Traffic Management for the Available Bit Rate (ABR) Service in Asynchronous Transfer Mode (ATM) Networks

DISSERTATION

Presented in Partial Fulfillment of the Requirements for

the Degree Doctor of Philosophy in the

Graduate School of The Ohio State University

By

Shivkumar Kalyanaraman, B.Tech, M.S.

* * * * *

The Ohio State University

1997

Dissertation Committee:

Raj Jain, Adviser

Ten-Hwang Lai

Wu-chi Feng

Approved by

Adviser
Department of Computer
and Information Sciences

ABSTRACT

With the merger of telecommunication, entertainment and computer industries, computer networking is adopting a new paradigm called *Asynchronous Transfer Mode (ATM)* networking. ATM networks have multiple service classes allow audio, video and data to share the same network. Of these, the *Available Bit Rate (ABR)* service class is designed to efficiently support data traffic.

Traffic management involves the design of a set of mechanisms which ensure that the network bandwidth, buffer and computational resources are efficiently utilized while meeting the various Quality of Service (QoS) guarantees given to sources as part of a traffic contract. The general problem of network traffic management involves all the available traffic classes. In this dissertation, we address the problem of designing traffic management mechanisms for one class - the ABR service class in ATM networks.

We consider five aspects of this problem in this dissertation. First, the ABR service requires a mechanism to carry rate feedback from the network switches to the sources. We design three switch algorithms (the OSU scheme, the ERICA and ERICA+ schemes) which calculate the rate allocations to satisfy different sets of goals. Second, we design a set of source end system mechanisms which respond to network feedback, and perform control in the case when feedback is disrupted or is stale. Third, we validate the performance of the service for various ABR and VBR demand

patterns. Specifically, we study the case of Internet traffic over ATM-ABR. Fourth, we consider the switch design issues for a specific ABR framework option called the "Virtual Source/Virtual Destination" option. Finally, we discuss cost/performance issues pertaining to the implementation of the service.

In summary, this dissertation work addresses fundamental issues in ATM ABR traffic management, and the techniques developed are applicable to a wider class of high-speed packet networks.

Traffic Management for the Available Bit Rate (ABR) Service in Asynchronous Transfer Mode (ATM) Networks

By

Shivkumar Kalyanaraman, Ph.D.

The Ohio State University, 1997

Raj Jain, Adviser

With the merger of telecommunication, entertainment and computer industries, computer networking is adopting a new paradigm called *Asynchronous Transfer Mode (ATM)* networking. ATM networks have multiple service classes allow audio, video and data to share the same network. Of these, the *Available Bit Rate (ABR)* service class is designed to efficiently support data traffic.

Traffic management involves the design of a set of mechanisms which ensure that the network bandwidth, buffer and computational resources are efficiently utilized while meeting the various Quality of Service (QoS) guarantees given to sources as part of a traffic contract. The general problem of network traffic management involves all the available traffic classes. In this dissertation, we address the problem of designing traffic management mechanisms for one class - the ABR service class in ATM networks.

We consider five aspects of this problem in this dissertation. First, the ABR service requires a mechanism to carry rate feedback from the network switches to the sources. We design three switch algorithms (the OSU scheme, the ERICA and ERICA+ schemes) which calculate the rate allocations to satisfy different sets of goals. Second, we design a set of source end system mechanisms which respond to network feedback, and perform control in the case when feedback is disrupted or is stale. Third, we validate the performance of the service for various ABR and VBR demand patterns. Specifically, we study the case of Internet traffic over ATM-ABR. Fourth, we consider the switch design issues for a specific ABR framework option called the "Virtual Source/Virtual Destination" option. Finally, we discuss cost/performance issues pertaining to the implementation of the service.

In summary, this dissertation work addresses fundamental issues in ATM ABR traffic management, and the techniques developed are applicable to a wider class of high-speed packet networks.

To my family

ACKNOWLEDGMENTS

I would like to thank my advisor, Prof. Raj Jain for the excellent guidance and inspiration he has provided for me throughout this work.

I could have never achieved this goal without the steadfast support from my parents.

I would like to thank my collegues, Rohit Goyal, Sonia Fahmy, Bobby Vandalore, Sohail Munir, Ram Viswanathan, Xianrong Cai, Fang Lu, Arun Krishnamoorthy, and Manu Vasandani for giving me an enjoyable and memorable time in the networking lab. The times I remember are usually late in the nights, when we were handling deadlines and had to review each others' work while photocopying tons of paper.

Thanks are due to my good friends and former roommates, C.V. Shankar, Shyamala, Satya, Kram, Sri, Ram Kaushik, Rekha, Rajeev, Saddam, Kishore, Balaji, Prakash, Jayanthi, Ajay, Vijay, Ashok, Lalitha, Revathi, Subra, the Sangam music band, and a host of Columbus folks for putting up with my variable moods and for the fun time I had with them.

A big hi to friends, especially Oil and PV whom I know will read this page.

Finally I'd like to thank Prof. Steve Lai, Prof. Wuchi Feng, and Prof. Anish Arora for their interest in my work and for their valuable suggestions.

VITA

April 30, 1971	Born - Madras, India
1993	B.Tech Computer Science and Engineering, Indian Institute of Technology, Madras, India
1994	M.S Computer and Information Sciences, The Ohio State University
1994-present	Graduate Research Associate, The Ohio State University

PUBLICATIONS

Research Publications

R. Jain, S. Kalyanaraman, R. Goyal and S. Fahmy, "Source Behavior for ATM ABR Traffic Management: An Explanation". *IEEE Communications Magazine*, November 1996

R. Jain, S. Kalyanaraman and R. Viswanathan, "Method and Apparatus for Congestion Management in Computer Networks using Explicit Rate Indication". *U.S. Patent Number 5,633,859*, granted May 27th, 1997

R. Jain, S. Kalyanaraman and R. Viswanathan, "The OSU Scheme for Congestion Avoidance in ATM Networks: Lessons Learnt and Extension". *Performance Evaluation Journal*, Vol. 31/1-2, December, 1997

FIELDS OF STUDY

Major Field: Computer and Information Sciences

TABLE OF CONTENTS

Page

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1

INTRODUCTION AND PROBLEM STATEMENT

1.1 Asynchronous Transfer Mode (ATM) Networks

With the convergence of telecommunication, entertainment and computer industries, computer networking is adopting a new paradigm called *Asynchronous Transfer Mode (ATM)* [14, 22]. ATM was selected by the telecommunication (carrier) industry as the technology to deliver the Broadband Integrated Services Digital Network (B-ISDN) carrier service. ATM is designed to handle different kinds of communication traffic (voice, audio, video and data) in an integrated way. It is first technology to promise seamless interworking between the LAN and WAN network environments. The international standards for ATM networks are being formulated by the ATM Forum [31] and ITU-T [78].

ATM uses short, fixed size (53-byte) packets, called "cells" which is an attractive option because: a) the transmission time per cell is fixed (which reduces the variability in queuing delays), and b) the transmission time is small (which allows building pipelined hardware architectures to process cells in switches). The resulting low mean delay, and low delay variance characteristics are the features that facilitate cell-based voice and video transmissions. However, each cell has five bytes (or 9.43%)

header information which limits the maximum possible efficiency of data transmission, especially on LANs. Further, the loss of one cell results in the loss of an entire packet (which may consist of several cells). But the cell switching (as opposed to expensive packet routing) and sophisticated traffic management technology in ATM networks allows the real efficiency to be close to the maximum possible (unlike the Ethernet technology where the efficiency drops off rapidly as load increases). This feature makes ATM attractive for data communications as well.

The development of the ATM technology has also resulted in several elegant total or compromise solutions to facilitate high-speed integrated services networking. These include: the use of shared switches (as opposed to using shared media), connection-oriented technology (to deliver guarantees, and simplified management and control), the use of short switch-assigned labels in cell headers instead of addresses (for scalability), the development of a true QoS-based routing (PNNI) protocol, and introduction of features such as LAN Emulation (LANE) and Multiprotocol over ATM (MPOA) which has triggered off work in the field of internetworking (running technology "X" over technology "Y") [22, 4].

In this dissertation, we focus on the problem of supporting data applications efficiently within the integrated services framework. Note that, in addition to providing a viable solution for any one of voice, video, or data transmission in isolation, ATM allows all these applications to be supported efficiently in a single network. This is a key feature differentiator when compared with current data network technologies like Ethernet. This feature, when complemented with traffic management capabilities allows the integrated network to be fully utilized while delivering the quality of service requested by applications.

1.2 The Available Bit Rate (ABR) Service

ATM networks provide multiple classes of service to support the quality of service (QoS) requirements of diverse applications, [32]. The current set of classes specified are: the constant bit rate (CBR), real-time variable bit rate (rt-VBR), non-real time variable bit rate (nrt-VBR), available bit rate (ABR), and unspecified bit rate (UBR). The CBR service is aimed at supporting voice and other synchronous applications, the VBR (rt- and nrt-) service are designed to support video and audio applications (which do not need isochronous transfer), while the ABR and UBR services are designed to primarily support data applications.

