A SIMULATION STUDY ON CONGESTION CONTROL FOR THE ATM ABR SERVICE

A THESIS
SUBMITTED TO THE DEPARTMENT OF ELECTRICAL AND
ELECTRONICS ENGINEERING
AND THE INSTITUTE OF ENGINEERING AND SCIENCE
OF BILKENT UNIVERSITY
IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE DEGREE OF
MASTER OF SCIENCE

By
Sezer Ülkü
July 1997

TK
5105.35
·U45
1997

A SIMULATION STUDY ON CONGESTION CONTROL FOR THE ATM ABR SERVICE

A THESIS

SUBMITTED TO THE DEPARTMENT OF ELECTRICAL AND

ELECTRONICS ENGINEERING

AND THE INSTITUTE OF ENGINEERING AND SCIENCES

OF BILKENT UNIVERSITY

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS

FOR THE DEGREE OF

MASTER OF SCIENCE

Sezer Ülkü.

tarafından bağışlanmıştır

By

Sezer Ülkü

July 1997

TK
5105.35
-U45
1997

B038246

I certify that I have read this thesis and that in my opinion it is fully adequate, in scope and in quality, as a thesis for the degree of Master of Science.

[illegible]

Prof. Dr. Erdal Arıkan(Supervisor)

I certify that I have read this thesis and that in my opinion it is fully adequate, in scope and in quality, as a thesis for the degree of Master of Science.

[illegible]

Assoc. Prof. Dr. Mustafa Akgül

I certify that I have read this thesis and that in my opinion it is fully adequate, in scope and in quality, as a thesis for the degree of Master of Science.

[illegible]

Assist. Prof. Dr. Tuğrul Dayar

Approved for the Institute of Engineering and Sciences:

[illegible]

Prof. Dr. Mehmet Baray Y.
Director of Institute of Engineering and Sciences

ABSTRACT

A SIMULATION STUDY ON CONGESTION CONTROL FOR THE ATM ABR SERVICE

Sezer Ülkü
M.S. in Electrical and Electronics Engineering
Supervisor: Prof. Dr. Erdal Arıkan
July 1997

In this thesis, we have performed a simulation study on congestion control for the asynchronous transfer mode (ATM) available bit rate (ABR) service. Even though ABR is primarily intended for applications that can not describe their characteristics appropriately, it can be used by a wider range of applications since it provides some minimal guarantees for bandwidth. For the simulations, the ABR mechanisms specified in The ATM Forum Specification, Version 4.0 have been implemented to a great extent. Relative marking, enhanced proportional rate control (EPRCA) and efficient rate allocation algorithms (ERAA) have been realized, and their performances at ATM, TCP and application layers have been examined based on robustness, efficiency, fairness, buffer requirements and response time. The beat-down problem and large buffer requirements for the relative marking scheme have been illustrated. EPRCA was shown to be sensitive to parameters and result in oscillations in allowed cell rate. Finally, ERAA was shown to work efficiently with small buffers.

Keywords : asynchronous transfer mode (ATM), available bit rate (ABR), congestion control, relative marking, EPRCA, ERAA, TCP/IP

ÖZET

EŞZAMANSIZ AKTARIM MODU MEVCUT BANT GENİŞLİĞİ HİZMET SINIFINDA KARMAŞA KONTROLÜ ÜZERİNE BİR SİMULASYON ÇALIŞMASI

Sezer Ülkü
Elektrik ve Elektronik Mühendisliği Bölümü Yüksek Lisans
Tez Yöneticisi: Prof. Dr. Erdal Arıkan
Temmuz 1997

Bu tezde eşzamansız aktarım modu (ATM) mevcut bant genişliği (ABR) hizmet sınıfında karmaşa kontrolü üzerine bir simulasyon çalışması yapılmıştır. ABR özellikle kendi trafik karakteristiklerini önceden belirleyemeyen uygulamalar için geliştirilmiştir. Buna rağmen, uygulama alanları genişletilebilir. Simulasyonlar için ATM Forumu Spesifikasyonunun dördüncü versiyonundaki ABR mekanizmaları büyük ölçüde gerçekleştirilmiştir. Göreceli işaretleme, ilerletilmiş orantılı hız kontrolü (EPRCA) ve etkin hız dağıtımı (ERAA) algoritmaları gerçeklenmiş; ATM, TCP ve uygulama tabakalarındaki başarımları etkinlik, eşitlik, tepki zamanı, parametre değişimlerine karşı dayanıklılık ve kuyruk uzunlukları ölçü alınarak incelenmiştir. Göreceli işaretleme mekanizmasında büyük sıra uzunlukları ve eşitsizlik gözlemlenmiştir. EPRCA algoritmasının parametre değişlerine karşı hassas olduğu ve izin verilen bant genişliğinde dalgalanmalara yol açtığı görülmüştür. Son olarak, ERAA'nın kısa kuyruklarla etkin olarak çalıştığı gösterilmiştir.

Anahtar Kelimeler : eşzamansız aktarım modu (ATM), mevcut bant genişliği (ABR), karmaşa kontrolü, göreceli işaretleme, EPRCA, ERAA, TCP/IP

ACKNOWLEDGEMENT

I would like to express my deep gratitude to my supervisor Prof. Dr. Erdal Arıkan for his guidance, suggestions and valuable encouragement throughout the development of this thesis.

I would like to thank Assist. Prof. Dr. Tuğrul Dayar and Assoc. Prof. Dr. Mustafa Akgül for reading and commenting on the thesis and for the honor they gave me by presiding the jury.

I am indebted to Dr. Y. Ahmet Şekercioğlu for his precious help on my simulation programs.

Sincere thanks are also extended to all the friends who have helped in the development of this thesis and made life more enjoyable.

To my family
and İlkay...

Contents

List of Figures

List of Tables

Glossary

α	averaging parameter (EPRCA)
ABR	available bit rate
ACR	allowed cell rate
A_i	rate allocation (ERAA)
A_{max}	maximum available share for bottlenecked VCs (ERAA)
ATM	asynchronous transfer mode
B	total bandwidth
B_{eq}	equal bandwidth share (ERAA)
B_f	free bandwidth (ERAA)
BRM	backward RM
BW_i	bandwidth for connection i
BW_{link}	bandwidth of a link
C	capacity
CAC	connection admission control
CAPC	congestion avoidance by proportional control
CBR	constant bit rate
CCR	current cell rate
CCERI	congestion control with explicit rate indication
CDF	cut-off decrease factor
CI	congestion indication
CLR	cell loss ratio
$Cr(t)$	credits
CRM	missing RM cell count
Cr_{max}	maximum credits
DES	destination end system
DIR	direction field
DMRCA	dynamic max rate allocation algorithm
DPF	down pressure factor (EPRCA)
EFCI	explicit forward congestion indication
EPD	early packet discard

EPRCA	enhanced proportional rate control algorithm
ER	explicit rate
ERAA	efficient rate allocation algorithm
ERF	reduction factor (EPRCA)
ERICA	explicit rate indication for congestion avoidance
ERU	upper limit on the rate increase factor (CAPC)
ERF	lower limit on the rate decrease factor (CAPC)
FERM	fuzzy explicit rate marking
FCVC	flow controlled virtual circuit
FMMRA	fast max-min rate allocation algorithm
FRM	forward RM
$FRTT$	fixed round-trip time
HT	high queue threshold
ICR	initial cell rate
ICR_N	negotiated ICR
λ	load factor (ERAA)
LAN	local area network
LT	low queue threshold
μ	average service time
$MACR$	mean allowed cell rate (estimate of the fair shares)
MCR	minimum cell rate
MMRCA	max-min rate control algorithm
MRF	Major reduction factor (EPRCA)
Mrm	minimum number of cells between RM-cell generation
mss	maximum segment size
N	number of connections through a switch
N_b	number of bottlenecked connections
NI	no increase
Nrm	number of data cells before an RM cell is sent +1
nrt-VBR	non-real-time variable bit rate
N_u	number of satisfied connections
OSU	Ohio State University (scheme)
PCR	peak cell rate
PPD	partial packet discard
PRCA	proportional rate control algorithm
PTI	payload type indicator

$Q(t)$	queue length as a function of time
Q_{max}	maximum queue size
QoS	quality of service
RDF	rate decrease factor
Rdn	slope parameter for rate decrease (CAPC)
RIF	rate increase factor
R_i	input rate of connection i
R_{in}	total input rate
RM	resource management (cells)
R_T	target rate (OSU)
RTT	round trip time
RTT_{link}	round trip time of a link
rt-VBR	real-time variable bit rate
Rup	slope parameter for rate increase (CAPC)
SES	source end system
$SN(t)$	sequence numbers over time
ss_thresh	slow-start threshold
STM	synchronous transfer mode
τ_p	propagation delay
T	average waiting time
TBE	transient buffer exposure
TCP	transport control protocol
UBR	unspecified bit rate
U	average utilization
$U(t)$	utilization over time
UPC	usage parameter control
VC	virtual circuit
VCI	virtual circuit identifier
VPI	virtual path identifier
W	window size
WAN	wide area network
z	load factor (ERICA, CAPC, OSU)

Chapter 1

Introduction

1.1 Asynchronous Transfer Mode

The Asynchronous Transfer Mode (ATM) is the prospective transmission, multiplexing and switching technology that will be used by the future high speed networks. It is basically a connection oriented cell-relay technology, where fixed size cells of 53 bytes are the units of transmission. Each cell is composed of a 5 byte header and an information field of 48 bytes. Routing is based on the virtual path and virtual circuit identifier (VPI and VCI) fields in the header.

An important feature of ATM is the efficient use of resources through statistical multiplexing. As opposed to the Synchronous Transfer Mode (STM) in which connections use the bandwidth on periodical slots, ATM allows transmission on arbitrary slots, as long as bandwidth is available. Hence, bandwidth unused by a connection is grabbed by another.

Another feature is the hierarchy-free architecture. In an STM network, channels of lower capacity are multiplexed into channels of larger capacity. As the number of layers forming the hierarchy increases, the operations prior to and after switching become more demanding since the whole stack is demultiplexed before switching and re-multiplexed thereafter. In addition, synchronization problems arise. In ATM networks, there is no digital hierarchy [1], and as the name implies, no synchronization is required. These features compensate to a certain extent for the overhead incurred by using a header.

Finally, the small fixed size of cells reduces delay and jitter, thereby allowing ATM to merge data, voice and video applications over a single network. The importance of this feature is seen, when one considers the current networks. As of today, there exist separate networks for telephony, data communications and cable TV. ATM promises a single network for all information transfer in digital form.

There exists an issue worth noticing, to realize the gains offered by ATM. Effective traffic management is a must, in order to maximize performance observed by the application layers. Otherwise, we may get very low quality of service (QoS) despite full utilization of the resources.

1.2 Service Categories

A complexity that arises with the *one-network-fits-all* approach is the necessity to differentiate between requirements for different applications. Whereas real-time applications such as multi-media demand low delay and delay variance, data applications require data integrity, while being more tolerant to delays. Five categories were defined by the ATM Forum to facilitate the provision of services with different requirements:

- CBR : constant bit rate
- rt-VBR : real-time variable bit rate
- nrt-VBR : non-real-time variable bit rate
- UBR : unspecified bit rate
- ABR : available bit rate

These service categories relate traffic characteristics and QoS requirements to network behavior. Functions such as routing, connection admission control (CAC) and resource allocation are generally structured differently for each category [2].

Among these classes, CBR, rt-VBR and nrt-VBR are intended for applications that require some form of bandwidth guarantees. Thus a fraction of the total capacity is booked for these connections before transmission starts. CBR basically emulates circuit switching. The allocated bandwidth is equal to the peak cell rate (PCR) of the connections. Since statistical multiplexing gain is allowed, rt-VBR or nrt-VBR is appropriate when the bit rate varies over time. The notion of *effective bandwidth* [3], [4] is used in the bandwidth allocation process for these classes. The effective bandwidth is a traffic descriptor, estimating the true bandwidth used by a bursty application.

The remaining service categories, UBR and ABR, were defined for applications that can not describe their requirements properly, and that do not have strict delay and jitter requirements, e.g, bursty data applications. Making a-priori reservations for such applications would lead to a waste of resources as the requirements are unpredictable. Thus, the choice has been to serve these applications on a *best effort* basis. UBR and ABR connections are allocated resources only if resources are not used by the higher priority connections, that is CBR and VBR connections.

Even though ABR and UBR are both best-effort services, a distinction between these two categories is essential. UBR is the *plain best effort* mode [5], providing absolutely no guarantees on cell loss, delay and jitter. Therefore, UBR connections are not rejected on the basis of bandwidth shortage. A typical example using UBR would be e-mail. ABR is the *better best-effort* mode, and has higher priority compared to UBR in that ABR connections are provided minimal guarantees. They are offered minimum possible delay and cell loss in addition to a fair share of the available bandwidth. Moreover, a minimum cell rate (MCR) can be reserved at connection set-up, but connections with $MCR > 0$ take the risk of being rejected. ABR connections grab the available bandwidth dynamically, thus the resources are utilized efficiently. Telnet applications, for instance, might work over ABR rather than UBR.

1.3 Congestion Control

Congestion is one of the major issues in packet-switched networks. As the service rates and buffer sizes are limited, the load of the network should be prevented from taking arbitrarily large values in order to provide a reasonable QoS with low delay and cell loss ratio. More specifically, congestion is formed at a network node when

$$R_{in} = \sum_i R_i(t) > C,$$

where $R_i(t)$ is the rate of transmission for connection i, C is the service rate, or the capacity of an outgoing link from the node under consideration and R_{in} is the total rate of flow into the node. In a high-speed network, it may take a short time for the buffers of a node to fill due to a rate mismatch, leading to high delays and serious overflows. Depending on the service class, the packets corresponding to the lost cells might have to be retransmitted. As the cells of different connections are multiplexed on the incoming links, lost cells would most probably belong to different packets. Hence, retransmissions might lead to a total throughput collapse. In the case of real-time services, it might be meaningless to retransmit lost cells but the QoS would be degraded due to missing information.

Congestion control is a mechanism to prevent or resolve the situations, where the network is overloaded [6]. Depending on the traffic characteristics, different means for congestion control are used. For applications that can describe their characteristics, i.e, CBR and VBR, the appropriate mechanism is to use CAC at connection set-up, admitting connections only if their requirements can be met. In addition to CAC, a usage parameter control (UPC) scheme can be used to force users to comply with the agreement with the network. For UBR connections, the procedure is to drop cells in cases of congestion and let the higher layers provide reliable delivery.

Among the five service categories, ABR poses the most challenging problems concerning traffic management. ABR connections are unable to describe their bandwidth requirements appropriately, thus allocating a fixed bandwidth to these connections could lead to a waste of resources. Moreover, the bandwidth available is time-varying since the ABR class uses the left-over bandwidth from the higher priority traffic. Still, ABR customers are offered a low cell loss ratio and fairness, if they adapt their rates according to the network feedback.

The problem of congestion control for ABR category is one of allocating the available bandwidth fairly and efficiently among the connections. Had the buffer capacity been free of cost and the applications tolerant to excessive delays, the solution to this problem could have been *use large buffers and serve the VCs fairly.* Even though excessive delays could have been observed at certain times, no cells would have been dropped and the utilization would always have been maximum in such a case. That could have been managed by fair scheduling algorithms, without requiring a feed-back mechanism. However, in reality, the buffer sizes are limited by cost and excessive delays are not desirable. As a result, a mechanism is necessary to inform the sources about the network state of congestion, so that they regulate their transmission rates. After long discussions in the ATM Forum, a distributed, end-to-end, rate-based feedback algorithm as specified in [2], has been selected as an effective approach.

1.3.1 Efficiency

Efficiency in the congestion control context has two measures. While it is desirable to use the network to the fullest extent, it is also desirable to bound the end-to-end delay to small values. These two are conflicting goals, since queuing delay is an increasing function of utilization as given by the *Pollaczek-Khinchin formula* for an *M/D/1* system, where T, U and μ denote mean waiting time, utilization and mean service time respectively:

$$T = \frac{U}{2\mu(1-U)}.$$

For a fixed service rate, the average waiting time (T) increases with load. With finite buffers, a major component of delay is the waiting time for the lost packet retransmissions.

In an extreme case, where sources transmit at an arbitrarily large rate, one would observe full utilization of the network. However, the delay values would also be arbitrarily large. If the finite size of the buffers is also taken into account, there would be packet losses in the buffers due to overflows, and a throughput collapse would be possible due to retransmissions, making the effective throughput close to zero, and delays arbitrarily large. Obviously, the load should be adjusted to make a good compromise between achieving high throughput and limiting delay and cell losses.

1.3.2 Fairness

Fairness is the other major issue, which is not totally independent of the efficiency consideration. A definition of fairness is made in [2] as follows : "No set of connections should be arbitrarily discriminated against and no set of circuits should be arbitrarily favored, although resources may be allocated according to a defined policy." At first thought, fairness might be taken as providing equal rates to all connections belonging to a certain service class, however, such an approach may lead to under-utilization of resources.

Max-min fairness will be the criterion of fairness throughout this thesis .It is a widely used definition of fairness, which does not ignore efficiency. The idea behind max-min fairness is to maximize the resources allocated to the sessions with the minimum allocation, as stated in [7]. For the network topology in Fig. 1.1, one would allocate a rate of 1/3 to the sources S1, S2 and S3, but it would be a waste of resources to assign the same rate to S4, since it is possible for S4 to transmit at a rate 2/3 without degrading the overall fairness.

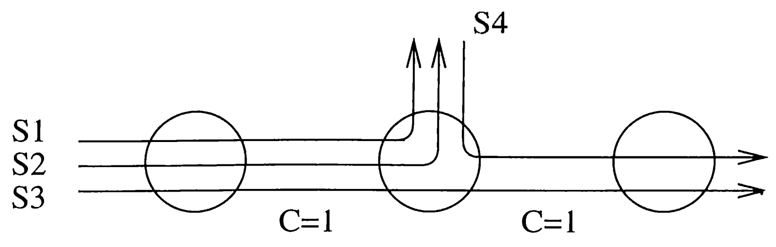


Figure 1.1: Max-min fairness example

1.4 TCP

Currently, the services on the best-effort basis are supported by the TCP/IP protocols. This family is the standard which forms the backbone of the current *Internet*, and there are no signs that it will be replaced by other transport/network layer protocols, discounting changes on the original TCP/IP. Thus, it is imperative for ATM that it coexist and work efficiently with TCP/IP.

The most important problem with TCP/IP over ATM is the fragmentation problem. A study by Romanov et al [8] has shown that throughput collapse is possible for TCP over ATM, when measures are not taken against congestion at the ATM layer. As the TCP segments are divided into ATM cells, the loss of a single cell requires the retransmission of a whole TCP packet. Even if the rest of the cells safely reach their destination, they are discarded by the destination, thus, the bandwidth is wasted for transmitting useless cells. Hence even for small cell loss ratio (CLR), the effective throughput might be very small. This problem is also seen in IP networks, however the small size of ATM cells worsens the situation. Examples of solutions for this problem are intelligent cell discarding techniques like early packet discard (EPD) and partial packet discard (PPD) as discussed in [8]. These mechanisms discard the cells belonging to a corrupted IP packet or drop them even before the buffer is full. Nevertheless, the ideal solution to the problem would avoid cell losses completely, as throughput would be maximized in that case. An effective ATM layer congestion control algorithm can increase good-put significantly when applied with appropriate buffer sizes as shown in [9, 10, 11].

Another issue requiring attention is the interaction between the rate based congestion control algorithm for ABR services and the window control of TCP. The TCP algorithm was designed especially for networks which does not provide any information on the network state of congestion, and it works, at least currently, independent of the ABR flow control. Meanwhile, the ATM layer makes use of the network information for regulating the transmission rates of the ATM cells into the network. TCP ignores the information that comes at no cost, causing inefficiency in the use of resources, and the danger of buffer overflows at the network interfaces due to rate mismatches between the two algorithms. Thus, a coupling between the two mechanisms seems necessary for a more efficient operation. Such an approach with binary feedback is mentioned by Floyd in [12].

1.5 Synopsis and Organization

In this work we examine the *Rate Based Congestion Control Specification* [2] which was finalized by the ATM Forum Traffic Management Working Group in April 1996. We perform a simulation study to evaluate the effectiveness of the end-to-end, rate based feedback algorithm for congestion control. To this end, we have implemented the mechanisms proposed by the specification except some minor points, and we have performed extensive simulations. Several fair rate allocation algorithms have been implemented, ranging from the simplest to the most sophisticated, and their performances have been tested under different network conditions. Both transient and steady state behavior have been observed; the effect of control parameters, connection parameters and delay have been examined.

In addition to the ATM layer performance, the service perceived by TCP and application layers are also of interest. Hence, the next stage of simulations has been performed with TCP end systems communicating over an ATM network. A comparison is made between the ABR congestion control schemes based on their abilities to provide fast, fair and smooth transfer and on their buffering requirements. Finally, a method for coupling ATM and TCP layer congestion control mechanisms is proposed.

Our work is organized as follows. We start in Chapter 2 by giving a brief review of the literature on ABR congestion control, including the credit based, rate based and integrated proposals and some background on the ABR congestion control framework. Chapter 3 describes the network models used in the simulations, and illustrates their virtues and limitations. In Chapter 4 we explore the ATM layer performances of several rate allocation algorithms, namely relative marking, enhanced proportional rate allocation algorithm (EPRCA), and efficient rate allocation algorithm (ERAA). Chapter 5 is on the performance observed at TCP and application layers. Finally in Chapter 6, concluding remarks are put forward together with the proposed future work.

Chapter 2

Background on ABR Congestion Control

2.1 Evolution of the Congestion Control Framework for the ATM ABR Service

The ABR Service was developed for the economical support of applications with vague requirements [13]. This service category might be suitable for many different applications, however it was primarily defined for bursty data applications. It allows connections to specify a range of rates for proper operation in terms of minimum cell rate (MCR) and peak cell rate (PCR), which is an efficient way of expressing the requirements for data applications.

After studies by Romanov et al [8], which indicated that congestion collapse due to fragmentation was possible in packet based transmission, a feedback algorithm has been included in order to tightly control cell losses within the network, as efficiency is a concern and as cell losses and excessive delays are undesirable. In addition, the fair distribution of the bandwidth to connections was an issue, and the mechanism was expected to work in a wide range of environments due to the recurrent progress in networking technology, leading to increases in transmission speeds.

The bandwidth usage for ABR along with VBR, CBR, and UBR are seen in Fig. 2.1. The priority for bandwidth usage is CBR, VBR, ABR and UBR in a decreasing order. Whereas the first two use the bandwidth on a reservation basis, UBR and ABR try to grab the bandwidth left unused by the higher priority traffic. UBR forwards all the information received from upper layers into the network without performing any control on the ATM layer, ignoring any overflows due to transmission in excess of the capacity. ABR tries to use the available bandwidth as much as possible while holding

$$\sum_i R_i < C,$$

hence avoiding buffer overflows as much as possible. This is achieved by the feedback that controls the rate of flow into the network, according to the state of the network.

