance phase, and slows down the rate of increment. The purpose is to probe for additional available bandwidth in the network and at the same to avoid causing additional congestion.

The retransmission and exponential backoff mechanism retransmits the packet after a 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.

2.1. 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 these 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 which cannot be recognized by the network 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 networks. 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 amount of 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 [6] presented a simulation result showing that the TCP effective throughput over plain ATM networks 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 cell 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 congestion can be reduced by reducing the TCP packet length and window size, and increasing the switch buffer size, congestion caused by the 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 implemented in the ATM switches is necessary.

3. ABR rate-based flow control mechanism

Although ABR can improve total link utilization by dynamically and efficiently sharing bandwidth, it may cause 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.

3.1. ATM forum congestion control framework for ABR service

In the ABR service, the source adapts its rate to 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 the 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 Fig. 1. Details of the RM cell format can be found in [2]. A switch shall implement at least one of the following methods to control congestion at a 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. Details of the ATM Forum congestion control framework for ABR service are beyond the scope of this paper and can be found in [2].

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 paper: Explicit Forward Congestion Indication (EFCI), Explicit Rate Indication for Congestion Avoidance (ERICA), and 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 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 con-

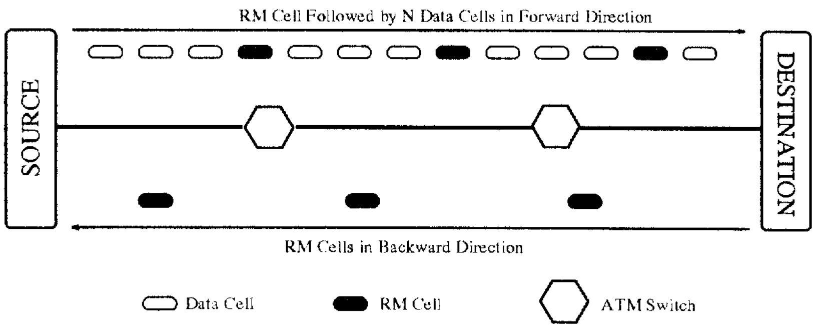

RM Cell Followed by N Data Cells in Forward Direction
SOURCE
DESTINATION
RM Cells in Backward Direction
Data Cell
RM Cell
ATM Switch

Fig. 1. Rate-based end-to-end congestion control scheme.

gested 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 short-hop connections.

3.3. Explicit rate indication for congestion avoidance algorithm

The ERICA algorithm proposed in [7] is an approximation fair rate computation and congestion avoidance algorithm. A switch is opted to operate at a congestion avoidance status by specifying a less than 100% target link utilization factor, upon which the available bandwidth for ATM connections is determined. The target link utilization is normally chosen to be 0.9, implying that only 90% of the total bandwidth is available to ATM connections and the remaining 10% is used to drain the ABR queue when sustained congestion occurs.

Instead of directly calculating the max-min fair rate, the switch calculates the fair-share rate for each VC connection. If any VC connection cannot use the fair-rate due to bottlenecked elsewhere, in the next round trip time, the switch will experience a traffic load below the target link utilization. When under utilization is detected at a switch, the unused bandwidth is reallocated to the unbottlenecked VC connections, and the traffic load will hopefully, after several round trip times, converge to the target link utilization, and each individual VC connection will reach its max-min fair rate.

Since ERICA operates in a congestion-avoidance state, it is insensitive to parameter variations, and proves to be very robust. The rate converges very quickly with little oscillation. Also because it does not have to keep bottleneck information for each VC connection, the switch implementation is relatively simple compared to other ER algorithms. 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, though acquiring its equal link share, may 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 [7,8].

3.4. Fast max-min rate allocation algorithm

The Fast Max-Min Rate Allocation (FMMRA) [3] algorithm is based on measurement of available capacity and exact calculation of max-min 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. 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 of 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. 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. Details of FMMRA can be found in [3].

4. Simulations and observations

Fig. 2 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

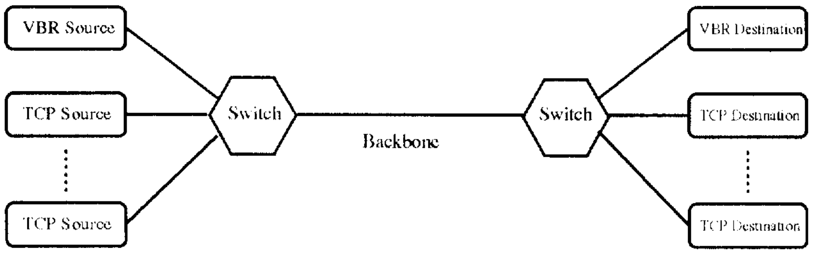


Fig. 2. Network configuration.