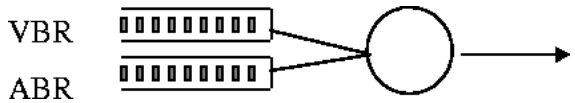


Figure 1.1: ATM ABR and VBR traffic sharing a link

Typically, the CBR and VBR classes are assigned higher "priority" by the network switches and get a share of the link bandwidth first. The "left-over" capacity is used by the ABR and UBR services, with ABR typically having priority over UBR. In figure 1.1, we show a link being shared by a "higher priority" VBR class and a "lower priority" ABR class. Note that VBR and ABR cells are queued separately.

The ABR service class includes an elaborate traffic management framework which allows the efficient handling of data traffic. On the other hand, there exist no standard method of managing traffic on the UBR service. Switches can provide proprietary traffic management mechanisms for UBR, but they cannot coordinate with other

switches since a standard does not exist. In the next section, we define the traffic management problem and discuss its role in the success of ATM as an integrated services networking technology.

1.3 Traffic Management vs Congestion Control

A key issue in ATM and in *any* network architecture design is resource management, i.e., how to make the best use of available resources. Maintaining high utilization of resources while satisfying the users' traffic contracts is the only way the high investment on the networking infrastructure can be recouped. However, striving for high utilization of a resource without proper allocation may lead to long queuing delays, and losses resulting in a low throughput (degradation of user-perceived quality of service).

Traffic management is a resource management problem which deals exclusively with the mechanisms required to control traffic on the network. A related problem is *"congestion"* which occurs when the aggregate demand for a resource (typically link bandwidth) exceeds the available capacity of the resource. In other words, congestion happens whenever the demand is more than the available capacity:

$$\sum_i \text{Demand}_i > \text{Available Capacity}$$

. There are two sets of mechanisms to handle congestion. *"Congestion control"* mechanisms typically come into play *after* the network is overloaded, i.e., congestion is detected. *"Congestion avoidance"* mechanisms come into play *before* the network becomes overloaded, i.e., congestion is predicted. "Congestion management" is a term used to to denote the combination of congestion avoidance and control mechanisms [50].

Congestion management involves the design of mechanisms and schemes to statically limit the demand-capacity mismatch, or dynamically control traffic sources when such a mismatch occurs. Congestion is a problem associated with the dynamics of the network load and capacity, it has been shown that static solutions such as allocating more buffers, or providing faster links, or faster processors does not solve the problem [50, 48]. In fact, the partial deployment of these static alternatives has led to more heterogeneity in the network and increased the possibility of congestion.

Observe that congestion management deals with the problem of matching the demand and capacity for a *single* network traffic class. Traffic management, even for a single traffic class, deals with the problem of ensuring that the network bandwidth, buffer and computational resources are efficiently utilized while meeting the various Quality of Service (QoS) guarantees given to sources as part of a traffic contract. The general problem of network traffic management involves all the available traffic classes. In ATM networks, the general traffic management problem involves the mechanisms needed to control the multiple classes of traffic (like CBR, VBR, ABR and UBR) while ensuring that *all* the traffic contracts are met. The components of traffic management other than congestion management schemes include scheduling mechanisms, traffic contract negotiation, admission control, and traffic policing. In this dissertation, we address the problem of designing traffic management mechanisms for one class - the ABR service class in ATM networks.

Historically, traditional data networks supported only one class of service (data). In such networks, the term "traffic management" was synonymous with "congestion control." In passing, we also note the difference between "flow control" and "congestion control." Flow control deals with the control of a particular flow, whereas

congestion control deals with the control of a group of flows sharing a group of network resource. It is possible to design congestion control schemes which essentially control flows individually at every hop. This makes the problem similar to flow control. An example of such a design is the hop-by-hop flow-controlled virtual circuit [64] or credit-based framework proposal for ATM discussed later in the dissertation.

1.4 Traffic Management for the ABR Service

In this dissertation, we shall address the problem of designing traffic management mechanisms for the ABR service class of ATM networks.

1.4.1 Problem Statement

Traffic management for ABR involves using end-to-end feedback control to match the variable ABR bandwidth at the network ABR queuing points with the variable demand of ABR sources. The statement of the abstract control problem(s) is(are) as follows:

Consider a bottleneck queuing point fed by a set of ABR sources. Define the following per-source variables:

$d_i(t)$ **:** is the desired demand (rate) of the i^{th} source at time t

$r_i(t)$ **:** is the network-assigned rate of the i^{th} source at time t

$T^i{}_d$ **:** is the propagation delay from the i^{th} source to the bottleneck ($T_d^- \leq T_d^i \leq T_d^+$).

Define the following bottleneck variables:

B : The buffer size at the bottleneck (constant)

C : The total capacity of the bottleneck (constant)

N : The total number of ABR virtual circuits through the bottleneck (constant).

$q_a(t)$ **:** the number of ABR cells at the bottleneck buffer at time t ($q_a(t) < B$)

$q_v(t)$ **:** the number of VBR cells at the bottleneck buffer at time t ($q_v(t) = 0$ since VBR is immediately serviced and never queued).

$C_a(t)$ **:** the available capacity for the ABR service ($K \times C < C_a(t) < C$, where $0 \leq K \leq 1$).

$C_v(t)$ **:** the capacity used by the VBR service ($0 \leq C_v(t) \leq (1 - K) \times C$).

$d_v(t)$ **:** the aggregate VBR demand. Note that $C_v(t) = d_v(t)$.

$\rho_a(t)$ **:** the ABR utilization factor at time t ($0 \leq \rho_a(t) \leq 1$, and, $\rho_a(t) = 1$, when $q_a(t) > 0$)

$n_a(t)$ **:** the number of active ABR sources at time t : $(0 \leq n_a(t) \leq N)$

The "open-loop" system is defined as follows. The bottleneck is loaded by both VBR and ABR. VBR is not controllable, whereas the ABR load and capacity display the following relation:

$$\sum_{i=1}^{n_a(t)} min(r_i(t - T^i{}_d), d_i(t - T^i{}_d)) = \frac{dq_a(t)}{dt} + \rho_a(t) \times C_a(t)$$

The equation gives a relation between ABR demand, capacity, queues. and utilization. The *left hand side* of the equation is the *aggregate ABR demand* at time t. It is simply the sum of the demands of the active ABR sources ($\sum_{i=1}^{n_a(t)}$) staggered by their respective time delays ($\tau = t - T^i{}_d$). Each ABR source demand is the minimum of the network-assigned rate ($r_i(\tau)$) and the desired source demand ($d_i(\tau)$). The *right*

hand side of the equation is the sum of the rate of growth of the ABR queue ($\frac{dq_a(t)}{dt}$) and the capacity ($C_a(t)$) scaled by the utilization factor ($\rho_a(t)$). In other words, the ABR demand directly affects the ABR utilization and the rate of growth of the ABR queue.

We desire that the system calculate and feedback rate assignments $r_i(t)$ which satisfy a desired set of goals. Since the goals are many, we elaborate the goals later in chapter 3. More generally, the problem we consider is the design of traffic management mechanisms for the ABR service. We consider five aspects of this problem in this dissertation. Firstly, the service requires a mechanism to carry rate feedback from the switches to the sources. We also design switch algorithms which calculate the rate allocations $r_i(t)$ to satisfy a given set of goals. Secondly, we design a set of source mechanisms which respond to feedback, and perform control when feedback is disrupted or is stale. Thirdly, we validate the performance of the service for various ABR and VBR demand patterns ($d_i(t)$ and $d_v(t)$). Specifically, we study the case of Internet traffic over ABR. Fourthly, we consider the switch design issues for a specific ABR framework option called the "Virtual Source/Virtual Destination" option. The detailed problem specifications and goals are considered in the respective portions of the dissertation.

Our general methodology for tackling this problem is the use of experimentation and simulation techniques, rather than rigorous mathematical analysis. This technique helps us build models which are closer to the real-world systems than mathematical models. However, we rely on simple analytical tools and techniques (such as metric design, and correlation of feedback with control) to ensure stability of the designed system.

1.4.2 Thesis Organization

The ATM Forum has defined a traffic management standard includes a rate-based framework to facilitate end-to-end feedback control. In this framework, ABR sources are allowed to send data at a network-directed rate (r_i, also called the "Allowed Cell Rate"). Periodically, the sources send control cells which are used by the switches to give feedback to the sources. We present the ABR traffic management framework in Chapter 2.

The first part of this dissertation covers the design and performance analysis of distributed algorithms (or "schemes") whose components run independently at different switches in the ATM network, and calculate the feedback for sources. In Chapter 3, we enumerate the goals and limitations of switch schemes.