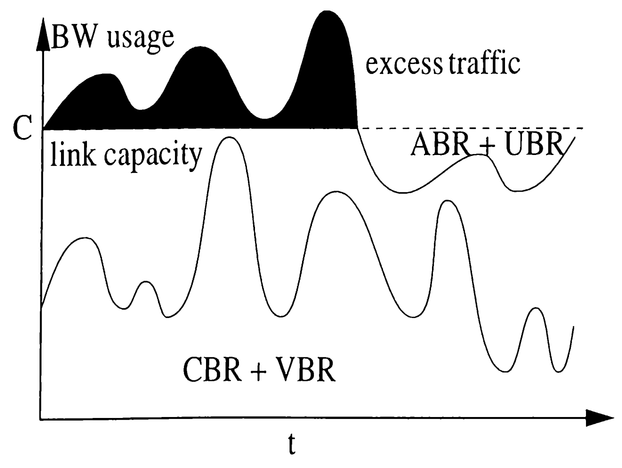


Figure 2.1: Bandwidth usage for CBR, VBR, ABR and UBR

There are several issues in evaluating the power of feedback mechanisms used for congestion control. These are fairness (optimality, convergence), low complexity, scalability with the number of connections and transmission rates, robustness, fast response to changes in the available bandwidth, low buffer requirements, stable operation and inter-operability. It is desirable for the congestion control architecture to be flexible, allowing a compromise between implementation complexity and efficiency. Such an architecture makes different implementations for different environments possible, which can inter-operate.

The adoption of the feedback algorithm for the ABR congestion control has taken a long evolution process, subject to deep discussions in the ATM Forum Traffic Management Working Group. There have been an abundance of proposals which can be classified in three main groups :

- Credit Based Schemes

- Rate Based Schemes
- Integration Proposals

The rate based schemes aim at the direct control of source rates [13, 14], whereas the credit based ones aim at the control of the available trunk buffers[15]. Integration proposals advocate the co-existence of both mechanisms, based on the fact that each class has its advantages. The credit based approaches provide zero cell loss, efficiency, fast response, and robustness, while requiring per-VC buffering. The rate based approaches offer flexibility in implementations and scalability. The race has been mainly between the credit based and rate based schemes and in the final specification, an end-to-end, rate based feedback mechanism has been accepted as the framework for ABR congestion control [2]. In what follows, we present a brief review of these proposals.

2.1.1 Credit Based Congestion Control Proposal

Credit based schemes are basically link-by-link window flow control mechanisms. This approach to congestion control offers precise control over buffer use. Upstream nodes are required to wait till they receive a signal noting that downstream nodes can accept their packets without dropping them (Fig. 2.2). In the ATM context, this signal is the *credit*, i.e, the empty buffer space in the downstream nodes. Each receiver monitors its queue-length, and informs the sender about how many cells it can receive. A sender is required to have a non-zero credit, i.e., $Cr(t) > 0$ to transmit cells. For the links to be fully utilized, the maximum credit should be large enough to fill the pipe, that is

$$Cr_{max} \geq BW_{link} \cdot RTT_{link},$$

where Cr_{max}, BW_{link} and RTT_{link} are maximum credit, link bandwidth and link round-trip-time respectively. Otherwise, the attainable bandwidth is limited to

$$BW_i = \frac{Cr_{max}}{RTT_{link}}.$$

As seen, sustainable bandwidth depends on buffer size.

Such schemes result in high link utilization, fair sharing of the bandwidth and zero cell loss with appropriate buffering, where the sense of "appropriate buffering"

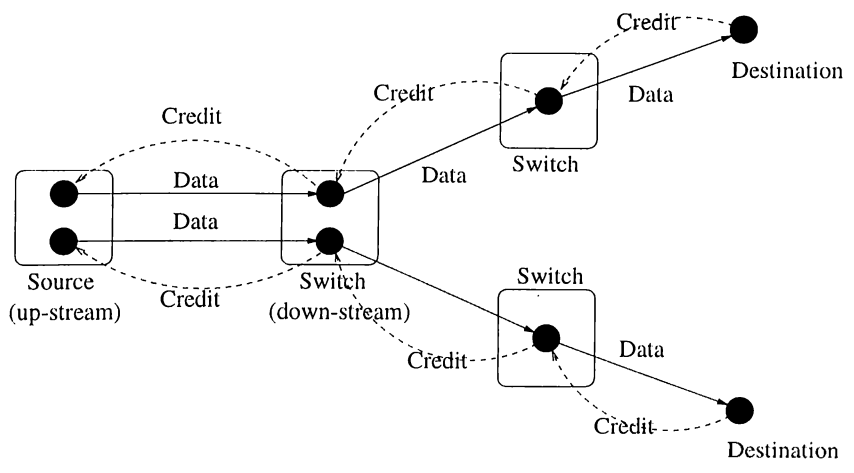


Figure 2.2: Credit-based flow control

depends on the bandwidth and the round-trip time. In addition, due to the link by link approach, the congestion is spread to the network, instead of localizing in a single node. Thus, the load of congestion is shared by nodes in the network.

There are two approaches to the use of buffers, and credit allocation. The first one is the strictly partitioned buffer approach. Flow controlled virtual connection (FCVC) proposed by Kung [16] is an instance of this class. In FCVC, per-VC queuing is required. Buffers are partitioned for the connections, thus separate flows are isolated from each other, leading to fair allocation of the bandwidth and efficient utilization of the links. In addition, the response to changes in network conditions is very fast. However, buffering might turn out to be a big problem, especially in cases where propagation delay, bandwidth and number of connections are large, i.e., in WANs. For instance, for a 622 Mb/s link with a length of 1000 km and 4096 connections [17], the buffer requirement is 3.2 GBytes, which is a prohibitively large figure. Such an approach might lead to inefficient and wasteful use of memory.

N23 scheme is one of the several implementations of FCVC. Two levels of buffer vacancy are used for rate allocation . The first one, $N2$, which is chosen as a design parameter, signifies the number of cells forwarded by the downstream node before credits are returned to the upstream node. $N2$ serves to limit the overhead due to transmitting credit information. $N3$ determines the sustainable bandwidth of a connection. Given the link round trip time RTT_{link}, the size of $N3$ is

$$N3 = RTT_{link} \cdot BW_i.$$

The second approach is based on buffer sharing [18, 19] . Memory requirements are reduced by dynamical allocation of credits to connections. Active VCs are given a larger share, whereas inactive connections are given a small fixed credit. However, this advantage comes at the expense of degrading the utilization and responsiveness. As the allocation of credits is based on the estimated use of VCs, it may take a long duration for a silent source to ramp up. Moreover, the implementation complexity increases.

Despite its advantages over the rate based proposal such as zero cell-loss, fast ramp-up and isolation of misbehaving users, the credit based congestion control proposal has been rejected by the ATM forum mainly due to its inability to scale with an increasing number of connections, and its inflexibility.

2.1.2 Rate Based Congestion Control Schemes

Rate based schemes perform congestion control by directly regulating source rates. The network nodes determine their congestion status and send congestion information to sources through a closed loop mechanism. Upon reception of feedback from the network, the sources update their rates appropriately. Such a process can be described by a leaky bucket with token rates changing over time, in reaction to the the amount of excess bandwidth along an ABR connection's path [20], as illustrated in Fig. 2.3.

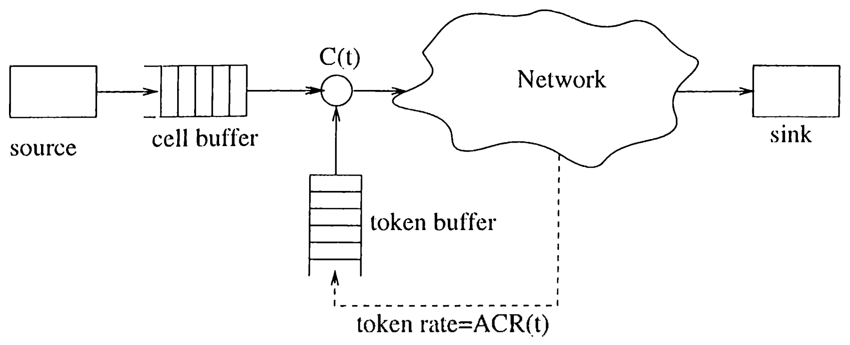


Figure 2.3: Rate-based flow control

The information that is fed to the sources can be in two forms. In the first form, sources are sent binary information indicating whether the network is congested or not. The provision of such information requires switch mechanisms with minimal complexity. The second form reports an explicit rate (ER) at which the sources are expected to operate. Such an approach brings in additional complexity due to the necessity to calculate rates, but results in better performance in terms of efficiency and fairness.

The early methods proposed for rate based congestion control in ATM networks have been in the form of binary feedback. The first proposal for rate-based congestion control in the ATM context has come from Newman [21]. In this method switches detected congestion by comparing their queue-lengths with a threshold, and periodically sent resource management (RM) cells to the sources in backward direction in cases of congestion. Sources were expected to halve their rates, upon reception of the RM cells, and to increase multiplicatively if no RM cells were received over a period.

Another binary feedback scheme was the explicit forward congestion indication (EFCI) marking scheme by Hluchyj [22]. In this scheme, switches monitored their queue-lengths and marked the EFCI field in data cells in cases of congestion. Destinations observed the EFCI fields periodically and sent RM cells to sources when they encountered EFCI=1. The sources decreased their rates multiplicatively when they received congestion indication and increased their rates otherwise. However, it was shown that his *negative polarity feedback* lead to congestion collapse, when RM cells experienced congestion in the backward direction. This lead to the development of the proportional rate control algorithm (PRCA) [23]. PRCA required the connections to increase their rates only when they received feedback from the network, and to decrease their rates otherwise. This approach solved the congestion collapse problem. In the final form of the specification, sources does not change their rates until feedback is received from the network, unless a time-out occurs.

The price paid for the simplicity of the binary schemes is slow response, unfairness, oscillations at steady state and linear dependence of queue sizes on the number of connections. These problems inherent to binary schemes make the use of more complex ER schemes feasible.

The provision of explicit rates necessitates the use of additional algorithms to calculate the fair rates for each connection. Certain algorithms approximate the optimal rates. Enhanced proportional rate control (EPRCA) [24], max-min rate control (MMRCA) [25], dynamic max rate control (DMRCA) [26], congestion avoidance with proportional control (CAPC) [27] and fuzzy explicit rate marking (FERM) [28] algorithms are examples of approximate rate calculation algorithms. Another approach is to calculate the fair rates exactly, as in congestion control with explicit rate indication (CCERI) [29], efficient rate allocation (ERAA) [17], Ohio State University (OSU) [30], explicit rate indication for congestion avoidance (ERICA), and ERICA+ [31, 32, 33] schemes.

The first proposal for an ER scheme has been the congestion control with explicit rate indication (CCERI) scheme [29] which formed the MS thesis of Charny. Charny proposed a scheme that exactly calculated the max-min fair rates for the VCs, by making use of a distributed algorithm. Charny's algorithm makes use of the desired and current cell rates. It computes a fair share as given by

$$FairShare = \frac{C - \sum_{S_u} R_i}{N - N_u},$$

where C is the channel capacity, S_u is the set of unbottlenecked connections, R_i is the rate allocation for unbottlenecked connection i, N is the number of connections and N_u is the number of unbottlenecked connections. CCERI allocates the desired rate to connections with desired rate less than the fair share, while allocating an equal share to others. It was shown that this algorithm converges to the fair rates very fast, but the complexity of the algorithm is $O(N)$, which makes its implementation in hardware difficult. Later Kalampoukas has developed the efficient rate allocation algorithm (ERAA) in [17], reducing the complexity of CCERI to $O(1)$, while keeping the benefits of the previous scheme.

Enhanced proportional rate control algorithm (EPRCA) proposed by Roberts [24] was the result of an attempt to develop an $O(1)$ complexity algorithm providing ER feedback. This algorithm has been approved by a large audience, and has gone through several modifications. EPRCA keeps an estimate of the fair share obtained by the exponential average given by

$$MACR = (1 - \alpha)MACR + \alpha CCR,$$

where CCR is the current cell rate and α is the averaging parameter. The sources adapt their rates to MACR multiplied by a constant in cases of congestion, which is detected by thresholds on queue-length. There are several variations of this scheme using multiple thresholds and queue-length derivative for congestion detection. MMRCA developed by Muddu et al [25] and DMRCA by Chiussi et al [26] are improvements on EPRCA.

Starting with the OSU scheme developed by Jain et al [30], another approach has been to attempt to hold link utilization at a desired level, and thereby avoiding congestion. In the OSU scheme, the switches measure their input rates over a fixed interval and compute a load factor given by

$$z = \frac{\sum R_i}{R_T},$$

where R_T is the target rate and R_i are the input rates for each VC. Source rates are updated by this load factor:

$$R_i = \frac{R_i}{z}.$$

ERICA and ERICA+ [31, 32, 33] schemes are variations on the OSU scheme. CAPC scheme is another congestion avoidance scheme, proposed by Barnhardt [27]. This algorithm is inspired by the OSU scheme. In addition to calculating a load factor, queue-length information is used for congestion indication in CAPC. The load factor z is used to update the fair share estimate by

$$FS = min(ERU, 1 + (1 - z)Rup)(FS)$$

in case of under-load and by

$$FS = max(ERF, 1 - (z - 1)Rdn)(FS)$$

in case of overload. ERU and ERF are bounds on increase and decrease factors, Rup and Rdn are slope parameters.

There have been novel approaches to the congestion control problem, incorporating computational intelligence into the algorithms. Pitsillides et al make use of fuzzy logic in FERM [28]. Jagielski et al applies genetic programming techniques to congestion control [34]. Taraafi et al have proposed the use of neural networks in this problem [35].

Thanks to the final version of the congestion control specification, most of the proposed algorithms can be implemented with modifications, and co-exist in a network. This specification will be presented in Section 2.2.

2.1.3 Integrated Proposal

In the WAN, it is not possible to use credit based schemes due to large buffer requirements, making the use of rate based schemes a must. In the LAN, as the propagation delays are low, the credit schemes, which provide superior performance in terms of cell loss, responsiveness and utilization can be used since the buffer requirements to support proper operation are at feasible levels. Thus, the integration of the rate based and credit based schemes has been proposed by Ramakrishnan and Newman [36] to exploit the advantages of both, and to offer a more flexible architecture. There were three proposals:

- Rate in the WAN/credit in the LAN
- Rate is default, credit is optional
- One size fits all

However, this approach has not been accepted in the ATM Forum due to the interoperability problems and additional costs.

2.2 ABR Congestion Control in the ATM Forum Traffic Management Specification V. 4.0

ATM Forum Traffic Management Working Group has focused on the flow control related issues in ATM networks, and a major part of this work has been devoted to the control of ABR traffic. It has taken about three years for the specification to be finalized.

The resulting framework for ABR has been a rate-based, distributed, closed loop control mechanism to dynamically regulate the inflow of traffic into the network [2]. This control loop is seen in Fig. 2.4. It is composed of source end systems (SES), destination end systems (DES) and the switches in-between. Special cells named resource management (RM) cells are used to probe the network. The behavior for sources and destinations have been precisely described by the committee whereas the switch behavior has been defined loosely. The reason for avoiding a rigid description is to allow the implementors certain flexibility in switch designs.

The framework will be described briefly in the following sections.

2.2.1 Source End System Behavior

Each connection starts by a connection set-up phase, during which the connection parameters are negotiated. These parameters include MCR, PCR, initial cell rate (ICR), transient buffer exposure (TBE), rate increase factor (RIF) and rate decrease factor (RDF). A CAC mechanism is used for accepting connections with $MCR > 0$, thus such connections might be rejected.

SES are expected to regulate their rates in compliance with the network feedback. In exchange, they are offered low cell loss, bounded delay and an MCR, if one has been negotiated. Usage parameter control (UPC) mechanisms are used to deal with non-cooperative SES, and discard the inflow that is in excess of the allowable traffic.

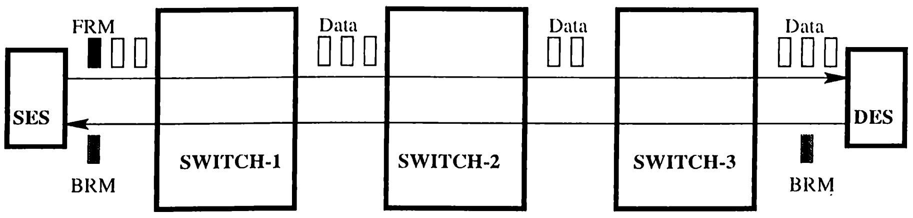


Figure 2.4: The control loop

SES initiates the control process by periodically emitting forward resource management cells (FRM), to probe the congestion status of the network. After every $Nrm - 1$ data cells, the source creates an FRM cell and writes the current allowed cell rate into the current cell rate (CCR) field and the desired cell rate into the explicit rate (ER) field. The congestion indication (CI) is set to 0 and payload type indicator (PTI) to 110, and then the FRM cell is forwarded into the network. The FRM cells are transmitted *in-rate*, i.e, they are counted in the bandwidth usage of

NI	CI	Action
0	0	$ACR := max(MCR, min(ER, ACR + RIF \times PCR, PCR))$
0	1	$ACR := max(MCR, min(ER, ACR - ACR \times RDF))$
1	0	$ACR := max(MCR, min(ER, ACR))$
1	1	$ACR := max(MCR, min(ER, ACR - ACR \times RDF))$

Table 2.1: Source reaction to network feedback

the source. *Out-of-rate* RM cells are used by the source only when the allowed cell rate is 0. Switches and destination end systems can also generate out-of-rate RM cells in special cases.

Upon receiving feedback from the network carried by the backward RM cells (BRM), which might be in the form of binary information or an explicit rate, the source regulates its allowed cell rate (ACR). The adaptation of ACR by the source is shown in Table 2.1. First, a new ACR is computed according to the CI and NI fields in the received RM cells. In case of congestion, i.e., if CI is set to 1, rate is decreased multiplicatively. If both no increase (NI) and CI bits are clear, ACR is increased additively. If the NI bit is set but CI is 0 then no action is taken. After this update, ACR is compared to ER and PCR with a min operator and to the MCR with a max operator so that it takes values between MCR and PCR.

In addition to the above stated mechanism, several procedures have been devised to protect the network in cases where the source is not able to check the network appropriately. First of all, if more than 100 ms has elapsed between two FRM cells, the source has to send another FRM cell after the first data cell, without waiting for Nrm-1 data cells. This mechanism avoids the breaking of the feed-back loop for a long time.

The second safety feature is the so called *use it or lose it* behavior intended to solve the ACR retention problem. If a source sends an RM cell when the network is lightly loaded and does not receive feedback from the network for a long time, the network might have become congested, and the source might still try to send cells with a high rate. To prevent that, the allowed cell rate is valid for about 500 msec and after that period the ACR is decreased to $min(ACR, ICR)$.

Apart from the above-stated two cases, the source may have difficulty in receiving feedback due to blocked RM cells in a queue in cases of congestion or a broken link.

In order to protect the network from a continuous inflow of traffic, the sources are required to decrease their rates if they have more than a number of FRM cells for which they have not received a BRM. Once this mechanism is fired, rate is decreased for each FRM, and that leads to an exponential decrease in the rate. This mechanism might cause problems when the round trip delay is large. Thus the threshold for activating this mechanism should be held high enough to avoid false alarms. The threshold, CRM is computed from another quantity that is negotiated at connection set-up, namely the transient buffer exposure, TBE as

$$CRM = \lceil \frac{TBE}{Nrm} \rceil.$$

TBE is also used for setting the ICR. ICR is set to

$$min(ICR_N, \frac{TBE}{FRTT}),$$

where FRTT is the fixed part of the round trip delay and ICR_N the negotiated initial cell rate.

2.2.2 Switch Behavior

The switch behavior is equally important for the proper operation of a network supporting ABR service although it is not defined precisely by the ATM Forum. The duty of the switch is to monitor its queues, the incoming traffic rate or a mixture of both and provide information about the state of congestion to the sources by either binary feedback or an explicit rate, depending on the complexity choice of the implementor. This provision of information is accomplished by marking a single bit of the data cells, as in the case of binary schemes or by writing on the ER field of RM cells in the case of explicit rate switches.

The first generation switches use binary feedback for congestion control as these are the least complex mechanisms. However, due to their slow convergence and large buffer requirements it seems that the second generation switches will be based on more complex explicit rate schemes.

In addition to marking cells, the switches themselves might create and send back out-of-rate RM cells (BRM) in cases of extreme congestion, or perform Virtual Source/Virtual Destination behavior, that is, partition the control loop into smaller segments to reduce control delay. Moreover, per VC queueing can be implemented to isolate flows from each other. However, these are all optional, and left to the consent of implementors.

As a result of this flexible switch definition, switches made by different implementors can work together.

2.2.3 Destination End System Behavior

The final component of the feedback loop is the destination. The main duties of the destination are to monitor the EFCI fields of incoming data cells and return the FRM cells to the network after setting the direction (DIR) bit and the CI bit if necessary. In addition to these, the destinations might also reduce the ER of the incoming cells in case of internal congestion and create BRM cells.

Chapter 3

Simulation Models

In this study, our objective is a performance study of the ATM ABR congestion control framework. We will cover both ATM and transport layer performances, taking fairness and efficiency as the basis of our evaluation.

Due to the complexity and non-linearity of the systems under consideration, simulation is our preferred methodology. To this end, rate allocation graphs $ACR(t)$, queue size over time $Q(t)$, link utilizations and individual connection utilizations $U(t)$ are used. In addition, sequence numbers $SN(t)$ and response times are taken as measures of performance for the TCP and application layers respectively.

Throughout our simulations OPNET, an event-driven simulation tool, has been used. The SES, DES and switch components have been modeled on OPNET and mechanisms essential to the ABR congestion control framework as specified by the ATM Forum in [2] have been implemented to a great extent. The OPNET model library used in our studies was formed, taking the models used in [28] as a basis. Extensive modifications have been done over these modules, and new modules have been created.

Two network topologies were used for the experiments. The R_1 topology, which is seen in Fig. 3.1 was used to test mainly the max-min fairness of the allocation algorithms. Sensitivity to parameters was tested over the second network R_2 (Fig. 3.2). The end-stations, whose names start with xmt are the sources ends and the others are destination ends. The first digit in a name refers to the number of hops and the second one is used for identifying different end-systems with VCs traversing the same number of hops. Each virtual circuit is named with the extension of its source-destination pair.

All the links are 155 Mb/s duplex links. The propagation delay for links are $\tau_p = 1\ \mu s$ for links between switches and end systems, $\tau_p = 0.01\ ms$ for inter-switch links in LAN configurations and $\tau_p = 1\ ms$ for WAN inter-switch links. Error rates on the links have been taken as 0.

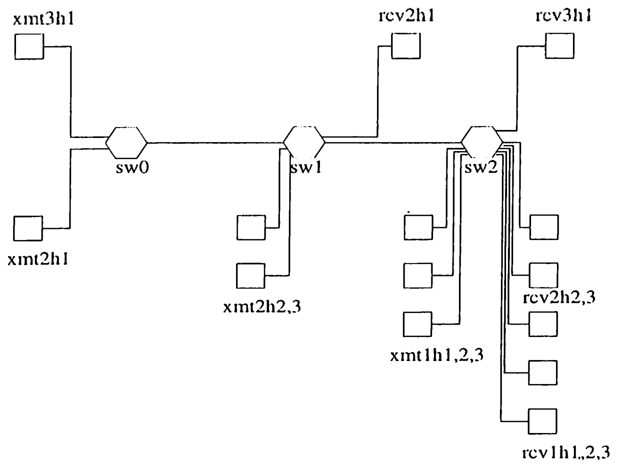


Figure 3.1: R_1 configuration

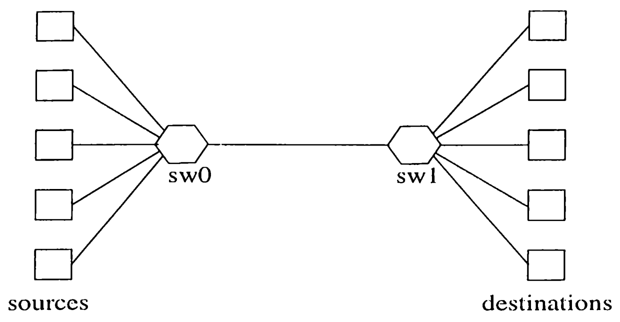


Figure 3.2: R_2 configuration

In the following sections, the virtues and limitations of the modules used are examined.