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 1 km represent typical Local Area Network (LAN) situations. The VBR background traffic is running at 100 Mbps 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 awaiting ABR cells 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 informs the ABR source to reduce its transmission rate. Different ABR congestion control algorithms will result in different buffer requirements and TCP performance.

The simulation tool used here is the NIST ATM Network Simulator. The simulator is developed at the National Institute of Standards and Technology (NIST). Its purpose is to provide a flexible test bed 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.

We have implemented ERICA and FMMRA in the ATM switch component, and modified the TCP component to facilitate TCP over ATM simulations. In the NIST simulator, the TCP model is based on Jacobson's congestion control mechanisms [4], exponential backoff, enhanced round-trip time (RTT) estimation based on both the mean and variance of the measured RTT. Since RTT values in our simulation configuration were in the order of ms, 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.

4.1. ABR parameters and TCP parameters

We used an infinite source mode at the application layer running on top of TCP, implying that TCP

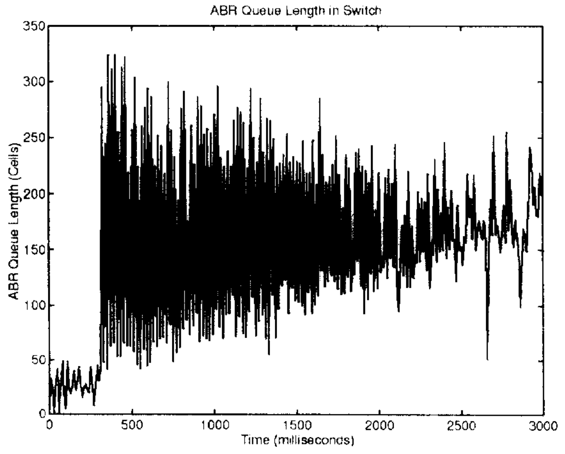


Fig. 3. ABR queue length using FMMRA, 2 TCP and 1 VBR connections.

always had a packet to send as long as its window permitted it. Our goal was to explore the possible limitations that ATM networks placed on the performance of the TCP protocol. The infinite source mode best met our requirement. In our simulations, the "fast retransmit and recovery" mechanism has not been implemented. The ATM source end system (SES) parameters [3] of ABR were selected to maximize the Available Cell Rate (ACR). The source TCP and SES parameters were chosen as follows:

Source TCP parameters	SES parameters
TCP maximum segment size = 9180 bytes	Peak cell rate = 155 Mbits
Mean packet processing time = 200 μs	N_{rm} = 32 cells
Packet processing time variation = 50 μs	M_{rm} = 2 cells
Receive window size = 64 K bytes	ICR = 10 Mbits
Bit rate = 155 Mbit/s	MCR = 0
Delay-ack timer = 0	CRM = TBE/N_{rm} = 20 cells
	CDF = 0.5
	TRM = 100 ms
	TCR = 0.00424 Mbits

For ERICA and FMMRA, RDF = 1/512, RIF = 1; for the binary scheme EFCI algorithm, RDF = 1/16, RIF = 0.1. For ERICA, the Target Utilization Factor was set at 90%, a level recommended by the proposer, but the Target Utilization Factor for FMMRA was set at 100% since FMMRA aims to achieve 100% utilization.

4.2. Simulation results and observations

In our simulations, the following TCP and ABR performance metrics were evaluated:
- ABR queue length in the 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 the 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.
4. Source ACR should reflect the changes of the network condition.

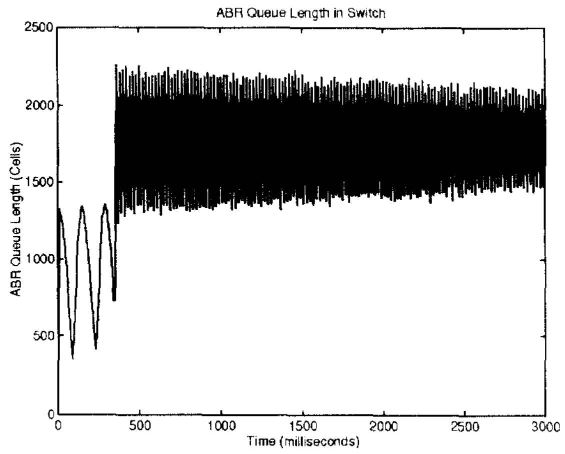


Fig. 4. ABR queue length using EFCI, 2 TCP and 1 VBR connections.