Typical performance goals are: **a)** "efficiency" - to provide maximum link bandwidth utilization while minimizing queue length and computational overhead; **b)** "fairness" - to divide the available bandwidth fairly among all active sources; **c)** "transient response" - to respond quickly to changes in the load; and **d)** "steady state" - to be stable with minimal load oscillations. Finally, the system should be tuned to work for a wide variety of realistic workloads, and should provide a cost-effective implementation option.

We survey related work in the area of ABR switch scheme design in Chapter 4. We then describe the three switch schemes designed as a part of this dissertation work: the OSU scheme (Chapter 5, the ERICA and the ERICA+ schemes (Chapter 6). The work done as part of the development of these schemes helped design the ATM Forum Traffic Management Specification 4.0 [32], and introduced several concepts which are part of later switch schemes. These chapters also include extensive performance

analyses which are used to validate the schemes, and to illustrate the methodology of switch scheme testing.

The second part of this dissertation deals with design of source end-system control mechanisms (Chapter 7). Such mechanisms are inherently "open-loop" in the sense that sources may unilaterally reduce rates without feedback from switches. Cases where such an approach will be useful includes : a) the case when a network link becomes broken and feedback does not reach the sources, b) the case when a source which is granted a high rate becomes idle temporarily and later uses its retained rate. If the network is heavily loaded, both these cases may result in unpredictably large queuing delays. The mechanisms also determine how RM and data cells are scheduled, especially for low bit-rate sources.

The third part of the dissertation deals with issues in supporting Internet applications like file transfer and world wide web (which run over the TCP/IP protocol) over ATM ABR, with different models of higher priority VBR background traffic (Chapter 8). We study the dynamics and quantify buffer requirements to support zero-loss transmission under such conditions.

The fourth part of this dissertation deals with the switch design issues for a specific ABR framework option called the "Virtual Source/Virtual Destination" option (Chapter 9). In this option, the switch splits the network into two segments and shortens the feedback loop for both segments.

We briefly look at implementation issues in Chapter 10 and proceed to summarize and conclude this dissertation in Chapter 11.

Appendix A quotes the source, destination and switch rules from the ATM Traffic Management 4.0 specification. Appendices B, C and C.3 detail the complete

pseudo-code for the OSU scheme, ERICA schemes and VS/VD alternatives, including the optional features of each. Finally, appendix D provides a glossary of common acronyms used in this dissertation.

CHAPTER 2

THE ABR TRAFFIC MANAGEMENT FRAMEWORK

ABR mechanisms allow the network to divide the available bandwidth fairly and efficiently among the active traffic sources. In the ABR traffic management framework, the *source end systems* limit their data transmission to rates allowed by the network. The network consists of *switches* which use their current load information to calculate the allowable rates for the sources. These rates are sent to the sources as feedback via *resource management (RM)* cells. RM cells are generated by the sources and travel along the data path to the *destination end systems*. The destinations simply return the RM cells to the sources. The components of the ABR traffic management framework are shown in Figure 2.1. In this tutorial, we explain the source and destination end-system behaviors and their implications on ABR traffic management.

The ABR traffic management model is called a "rate-based end-to-end closed-loop" model. The model is called "rate-based" because the sources send data at a specified "rate." This is different from current packet networks (for example, TCP), where the control is "window based" and the sources limit their transmission to a particular number of packets. The ABR model is called "closed-loop" because there is a continuous feedback of control information between the network and the source.

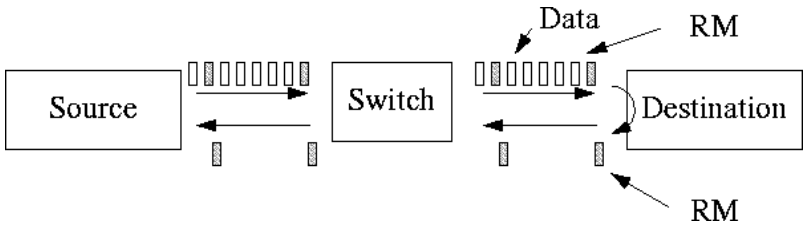


Figure 2.1: ABR Traffic Management Model: Source, Switch, Destination and Resource Management Cells

If more sources become active, the rate allocated to each source is reduced. The model used for CBR and VBR traffic, on the other hand, is "open-loop" in the sense that rates are negotiated at the beginning of the connection and do not change dynamically. Finally, the model is called "end-to-end" because the control cells travel from the source to the destination and back to the source. The alternative of "hop-by-hop" control in which each switch would give feedback to the previous switch [64] was considered and not accepted due to its complexity. However, one can achieve the hop-by-hop control in TM4.0 using the virtual source/virtual destination (VS/VD) feature discussed later in this section.

When there is a steady flow of RM cells in the forward and reverse directions, there is a steady flow of feedback from the network. In this state, the ABR control loop has been established and the source rates are primarily controlled by the network feedback (closed-loop control). However, until the first RM cell returns, the source rate is controlled by the negotiated parameters, which may or may not relate to the current load on the network. The virtual circuit (VC) is said to be following an "open-loop" control during this phase. This phase normally lasts for one round-trip

time (RTT). As we explain later, ABR sources are required to return to the open-loop control after long idle intervals. Traffic sources that have active periods (bursts) when data is transmitted at the allocated rate and idle periods when no data is transmitted are called "bursty sources" Open-loop control has a significant influence on the performance of bursty traffic particularly if it consists of bursts separated by long idle intervals.

There are three ways for switches to give feedback to the sources:

1. First, each cell header contains a bit called Explicit Forward Congestion Indication (EFCI), which can be set by a congested switch. This mechanism is a modification of the DECbit scheme [46]. Such switches are called "binary" or "EFCI" switches. The destination then aggregates these EFCI bit information and returns feedback to the source in an RM cell. An initial version of the binary feedback scheme is illustrated in figure 2.2. In the current specification, the RM cell is sent by the source periodically and is turned around by the destination with the bit-feedback.

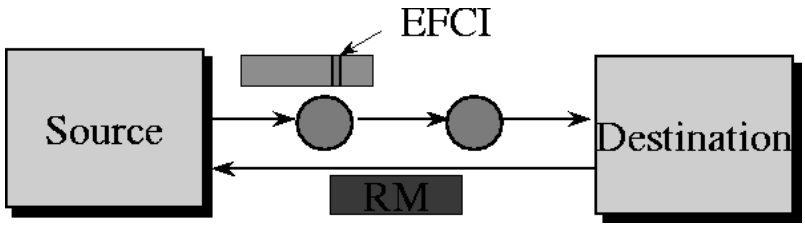


Figure 2.2: Initial Binary Feedback Scheme

2. Second, RM cells have two bits in their payload, called the Congestion Indication (CI) bit and the No Increase (NI) bit, that can be set by congested switches. Switches that use only this mechanism are called relative rate marking switches.

3. Third, the RM cells also have another field in their payload called explicit rate (ER) that can be reduced by congested switches to any desired value. Such switches are called explicit rate switches. The explicit rate mechanism is shown in figure 2.3.

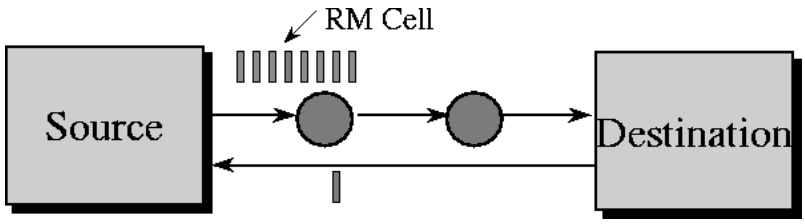


Figure 2.3: Initial Explicit Rate Scheme

Explicit rate switches normally wait for the arrival of an RM cell to give feedback to a source. However, under extreme congestion, they are allowed to generate an RM cell and send it immediately to the source. This optional mechanism is called backward explicit congestion notification (BECN).

Switches can use the VS/VD feature to segment the ABR control loop into smaller loops. In a VS/VD network, the switches additionally behave both as a (virtual) destination end system and as a (virtual) source end system. As a destination end system, it turns around the RM cells to the sources from one segment. As a source end system, it generates RM cells for the next segment. This feature can allow

feedback from nearby switches to reach sources faster, and allow hop-by-hop control as discussed earlier.

2.1 ABR Parameters

At the time of connection setup, ABR sources negotiate several operating parameters with the network. The first among these is the peak cell rate (PCR). This is the maximum rate at which the source will be allowed to transmit on this virtual circuit (VC). The source can also request a minimum cell rate (MCR) which is the guaranteed minimum rate. The network has to reserve this bandwidth for the VC. During the data transmission stage, the rate at which a source is allowed to send at any particular instant is called the allowed cell rate (ACR). The ACR is dynamically changed between MCR and PCR. At the beginning of the connection, and after long idle intervals, ACR is set to initial cell rate (ICR).

During the development of the RM specification, all numerical values in the specification were replaced by mnemonics. For example, instead of saying "every 32nd cell should be an RM cell" the specification states "every Nrmth cell should be an RM cell." Here, Nrm is a parameter whose default value is 32. Some of the parameters are fixed while others are negotiated. A complete list of parameters used in the ABR mechanism is presented in Table 2.1. The parameters are explained as they occur in our discussion.