3.1 Source End Systems

The specification regarding SES behavior has been implemented except out-of-rate RM cells, TBE negotiation and FRTT computation. In addition to rate adaptation, the time-out mechanisms are implemented. The source parameters in Table 3.1 are used unless otherwise stated.

PCR	353,208 (cells/s)
MCR	4717 (cells/s)
ICR	70641.6 (cells/s)
RIF	0.0625
RDF	0.0625
CDF	0.0625
CRM	40
Nrm	32
Mrm	2
Trm	100 (ms)
ADTF	500 (ms)

Table 3.1: Source Parameters

In order to avoid unfairness resulting from cell drops at the switches, the transmission times of the sources are randomized by the random shift $\dot{x}$, which is uniformly distributed between 0 and $0.01 \cdot PCR^{-1}$. This corresponds to throwing an N faced dice when N cells arrive at the same time to the switch. If the switch is full, one cell is randomly kept, and the others are discarded. The motivation for such an approach is to prevent a single connection from always beating down the others.

Each source end comprises of a flow regulator, a rate controlled queue and a traffic generator. The flow regulator updates the ACR of the controlled queue with respect to network feedback, as given in Table 2.1. The controlled buffer creates RM cells, performs scheduling for transmission and time-outs. All the SES are assumed to be compliant sources, as network feed-back is fully obeyed. Thus a UPC mechanism is not implemented. In fact, the controlled queue can be thought of as the UPC mechanism. The traffic generators used in the simulations are either persistent cell generators, or TCP generators, whose packets are segmented to ATM cells. TCP sources can be bursty, and persistent depending on the inter-arrival times at the application layer. In addition, there exist CBR, VBR and bursty ABR sources in our OPNET library.

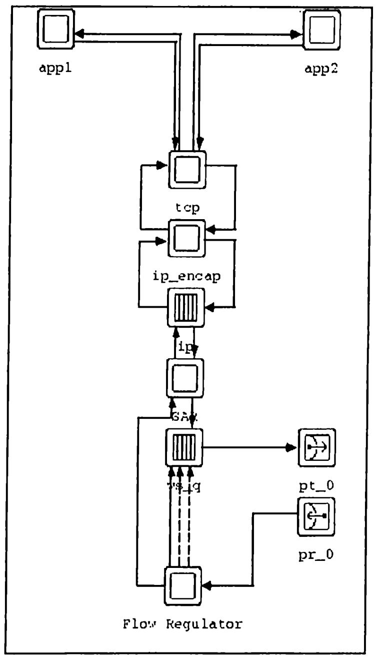


Figure 3.3: A typical TCP source

In case of TCP sources, the cell generator is replaced by a TCP source that includes a full protocol stack from the application layer to the IP layer. A typical TCP source is seen in Fig. 3.3.

The TCP layer of the OPNET simulation tool is used in the network model. This module, which is based on *RFC 793* and *RFC 1122* supports "end-to end reliability based on ACKs and retransmissions triggered by exponentially backed-off timers, where the retransmission time-outs are adaptively computed from segment round-trip times"[37]. The features include slow-start, Nagle Silly Window Syndrome Avoidance, Jacobson's algorithm for RTT estimation and re-sequencing. Transmissions are dynamically limited based on the availability of remote buffering resources. Connections are established by a three-part hand-shake.

Fast retransmit/recovery, protection against wrapped around sequence numbers

(PAWS), TCP time-stamps option and quiet time mechanism are not implemented, neither are TCP checksums computed.

Another lacking feature is the timer granularity. In real-life timers that are used for TCP time-out mechanisms, there exists a difference between the "ideal" timer expiration and the actual time that is processed, due to the sampling of the system clock, which can be as large as 200 to 500 ms. In addition to these, the processing delays for the TCP layer are not taken into account.

The TCP module has several parameters that can be changed, namely receive buffer size, maximum segment size, initial retransmission time-out, maximum ACK delay, persistence timeout and some RTT estimation parameters. In addition to these Nagle SWS Avoidance and end-to-end delay measurement can be turned on and off.

The IP layer is used only for encapsulation of the TCP segments. The OPNET IP model set allows the processing rate and the queueing capacity to be set. Even though the IP module set has a routing capability, it will not be used since whole network is composed of ATM components, and routing is performed at the ATM layer.

The segmentation and reassembly module simply segments packet's arriving from a higher layer into 48 byte cells and adds a header of 5 bytes, and performs re-assembly when cells are received from the network. No processing delay is taken into account for this module.

3.2 Switches

Input buffered switches with three separate queues, one for each service class, namely ABR, VBR and CBR is used in the simulations. All the cells belonging to a class are put into a single queue, and each queue is served in a FIFO fashion. The CBR queue has priority over the VBR queue, and the VBR queue is not served unless the CBR queue is empty. In the same manner the ABR queue is not served unless both VBR and CBR queues are empty. The buffer sizes at the switches are 128 cells for VBR and CBR cells and infinite for ABR cells, unless otherwise stated. Buffer requirement:

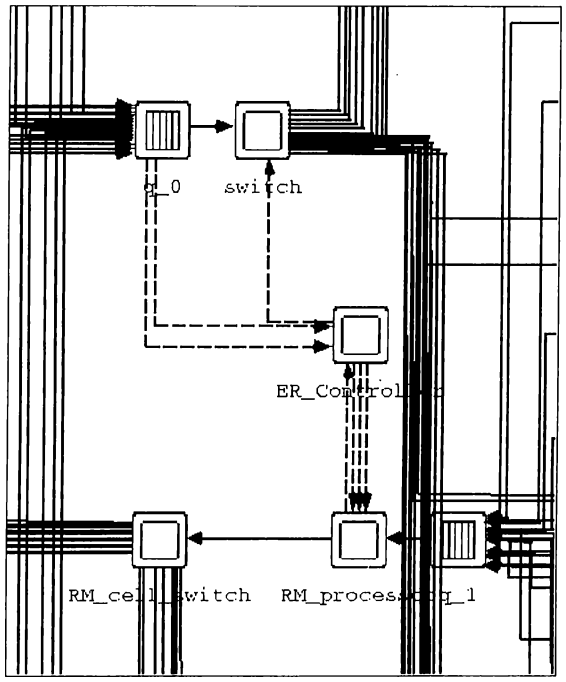


Figure 3.4: ATM switch

Our switch implementation and its surrounding elements for rate allocation are illustrated in Fig. 3.4. The main functions of the switches are to route cells to their destinations, to detect congestion and to allocate rates to connections depending of the state of congestion. The EPRCA, ERAA, relative marking and intelligent binary marking schemes have been implemented for congestion control.

The relative marking algorithm uses two thresholds for congestion detection and clearance. These thresholds are $LT = 50$ and $HT = 100$. The decision of the switch when $Q(t) > HT$ is to declare congestion, setting $CI = 1$, whereas congestion is cleared when $Q(t) < LT$. When $Q(t)$ is in-between, the NI bit is set to 1.

The EPRCA algorithm uses the same thresholds. The other parameters of EPRCA are set at: $\alpha = 0.0625$, $ERF = 0.875$, $DPF = 0.875$, $MRF_{LAN} = 0.75$ and $MRF_{WAN} = 0.25$.

The implementation of the ERAA algorithm has deviated from the original one to an extent. As connection set-up procedures do not exist in our modules, we do not update the number of connections before starting the transmission, hence do not allow the queues to drain in this period. Thus, the queue-length results that we obtain are larger than the values attainable by using the original version. Instead, we use a scaling factor λ to make the maximum utilization smaller than 1, which in turn ensures the stable operation of the queues.

Traffic flow in the forward direction and BRM flow in the reverse direction have been assumed. Currently 20 ABR, 5 VBR and 3 CBR connections can be supported by each switch. The support of bi-directional traffic and increase in the number of connections are possible with minor modifications.

3.3 Destination End Systems

At the destination, the RM cells are turned around and sent back with the CI bit set if the EFCI bit of the last data cell was 1. No other action is taken at the destination. It should be noted that the TCP source module in Fig. 3.3 also acts as a destination.

Chapter 4

Evaluation of Rate Allocation Algorithms

ABR sources are not capable of describing their bandwidth requirements precisely. Moreover the bandwidth that is available to the ABR users changes over time due to the co-existence of higher priority traffic and the opening and closing of ABR connections. Under these non-stationary conditions, the network provides proper operation, thanks to the rate-based feedback algorithm.

The congestion control algorithm implemented at the switch has a major role in the extent to which the network resources are used efficiently and allocated fairly. In this section, performances of enhanced proportional rate control algorithm (EPRCA), efficient rate allocation algorithm (ERAA) and two binary schemes, namely relative and intelligent marking schemes will be examined. The evaluation of these algorithms at the network layer have been performed on the basis of ability to provide max-min fairness, and efficiency in using the resources. In addition, the effects of propagation delay, control parameters and ICR have been examined.

4.1 Binary Schemes and Their Deficiencies

Binary schemes form the least complex mechanisms for flow control. They make use of one bit information for regulating the flow rates of the connections. In the simplest case, when the queue length of a switch is above a predefined threshold value, the switch informs sources, by setting the EFCI bits of the incoming data cells. When a destination end system receives an RM cell, it copies the EFCI of the last data cell to the CI field of the first RM cell that arrives. Finally, the source increases its rate additively if the CI bit is clear, and decreases it multiplicatively otherwise.

These simple mechanisms have several drawbacks. The first and perhaps the most important, the convergence to optimal rates is not guaranteed, that is fairness is not guaranteed. Usually, the connections that traverse fewer links are discriminated against those that traverse many links. This is called the *beat-down problem*. The explanation of the beat-down problem is straight-forward. If the probability of a cell being marked at a switch is p, then this probability is

$$P(marked) = 1 - (1 - p)^n.$$

for a connection traversing n switches. This probability is roughly np for small p.

The relative marking algorithm has been used to illustrate an instance of this problem in the R_1 configuration. With this algorithm, switches mark the CI and NI fields of the RM cells in the backward direction according to their congestion states. The algorithm and the parameters used are provided in Table 4.1. The resulting allowed cell rates and individual utilizations are shown in Fig. 4.1. While the first graph shows the rates allowed by the network for each VC, the latter indicates the utilizations for each VC resulting from transmitting at that rate. Given that the max-min fair rates are $\frac{C}{2}$ for VC_{2h1} and $\frac{C}{6}$ for the rest, it is obvious that the allocated rates for the VC_{2h1} and one-hop connections are above their fair levels, and below fair levels for the rest of the connections.

The queue sizes and link utilizations over time for relative marking can be seen in Fig. 4.2. The queues are not stable, since the algorithm is not able to exactly match the inflow rates to the service rate of the queues using one bit of information. In addition, there is the control delay, which makes the task harder. The high link utilizations in Fig. 4.2.b are attributable to VC_{2h1} which grabs about 90 % of the bandwidth.

The beat-down problem can be alleviated either using per VC-queuing, which is a demanding solution in terms of hardware requirements, or by performing intelligent marking, that is regulating the rates of connections that transmit above their fair rates. The second solution makes use of a scheme that calculates fair rates for connections, and reduces the rates of only those whose rates are in excess of their

Switch Behavior:	**Simulation Set-up:**
if $Q(t) \geq HT$	configuration: R1
set CI=1	$LT = 50$ cells, $HT = 100$ cells
if $Q(t) < LT$	$RIF = RDF = 0.0625$
congestion cleared	$ICR = 70641$ cells/s,
else	$MCR = 4717$ cells/s
set NI=1	$PCR = 353208$ cells/s
	starting times:
SES and DES Behavior:	3h1 at $t = 0$
	2h1 at $t = 20ms$
As stated in Section 2.2	2h2, 2h3 at $t = 40ms$
	1h1, 1h2, 1h3 at $t = 60ms$

Table 4.1: The pseudo-code and simulation parameters used for simulating the relative marking algorithm

fair rates.

$ACR(t)$ and $Q(t)$ for the intelligent marking solution on the same topology are illustrated in Fig. 4.3. The algorithm and parameters for this scheme are provided in Table 4.2. As seen in Fig. 4.3.a, the beat down problem is alleviated. Still, there exist other problems. The convergence to the optimal rates is gradual since the source is expected to increase the rate additively or decrease it multiplicatively depending on the value of the binary feedback. It takes several round-trips for convergence to optimal rates. As a result, the response time to congestion is long and oscillatory behavior is observed both for $ACR(t)$ and $Q(t)$. Moreover Q_{max} takes fairly large values. A large Q_{max} is a disadvantage since for a given buffer size, a larger Q_{max} implies a higher CLR. When one considers that the higher layers retransmit the segments that are not received completely, it is clear that a high CLR might lead to a throughput collapse.

The binary algorithms are not robust at all, and are sensitive to the control parameters, RIF and RDF, and the congestion thresholds. Hasegawa et al [11] have shown the importance of tuning these parameters depending on the buffer size and delay for ensuring fairness and better throughput for higher layers. Increases

Switch Behavior:

```
Estimate fair share:
    $MACR = (1 - \alpha)MACR + \alpha CCR$
    $fair\_share = MACR \times DPF$

if $Q(t) \geq HT$
    set CI=1 for all connections
if $Q(t) < LT$
    congestion cleared
else
    set CI=1 for VC with $CCR > fair\_share$
```

SES and DES Behavior:

As stated in Section 2.2

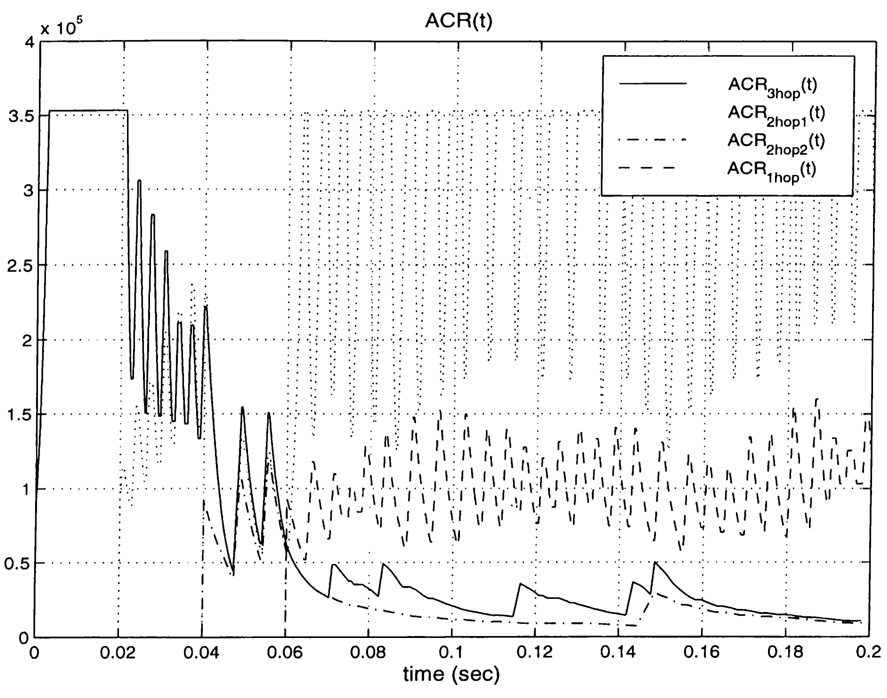
Simulation Set-up:

configuration: R1
$LT = 50$ cells, $HT = 100$ cells
$RIF = RDF = 0.0625$
$\alpha = 0.0625$, $DPF = 0.875$
$ICR = 70641$ cells/s
$MCR = 4717$ cells/s
$PCR = 353208$ cells/s
starting times:
3h1 at $t = 0$
2h1 at $t = 20ms$
2h2, 2h3 at $t = 40ms$
1h1, 1h2, 1h3 at $t = 60ms$

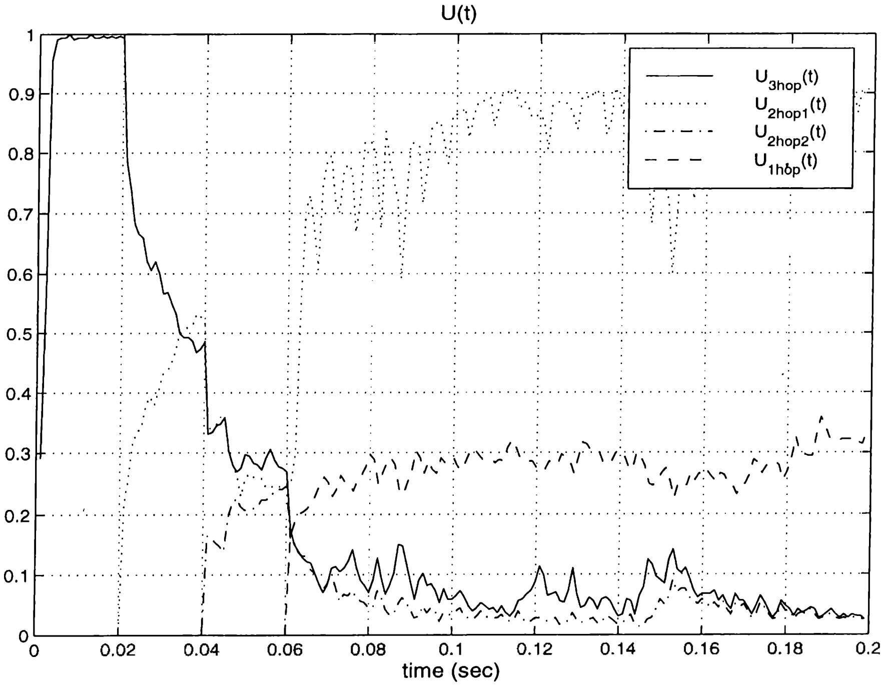
Table 4.2: The pseudo-code and simulation parameters used for simulating the intelligent marking algorithm

in these two parameters disturb the stability of the system. Moreover an increase in the buffer requirements is induced by an increase in RIF and decrease in RDF. Fig. 4.4 and Fig. 4.5 illustrate the simulation results over the R_2 regarding these problem. The first of the figures illustrates the dependence of the maximum buffer size on the source parameters and on ICR. The second one indicates $U_i(t)$, the utilization of the bottleneck link by each connection, for $RIF = RDF = 0.015625$ and $RIF = RDF = 0.125$.

As seen, the only motivation for using binary feedback is very low computational complexity. The provision of explicit rate information is an alternative to the binary feedback. Even though switches performing such a function are more complex compared to the binary switches, they may provide significant advantages in terms of fairness and efficiency. In the next sections, two of these algorithms, namely EPRCA and ERAA will be discussed.

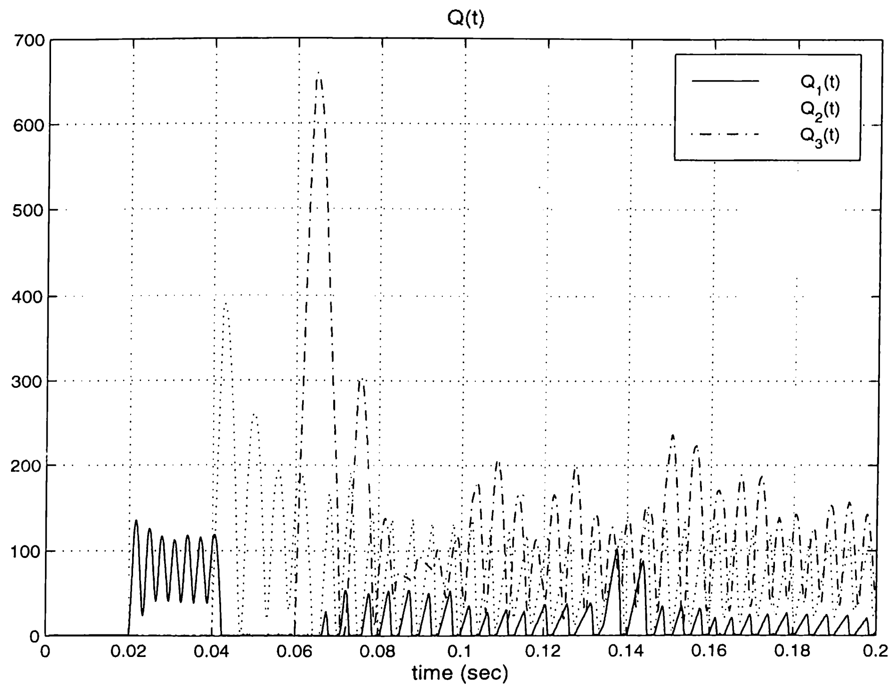


(a)

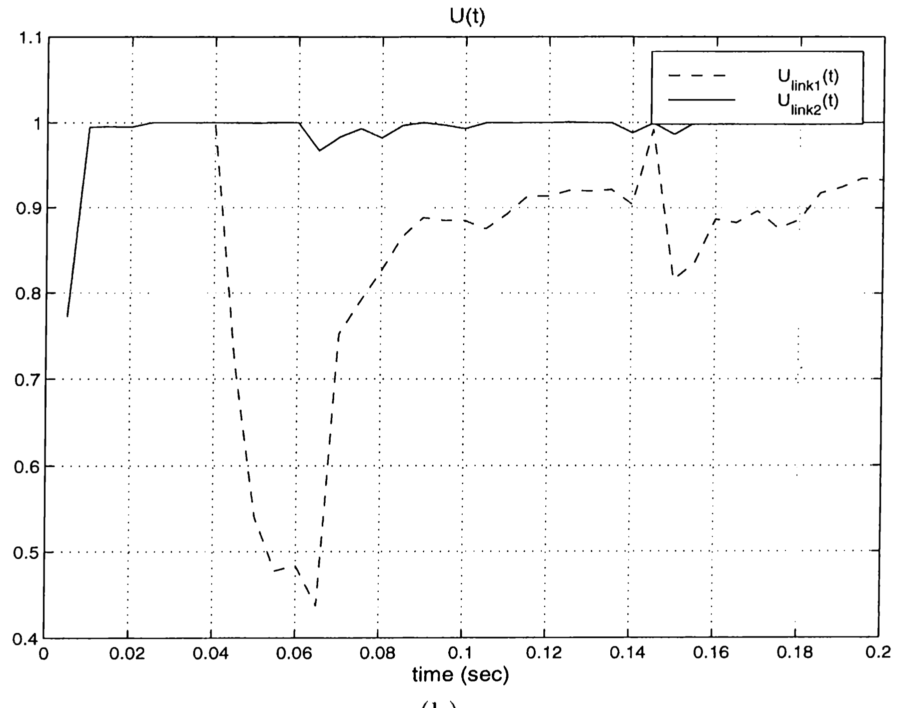


(b)