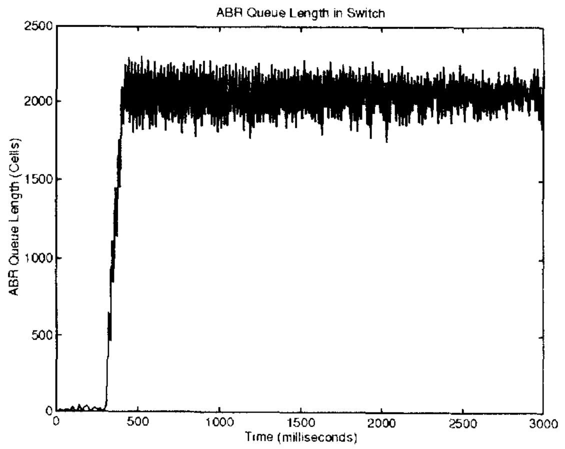


Fig. 5. ABR queue length using ERICA, 2 TCP and 1 VBR connections.

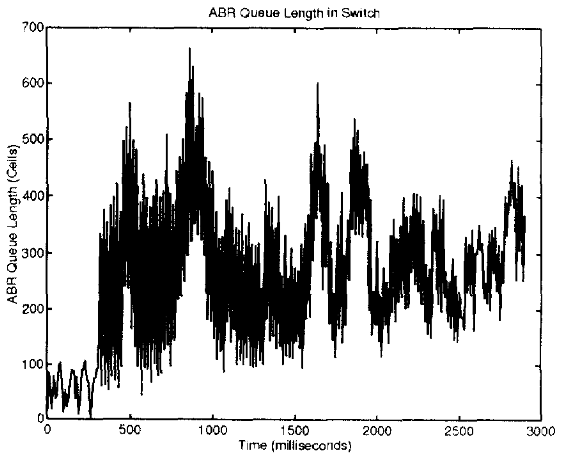


Fig. 6. ABR queue length using FMMRA, 5 TCP and 1 VBR connections.

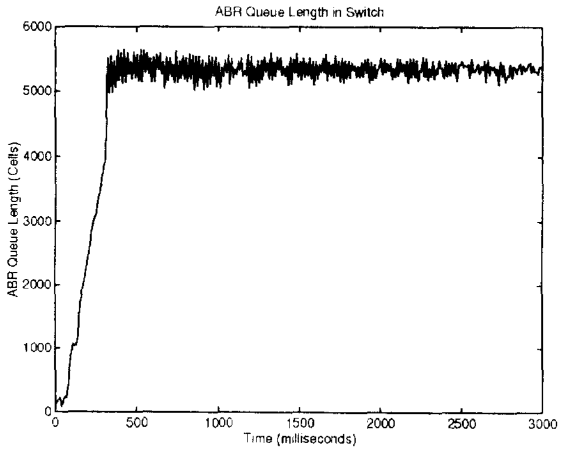


Fig. 8. ABR queue length using ERICA, 5 TCP and 1 VBR connections.

From the simulation results, the following observations were 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. Figs. 3–8 show the simulation results of ABR queue length vs. time for all of the three algorithms where two TCP sources and one VBR background traffic were employed in Figs. 3–5, and five TCP sources and one VBR background traffic in Figs. 6–8. From these figures, under the same network configurations, ATM switches implemented with FMMRA have the smallest ABR queue length, implying that FMMRA has the least buffer requirement for zero cell loss.