2.2 In-Rate and Out-of-Rate RM Cells

Most resource management cells generated by the sources are counted as part of their network load in the sense that the total rate of data and RM cells should not

Label	Expansion	Default Value
PCR	Peak Cell Rate	-
MCR	Minimum Cell Rate	0
ACR	Allowed Cell Rate	-
ICR	Initial Cell Rate	PCR
TCR	Tagged Cell Rate	10 cells/s
Nrm	Number of cells between FRM cells	32
Mrm	Controls bandwidth allocation between FRM, BRM and data cells	2
Trm	Upper Bound on Inter-FRM Time	100 ms
RIF	Rate Increase Factor	1/16
RDF	Rate Decrease Factor	1/16
ADTF	ACR Decrease Time Factor	0.5 ms
TBE	Transient Buffer Exposure	16,777,215
CRM	Missing RM-cell Count	$\lceil$ TBE/Nrm $\rceil$
CDF	Cutoff Decrease Factor	1/16
FRTT	Fixed Round-Trip Time	-

Table 2.1: List of ABR Parameters

exceed the ACR of the source. Such RM cells are called "in-rate" RM cells. Under exceptional circumstances, switches, destinations, or even sources can generate extra RM cells. These "out-of-rate" RM cells are not counted in the ACR of the source and are distinguished by having their cell loss priority (CLP) bit set, which means that the network will carry them only if there is plenty of bandwidth and can discard them if congested. The out-of-rate RM cells generated by the source and switch are limited to 10 RM cells per second per VC. One use of out-of-rate RM cells is for BECN from the switches. Another use is for a source, whose ACR has been set to zero by the network, to periodically sense the state of the network. Out-of-rate RM cells are also used by destinations of VCs whose reverse direction ACR is either zero or not sufficient to return all RM cells received in the forward direction.

Note that in-rate and out-of-rate distinction applies only to RM cells. All data cells in ABR should have CLP set to 0 and must always be within the rate allowed by the network.

2.3 Forward and Backward RM cells

Resource Management cells traveling from the source to the destination are called "forward RM" (FRM) cells. The destination turns around these RM cells and sends them back to the source on the same VC. Such RM cells traveling from the destination to the source are called Backward RM (BRM) cells. Forward and backward RM cells are illustrated in Figure 2.4. Note that when there is bi-directional traffic, there are FRMs and BRMs in both directions on the Virtual Channel (VC). A bit in the RM cell payload indicates whether it is an FRM or BRM. This direction bit (DIR) is changed from 0 to 1 by the destination.

2.4 RM Cell Format

The complete format of the RM cells is shown in figure 2.5. Every RM cell has the regular ATM header of five bytes. The payload type indicator (PTI) field is set to 110 (binary) to indicate that the cell is an RM cell. The protocol id field, which is one byte long, is set to one for ABR connections. The direction (DIR) bit distinguishes forward and backward RM cells. The backward notification (BN) bit is set only in switch generated BECN cells. The congestion indication (CI) bit is used by relative rate marking switches. It may also be used by explicit rate switches under extreme congestion as discussed later. The no increase (NI) bit is another bit available to explicit rate switches to indicate moderate congestion. The request/acknowledge,

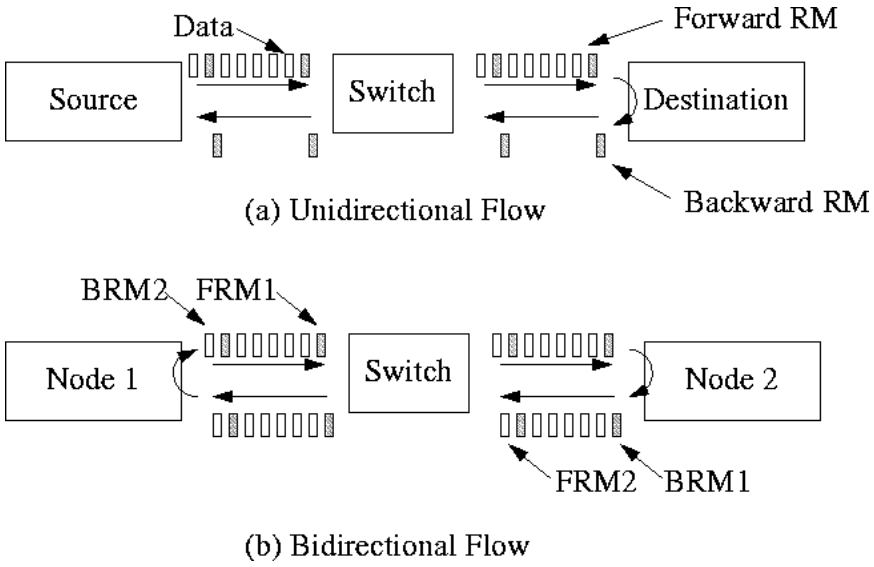


Figure 2.4: Forward and Backward Resource Management Cells (FRMs and BRMs)

queue length, and sequence number fields of the RM cells are for compatibility with the ITU-T recommendation I.371 and are not used by the ATM Forum.

The Current Cell Rate (CCR) field is used by the source to indicate to the network its current rate. Some switches may use the CCR field to determine a VC's next allocation while others may measure the VC's rate and not trust CCR. The minimum cell rate (MCR) field is redundant in the sense that like PCR, ICR, and other parameters it does not change during the life of a connection. However, its presence in the RM cells reduces number of lookups required in the switch.

The ER, CI and NI fields are used by the network to give feedback to the sources. The ER field indicates the maximum rate allowed to the source. When there are multiple switches along the path, the feedback given by the most congested link is the one that reaches the source.

Data cells also have an Explicit Forward Congestion Indication (EFCI) bit in their headers, which may be set by the network when it experiences congestion. The destination saves the EFCI state of every data cell. If the EFCI state is set when it turns around an RM cell, it uses the CI bit to give (a single bit) feedback to the source. When the source receives the RM cell from the network, it adjusts its ACR using the ER, CI, NI values, and source parameters.

ATM Header	5 Bytes	
Protocol ID	1 Byte	1 = ABR
Direction	1 bit	0 = Forward
Backward Notification	1 bit	1 = Switch/dest generated
Congestion Indication	1 bit	1 = High Congestion
No Increase	1 bit	1 = Mild congestion
Request/Acknowledge*	1 bit	
Reserved	3 bits	
Explicit Rate	2 Bytes	
Current Cell Rate	2 Bytes	
Minimum Cell Rate	2 Bytes	
Queue Length*	4 Bytes	
Sequence Number*	4 Bytes	
Reserved	30.75 Bytes	
CRC-10	10 bits	

Figure 2.5: Resource Management (RM) Cell Fields

All rates (e.g., ER, CCR, and MCR) in the RM cell are represented using a special 16-bit floating point format, which allows a maximum value of 4,290,772,992 cells per second (1.8 terabits per second). During connection setup, however, rate parameters are negotiated using an 24-bit integer format, which limits their maximum value to 16,777,215 cells per second or 7.1 Gb/s.

2.5 Source End System Rules

TM4.0 specifies 13 rules that the sources have to follow. This section discusses each rule and traces the development and implications of certain important rules. In some cases the precise statement of the rule is important. Hence, the source and destination rules are quoted from the TM specification [32] in appendix A.

- **Source Rule 1**: Sources should always transmit at a rate equal to or below their computed ACR. The ACR cannot exceed PCR and need not go below MCR. Mathematically,

$$\text{MCR} \leq \text{ACR} \leq \text{PCR}$$

$$\text{Source Rate} \leq \text{ACR}$$

- **Source Rule 2**: At the beginning of a connection, sources start at ICR. The first cell is always an in-rate forward RM cell. This ensures that the network feedback will be received as soon as possible.

- **Source Rule 3**: At any instant, sources have three kinds of cells to send: data cells, forward RM cells, and backward RM cells (corresponding to the reverse flow). The relative priority of these three kinds of cells is different at different transmission opportunities.

  First, the sources are required to send an FRM after every 31 cells. However, if the source rate is low, the time between RM cells will be large and network feedback will be delayed. To overcome this problem, a source is supposed to send an FRM cell if more than 100 ms has elapsed since the last FRM. This introduces another problem for low rate sources. In some cases, at every transmission

opportunity the source may find that it has exceeded 100 ms and needs to send an FRM cell. In this case, no data cells will be transmitted. To overcome this problem, an additional condition was added that there must be at least two other cells between FRMs.

An example of the operation of the above condition is shown in the figure 2.6. The figure assumes a unidirectional VC (i.e., there are no BRMs to be turned around). The figure has three parts. The first part of the figure shows that, when the source rate is 500 cells/s, every 32nd cell is an FRM cell. The time to send 32 cells is always smaller than 100 ms. In the second part of the figure, the source rate is 50 cells/s. Hence 32 cells takes 640 ms to be transmitted. Therefore, after 100 ms, an FRM is scheduled in the next transmission opportunity (or slot). The third part of the figure shows the scenario when the source rate is 5 cells/s. The inter-cell time itself is 200 ms. In this case, an FRM is scheduled every three slots, i.e., the inter-FRM time is 600 ms. Since *Mrm* is 2, two slots between FRMs are used for data or BRM cells.