Figure 4.1: Beat-down problem for relative marking, (a) shows allowed cell rates and (b) utilizations by VCs

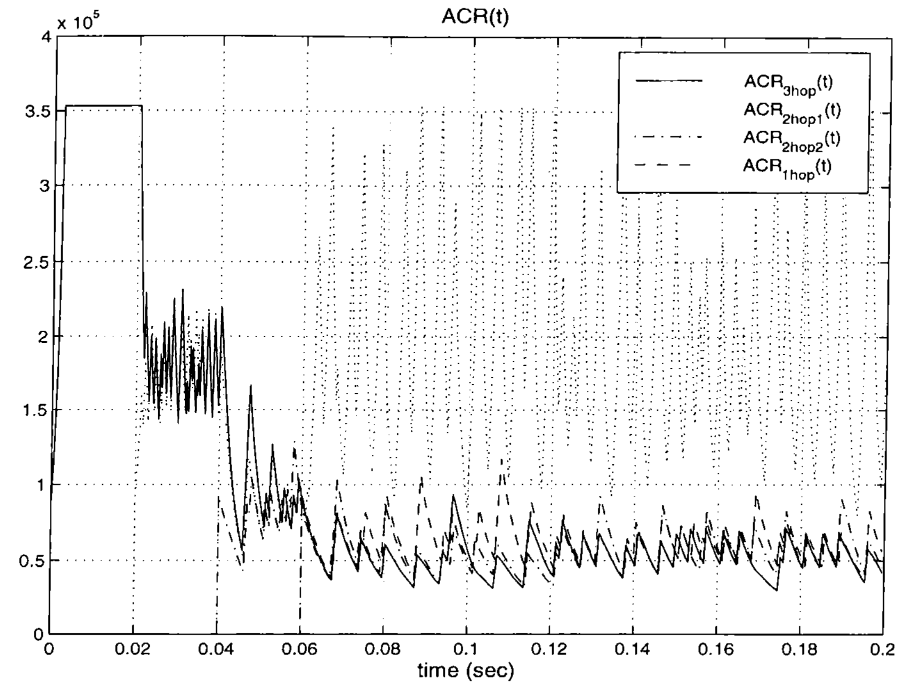


(a)

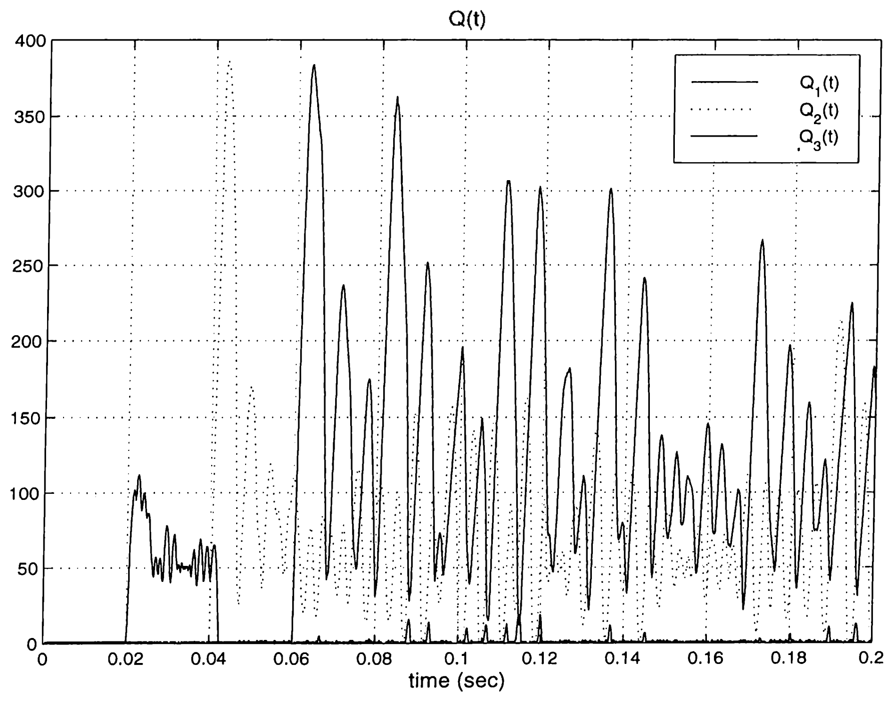


(b)

Figure 4.2: Queue size evolution (a) and link utilization (b) for relative marking

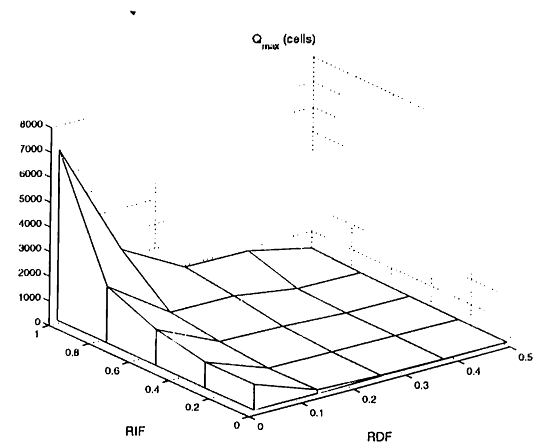


(a)

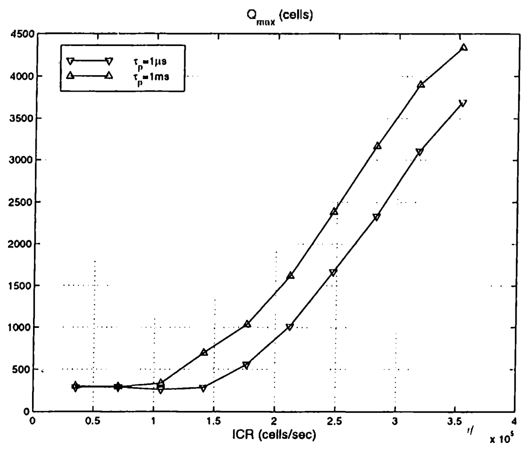


(b)

Figure 4.3: $ACR(t)$ (a) and $Q(t)$ for intelligent binary marking

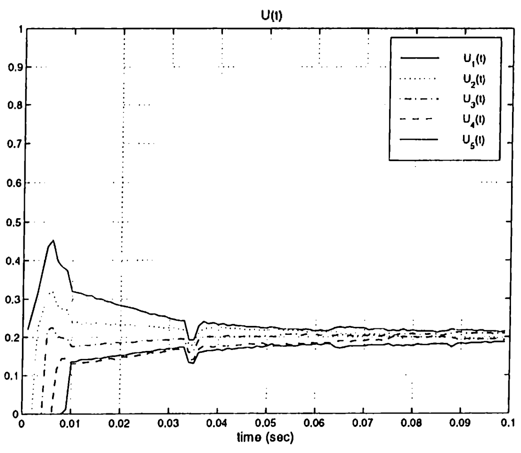


(a)

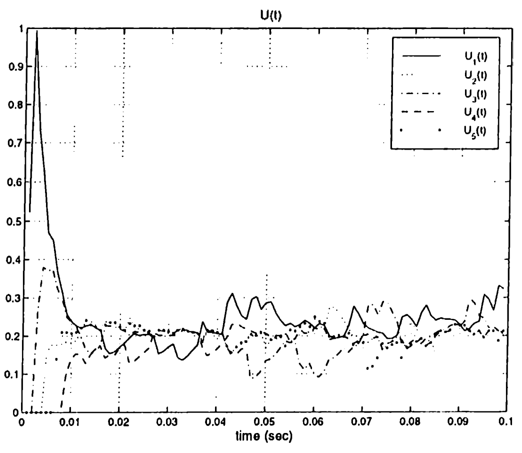


(b)

Figure 4.4: Effects of source parameters *RIF*, *RDF* (a) and *ICR* (b) on buffer requirements



(a)



(b)

Figure 4.5: Effects of source parameters on utilizations $U_i(t)$

4.2 Enhanced Proportional Rate Control Algorithm (EPRCA)

EPRCA is an *approximate fair rate calculation* algorithm [24, 38]. It is supposed to converge to the fair bandwidth allocation by performing intelligent marking. The decision for congestion is taken using two thresholds on queue lengths and a mean allowed cell rate ($MACR$) is computed after the reception of each forward RM cell. $MACR$ is a running exponential average that is computed as follows :

$$MACR = (1-\alpha)MACR + \alpha CCR$$

where α is the averaging factor. The fair share is computed as

$$fair\_share = MACR \times DPF$$

When the queue length $Q(t) \geq LT$, the decision is the existence of slight congestion, thus the rates of connections with $CCR \geq fair\_share$ are decreased to $MACR \times ERF$. When $Q(t) \geq HT$, the switch is in severe congestion and all the rates are reduced to $MACR \times MRF$. The rate information is marked in the ER field of the RM cells, and is signaled in the backward direction. The pseudo-code for EPRCA is given in Table 4.3.

The most important advantage of this algorithm is low complexity. The price paid for this low complexity is strong dependence on the control parameters, the possibility of convergence to rates other than the fair allocation and severe oscillations. There have been several improvements on this algorithm ([26], [25]), nevertheless the convergence to optimal rates takes time and fairness is not guaranteed under all circumstances. It is stated in [39] that even though the EPRCA and its enhanced versions can provide max-min fairness to the users, there can be cases where max-min fairness is violated. During our simulations, we have not encountered any case leading to such results.

The R_1 configuration has been used to examine the fairness, transient and steady state performances. Fig. 4.6 illustrates the allocated cell rates $ACR(t)$ and individual utilizations $U_i(t)$ over time. It is seen that max-min fairness is achieved in the

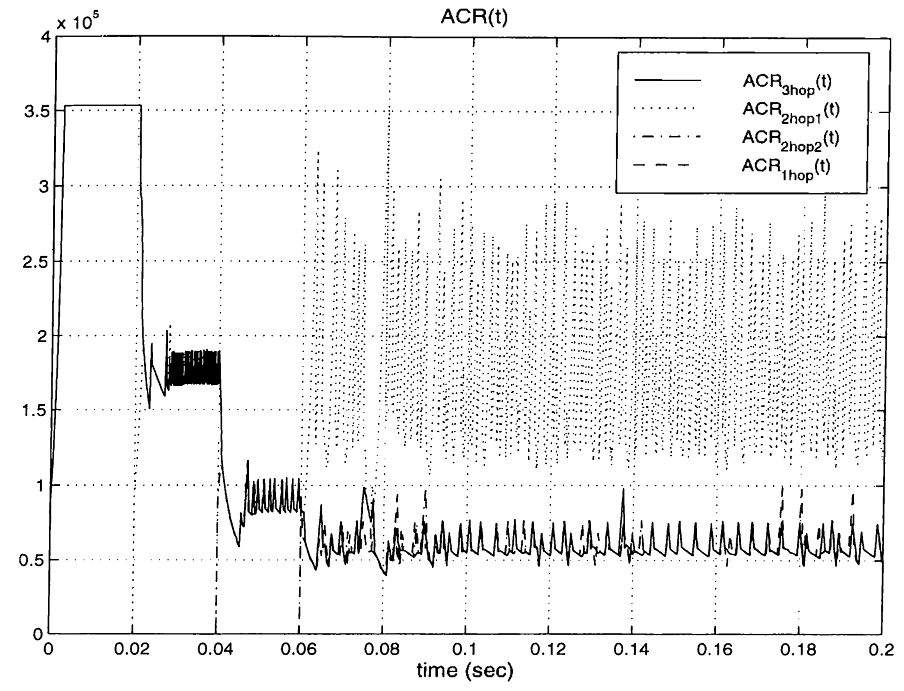
Switch Behavior:	**Simulation Set-up:**
Estimate fair share:	configuration: R1
$MACR = (1-\alpha)MACR + \alpha CCR$	$LT = 50$ cells, $HT = 100$ cells
$fair\_share = MACR \times DPF$	$RIF = RDF = 0.0625$
	$\alpha = 0.0625$, $ERF = 0.875$
if $Q(t) \geq HT$	$DPF = 0.875$
decrease all rates to $MACR \times MRF$	$MRF = 0.75$ in LANs
if $Q(t) < LT$	$MRF = 0.25$ in WANs
congestion cleared	$ICR = 70641$ cells/s
else	$MCR = 4717$ cells/s
set the rates of VCs with	$PCR = 353208$ cells/s
$CCR > fair\_share$ to	starting times:
$MACR \times ERF$	3h1 at $t = 0$
	2h1 at $t = 20ms$
SES and DES Behavior:	2h2, 2h3 at $t = 40ms$
As stated in Section 2.2	1h1, 1h2, 1h3 at $t = 60ms$

Table 4.3: The pseudo-code and simulation parameters used for simulating EPRCA

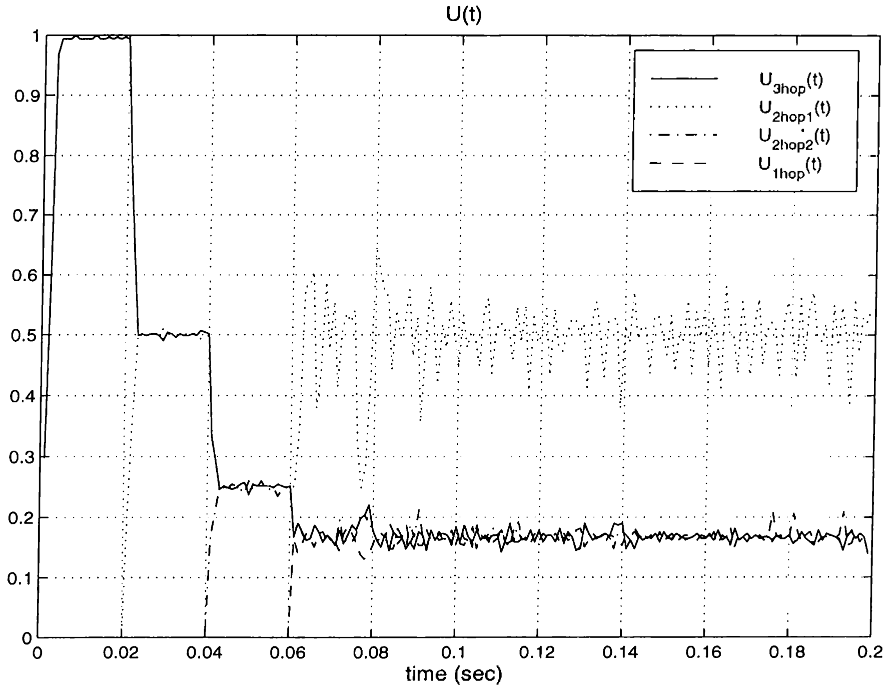
mean sense, in spite of the oscillations that exist at steady state. This oscillatory behavior can be reduced to some extent by careful tuning of the parameters, but may not be avoided completely. Fig. 4.7.a shows that the links are used efficiently by EPRCA.

Fig. 4.7.b illustrates the $Q(t)$ function over time. From this graph, we see that EPRCA can work with smaller buffers, compared to the binary marking schemes. Still, this is also related to the appropriate setting of the parameters. As opposed to the binary schemes, the buffers are never empty, and exhibit oscillations with small amplitudes around the queue thresholds.

Strong dependence of EPRCA performance on the choice of the control parameters is a major problem inherent to this algorithm. The first parameter that is effective on performance is the *rate increase factor* (RIF). Larger values of this parameter imply faster response to changes, however such values degrade the steady state performance by worsening the oscillations in $ACR(t)$ and $Q(t)$, and lead to a

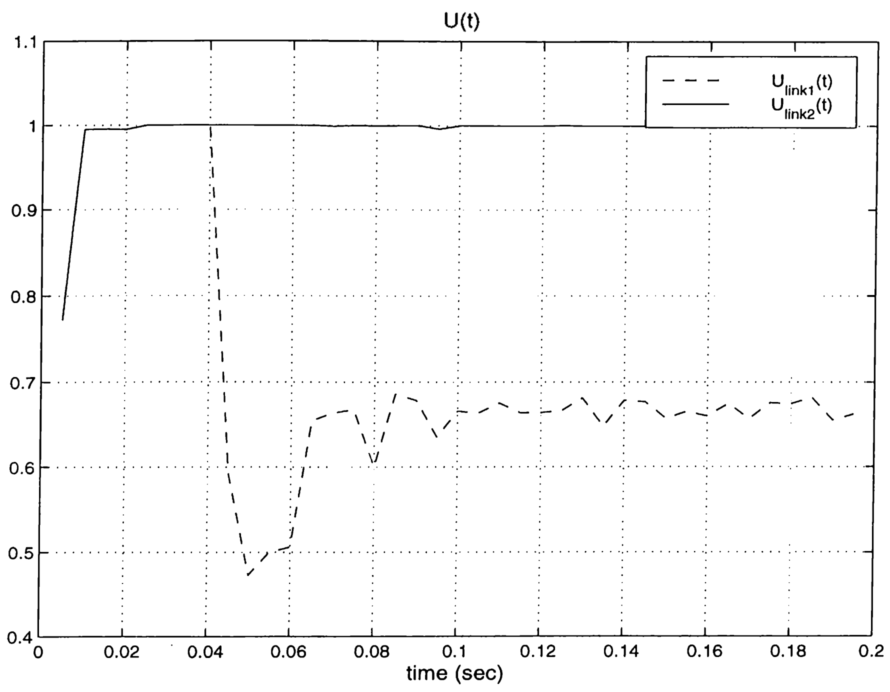


(a)

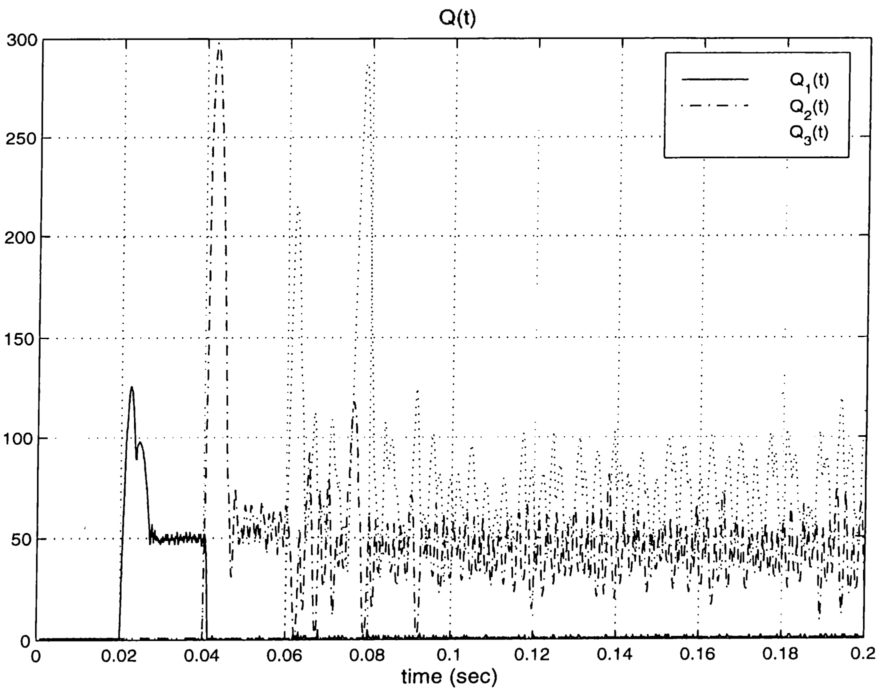


(b)

Figure 4.6: $ACR(t)$ (a) and $U(t)$ (b) with EPRCA over the R_1 configuration



(a)



(b)

Figure 4.7: Link utilizations (a) and buffer requirements (b) for EPRCA over the R_1 configuration

higher Q_{max}, as was the case in binary feedback systems. This fact is seen in Fig. 4.8. Two simulations were run on the R_2 configuration with RIF and RDF equal to 1/4 and 1/64. It is seen that, for larger values of RIF, the oscillation amplitudes increase, and the buffer requirements are higher. Fig. 4.9.a illustrates further the dependence of Q_{max} on the RIF. The second SES parameter RDF does not effect the performance since explicit rates are fed back.

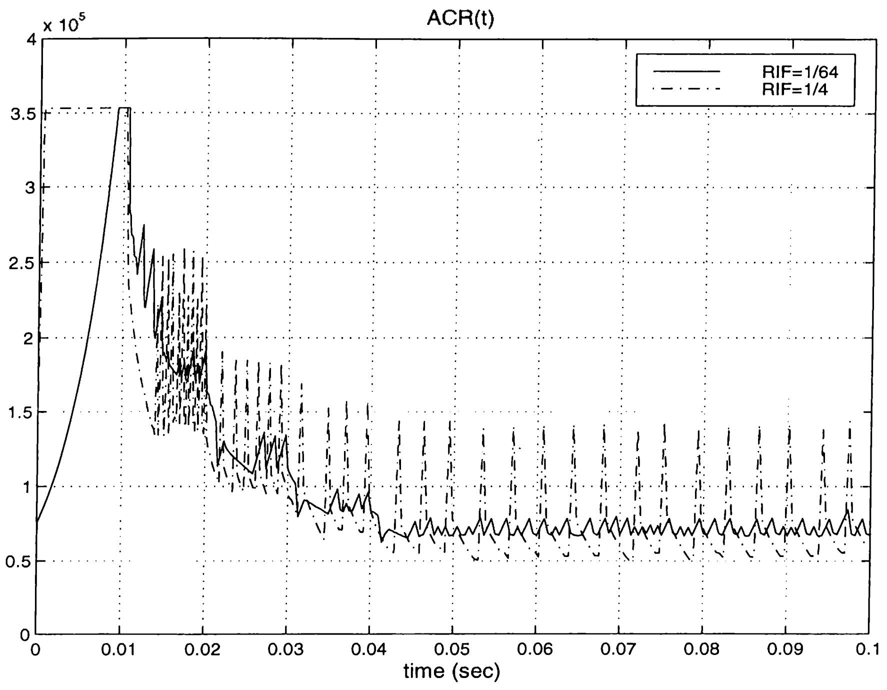
In addition to the source parameters RIF and RDF, EPRCA has its own parameters, namely, α, ERF, MRF, DPF, LT and HT. Among these, the averaging constant α determines the weight of the new CCR values in the $MACR$ estimate. It determines the responsiveness of the algorithm along with MRF and ERF. ERF is the rate reduction factor for moderate congestion levels and MRF is used for reducing rate in cases of more severe congestion. The value of MRF is chosen depending on the inter-switch distances, lower for high τ_p and vice versa. The reason for such a choice is to avoid buffer overflows that may result due to late feed-back. DPF is used to avoid potential oscillations that can be caused by connections with CCR close to $MACR$. Finally, the choice of LT effects the mean queue length, larger values implying larger queuing delays.

With a good parameter setting, the maximum buffer size should be observed at transience, since the rates have not reached the optimal values yet. The buffer requirements depend on ICR and the RTT which is the time required for the feedback to arrive at the source. Fig. 4.9.b shows this relation for the R_2 configuration. Q_{max} stays constant till a threshold is exceeded and increases with ICR after this threshold. This threshold is the idle capacity. Obviously, transmission at a rate above the link capacity would swell the queues until the reception of feedback from the network. When we check the propagation delay dimension of the picture, we see that the larger value of delay allows for more cells to be transmitted before information arrives and thus to a larger Q_{max}.