(2) FMMRA has the best performance in fairly allocating available network bandwidth to individual TCP sources. Figs. 9–11 show the results of the effective TCP throughput vs. time for the three algorithms. There were five TCP sources and one VBR background traffic with 10 ms ON and 10 ms OFF bursty nature. From these figures, both ERICA and EFCI lead to some unfairness among the individual TCP sources during the transient period. Some

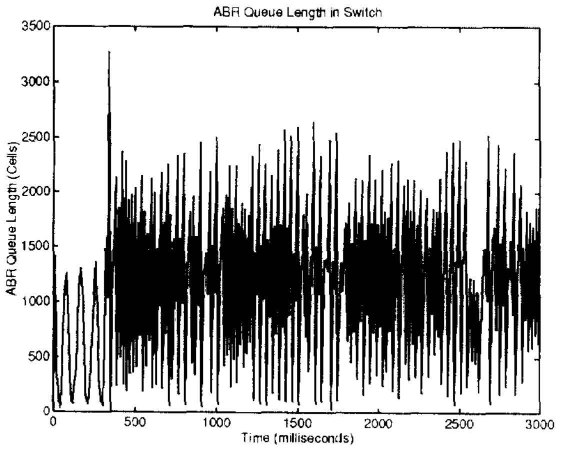


Fig. 7. ABR queue length using EFCI, 5 TCP and 1 VBR connections.

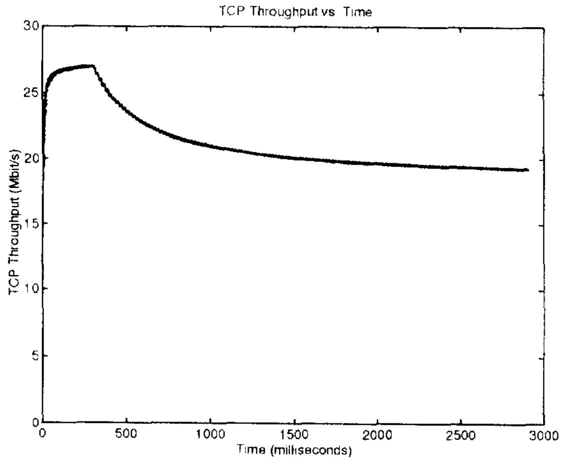


Fig. 9. TCP effective throughput using FMMRA, 5 TCP and 1 VBR connections.

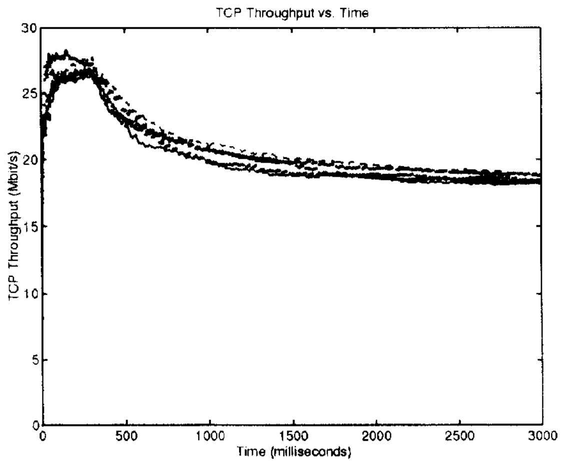


Fig. 10. TCP effective throughput using EFCI, 5 TCP and 1 VBR connections.

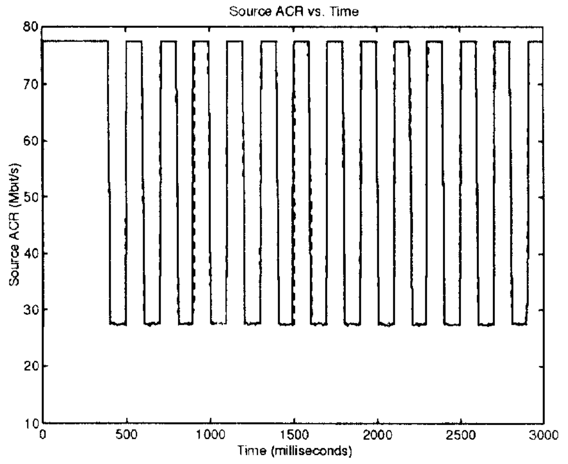


Fig. 12. TCP source Allowed Cell Rate (ACR) using FMMRA, 2 TCP and 1 VBR connections.

sources acquire higher throughputs than the others. Using FMMRA, every TCP source achieves the same effective throughput.

(3) FMMRA has the fastest and most accurate response to the source ACR rate in response to the changes in network traffic. Figs. 12–14 show results of ABR source Allowed Cell Rate (ACR) vs. time for the three algorithms. There were two TCP sources and one VBR background traffic with 100 ms ON and 100 ms OFF bursty nature. FMMRA has the fastest and most accurate response to the changes of available network ABR capacity.