Second, a waiting BRM has priority over waiting data, given that no BRM has been sent since the last FRM. Of course, if there are no data cells to send, waiting BRMs may be sent.

Third, data cells have priority in the remaining slots.

The second and third part of the this rule ensure that BRMs are not unnecessarily delayed and that all available bandwidth is not used up by the RM cells.

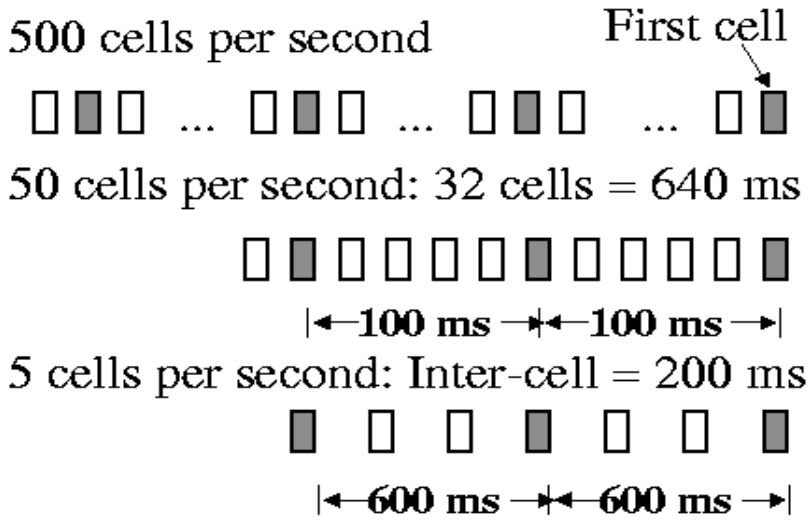


Figure 2.6: Frequency of forward RM cells.

Figure 2.7 illustrates the scheduling of FRMs, BRMs and data cells. In the first slot, an FRM is scheduled. In the next slot, assuming that a turned around BRM is awaiting transmission, a BRM is scheduled. In the remaining slots data is scheduled. If the rate is low, more FRMs and BRMs may be scheduled.

- **Source Rule 4**: All RM cells sent in accordance with rules 1-3 are in-rate RM cells and have their cell loss priority (CLP) bit set to 0. Additional RM cells may be sent out-of-rate and should have their CLP bit set to 1. For example, consider the third unidirectional flow of Figure 2.6. It has an ACR of 5 cells/s.

Figure 2.7: Scheduling of forward RM, backward RM, and data cells.

It is allowed to send only one in-rate RM cell every 400 ms. If necessary, it can send a limited number of out-of-rate RM cells with CLP set to 1.

The frequency of FRM is determined by parameters Nrm, Trm, and Mrm, whose default values are 32, 100 ms, and 2, respectively. During the debate on credit vs rate based alternatives for traffic management [52], the rate based group selected a default value of 32 for Nrm. This ensured that the control overhead was equivalent to that of credit based alternative which claimed an overhead of approximately 6%. During normal operation 1/32th or 3% of all cells are FRM cells. Similarly, another 3% of cells are BRM cells resulting in a total overhead of 6%.

In practice, the choice of Nrm affects the responsiveness of the control and the computational overhead at the end systems and switches. For a connection running at 155 Mb/s, the inter-RM cell time is 86.4 μs while it is 8.60 ms for the same connection running at 1.55 Mb/s. The inter-RM interval determines the responsiveness of the system. While most end-systems and switches will do ABR computations in hardware, it has been shown that it is possible to do them in software on a Pentium^{TM} system provided Nrm is set to 192 or higher on a 155 Mb/s link.

- **Source Rule 5**: The rate allowed to a source is valid only for approximately 500 ms. If a source does not transmit any RM cells for this duration, it cannot use its previously allocated ACR particularly if the ACR is high. The source should re-sense the network state by sending an RM cell and decreasing its rate to the initial cell rate (ICR) negotiated at connection setup. If a source's ACR is already below ICR, it should stay at that lower value (and not increase it to ICR).

  The timeout interval is set by the ACR Decrease Time Factor (ADTF). This parameter can be negotiated with the network at connection setup. Its default value is 500 ms.

  This simple rule was the cause of a big debate at the Forum. It is intended to solve the problem of *ACR retention*. If a source sends an RM cell when the network is not heavily loaded, the source may be granted a very high rate. The source can then retain that rate and use it when the network is highly loaded. In fact, a source may set up several VCs and use them to get an unfair advantage. To solve this problem, several so called *use it or lose it* (UILI) solutions were proposed. Some of them relied on actions at the source while others relied on actions at the switch. The source based solutions required sources to monitor their own rates and reduce ACR slowly if was too high compared to the rate used.

  UILI alternatives were analyzed and debated for months because they have a significant impact on the performance of bursty traffic that forms the bulk of data traffic. The ATM Forum chose to standardize a very simple UILI policy at the source. This policy provided a simple timeout method (using ADTF

as the timeout value) which reduces ACR to ICR when the timeout expires. Vendors are free to implement additional proprietary restraints at the source or at the switch. A few examples of such possibilities are listed in the Informative Appendix I.8 of the specification [32]. We survey the proposed UILI alternatives and present our design later in this dissertation.

- **Source Rule 6**: If a network link becomes broken or becomes highly congested, the RM cells may get blocked in a queue and the source may not receive the feedback. To protect the network from continuous in-flow of traffic under such circumstances, the sources are required to reduce their rate if the network feedback is not received in a timely manner.

  Normally under steady state, sources should receive one BRM for every FRM sent. Under congestion, BRM cells may be delayed. If a source has sent CRM FRM cells and has not received any BRM, it should suspect network congestion and reduce its rate by a factor of CDF. Here, CRM (missing RM cell count) and CDF (cutoff decrease factor) are parameters negotiated at the time of connection setup. BECN cells generated by switches (and identified by BN=1) are not counted as BRM.

  When rule 6 triggers once, the condition is satisfied for all successive FRM cells until a BRM is received. Thus, this rule results in a fast exponential decrease of ACR. An important side effect of this rule is that unless CRM is set high, the rule could trigger unnecessarily on a long delay path. CRM is computed from another parameter called transient buffer exposure (TBE) which is negotiated at connection setup. TBE determines the maximum number of cells that may

suddenly appear at the switch during the first round trip before the closed-loop phase of the control takes effect. During this time, the source will have sent TBE/Nrm RM cells. Hence,

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

The fixed part of the round-trip time (FRTT) is computed during connection setup. This is the minimum delay along the path and does not include any queuing delay. During this time, a source may send as many as ICR $\times$ FRTT cells into the network. Since this number is negotiated separately as TBE, the following relationship exists between ICR and TBE:

$$ICR \times FRTT \leq TBE$$

or

$$ICR \leq TBE/FRTT$$

The sources are required to use the ICR value computed above if it is less than the ICR negotiated with the network. In other words:

ICR used by the source =

Min{ICR negotiated with the network,

TBE/FRTT}

In negotiating TBE, the switches have to consider their buffer availability. As the name indicates, the switch may be suddenly exposed to TBE cells during the first round trip (and also after long idle periods). For small buffers, TBE should be small and vice versa. On the other hand, TBE should also be large enough to prevent unnecessary triggering of rule 6 on long delay paths.

It has been incorrectly believed that cell loss could be *avoided* by simply negotiating a TBE value below the number of available buffers in the switches. We have shown [53] that it is possible to construct workloads where queue sizes could be unreasonably high even when TBE is very small. For example, if the FRM input rate is x times the BRM output rate (see Figure 2.8), where x is less than CRM, rule 6 will not trigger but the queues in the network will keep building up at the rate of $(x-1) \times$ ACR leading to large queues. The only reliable way to protect a switch from large queues is to build it in the switch allocation algorithm. The ERICA+ algorithm presented in this dissertation is an example of one such algorithm.

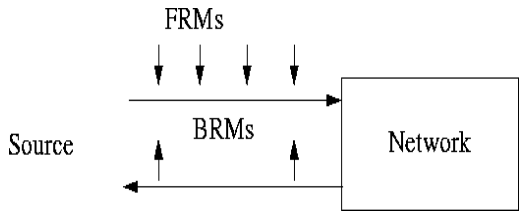


Figure 2.8: Source Rule 6 does not trigger if BRM flow is maintained

Observe that the FRTT parameter which is the sum of fixed delays on the path is used in the formula for ICR. During the development of this rule, an estimate of round trip time (RTT), including the fixed and variable delays was being used instead of FRTT in the ICR calculation. We argued that RTT estimated at connection setup is a random quantity bearing little relation to the round trip delays during actual operation [55]. Such parameter setting could trigger source

Rule 6 unnecessarily and degrade performance. Hence, the Forum decided to use FRTT parameter instead of RTT.