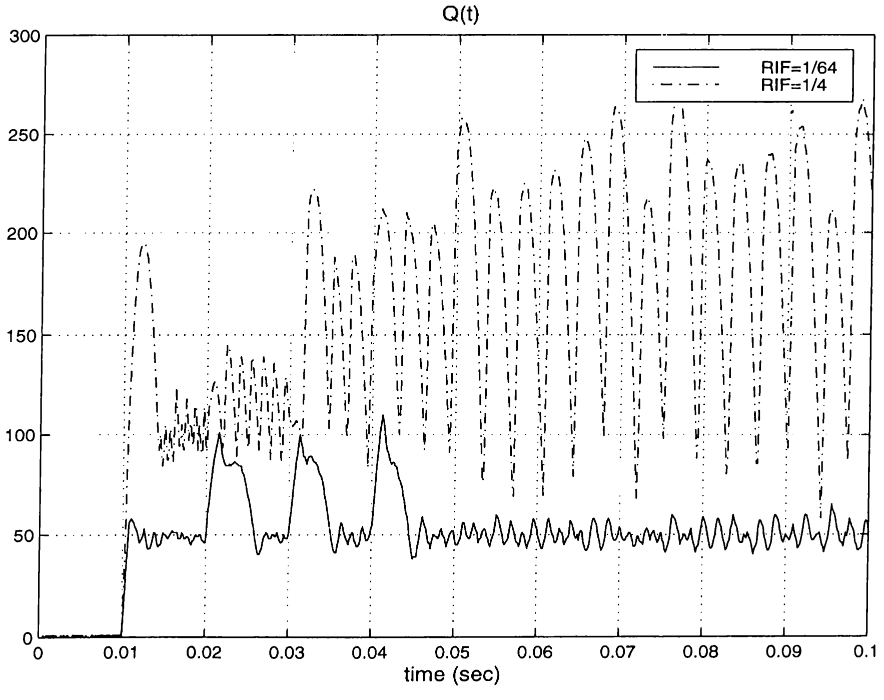
The performance of EPRCA is distorted in a WAN due to the delay between the fair rate estimation and the reaction by the source to this estimate. Oscillation amplitudes and buffer requirements are increased, still max-min fairness is achieved in the mean sense. Rate allocations and buffer use are illustrated in Fig. 4.10.

To summarize, EPRCA is a simple rate allocation algorithm. It is not robust, especially when the propagation delay is large, that is in the WAN environments.

Still, if the parameters are well tuned, it provides better performance compared to the binary feedback algorithms.

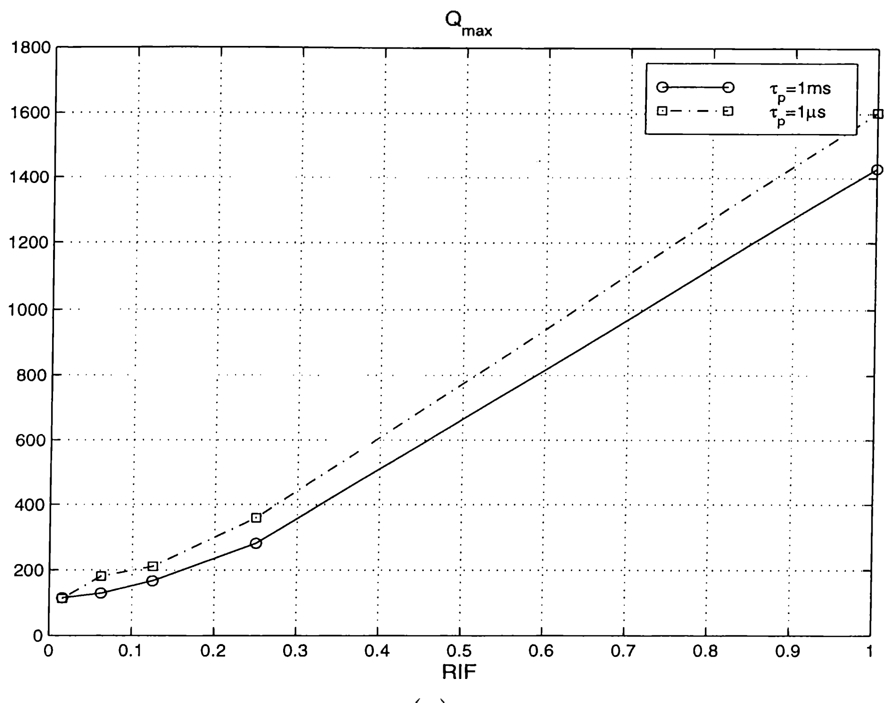


(a)

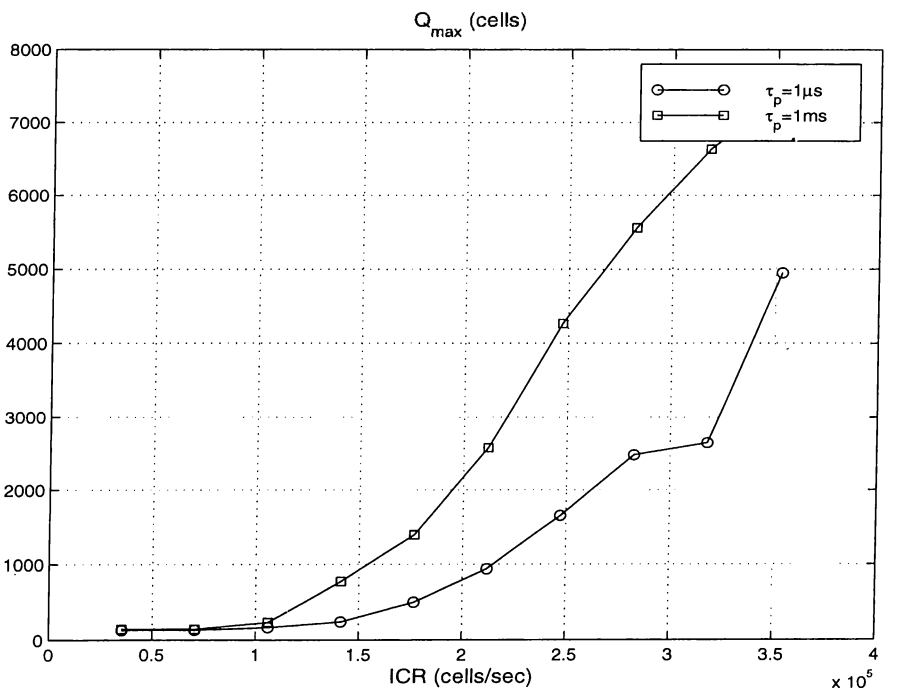


(b)

Figure 4.8: Effects of RIF on transient and steady state behaviors of $ACR(t)$ (a) and $Q(t)$ (b)

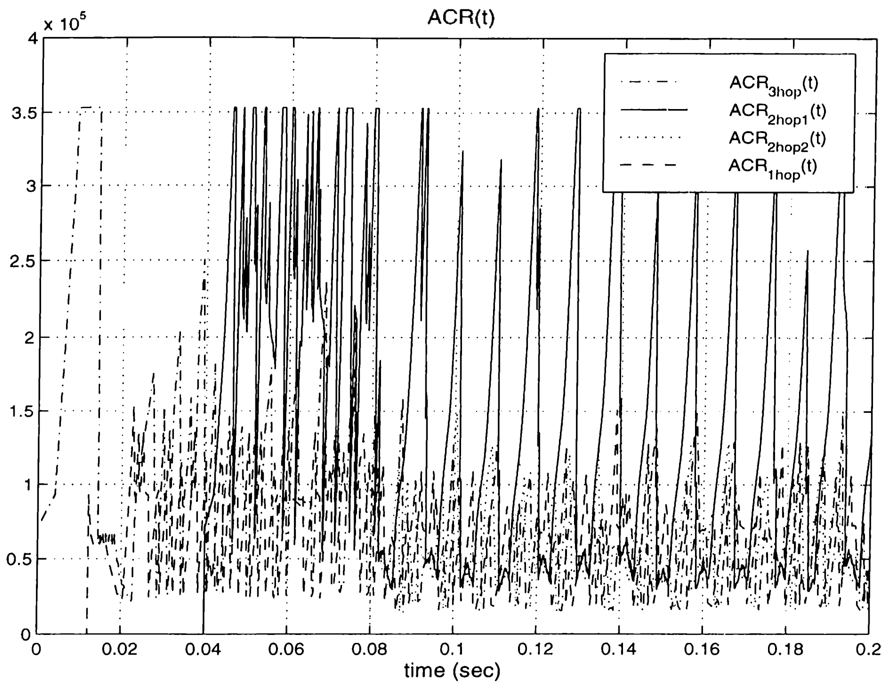


(a)

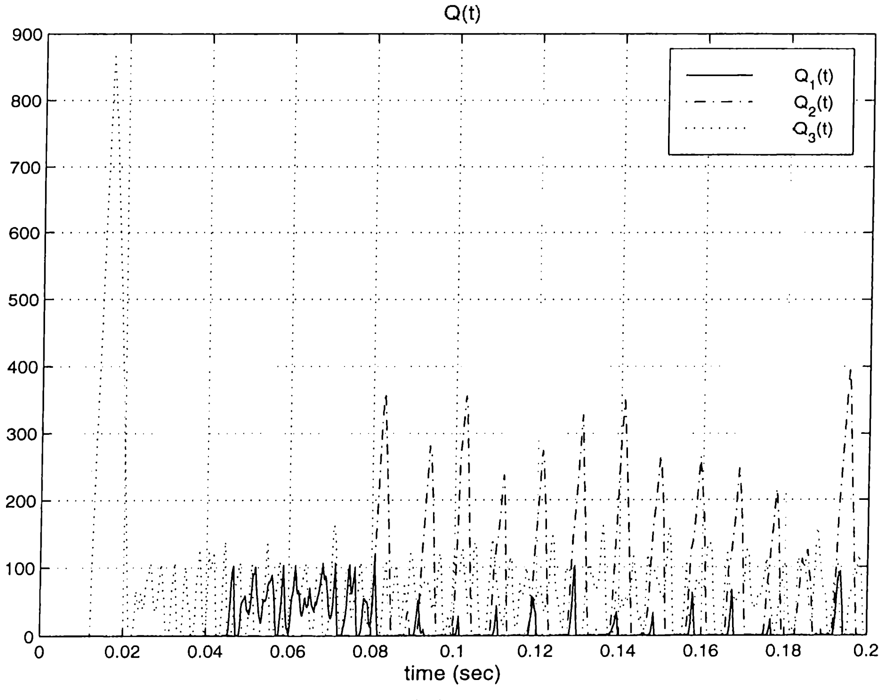


(b)

Figure 4.9: Effects of *RIF* (a) and *ICR* (b) on EPRCA buffer requirements



(a)



(b)

Figure 4.10: EPRCA performance in a WAN in terms of rate allocation (a) and buffer use (b)

4.3 Efficient Rate Allocation Algorithm (ERAA)

ERAA [17] is an exact rate allocation algorithm, an improvement to the CCERI scheme proposed by Charny [29]. This algorithm distributes the bandwidth that is available for the ABR service fairly to the users, in the sense of max-min fairness. This is achieved by a distributed algorithm of storage complexity $O(N)$ and computations complexity $O(1)$. The worst case convergence time of the algorithm is on the order of $O(M(2D + \frac{1}{R}))$, where D is the round trip delay, M is the number of distinct values in the max-min allocation, and R is the RM cell rate.

Switches performing ERAA differentiate between the bottle-necked and satisfied connections. This decision is based on $\rho_j(t) = min(ER_j(t), CCR_j(t))$, where $ER_j(t)$ is the desired rate and $CCR_j(t)$ is the current rate. When the request $\rho_j(t)$ is larger than the maximum allocation $A_{max}(t)$, a connection is marked as bottle-necked and satisfied otherwise.

Even though all the sources are entitled to the same bandwidth, $B_{eq}(t) = B(t)/N(t)$, the satisfied ones are limited in their use of this equal share, due to being bottle-necked elsewhere in the network. Thus a fraction of the capacity is available for the use of others. This fraction is named the free bandwidth and is equal to

$$B_f(t) = N_u(t)B_{eq}(t) - \sum_{i \in S_u(t)} A_i(t). \tag{4.1}$$

After the satisfied connections receive their shares, the bottle-necked ones each receive an equal share given by

$$A_{max}(t) = B_{eq}(t) + \frac{B_f(t)}{N_b(t)} \tag{4.2}$$

where $B(t)$ is the total bandwidth available for ABR use, $A_i(t)$ is the allocation for the connection i belonging to the set of satisfied connections $S_u(t)$ and $N_b(t)$ is the number of bottle-necked connections.

The first step performed on the reception of an RM cell in the forward direction is to check whether the state of that connection has changed. For a bottle-necked connection, $A_{max}(t)$ is compared with $\rho_j(t)$ to see whether $A_{max}(t) < \rho_j(t)$ still holds. When that is not the case, the state of that connection is changed to satisfied.

For the satisfied connections, a similar procedure is followed. The state of the connection is temporarily changed to bottle-necked and the bandwidth that would be allocated to it in that case is determined by

$$A_{max}(t) = B_{eq}(t) + \frac{B_f(t) + A_j(t) - B_{eq}(t)}{N_b(t) + 1.} \tag{4.3}$$

If $A_{max}(t) < \rho_j(t)$, then the new state of the connection is bottle-necked. For the sake of simplicity, Eq. (4.2) and (4.3) are merged into

$$A_{max}(t) = B_{eq}(t) + \frac{B_f(t) + A'_j(t) - B_{eq}(t)}{N_b(t) + w.} \tag{4.4}$$

where

$$A'_j(t) = \begin{cases} A_j(t), & \forall j \in S_u(t) \\ B_{eq}(t), & \forall j \in S_b(t) \end{cases}$$

and

$$w = \begin{cases} 1, & \forall j \in S_u(t) \\ 0, & \forall j \in S_b(t). \end{cases}$$

The state update is followed by an update in the allocation for the connection j and the free bandwidth. The new allocation for $A_j(t+)$ for connection j is the minimum of the rates $A_{max}(t)$ and $\rho_j(t)$

$$A_j(t+) = min(A_{max}(t), \rho_j(t)).$$

$B_f(t)$ is also updated as

$$B_f(t+) = B_f(t) + A'_j(t) - A'_j(t+). \tag{4.5}$$

The next step is to update the rate that will be fed-back to the source. If $A_{max}(t) < \rho_j(t)$, the ER field is changed, otherwise, it is left unchanged. As a result of this update, the rate allocated to a certain connection is equal to the minimum of the rates provided by the switches along the path. The pseudo-code for ERAA is given in Table 4.4.

Switch Behavior:

```
$\rho_j = min(ER_j, CCR_j)$
$B_{eq} = B/N$
if $state_i = bottlenecked$
    $A_{max} = B_{eq} + B_f/N_b$
else
    $A_{max} = B_{eq} + (B_f + A'_i - B_{eq})/(N_b + 1)$
$A_{max} = max(A_{max}, B_{eq})$
if $(A_{max} < \rho_j)$ and $(state_i = satisfied)$
    $state_i = bottlenecked$; $N_b = N_b + 1$
if $(A_{max} > \rho_j)$ and $(state_i = bottlenecked)$ and $(i <> MaxVC)$
    $state_i = satisfied$; $N_b = N_b - 1$
if $state_i = satisfied$
    $A_i = \rho_i$
else
    $A_i = A_{max}$
$ER = min(A_{max}, ER_i)$
if $(\rho_j > MaxAlloc)$
    $MaxVC = j$; $MAxAlloc = \rho_j$
    if $state_j = satisfied$
        $state_j = bottlenecked$;$N_b = N_b + 1$
if $(MaxVC = j)$ and $(rho_j < MaxAlloc)$
    $MaxAlloc = \rho_j$
$A_{old} = A'_j$
if $(state_j = satisfied)$
    $A'_j = A_j$
else
    $A'_j = B_{eq}$
$B_f = B_f + A_{old} - A'_j$
```

SES and DES behavior:

As stated in Section 2.2.

Table 4.4: The pseudo-code and simulation parameters used for simulating ERAA

For the proper operation of the mechanism, several parameters have to be kept in addition to the states of connections. These are available bandwidth $B(t)$, free bandwidth $B_f(t)$, total number of connections $N(t)$, the number of bottle-necked connections $N_b(t)$ and $A'_j(t)$. In addition, the index of the connection receiving the maximum rate and its rate is held in order to prevent misallocation of resources when all the connections are satisfied, and the state of this connection is marked as bottle-necked even if its request is satisfied completely.

As opposed to EPRCA and binary marking algorithms, ERAA has the ability to allocate the link capacity with max-min fairness at all times, except at the transient periods. The former two were not guaranteed to converge to max-min rates, and when they did, they achieved max-min fairness only in the mean sense. The transient and steady state behavior of this algorithm regarding the allowed cell rate $ACR(t)$ and individual utilizations $U(t)$ for the R_1 configuration are illustrated in Fig. 4.11. No oscillations are observed at steady state, and links are fully utilized, as allowed by max-min allocations (Fig. 4.12.a).

When the target bandwidth is taken as equal to the trunk capacity, the buffer of the bottleneck switch neither drains, nor swells as the inflow is equal to service rate, when the sources are all persistent sources. As the utilization is %100, the buffers may become unstable. In transient periods, buffer overflows can take place since the new connection pumps cells into the network at a rate of ICR, which makes the total inflow rate

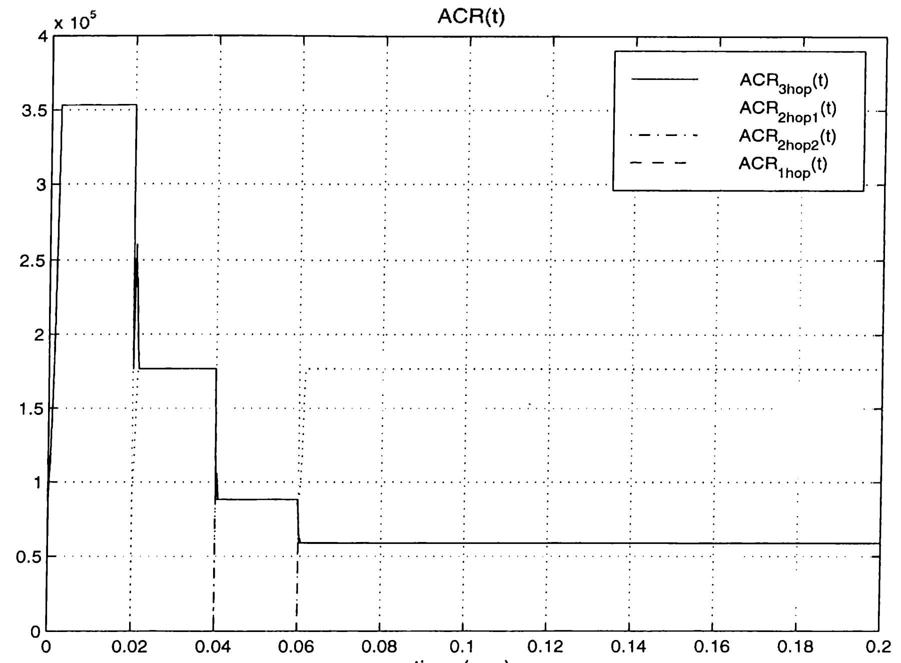
$$\sum_{i \in S(t^-)} ACR_i + ICR_j > B(t),$$

where $S(t^-)$ is the set of connections prior to the opening of connection j. The queue-length evolution for the R_1 configuration is illustrated in Fig. 4.12.b.

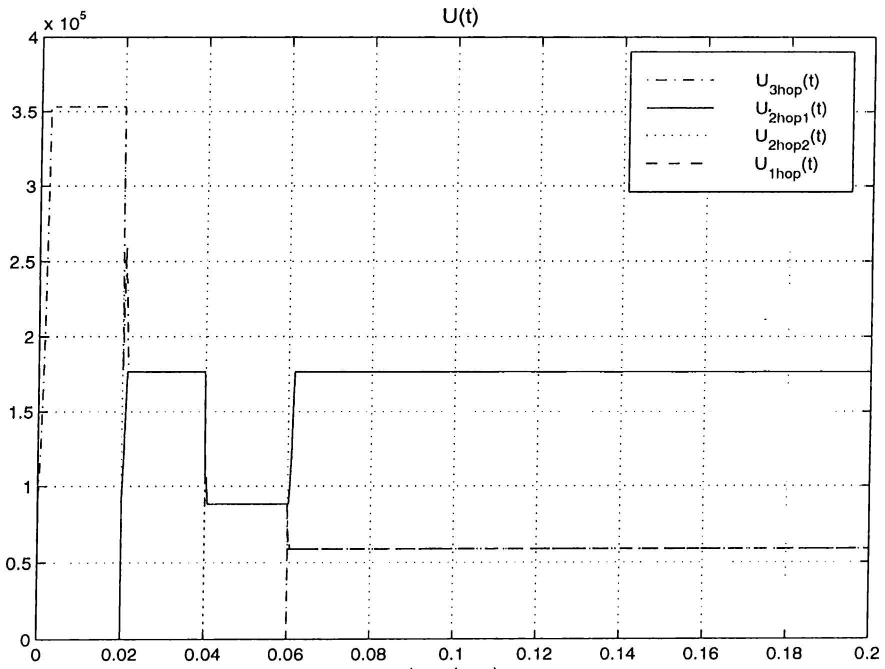
A precaution that has been taken by the original algorithm has been to update $N(t)$ and $N_b(t)$ at connection set-up, decreasing the rate allocations to the connections before the new connection starts transmission, hence allowing the queues to drain. The fraction of the capacity left idle until the new connection starts transmission is equal to

$$\frac{1}{N+1} B(t).$$

Thus, when the number of connections is small and the round trip delay is large as in the case of a WAN, this approach might lead to under-utilization of the links.