(4) There is a trade-off between link utilization and ABR queue length. High ABR queue length can result in high link utilization. From the simulation results, EFCI and ERICA have relatively bigger ABR queue lengths than FMMRA, and thus higher overall link utilization. However, keeping the ABR queue length as small as possible is much more important than getting a high link utilization. Figs. 15–17 show the effective TCP throughput of the three algorithms under the limited buffer size condition. With a switch buffer size of 1500 cells, FMMRA can achieve zero cell loss, hence yielding the highest

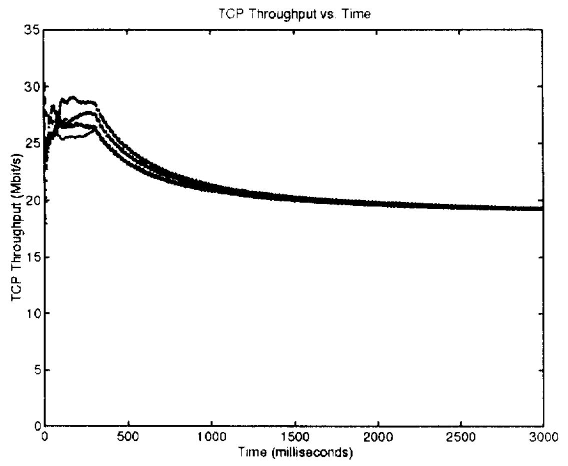


Fig. 11. TCP effective throughput using ERICA, 5 TCP and 1 VBR connections.

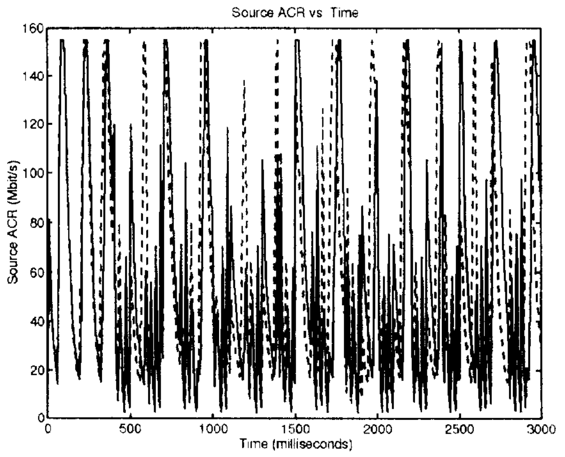


Fig. 13. TCP source Allowed Cell Rate (ACR) using EFCI, 2 TCP and 1 VBR connections.

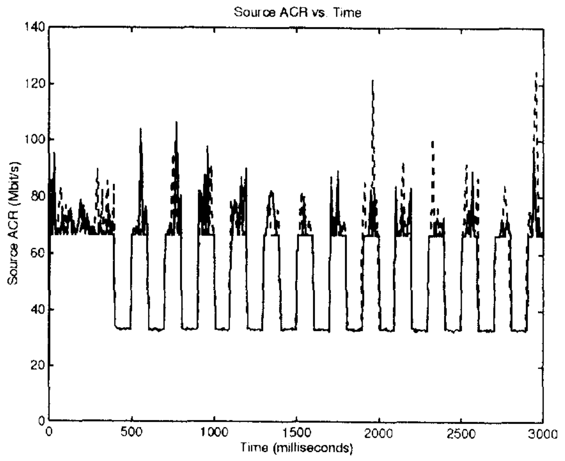


Fig. 14. TCP source Allowed Cell Rate (ACR) using ERICA. 2 TCP and 1 VBR connections.

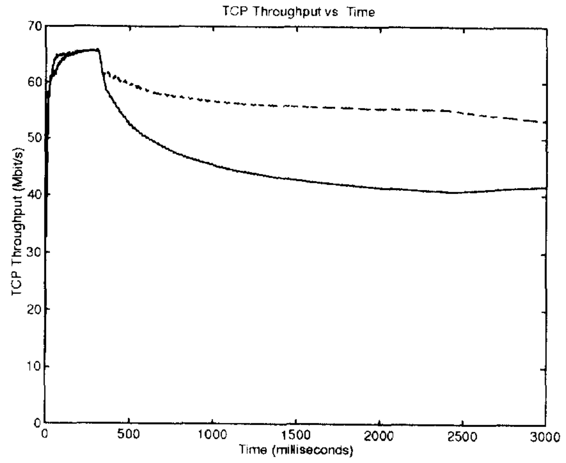


Fig. 16. Effective TCP throughput using EFCI with limited buffer size of 1500 cells. 2 TCP and 1 VBR connections.

throughput. Although we can choose proper High Queue length Threshold (HQT), RIF, and RDF to control the cell loss 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 the 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 unfavorable situation.