Note that it is possible to disable source Rule 6, by setting CDF to zero.

- **Source Rule 7**: When sending an FRM, the sources should indicate their current ACR in the CCR field of the RM cells.

- **Source Rules 8 and 9**: Source Rule 8 and 9 describe how the source should react to network feedback. The feedback consists of explicit rate (ER), congestion indication bit (CI), and no-increase bit (NI). Normally, a source could simply change its ACR to the new ER value; but this could cause a few problems as discussed next.

  First, if the new ER is very high compared to current ACR, switching to the new ER will cause sudden queues in the network. Therefore, the amount of increase is limited. The rate increase factor (RIF) parameter determines the maximum allowed increase in any one step. The source cannot increase its ACR by more than RIF $\times$ PCR.

  Second, if there are any EFCI switches in the path, they do not change the ER field. Instead, they set EFCI bits in the cell headers. The destination monitors these bits and returns the last seen EFCI bit in the CI field of a BRM. A CI of 1 means that the network is congested and that the source should reduce its rate. The decrease is determined by rate decrease factor (RDF) parameter. Unlike the increase, which is additive, the decrease is multiplicative in the sense that

$$\text{ACR} \leftarrow \text{ACR}(1 - \text{RDF})$$

NI	CI	Action
0	0	ACR $\leftarrow$ Min(ER, ACR + RIF $\times$ PCR, PCR)
0	1	ACR $\leftarrow$ Min(ER, ACR $-$ ACR $\times$ RDF)
1	0	ACR $\leftarrow$ Min(ER, ACR)
1	1	ACR $\leftarrow$ Min(ER, ACR $-$ ACR $\times$ RDF)

Table 2.2: Source End System actions upon CI and NI bits

It has been shown that additive increase and multiplicative decrease is sufficient to achieve fairness [19]. Other combinations such as additive increase with additive decrease, multiplicative increase with multiplicative decrease, and multiplicative increase with additive increase are unfair.

The no-increase (NI) bit was introduced to handle mild congestion cases. In such cases, a switch could specify an ER, but instruct that, if ACR is already below the specified ER, the source should not increase the rate. The actions corresponding to the various values of CI and NI bits are listed in Table 2.2.

$$\text{ACR} \leftarrow \text{Max(ACR,MCR)}$$

If there are no EFCI switches in a network, setting RIF to 1 allows ACRs to increase as fast as the network directs it. This allows the available bandwidth to be used quickly. For EFCI networks, or a combination of ER and EFCI networks, RIF should be set conservatively to avoid unnecessary oscillations.

Once the ACR is updated, the subsequent cells sent from the source conform to the new ACR value. However, if the earlier ACR was very low, it is possible that the very next cell is scheduled a long time in the future. In such a situation,

it is advantageous to "reschedule" the next cell, so that the source can take advantage of the high ACR allocation immediately [54] (also see chapter 7).

- **Source Rule 10**: Sources should initialize various fields of FRM cells as follows. For virtual path connections (VPCs), the virtual circuit id (VCI) is set to 6. For virtual channel connections (VCCs), the VCI of the connection is used. In either case, the protocol type id (PTI) in the ATM cell header is set to 6 (110). The protocol id field in the payload of the RM cell is set to 1. The direction bit should be set to 0 (forward). The backward notification (BN) bits should be set to 0 (source generated). Explicit rate field is initialized to the maximum rate below PCR that the source can support. Current cell rate is set to current ACR. Minimum cell rate is set to the value negotiated at connection setup. Queue length, sequence number, and request/acknowledge fields are set in accordance with ITU-T recommendation I.371 or to zero. All reserved octets are set to 6A (hex) or 01101010 (binary). This value is specified in ITU-T recommendation I.610 (whose number coincidently is also 6-A in hex). Other reserved bits are set to 0. Note that the sources are allowed to set ER and NI fields to indicate their own congestion.

- **Source Rule 11**: The out-of-rate FRM cells generated by sources are limited to to a rate below the "tagged cell rate (TCR)" parameter, which has a default value of 10 cells per second.

- **Source Rule 12**: The EFCI bit must be reset on every data cell sent. The alternative of congested sources being allowed to set EFCI bit was considered but rejected due to insufficient analysis.

- **Source Rule 13**: Sources can optionally implement additional Use-It-or-Lose-It (UILI) policies (see discussion of source Rule 5 and also later in this dissertation).

2.6 Destination End System Rules

- **Destination Rule 1**: Destinations should monitor the EFCI bits on the incoming cells and store the value last seen on a data cell.

- **Destination Rule 2**: Destinations are required to turn around the forward RM cells with minimal modifications as follows: the DIR bit is set to "backward" to indicate that the cell is a backward RM-cell; the BN bit is set to zero to indicate that the cell was not generated by a switch; the CCR and MCR fields should not be changed. If the last cell has EFCI bit set, the CI bit in the next BRM is set and the stored EFCI state is cleared.

  If the destination has internal congestion, it may reduce the ER or set the CI or NI bits just like a switch. Observe that this rule is used in the VS/VD configuration where the virtual destination is bottlenecked by the allowed rate in the next segment. In any case, the ER is never increased.

- **Destination Rules 3-4**: The destination should turn around the RM cells as fast as possible. However, an RM cell may be delayed if the reverse ACR is low. In such cases destination rules 3 and 4 specify that old out-of-date information can be discarded. The destinations are allowed a number of options to do this. The implications of various options of destination Rule 3 are discussed in the

Informative Appendix I.7 of the TM specification [32]. Briefly, the recommendations attempt to ensure the flow of feedback to the sources for a wide range of values of ACR of the reverse direction VC. If the reverse direction ACR is non-zero, then a backward RM cell will be scheduled for in-rate transmission. Transmitting backward RM cells out-of-rate ensures that the feedback is sent regularly even if the reverse ACR is low or zero (for example, in unidirectional VCs).

Note that there is no specified limit on the rate of such "turned around" out-of-rate RM cells. However, the CLP bit is set to 1 in the out-of-rate cells, which allows them to be selectively dropped by the switch if congestion is experienced.

- **Destination Rule 5**: Sometimes a destination may be too congested and may want the source to reduce its rate immediately without having to wait for the next RM cell. Therefore, like a switch, the destinations are allowed to generate BECN RM cells. Also, as in the case of switch generated BECNs, these cells may not ask a source to increase its rate (CI bit is set). These BECN cells are limited to 10 cells/s and their CLP bits are set (i.e., they are sent out-of-rate).

- **Destination Rule 6**: An out-of-rate FRM cell may be turned around either in-rate (with CLP=0) or out-of-rate (with CLP=1).

2.7 Switch Behavior

The switch behavior specifies that the switch must implement some form of congestion control, and rules regarding processing, queuing and generation of RM cells.

- **Switch Rule 1**: This rule specifies that one or more methods of feedback marking methods must be implemented at the switch. The possible methods include :

  EFCI Marking: This defines the binary (bit-based) feedback framework, where switches may set the EFCI bit in data cell headers. We have noted earlier that the destinations maintain an EFCI state per-VC and set the CI bit in backward RM cells if the VC's EFCI state is set. Note that the VC's EFCI state at the destination is set and reset whenever an incoming data cell has its EFCI set or reset respectively.

  Relative Rate Marking: This option allows the switch to set two bits in the RM cell which have a specific meaning to when they reach the source end systems. The CI bit when set asks the source to decrease, while the NI bit tells the source not to increase beyond its current rate, ACR. Observe that the source rate may be further reduced using the explicit rate indication field. These bits allow the switches some more flexibility than the EFCI bit marking. Specifically, the switches can avoid the "beat-down" fairness problem seen in EFCI marking scenarios. The problem occurs because connections going through several switches have a higher probability of their EFCI bits being set, than connections going through a smaller number of switches.

  Explicit Rate Marking: Allows the switch to specify exactly what rate it wants a source to send at. To ensure coordination among multiple switches in a connection's path, the switch may reduce (but not increase) the ER

field in the RM-cells (in the forward and/or backward directions). This dissertation deals mainly with explicit rate feedback from switches.

VS/VD Control: In this mode, the switch may segment the ABR control loop by appearing as a "virtual source" to one side of the loop and as a "virtual destination" to the other side. We study the implications of this mechanism on the ABR service later in this dissertation.

- **Switch Rule 2**: This rule specifies how a switch may generate an RM cell in case it is heavily congested and doesn't see RM cells from the source. Basically, the rule allows such RM cells to only decrease the source rate, and these RM cells are sent out-of-rate. This rule contains aspects of the Backward Explicit Congestion Notification proposal [69] and the OSU scheme proposal described later in this dissertation.