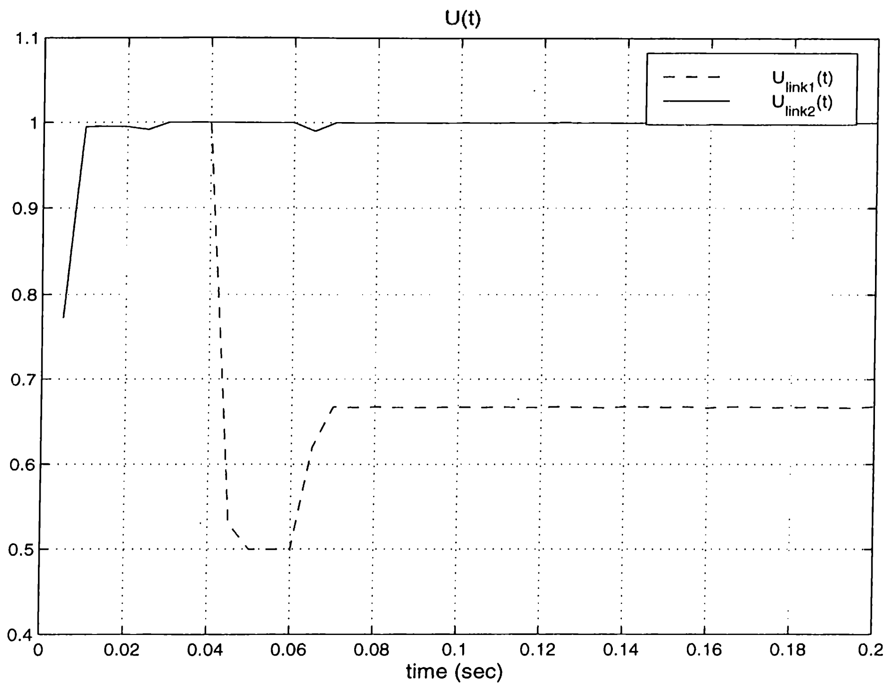


(a)

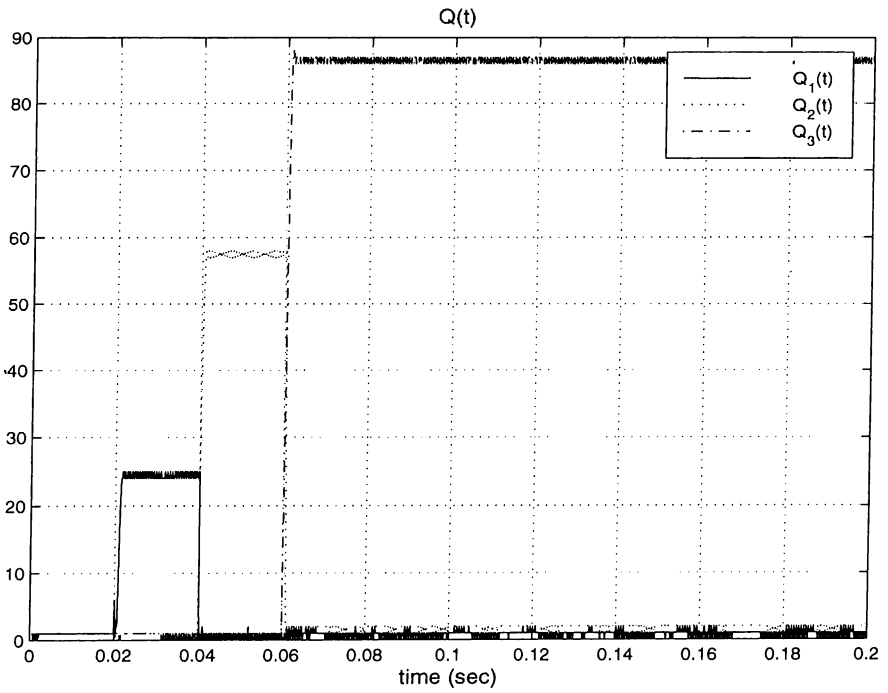


(b)

Figure 4.11: Fairness for ERAA: (a) allowed cell rates (b) utilizations for each VC



(a)



(b)

Figure 4.12: Buffer requirements (a) and link utilizations (b) for ERAA

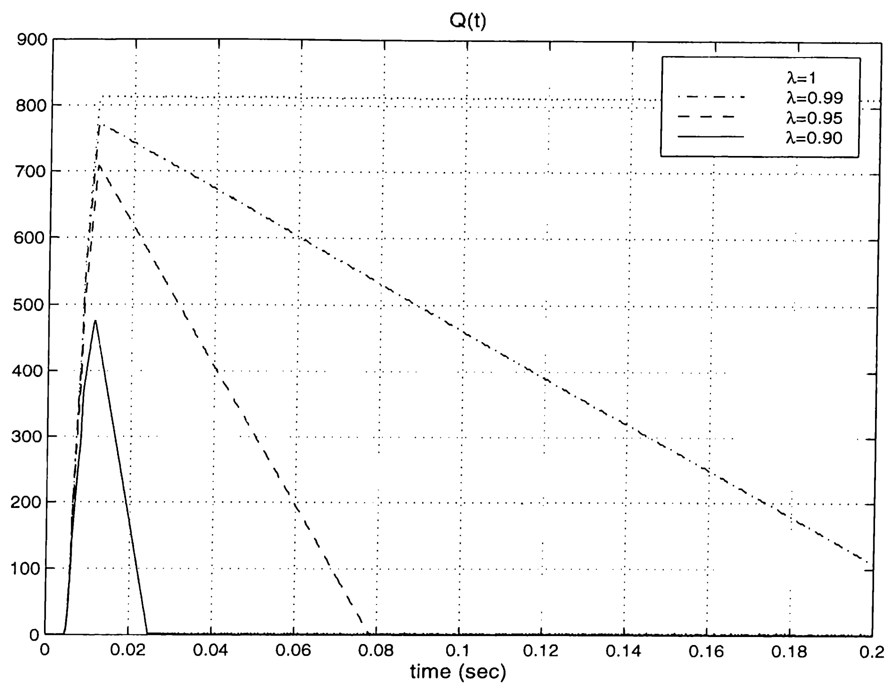


Figure 4.13: Effect of λ on buffer requirements

Still, if the number of connections is large and round-trip delay is small, that is a feasible mechanism.

Other approaches to providing stable operation might be to use a constant load factor λ smaller than one, or to adaptively change the λ depending on the queue occupancy. Fig. 4.13 illustrates the effect of λ on the buffer requirements in the R_2 configuration with staggered sources with τ_p = 1ms, i.e., in a WAN environment. In our simulations, a constant load factor $\lambda = 0.95$ is used to provide stable operation, hence network stability and buffer requirements are traded off with utilization.

An important feature of ERAA is its robustness. The algorithm itself does not have any parameters that require tuning. Moreover, the control parameters do not effect the steady state performance of this algorithm. Fig. 4.15 shows the effect of RIF on convergence to fairness and on buffer requirements. Higher RIF values can be used for faster convergence and higher utilization without degrading the steady state performance. As observed in the second part of this figure, even though RIF increases the buffer requirement, it is not as effective as for the binary schemes and EPRCA.

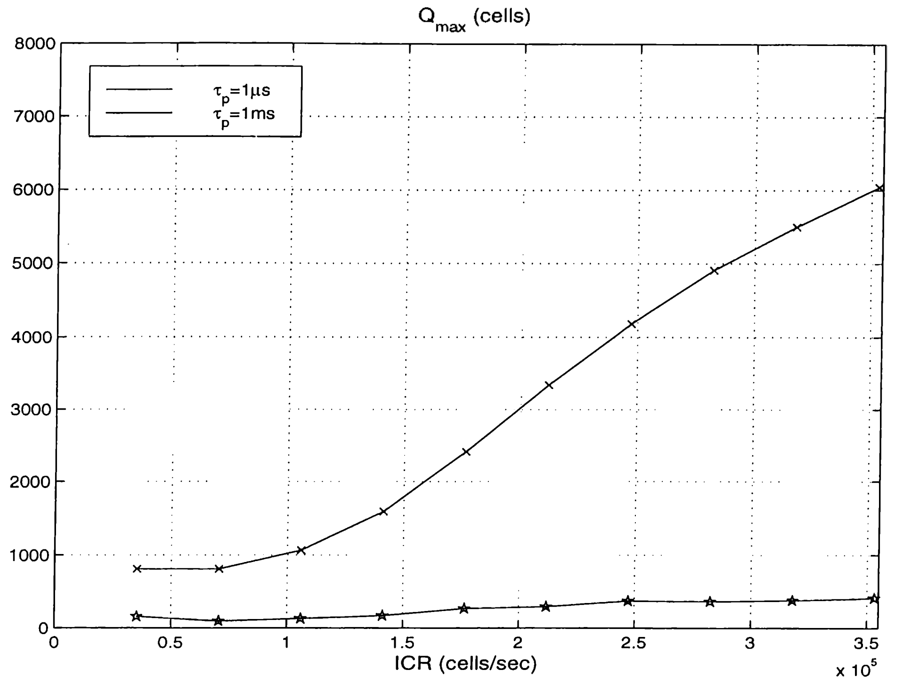


Figure 4.14: Effect of ICR on maximum queue size

Fig. 4.14 illustrates the dependence of Q_{max} on the initial cell rate (ICR). As seen, the dependence is weak in the LAN environment due to the fast settling of the rates. However, Q_{max} takes large values when both ICR and τ_p are large. As ERAA leads to full utilization of the links, new connections overload the network till the new max-min allocation is reached. With a larger $RTT \times ICR$, more excess cells are accumulated at the bottleneck switch buffers.

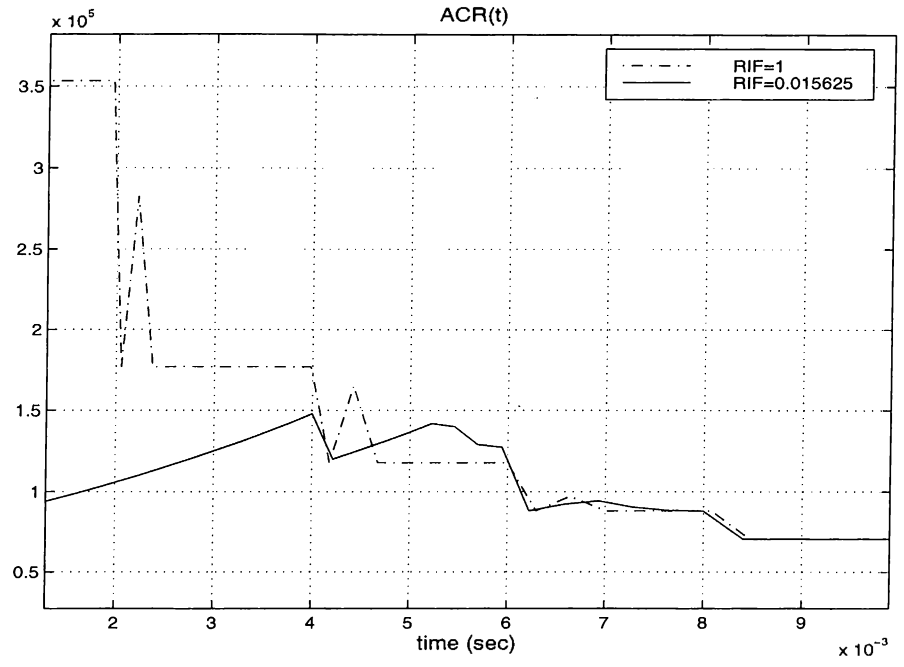
ERAA also works efficiently in the WAN environment. When the propagation delay is large between the end systems, congestion feedback arrives in a longer time interval, thus convergence takes a longer time. Nevertheless, at steady state, max-min rates are achieved. As it takes a longer time for the connections to detect the opening of a new connection, Q_{max} is larger for a WAN. Fig. 4.16 illustrates the rate allocations and the buffer sizes over time.

A major reason for queue growth in WANs is the marking discipline used by ERAA. The cells are marked in the forward direction, thus the information that reaches the SES is rather old. This can be alleviated to a certain extent by marking the cells in the backward direction. The ER value of the RM cells in the backward direction can be compared to the new A_{max} value, and reduced when $ER > A_{max}$.

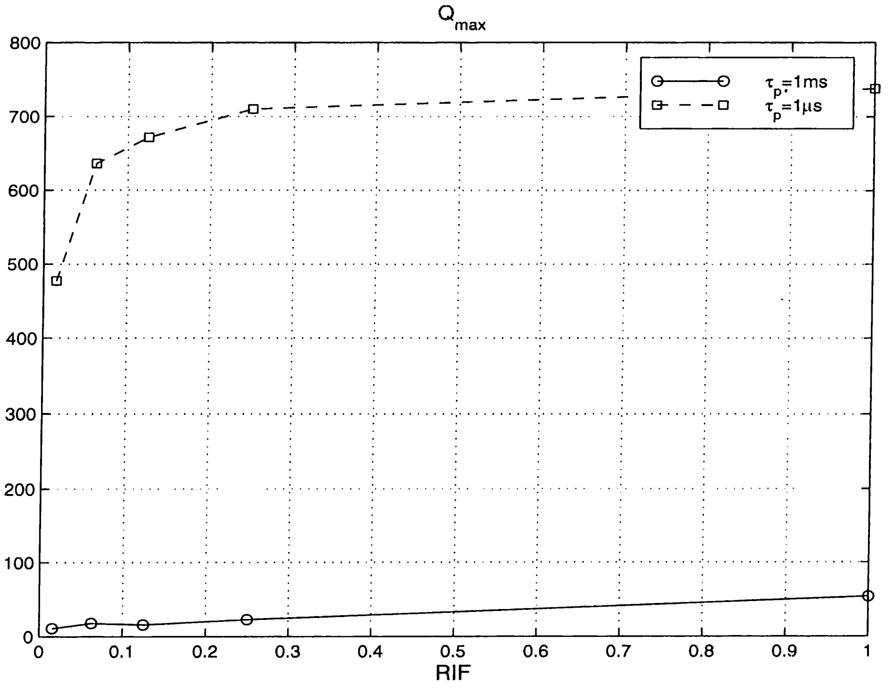
Such a solution comes at an additional cost, since ports in the reverse direction need to be informed about A_{max}.

Another feature of this algorithm is that it partitions the available bandwidth $B(t)$ into K sub-channels. While that can be considered an advantage in case the sources are all persistent, when a connection is idle or bursty, the other connections can not increase their rates to capture part of this bandwidth. That poses a problem for the efficient operation of ERAA. This is alleviated by fast max-min rate allocation algorithm (FMMRA) which does not rely on CCR values in the header.

To summarize, ERAA compensates for the additional computational complexity and storage requirements by achieving max-min fairness at all times at steady state, and keeping the mean delay low. It is a robust and efficient scheme for rate allocation.

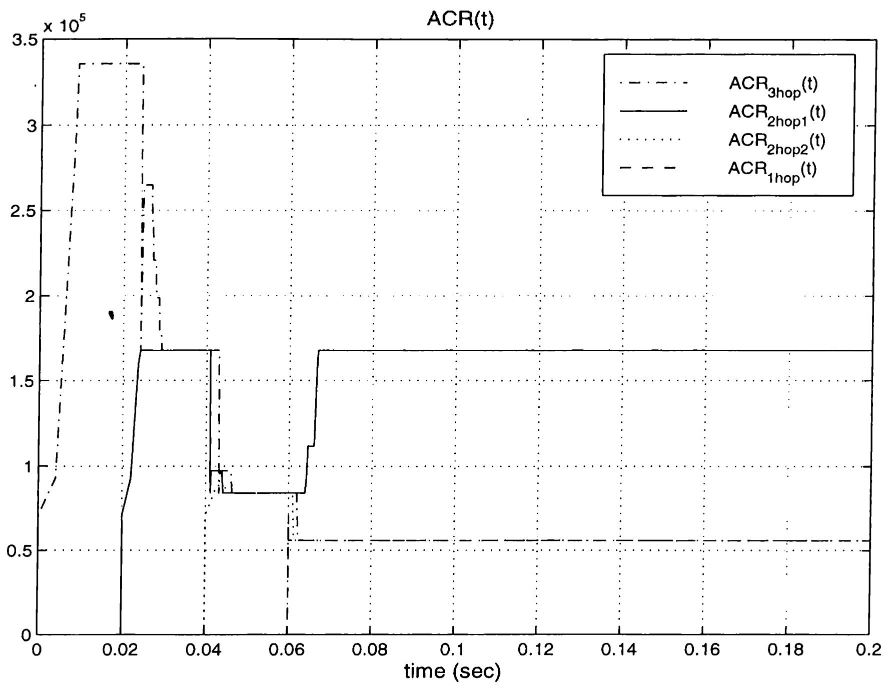


(a)

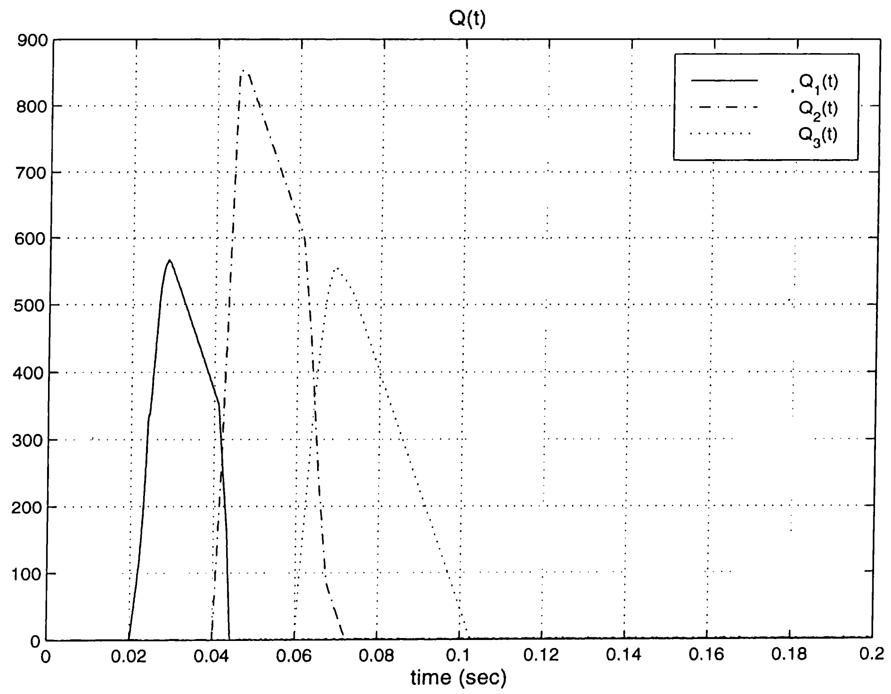


(b)

Figure 4.15: Effect of RIF on transient and steady state performance



(a)



(b)

Figure 4.16: The performance of ERAA over a WAN

Chapter 5

TCP/IP Performance Over ABR

5.1 Issues on TCP/IP over ATM

The transport control protocol (TCP) is the most commonly used transport layer as it forms the standard for the current Internet. It was primarily designed to work over connectionless networks, providing reliable delivery to the application layers. To accomplish this aim, an end-to-end sliding dynamic window mechanism is used. Lost packets are retransmitted after time-out periods. This window mechanism also has the effect of performing congestion control since in cases of congestion the ACKs return slower, thus the transmissions by the sources are throttled.

The *slow start algorithm* and round trip estimation for time-outs were added to increase the effectiveness of the TCP congestion control mechanism [40]. The source starts with a window size $W(t)$ equal to the maximum segment size (mss), and increases $W(t)$ by mss for every ACK received until a fixed threshold, ss_thresh, is reached. This corresponds to an exponential increase. Above this threshold, window size is increased by $\frac{mss^2}{W(t)}$ for each ACK. In addition to the window mechanism, RTT is estimated. These estimates are used to set timers, upon the expiration of which retransmission occurs. In later versions, fast retransmit and fast recovery mechanisms have been devised with the motivation to avoid waiting for the timers to expire and to sense packet losses faster.

Fragmentation problem is observed, when TCP packets are fragmented into smaller network entities. When these smaller units of transmission are lost, the corresponding packets need to be retransmitted due to the integrity requirement in data transmission. As the fragments belonging to different packets are interleaved over the links, at times of congestion, it is highly probable that fragments belonging to different packets will be lost, thereby leading to low good-put values. With smaller fragments, this situation becomes worse, as in the case of ATM. As shown in [8], a congestion collapse is possible when TCP is run over ATM UBR without any control mechanisms. In the same paper, EPD and PPD mechanisms have been proposed. However as shown in [11, 10], such an approach does not enforce fair operation, and requires pretty large buffers. ABR provides better performance by dynamically regulating the entry rates into the network, thereby keeping buffer sizes low.

For TCP/ABR, there exist two mechanisms for congestion control, one at the transport layer and one at the data link layer. Depending on the phase of the connection, one of the two controls becomes dominant. At the beginning of a connection, the transmission rate of a source is first limited by the window-size of the slow-start algorithm, assuming reasonable values for mss and the ICR. After a few round trip delays, when the TCP window is large enough to fill the virtual pipe for the corresponding VC with $C(t) = ACR(t)$, the rate control algorithm takes over. This evolution from being *window controlled* to *rate controlled* can be seen in Fig. 5.1. Thus, the allowed rate of transmission for connection i is given by,

$$R_i(t) = \min\left(\frac{W(t)}{RTT}, ACR(t)\right). \tag{5.1}$$

As seen from Eq. 5.1, unfairness against connections with larger RTTs is possible until the window size grows large enough to fill the bit pipe, or until the rate control takes over. When the dominant control mechanism is the ABR control, more aggressive sources are limited by the upper bound $ACR_i(t)$ and less aggressive ones are allowed to increase their rates up to the corresponding rate, allowing fairness at the TCP and application layers. Even though unfairness might be observed during the window controlled period, the throughput observed by the upper layers at steady state is fair, given that the maximum window size is sufficient to fill the corresponding virtual pipe. This evolution is illustrated in Fig. 5.2 for the R_1 configuration with large propagation delays between the switches. A small maximum segment size of 512 Bytes has been used to make the situation more clear. As seen in this figure, the sequence number $SN(t)$ for the 1-hop connection increases faster than for the one for the 3-hop connection for a while, but after a certain point, the two lines become parallel to each other.

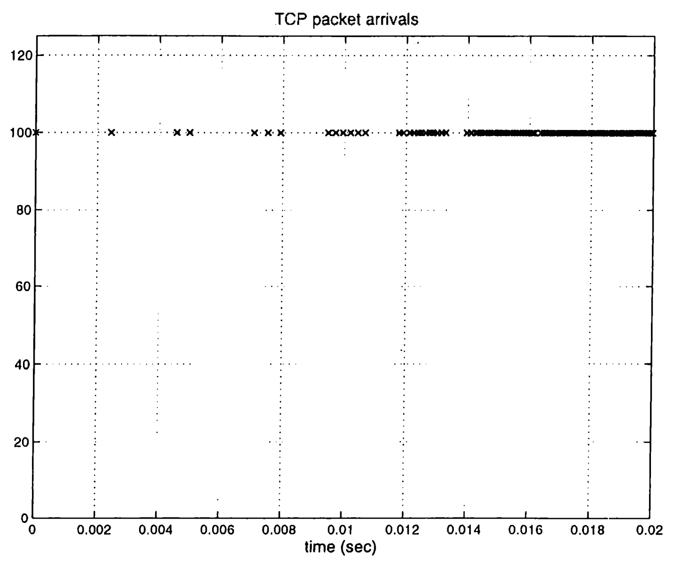


Figure 5.1: The transmission is first limited by window size and then by the allowed cell rate. Crosses indicate TCP packet arrivals at the IP layer

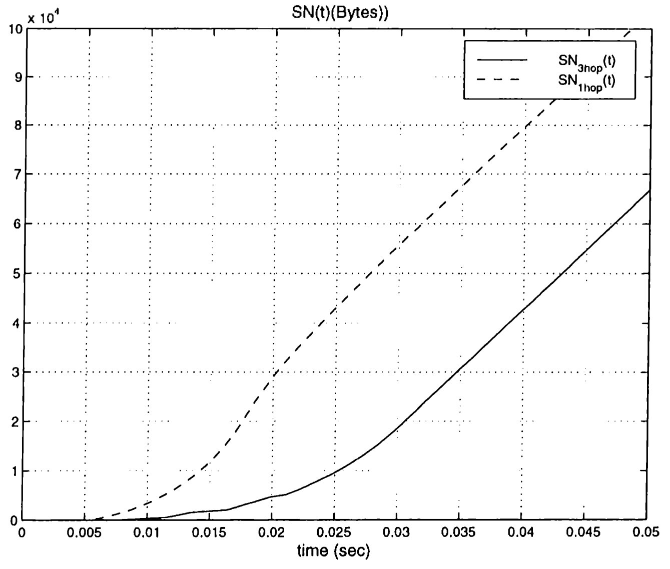


Figure 5.2: Connections with lower RTT increase their rates more aggressively till the rate controlled period is reached

In the following section, the performance perceived at the application and TCP layers are simulated for large file transfer.