(5) If queue length monitoring and control mechanism are not implemented in the ABR congestion control algorithm, the ABR queue length in a switch will increase unboundedly under the following situations: a large number of TCP connections, high frequency background VBR traffic, and long round trip delay. Figs. 18 and 19 show the ABR queue length for ERICA and FMMRA 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. A large ABR queue length pro-

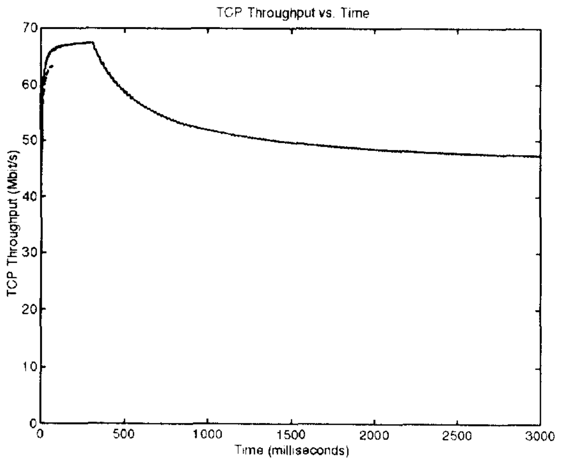


Fig. 15. Effective TCP throughput using FMMRA with limited buffer size of 1500 cells. 2 TCP and 1 VBR connections.

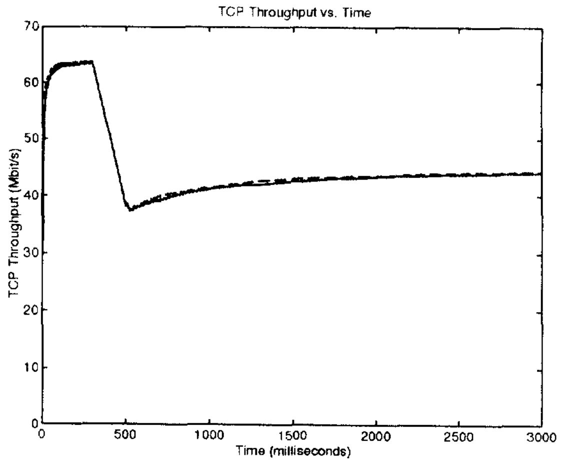


Fig. 17. Effective TCP throughput using ERICA algorithm with limited buffer size of 1500 cells. 2 TCP and 1 VBR connections.

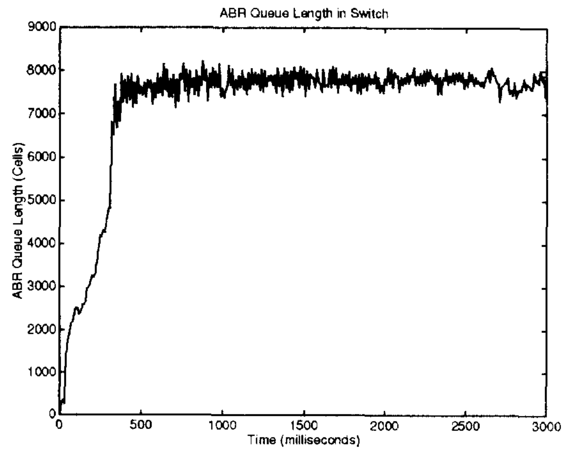


Fig. 18. ABR queue length using ERICA, 10 TCP and 1 VBR connections.