- **Switch Rule 3**: This rule says that the RM cells may be transmitted out-of-sequence, but the sequence integrity must be maintained. This rule allows the switch the flexibility to put the RM cells on a priority queue for faster feedback to sources when congested. However, by queuing RM cells separately from the data stream, the correlation between the quantities declared RM cells and the actual values in the data stream may be lost.

- **Switch Rule 4 and 5**: Rule 4 specifies alignment with ITU-T's I.371 draft, and ensures the integrity of the MCR field in the RM cell. Rule 5 allows the optional implementation of a use-it-or-lose-it policy at the switch. We treat the use-it-or-lose-it issue in greater detail later in this dissertation.

Observe that the ABR traffic management framework only specifies that the switch should implement a feedback marking mechanism and gives flexibility on how to handle RM cells. However, the specific schemes to calculate feedback are not standardized. Several other aspects (such as VS/VD, use-it-or-lose-it implementation, switch queuing and buffering architectures, and parameter selection) are implementation specific, and are an area for vendor differentiation. In this dissertation, we address issues in several of these non-standard areas. Towards this direction, the next chapter describes the design goals of switch algorithms.

2.8 Summary

We have presented the source, destination, switch rules, and parameters of the ABR traffic management model. Like any other standard, these rules reflect a compromise between several differing views. As observed, a key component in the traffic management specification is the switch scheme which calculates the feedback to be given to the sources.

The work presented in this dissertation helped develop the source, switch and destination rules of the ATM Forum Traffic Management standard. Specifically, we study and propose designs for switch rate feedback calculation, source rule design (especially SES Rules 3, 5, 9, 11, and 13), and address application performance and switch scheme implementation tradeoffs.

CHAPTER 3

SWITCH SCHEME DESIGN ISSUES

The most important part in ABR traffic management framework is the switch feedback calculation algorithm. The switch algorithm calculates the feedback to be given to the sources. We use the following switch model for further discussion:

3.1 Switch Model

A switch interconnects multiple links and supports multiple ports, typically an input port or/and an output port per-link. Each port may have some buffers associated with it. It is possible to put the buffers exclusively at the input port (an input-buffered architecture), exclusively at the output port (an output-buffered architecture), or at both the input and output ports. Popular switch architectures tend towards being exclusively output buffered [70] due to its superior performance when compared to input buffered switches. We choose to focus on output buffered switch architectures.

Buffers may be logically partitioned into queues, which are scheduled using a specific discipline. Queuing and scheduling at the buffers may be handled in a First In First Out (FIFO) manner where all the cells coming to the port are put into a common buffer (and later serviced) in the order they arrived at the port. On the other hand, a

complex method like per-VC queuing and scheduling (a separate queue for every VC) may be used. Minimally, a switch will have a separate FIFO queue for every traffic class supported (CBR, VBR, ABR, and UBR classes). The rate-based framework defined in the ATM Traffic Management 4.0 standards allows the switch designers total flexibility in choosing the buffer allocation, queuing, and scheduling policy. This was one of the key features that led to its acceptance compared to the credit-based proposal which required per-VC queuing and scheduling to be implemented at every switch. We assume a model of an output buffered switch implementing per-class queues at every output port. The ABR congestion control algorithm runs at every output port's ABR queue.

The capacity of the output link is assumed to be shared between the "higher priority" classes (constant bit rate (CBR), real-time variable bit rate (rt-VBR), and non-real time variable bit rate (nrt-VBR)) and the available bit rate (ABR) class. We bunch the higher priority classes into one conceptual class called "VBR." Link bandwidth is first allocated to the VBR class and the remaining bandwidth, if any, is given to ABR class traffic. The capacity allocated to ABR is called ABR capacity. We study the problem of controlling the ABR capacity and ABR queues of the output port. Note that, it is possible to have a number of separate subclasses within ABR which are queued and serviced separately. In such a case, the switch algorithm applies to each ABR class queue.

3.2 ABR Switch Scheme Goals

The ABR service was initially designed to achieve high throughput with control over cell loss, since early data users reported heavy loss of cells and throughput.

But the development of feedback mechanisms has led to an expansion of these goals. Specifically, switches can give feedback such that the sources are treated in a "fair" manner. Further, switches can control the queuing delays, provide a combination of quick response time, and a stable steady state. Switches today can also compensate efficiently for errors due to variation in network load and capacity. In this section, we will make these goals more concrete, and use these goals as a reference to evaluate switch schemes.

3.2.1 Congestion Avoidance

The goal of congestion avoidance is to bring the network to an operating point of high throughput and low delay [46]. Typically, there is a tradeoff between the link utilization and the switch queuing delay. For low utilization, the switch queue is small, and the delay is small. Once utilization is very high, the queues grow. Finally, when the queue size exceeds the available buffer size, cells are dropped. In this state, though the link utilization may be high (since the queue length is greater than zero), the effective end-to-end throughput is low (since several packets do not reach the destination). In general, we may replace the terms "utilization" and "switch queuing delay" can be replaced by "throughput" and "end-to-end delay" respectively when we consider entire networks.

Figure 3.1 shows the throughput and delay with varying load in the network. The operating point which has a utilization close to 100% and moderate delays is called the *knee* of the delay-throughput curve. Formally, the knee is the point where the ratio of the bottleneck throughput to bottleneck response time (delay) as a function of input load is maximized. In a network which is in a ideal operating point, typical utilization

graphs have a steady state with controlled oscillations close to 100% utilization, and typical queue length graphs have a steady state with controlled oscillations close to zero queue length. This is also illustrated in figure 3.1.

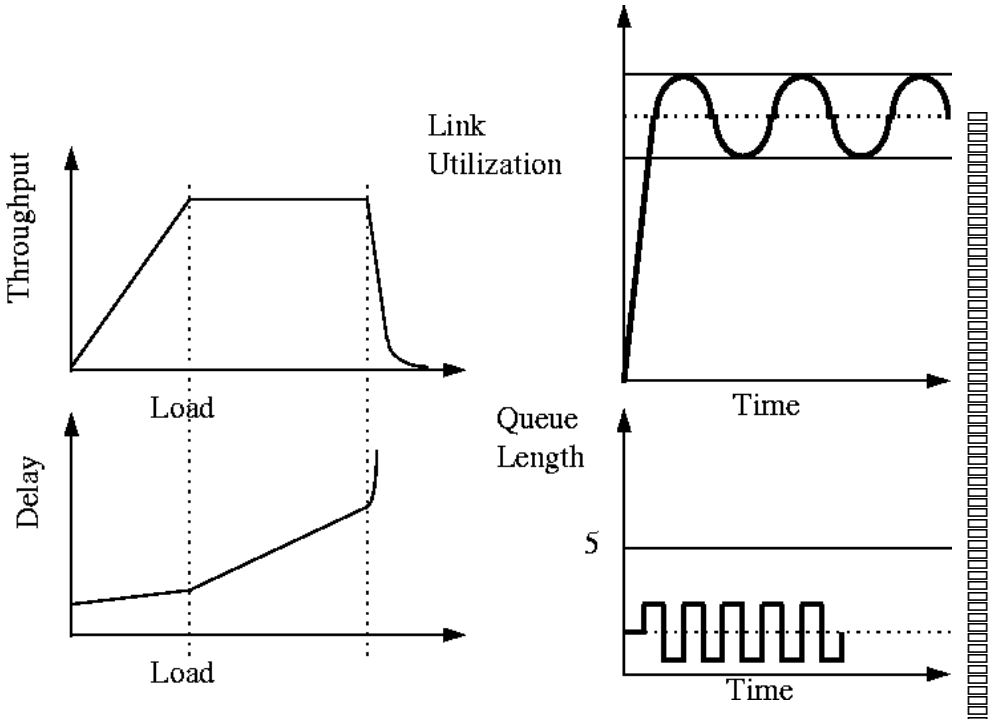


Figure 3.1: Throughput versus delay

The "knee" is a good choice for an operating point for congestion control schemes. Schemes which operate at (or close to) this point are called *congestion avoidance schemes*. Congestion avoidance can be considered as one notion of "efficiency" in the ABR service.

In the terminology of section 1.4.1, the goal could be stated as maximizing the steady state (or average) utilization, $\int \rho_a(t)dt$ while minimizing the average queue length, $\int q_a(t)dt$.

If the load is increased beyond the knee, the delay increases as a function of the load, but there is always a non-zero queuing delay. However, beyond a certain delay, the throughput drops again and the (end-to-end) delays rise sharply (due to higher layer mechanisms like timeout and retransmission). This point is called the "cliff" of the delay-throughput curve. The cliff is a highly unstable operating point, and has the disadvantage of large queuing delays.

Operating points between the knee and the cliff (as shown in figure 3.2) may also be desirable. Such operating points keep the network at 100% utilization in steady state and maintain a "pocket of queues" in the buffer. Further, as the queues grow beyond the desired value, additional capacity is allocated to drain the queues. In other words, the scheme has control over the queuing delay in the steady state, and the queue drain rate under transient conditions. Note that, in general, such an operating point is not very stable for rate-based control, unless the switch uses a function of input load as well in the control. This is because the bottleneck queue length is controlled not by a set of windows (which can at most result in finite queues), but by a set of rates (which can result in infinite queues if not controlled).