5.2 Large File Transfer

The first set of simulations for large file transfer have been run to examine efficiency and fairness observed at the application and TCP layers. To this end, R_1 configuration has been used with inter-switch propagation delays set to $\tau_p = 1\ ms$, and end-system interconnection propagation delays set to $\tau_p = 1\ \mu s$. Such an environment has been chosen to have different propagation delays in the network.

At the application layer of each source end system, a 1048576 byte file is created once throughout the simulation. Sources are staggered by 1 ms. At the TCP layer a maximum segment size of 9180 bytes was chosen. RIF and RDF were both chosen to be 0.015625. The switch parameters are as stated in Chapter 3.2.

For the optimal performance of TCP over ATM, buffer sizes should be adequate to prevent cell losses hence we assume the existence of appropriate buffering at both network interfaces and switches to guarantee lossless delivery. For the interested reader, there exists a wide literature on the effect of buffer sizes on performance [11, 10, 8, 41]. Buffer usage is taken as a parameter for the evaluation of the control schemes.

Fig. 5.3, 5.4, 5.5 illustrate the sequence numbers and allowed cell rates of the connections for relative marking, EPRCA and ERAA respectively. The first observation is that the beat-down problem is still observed at the TCP layer, when the ABR algorithm used was relative marking (Fig. 5.3). The one hop connections grab more bandwidth than their max-min fair shares, hence allowing VC_{2h1} to also increase its usage. The change in the slopes of $SN(t)$ for 2 and 3 hop connections after the termination of the first two transfers indicate that the idle bandwidth is grabbed by the remaining VCs. That is also seen in the $ACR(t)$ plot. Despite the large mss, oscillations in $ACR(t)$ are also observable at the TCP layer for VC_{2h1}, as the oscillations are very wild. We should expect to observe even wilder oscillations with smaller mss, as the averaging effect of the large mss would be lost.

Even though EPRCA seems to provide fairness in the mean sense at the ATM layer, the slope of $SN(t)$ for VC_{2h1} is lower than as required for max-min fairness. That is attributable to large oscillations (Fig. 5.4). The rest of the connections get their fair shares.

	relative marking	EPRCA	ERAA ($\lambda = 1$)	ERAA ($\lambda = 0.97$)
VC_{1h1}	0.3196	0.3995	0.3903	0.4009
VC_{1h2}	0.3302	0.3997	0.3908	0.4018
VC_{1h3}	0.3330	0.3997	0.3908	0.4021
VC_{2h1}	0.1308	0.2008	0.1478	0.1520
VC_{2h2}	0.3981	0.3934	0.3950	0.4061
VC_{2h3}	0.3900	0.3934	0.3952	0.4064
VC_{3h1}	0.4015	0.3945	0.3987	0.4100

Table 5.1: Delays observed at the application layer in seconds for large file transfer with relative marking, EPRCA and ERAA

ERAA provides very smooth and fair transfer at the TCP layer (Fig. 5.5). The link utilizations for the connections seen in Fig. 5.6 indicate that EPRCA and relative marking do not use the links as efficiently as ERAA.

The performance observed at the application layer has been measured by the response times. The results are provided in Table 5.1. It is seen that relative marking provides the fastest transfer to most of the connections. As 1 hop connections and VC_{2h1} grab larger bandwidth than their fair shares, the transmission delays for these connections are lower. Hence, this fast transfer is a consequence of unfairness. A slight unfairness is also observable between connections with the same number of hops. For EPRCA, the transmission time for VC_{2h1} is larger than those for the other schemes due to the oscillations in $ACR(t)$.

Finally, the results for ERAA have been taken for $\lambda = 1$ and $\lambda = 0.97$. ERAA with $\lambda = 1$ has a better delay performance compared to EPRCA, however when $\lambda = 0.97$ is used, buffer requirements are decreased drastically, a trade off that is desirable to decrease costs at the switches. With $\lambda = 0.97$, the performance of ERAA is still comparable to that of EPRCA, only with much lower buffer requirements. The buffer sizes required to provide the above stated performance are given in Table 5.2. If we fixed the buffer sizes, we would observe much larger delay values for the former schemes due to time-outs and retransmissions, hence ERAA might be preferable in a real network environment, when everything is taken into account up to now.

The ability to use any left over capacity is an important feature for the rate allocation algorithms. In order to observe the abilities of the algorithms in this

	relative marking	EPRCA	ERAA
sw0	91	2	2
sw1	658	410	5
sw2	839	495	21

Table 5.2: Buffer requirements at the switches for 0 cell loss operation with (a) relative marking, (b) EPRCA and (c) ERAA

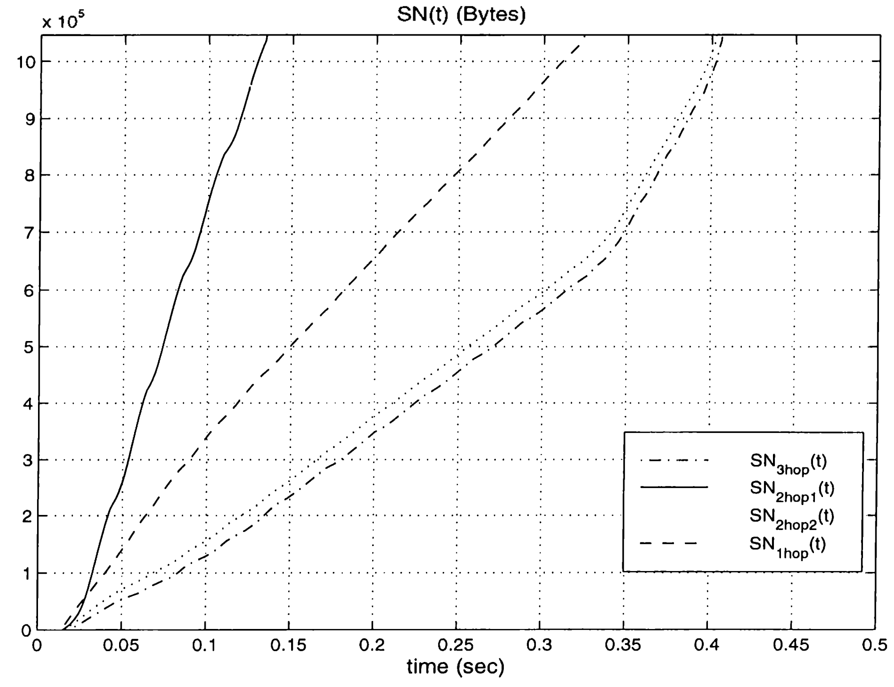
	relative marking	EPRCA	ERAA
VC_1	0.1092	0.1511	0.1596
VC_2	0.1452	0.1650	0.1847
VC_3	0.1575	0.1575	0.1912

Table 5.3: Delays observed at the application layer in seconds, with applications generating large bursts, staggered by 15 ms

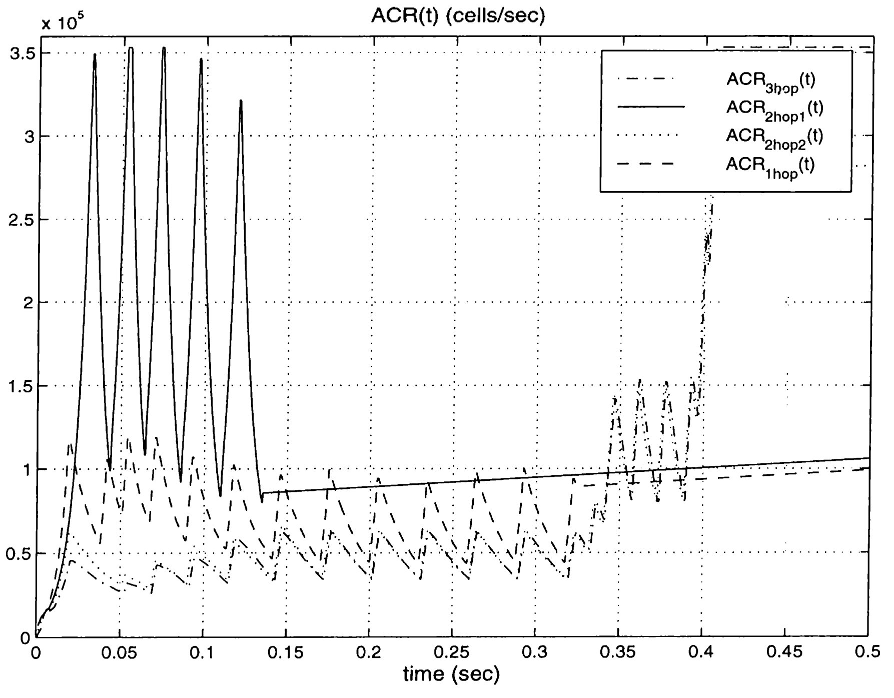
respect, a simulation has been performed on the R_2 configuration with three applications starting to transmit large bursts at different times. τ_p for inter-switch links has been chosen as 10 μs. RIF and RDF were chosen to be 0.0625. We assume that VCs are opened, but are not closed after the packet transfer, waiting for more packets to arrive.

Table 5.3 illustrates the resulting transfer delay values and $SN(t)$ and $ACR(t)$ for the connections are seen in Fig. 5.7 5.8, 5.9. The results are interesting : the slowest converging algorithm ironically performs the fastest packet delivery at the application layer. ERAA converges to the max-min fair rates very fast, when a new VC opens. However, as it has no capability to grab the idle bandwidth unused by ABR connections, the transfers take a long time. EPRCA also converges to the new max-min fair allocations fast. As EPRCA uses queue thresholds for congestion detection, it observes the decrease in link utilization and grabs a larger share when more bandwidth is available, thereby performing better than ERAA. Finally, relative marking converges very slowly as seen in Fig. 5.7. Hence, the VC_1 has the chance to use the high bandwidth allocation for a longer time. VC_2 grabs the bandwidth when VC_1 is finished, and the same reasoning applies for VC_3 .

In this section, we have observed application and TCP layer performances provided by several ABR layer rate control algorithms. The performance received depends on the algorithm used. As observed, ERAA is superior to the other algorithms in many respects, except that it can not grab idle bandwidth, unless the VCs are released by the end-stations.

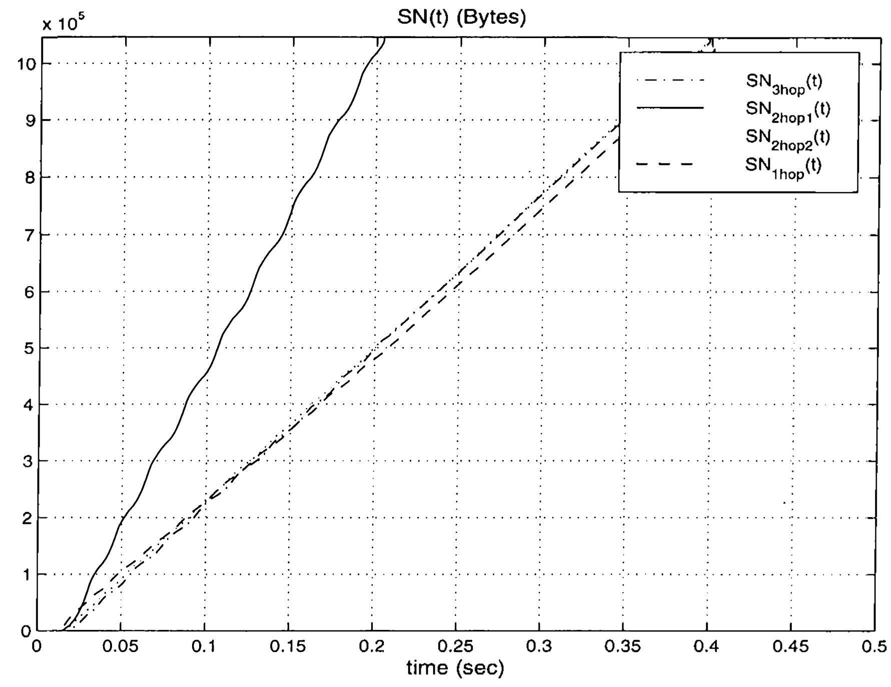


(a)

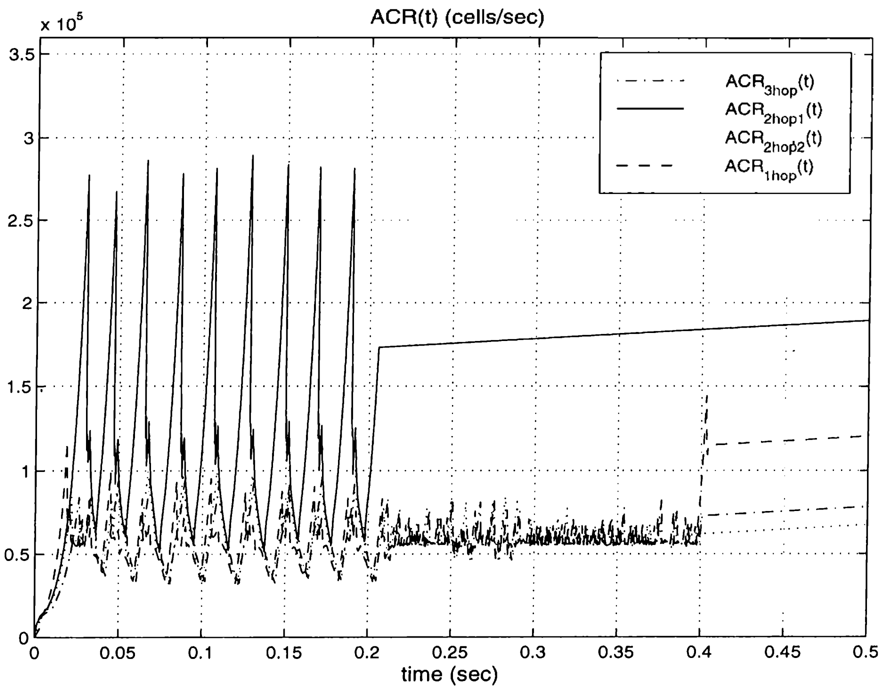


(b)

Figure 5.3: $SN(t)$ and $ACR(t)$ for TCP/ABR with relative marking

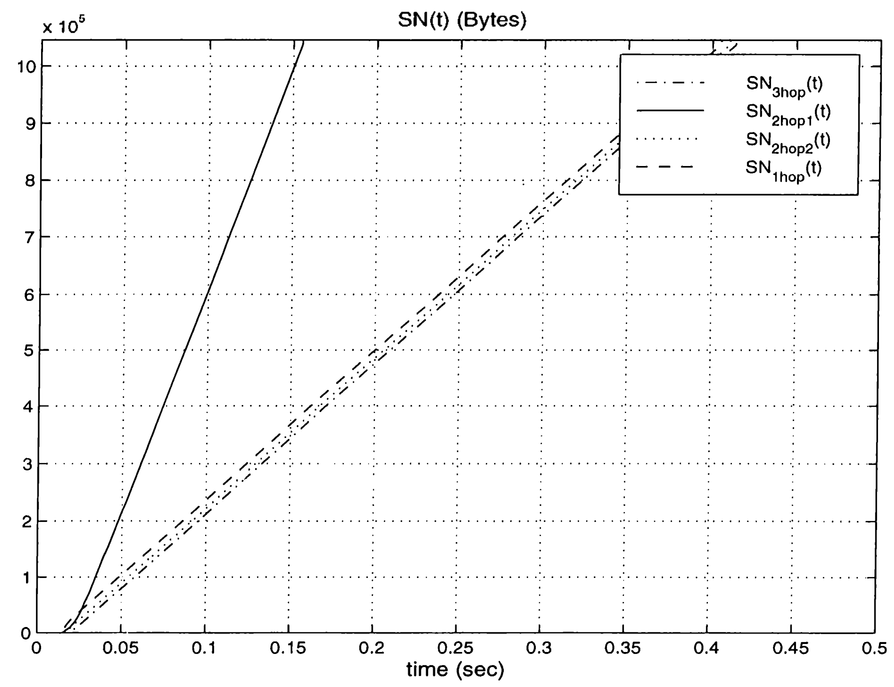


(a)

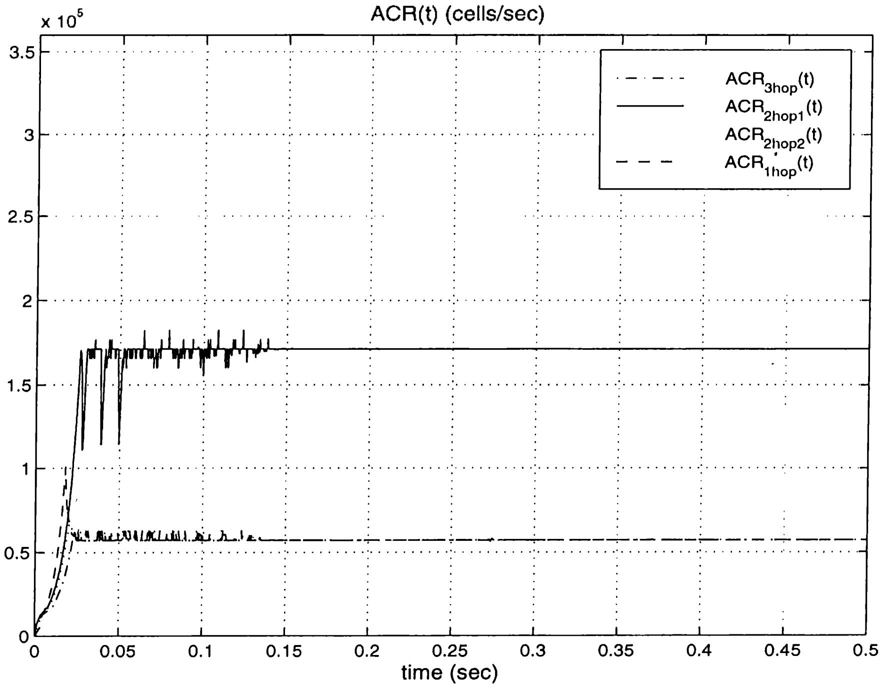


(b)

Figure 5.4: $SN(t)$ and $ACR(t)$ for TCP/ABR with EPRCA

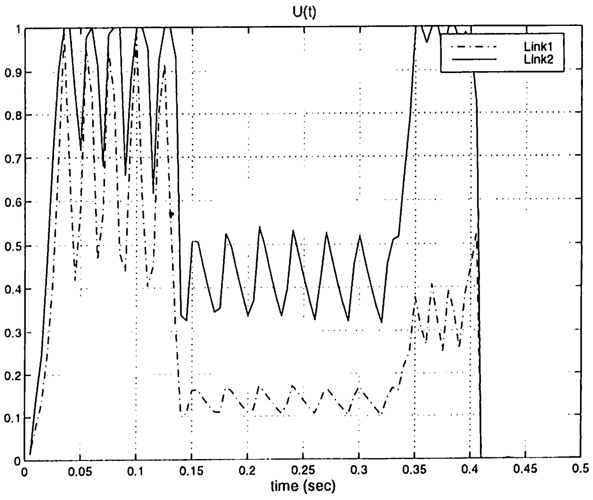


(a)

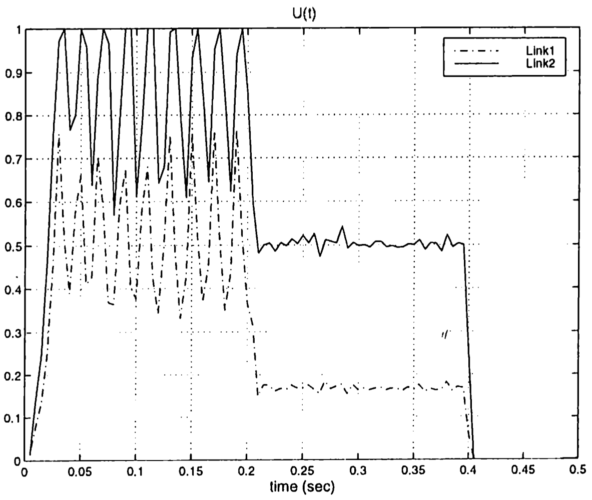


(b)

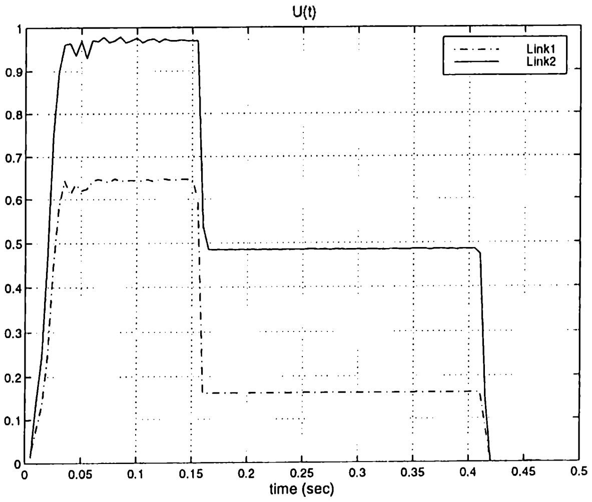
Figure 5.5: $SN(t)$ and $ACR(t)$ for TCP/ABR with ERAA



(a)

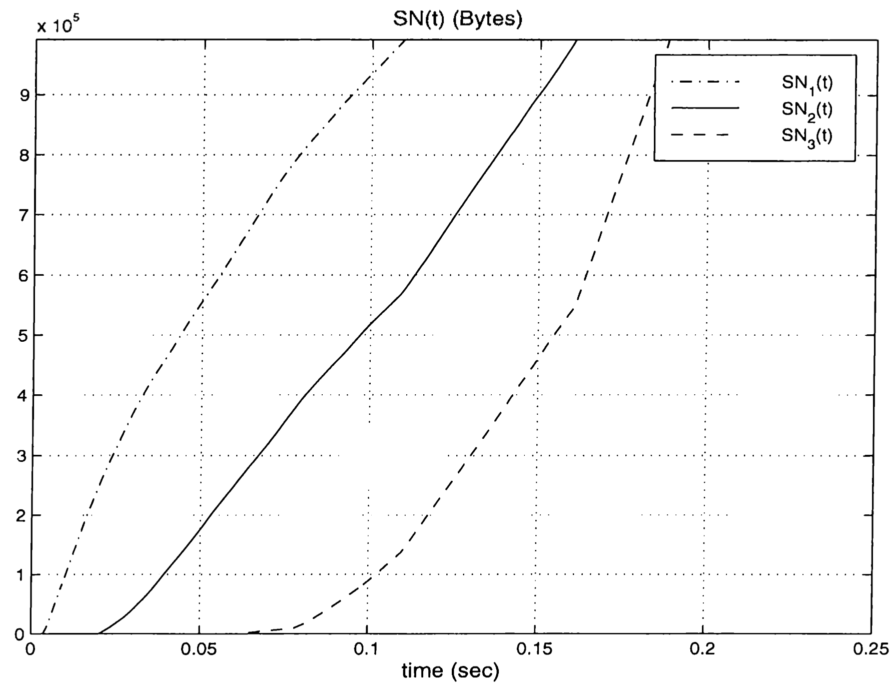


(b)