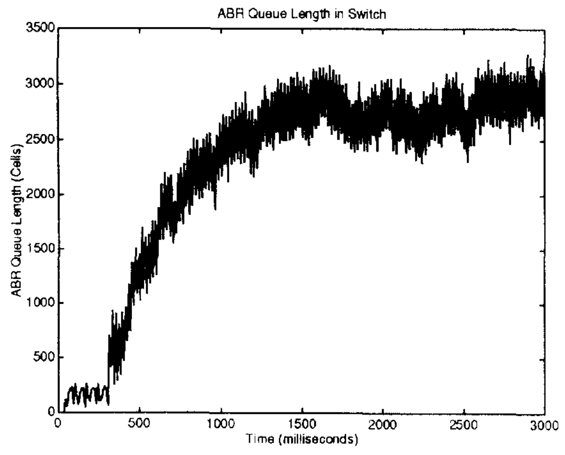


Fig. 19. ABR queue length using FMMRA, 10 TCP and 1 VBR connections.

duces long delay and makes the ATM switch expensive and complex. Some kind of queue length control mechanism is thus necessary.

5. Conclusions

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

- The least buffer requirement for zero cell loss;
- Fairly allocate available network bandwidth to individual TCP connections;
- Adjust source ACR rates fast and accurately in response to the changes of network traffic.
- Under limited switch buffer size situation, FMMRA 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.

Acknowledgements

The authors are grateful to Xiaoqiang Chen of Lucent Technologies for sharing his technical expertise on ATM.

References

[1] S. Kalyanaraman, R. Jain, R. Goyal, S. Fahmy, F. Lu, S. Srinidhi, Performance of TCP/IP over ABR, in: Proc. Globecom '96, London, UK, November 1996, pp. 468–475.

[2] The ATM Forum Traffic Management Specification, Version 4.0, The ATM Forum Specification, April 1996.

[3] A. Arulambalam, X. Chen, N. Ansari, Allocating fair rates for available bit rate service in ATM networks, IEEE Commun. Mag. 34 (1996) 92–100.

[4] V. Jacobson, Congestion avoidance and control, in: Proc. ACM-SIGCOMM '88, Stanford, CA, August 1988, Comput. Commun. Rev. 18 (1988) 314–329.

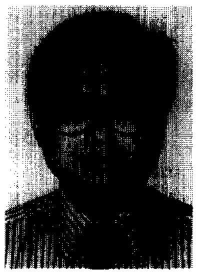
[5] V. Jacobson, R. Braden, D. Borman, TCP extensions for high performance, RFC 1323, November 1992.

[6] A. Romanow, S. Floyd, Dynamics of TCP traffic over ATM networks, IEEE J. Select. Areas Commun. 13 (1995) 633–641.

[7] R. Jain, S. Kalyanaraman, R. Viswanthan, Explicit rate indication for congestion avoidance, The ATM Forum Contribution, November 1995.

[8] R. Jain, S. Kalyanaraman, R. Goyal, S. Fahmy, R. Viswanthan, ERICA switch algorithm: a complete description, The ATM Forum Contribution 96-1172, August 1996.

Liping An received the B.S. from Tsinghua University in Beijing, China, in 1990, and the MSEE from the New Jersey Institute of Technology, Newark, USA, in 1996. He is currently working at Key Data Solutions, Inc. as a system analyst. His major research interests include high speed computer network, flow and congestion control in data communications, and massive parallel processing.



Nirwan Ansari received the B.S.E.E. (summa cum laude) from New Jersey Institute of Technology in 1982, the M.S.E.E. from the University of Michigan in 1983, and the Ph.D. degree from Purdue University in 1988 upon which he joined the ECE Department of New Jersey Institute of Technology, where he is a Professor and the Associate Chair for Graduate Studies. His current research interests include ATM networks, distributed and adaptive detection in CDMA, computational intelligence, and nonlinear signal processing. He is the chair of the North Jersey Chapter of the IEEE Communications Society which received the 1996 Chapter of the Year Award. He is a Senior Member of the IEEE, a special feature editor and an associate technical editor of the IEEE Communications Magazine, and he serves in various IEEE committees. He authored with E.S.H. Hou Computational Intelligence for Optimization, 1997, and edited with B. Yuhas Neural Networks in Telecommunications, 1994, both published by Kluwer Academic Publishers.

Ambalavanar Arulambalam received his B.S., M.S. and Ph.D. degrees in Electrical Engineering from the New Jersey Institute of Technology, Newark, New Jersey, in 1992, 1993 and 1996, respectively. In June 1996, he joined Bell Laboratories, Lucent Technologies as a member of technical staff in Murray Hill, New Jersey. His research interests include congestion and flow control, ATM network and switch performance evaluation, and the application of neural networks in telecommunications. He is a member of {\em Tau Beta Pi} and {\em Eta Kappa Nu} honor societies, and a member of IEEE Communications Society.