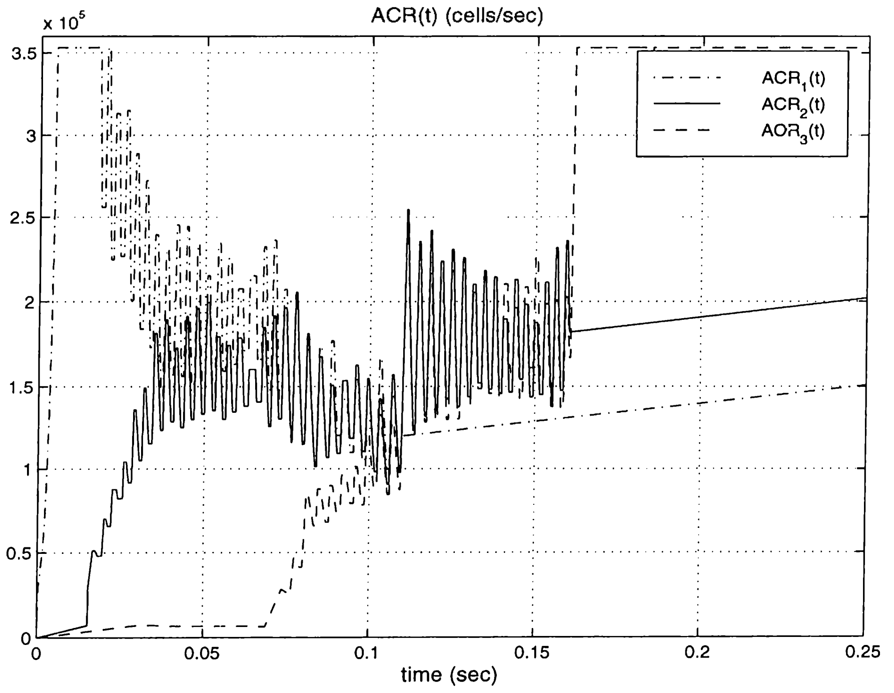


(c)

Figure 5.6: Link utilizations with (a) relative marking, (b) EPRCA and (c) ERAA

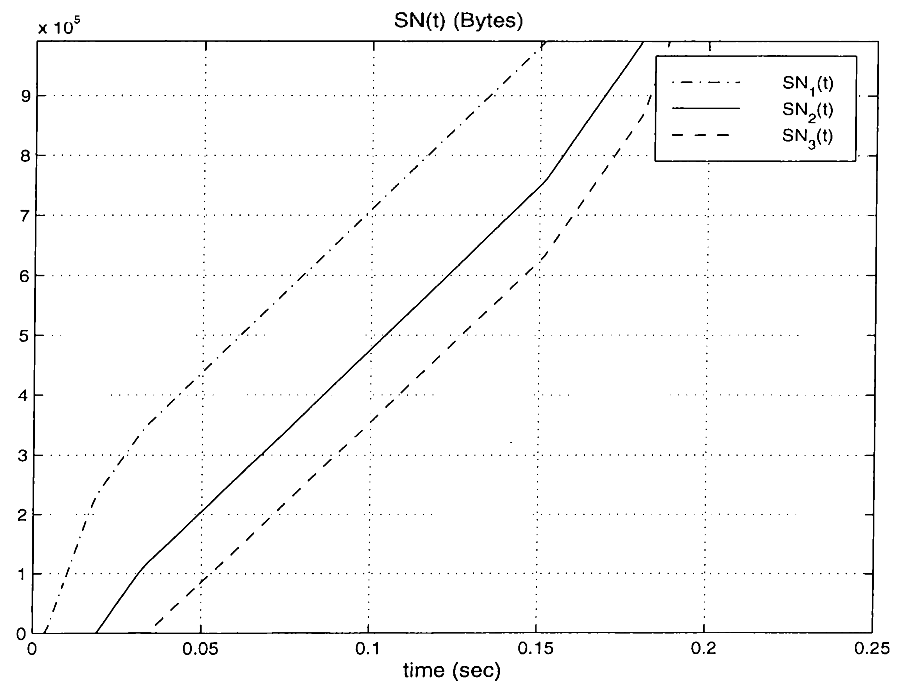


(a)

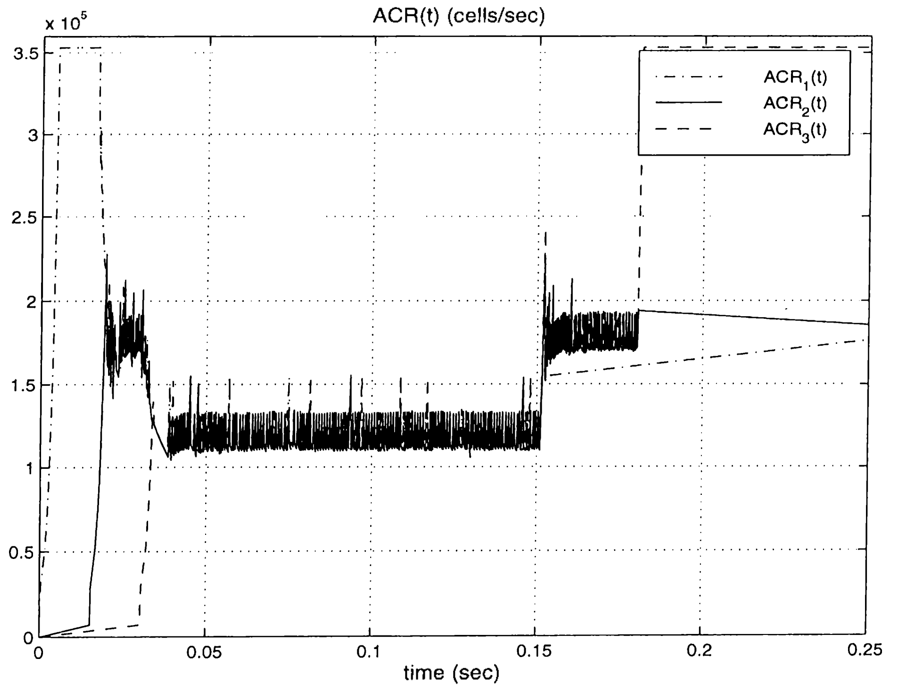


(b)

Figure 5.7: $SN(t)$ and $ACR(t)$ for TCP/ABR with relative marking

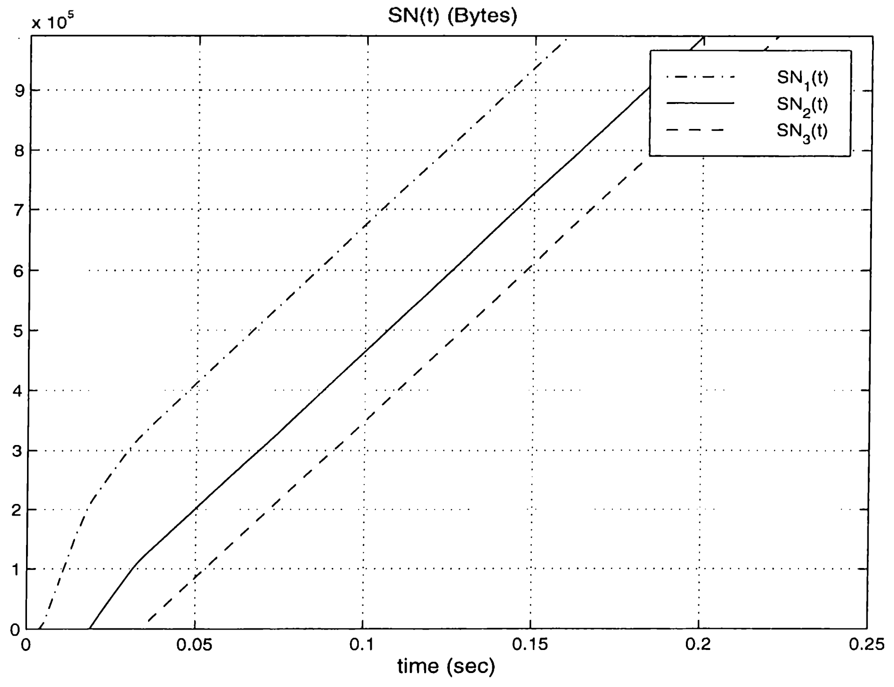


(a)

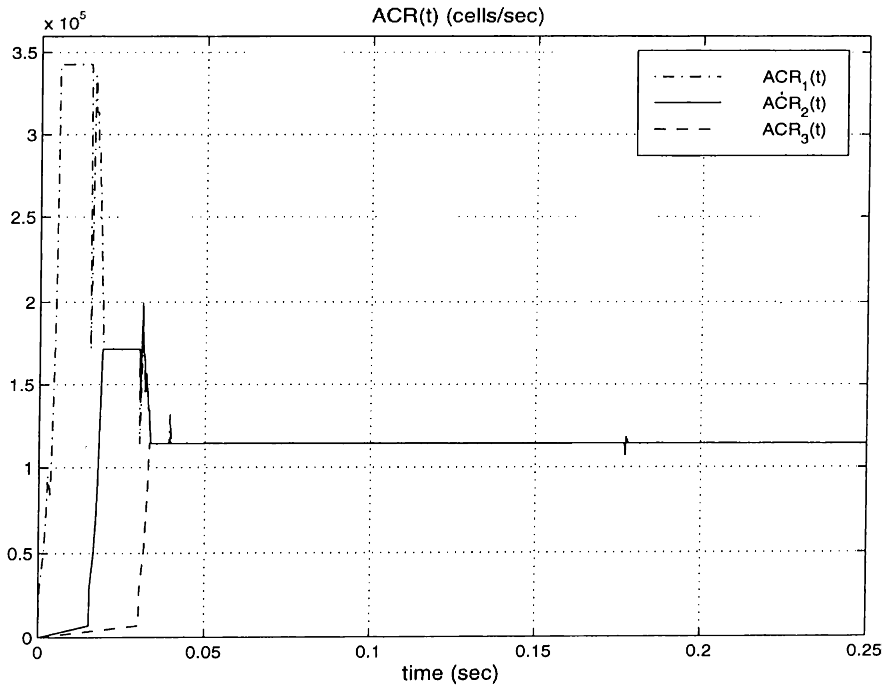


(b)

Figure 5.8: $SN(t)$ and $ACR(t)$ for TCP/ABR with EPRCA



(a)



(b)

Figure 5.9: $SN(t)$ and $ACR(t)$ for TCP/ABR with ERAA

5.3 Coupling TCP and ABR Control Algorithms

Considering the information available at the source and systems, the overall congestion control mechanism for TCP/ABR is not an optimal one, since the available information is not used by the TCP layer. The TCP layer relies on the time-outs and ACKs for the regulation of the sending rate, even though $ACR(t)$ provides information about $C(t)$, the bandwidth that is available for the TCP layer. Assuming that the fair rates $ACR(t)$ calculated by the network are good estimates and are free of oscillations, we could change the TCP window algorithm in a way to incorporate this information.

In this section we propose extensions for incorporating $ACR(t)$ information available at the ATM layer into the TCP window size update mechanism. We assume that the information obtained from the ATM layer is reliable and the real available bandwidth for the connection changes only slowly in the interval $[t_i, t_{i+1})$, where t_i and t_{i+1} are arrival times of consecutive RM cells.

Normally, TCP has a sliding window, the size of which is adjusted with the reception of ACKs, time-outs or multiply repeated ACKs. This dynamic window aims to fill the underlying bit pipe gradually, avoiding packet losses due to congestion. However in satellite links, that may be a waste of resources, as this process would take several round-trips. Furthermore, if the TCP layer forwards packets to the ATM layer without taking the draining rate of the source end ATM buffers, losses are possible outside the network. Thus incorporating the channel rate information might be useful.

For a link used by only one connection, the window size is equal to

$$W(t) = k \cdot C \cdot RTT$$

for full utilization of that link, where k is a scalar that takes the overhead incurred in lower layers into account. In the same line of thought, when we have VC_i with channel rate equal to $ACR_i(t)$, a window size smaller than

$$W_i(t) = k \cdot ACR_i(t) \cdot RTT_i$$

would lead to inefficient operation and a larger one would cause accumulation of cells at source buffers. RTT_i is already estimated by the TCP layer for setting time-outs. Hence, given the $ACR_i(t)$ information from the ATM layer, we can set $W_i(t)$

as stated above.

Another approach might be the use of an average for estimating the window size. The window size might be set to

$$W_i(t+) = (1-g)\ W_i(t) + g\ (k \cdot ACR_i(t) \cdot RTT_i)$$

upon the reception of an RM cell from the network. Finally, a thresholding operator can be used to sense significant changes in $ACR(t)$ so that window size is updated only in such cases. All the mechanisms above can be realized by making changes only at the source ends, thus TCP versions supporting such a change can inter-operate with ones that do not.

There might be two concerns about the above proposals. First of all, the layered architecture is broken, as information from the ATM layer is carried to the TCP layer. Vendors might not like to bear the additional complexity.

The second concern is the information used to estimate the window size. Reliable ACR and RTT values are assumed, however that is not for sure. The ACR values might be rather old due to long delays, or there may be large oscillations in the ACR values as in the case of EFCI marking. Simulations have to be performed to see the effectiveness of the above proposals. The additional information that is used should lead to equal or better performance at the TCP layer.

Chapter 6

Concluding Remarks and Future Directions

In this study, we performed extensive simulations on the rate-based congestion control framework specified by the ATM Forum. We have observed the performance at the ATM, TCP and application layers with three different schemes for fair rate calculation. To this end, an ATM simulation platform was constructed on OPNET. We have seen that the performance depends to a great extent on the congestion control scheme used.

The first and the simplest scheme was the relative marking scheme which provided binary feedback. This scheme suffers from slow convergence to max-min fair rates, sensitivity to parameters, oscillations at the steady state and large buffer requirements. We have also shown the existence of fairness problems for VCs traversing different number of hops, namely the beat-down problem. However, to our surprise, this algorithm was able to perform the fastest transfer observed at the application layer.

The second scheme implemented was the EPRCA algorithm. The beat-down problem and is eliminated by this algorithm, and smaller buffers are required. However, this algorithm is not robust either and oscillations were observed in the allowed cell rate. For the first two schemes, parameter tuning has been a major problem.

The final scheme was the ERAA scheme, which allocated the exact max-min fair rates to connections. This algorithm pays back for the additional complexity incurred at the switches by providing very fast convergence, robustness, max-min fair rates at all times, very low buffer requirements and oscillation free transmission at steady state. One deficiency of this algorithm is its incapability to grab bandwidth left idle by other ABR connections, as the switches compute fair rates without monitoring the link utilizations.

There remain a wide range of issues to be addressed. First of all, the performance with various traffic types should be examined. Short file transfers are particularly interesting since the response times might be longer than the life-time of these applications. In addition, the performance in the existence of higher priority traffic should be studied.

Interoperability is another matter of concern, since it is expected that there will exist a wide variety of switches, which will have to be part of a seamless network. Hence, it is essential that different algorithms coexist and inter-operate efficiently.

A third thread that can be followed is the QoS received by real-time services over ABR. As ABR allows an MCR to be specified, interactive communication is possible. As a matter of fact, some recent computer telephony systems run over the current Internet, i.e, the plain best effort service. ABR should be able to provide a better QoS compared to plain best effort as some guarantees are provided.

Finally, mechanisms for coupling TCP congestion control with the ATM layer control can be worked over, since the performance of TCP which currently works without any information from the network can be increased significantly by incorporating network information received from the ATM layer.

References

[1] P. Reilly. "PDH, Broadband ISDN, ATM, and All That : A guide to Modern WAN Networking, and How it Evolved,". Technical report, Silicon Graphics, Inc, April 1994.

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

[3] G. L. Choudry, D. M. Lucantoni, and W. Whitt "Squeezing the Most Out of ATM," *IEEE Transactions on Commnications*, vol. 44, Feb. 1996.

[4] T. Chahed. "On the Effective Bandwith for Resource Management in ATM Networks,". Master's thesis, Bilkent University, June 1997.

[5] M. W. Garret "A Service Architecture for ATM: From Applications to Scheduling," *IEEE Network*, vol. 10, pp. 6–14, May/June 1996.

[6] H. Kim. *A Non-feedback Based Congestion Control Framework for High-Speed Data Networks.* PhD thesis, University of Pennsylvania, 1995.

[7] D. Bertsekas and Gallager R. *Data Networks.* Prentice Hall, 1992.

[8] A. Romanov and S. Floyd "Dynamics of TCP Traffic over ATM Networks," *IEEE Journal on Selected Areas in Communications*, vol. 13, pp. 663–641, May 1995.

[9] S. Kalyanaraman, R. Jain, S. Fahmy, R. Goyal, and S. C. Kim. "Performance and Buffering Requirements of Internet Protocols over ATM ABR and UBR Services,". to be published, Sept. 1996.

[10] H. Li, K. S. Siu, H. Y. Tzeng, C. Ikeda, and H. Suzuki "A Simulation Study of TCP Performance over ABR and UBR Services in ATM LANs," *IEICE Transactions on Communications*, vol. E70-B, pp. 658–667, May 1996.

[11] G. Hasegawa, H. Ohsaki, M. Murata, and H. Miyahara "Performance Evaluation and Parameter Tuning of TCP over ABR Service in ATM Networks," *IEICE Transactions on Communications*, vol. E70-B, pp. 668–683, May 1996.

[12] S. Floyd "TCP and Explicit Congestion Notification," *Computer Communication Review*, vol. 24, pp. 8–23, Oct. 1994.

[13] F. Bonomi and K.W. Fendick "The Rate Based Flow Control Framework for the Available Bit Rate ATM Service," *IEEE Network*, vol. 9, pp. 25–39, April 1995.

[14] F. Bonomi "Flow Control for the ATM Available Bit Rate Service," in *Proceedings of the 34th Conference on Decision and Control*, Dec. 1995.

[15] H. T. Kung and R. Morris "Credit Based Flow Control for ATM Networks," *IEEE Network*, vol. 9, pp. 40–48, April 1995.

[16] H. T. Kung and A. Chapman "The FCVC (Flow Controlled Virtual Channels) Proposal for ATM NETWORKS," in *Proceedings of the 1993 Internation Conference on Network Protocols*, pp. 116–127, Oct. 1993.

[17] L. Kalampoukas. "An Efficient Rate Alocation Algorithm for ATM Networks,". Master's thesis, University of California Santa Cruz, June 1995.

[18] H. T. Kung, T. Blackwell, and A. Chapman "Credit Based Flow Control for ATM Networks: Credit Update Protocol, Adaptive Credit Allocation, and Statistical Multiplexing," in *Proc. ACM SIGCOMM '94 Symposium on Communications Architectures, Protocols and Applications*, pp. 101–114, Aug. 1994.

[19] G. Varghese, C. Ozveren, and R. Simcoe "Reliable and Efficient Hop-by-Hop Flow Control," in *Proc. of ACR SIGCOMM '94*, pp. 89–100, Dec. 1994.

[20] G. Kesidis. *ATM Network Performance.* Kluwer, 1996.

[21] P. Newman "Backward Explicit Congestion Notification for ATM Local Area Networks," in *Proc. GLOBECOM '93*, Dec. 1993.

[22] M. Hluchyj and N. Yin "On Closed-Loop Rate Control for ATM Cell Relay Networks," in *Proc. INFOCOM '94*, pp. 99–108, June 1994.

[23] L. Roberts, B. Makrucki, T. Tofigh, A. W. Barnhart, B. Holden, G. Fedorkow, J. Daigle, M. Hluchyj, H. Suzuki, G. Ramamurthy, P. Newman, N. Giroux, R. Kositpaiboon, S. Sathe, G. Garg, and N. Yin. "Closed Loop Rate-Based Traffic Management,". ATM Forum Contribution 94-0438R1, July 1994.

[24] L. Roberts. "Enhanced PRCA (Proportional Rate Control Algorithm),". AF-TM 94-0735R1, Aug. 1994.

[25] S. Muddu and et al "Max-Min Rate Control Algorithm for Available Rate Service in ATM Networks," in *ICC '96*, June 1996.

[26] F. Chiussi, Y. Xia, and V. P. Kumar "Dynamic Max Rate Control Algorithm for Available Rate Service in ATM Networks," in *GLOBECOM '96*, 1996.

[27] A. W. Barnhart. "Explicit Rate Performance Evaluation,". AF-TM 94-0983R1, Oct. 1994.

[28] A. Pitsillides, Y. A. Şekercioğlu, and G. Ramamurthy "Effective Control of Traffic Flow in ATM Networks Using Fuzzy Explicit Rate Marking (FERM)," *IEEE Journal on Selected Areas in Communications, special issue on Computational and Artificial Intelligence in High Speed Networks*, vol. 15, pp. 209–225, Feb. 1997.

[29] A. Charny, D. Clark, and R. Jain "Congestion Control with Explicit Rate Indication," in *Proc. ICC '96*, June 1995.

[30] R. Jain, S. Kalyanaraman, and R. Viswanatan. "The OSU Scheme for Congestion Avoidance Using Explicit Rate Indication,". AF-TM 94-0883, Sept. 1994.

[31] R. Jain, S. Kalyanaraman, R. Goyal, S. Fahmy, and R. Viswanathan "The ERICA Switch Algorithm for ABR Traffic Management in ATM Networks, Part I: Description," *IEEE/ACM Transactions on Networking*, Jan. 1997.

[32] R. Jain, S. Fahmy, S. Kalyanaraman, and R. Goyal "The ERICA Switch Algorithm for ABR Traffic Management in ATM Networks, Part II: Requirements and Performance Evaluation," *IEEE/ACM Transactions on Networking*, Jan. 1997.

[33] R. Jain, S. Kalyanaraman, R. Goyal, S. Fahmy, and F. Lu. "ERICA+: Extensions to the ERICA Switch Algorithm,". AF-TM 94-0384, Oct. 1995.

[34] R. Jagielski and Y. A. Şekercioglu "Genetic Programming Techniques for Efficient Estimation of ABR Queue Behavior in ATM Switches," in *Proc. ATNAC '96*, 1996.

[35] A. A. Taraafi, I. W. Habib, and T. N. Saadawi "Congestion Control Mechanism for ATM Networks Using Neural Networks," in *Proc. ICC '95*, pp. 206–215, June 1995.

[36] K.K. Ramakrishnan and P. Newman "Integration of Rate and Credit Schemes for ATM Flow Control," *IEEE Network*, vol. 9, pp. 49–56, April 1995.

[37] K. Archie, G. Campbell, G. Cathey, A. Cohen, R. Finn, A. Piironen, and R. Raghavan. *OPNET Example Models Manual/Protocol Models Vol. 0.* MIL3, 8.1.0 edition, 1994.

[38] K. Y. Siu and H. Y. Tzeng "Intelligent Congestion Control for ABR Services in ATM Networks," *Computer Communication Review*, vol. 24, pp. 81–106, Oct. 1994.

[39] A. Arulambalam, X. Chen, and N. Ansari "Allocating Fair Rates for Available Bit Rate Service in ATM Networks," *IEEE Communications Magazine*, vol. 34, pp. 92–100, November 1996.

[40] V. Jacobson "Congestion Avoidance and Control," in *Proc. of Sigcomm'88*, pp. 314–332, Aug. 1988.

[41] L. Kalampoukas and A. Varma. "Performance of TCP over Multi-Hop ATM Networks: A Comparative Study of ATM-Layer Congestion Control Schemes,". Technical Report UCSC-CRL-95-13, University of CA., Feb. 1995.