Note that this new operating problem poses a new control problem. In the terminology of section 1.4.1, we may state the new goal as maximizing the steady state (average) utilization, $\int \rho_a(t)dt$ while minimizing the difference of the queue length from a desired queue length, $|q_a(t) - q_{desired}|$.

3.2.2 Fairness

In a shared environment, the throughput for a source depends upon the demands by other sources. Ideally, a scheme should equally divide the available bandwidth

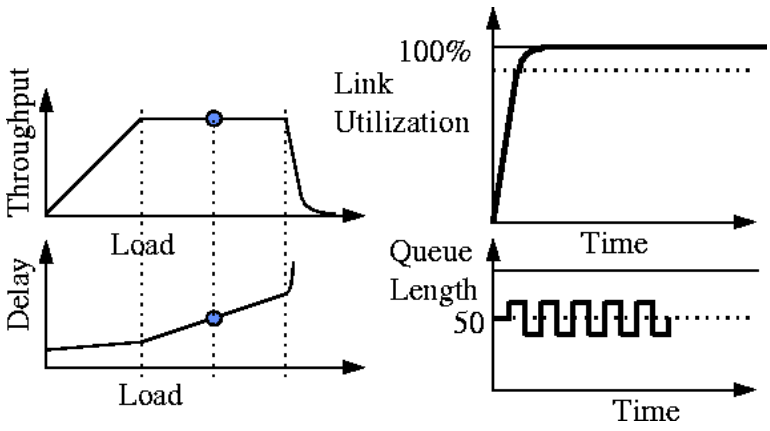


Figure 3.2: Operating point between the "knee" and the "cliff"

among sources which can use the bandwidth (active, unconstrained sources). The most commonly used criterion for what is the correct share of bandwidth for a source in a network environment, is the so called "max-min allocation [41]." It provides the maximum possible bandwidth to the source receiving the least among all contending sources.

Mathematically, the optimality criterion can be written as follows [15]:

Given a configuration with n contending sources, suppose the ith source gets a bandwidth η_i. The allocation vector $\{\eta_1, \ldots, \eta_n\}$ is feasible if all link load levels are less than or equal to 100%.

Consider vector $a = (a_1, \ldots, a_n)$. Let $\hat{a} = (\hat{a_1}, \ldots, \hat{a_n})$ be a permutation of a such that $\hat{a_i} \leq \hat{a_j}$ if $i < j$. Vector b is said to be lexicographically greater than a if either $\hat{a_1} < \hat{b_1}$ or $\exists 1 \leq j \leq n,\ s.t.\ \hat{a_i} = \hat{b_i}\ \forall 1 \leq i < j$ and $\hat{a_j} < \hat{b_j}$.

A vector $\eta = (\eta_1, \ldots, \eta_n)$ is a *max-min* fair vector if it is a feasible vector and it is lexicographically greater than any such feasible vector.

Observe that this definition means that the optimal vector is such that its smallest component is maximized over all feasible vectors, then, given the value of the smallest component, the next smallest component is maximized, etc.

In other words, we know that the total number of feasible vectors is infinite. For each allocation vector, the source that is getting the least allocation is in some sense, the "unhappiest source." Given the set of all feasible vectors, find the vector that gives the maximum allocation to this unhappiest source. Actually, the number of such vectors is also infinite although we have narrowed down the search region considerably. Now we take this "unhappiest source" out and reduce the problem to that of remaining $n-1$ sources operating on a network with reduced link capacities. Again, we find the unhappiest source among these $n-1$ sources, give that source the maximum allocation and reduce the problem by one source. We keep repeating this process until all sources have been given the maximum that they could get. In summary, a network is considered to be in a state of max-min fairness if it is impossible to increase the rate of any session without decreasing the rate of sessions whose rate is equal or smaller.

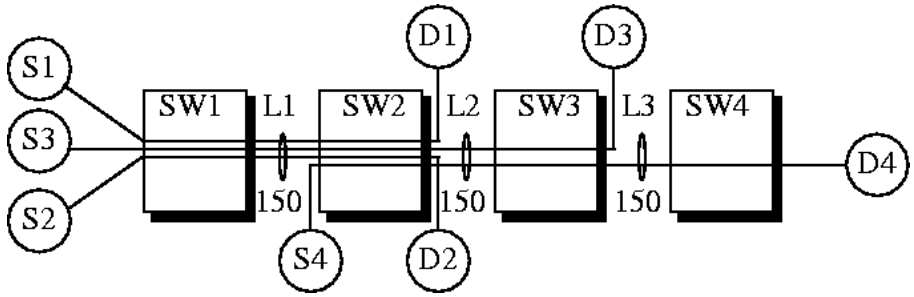


Figure 3.3: Sample configuration for max-min fairness

The following example illustrates the above concept of max-min fairness. Figure 3.3 shows a network with four switches connected via three 150 Mbps links. Four VCs are setup such that the first link L1 is shared by sources S1, S2, and S3. The second link is shared by S3 and S4. The third link is used only by S4. Let us divide the link bandwidths fairly among contending sources. On link L1, we can give 50 Mbps to each of the three contending sources S1, S2, and S3. On link L2, we would give 75 Mbps to each of the sources S3 and S4. On link L3, we would give all 155 Mbps to source S4. However, source S3 cannot use its 75 Mbps share at link L2 since it is allowed to use only 50 Mbps at link L1. Therefore, we give 50 Mbps to source S3 and construct a new configuration shown in Figure 3.4, where Source S3 has been removed and the link capacities have been reduced accordingly. Now we give 1/2 of the link L1's remaining capacity to each of the two contending sources: S1 and S2; each gets 50 Mbps. Source S4 gets the entire remaining bandwidth (100 Mbps) of link L2. Thus, the fair allocation vector for this configuration is (50, 50, 50, 100). This is the max-min allocation.

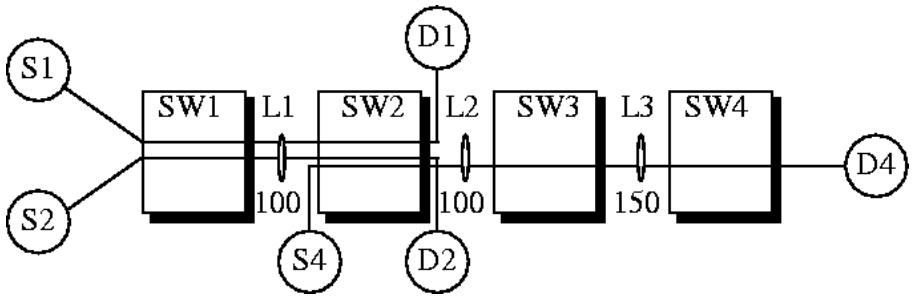


Figure 3.4: Configuration after removing VC 3

Notice that max-min allocation is both fair and efficient. It is fair in the sense that all sources get an equal share on every link provided that they can use it. It is efficient in the sense that each link is utilized to the maximum load possible.

When we take the minimum cell rate (MCR) of sources into account, there are several possible optimality criteria. Other criterion such as weighted fairness have been proposed to determine optimal allocation of resources over and above MCR [32].

Abraham and Kumar [1] develop a natural extension of the concept of max-min fair rate allocation to the case of ABR sessions with non-zero MCRs. Specifically, the feasibility condition includes the fact that *every VC's rate is at least its MCR*; the max-min criteria is the same: the network is considered to be in a state of max-min fairness if it is impossible to increase the rate of any session, while maintaining feasibility, without decreasing the rate of sessions whose rate is equal or smaller. The characterization in terms of rate vectors is also the same, i.e., a rate vector is max-min fair is it is lexicographically the largest among all feasible rate vectors. The authors also develop centralized and distributed algorithms to achieve this max-min allocation.

Finally, it should be pointed out that all definitions of fairness assume that the traffic sources always have data to send (i.e., are infinite sources). For traffic which is "bursty" (i.e., contains active and idle periods), the concept of fairness is ill-defined. As a heuristic, the definitions should be rephrased in terms of the throughputs achieved by sources. Source throughput is measured over a long time interval (covering many idle and active intervals) and not approximated as a series of instantaneous rate allocations. In other words, "fairness" is a long-term goal. While we

mention this concept of fairness for bursty traffic, we do not use the concept further in this dissertation.

3.3 Stable Steady State

The steady state of the system is a state where the goals of efficiency (congestion avoidance) and fairness (max-min) have been achieved. The scheme should first be able to converge to a steady state from any set of initial conditions, provided that the demand and capacity remain constant. Secondly, once the scheme reaches optimal steady state operation, it should stay close to the optimal operation in spite of asynchrony in feedback/response characteristics of the network. In other words, the steady state oscillations between overloaded and underloaded states should be minimal and bounded. An example of a desirable steady state operation is shown (with respect to congestion avoidance, or efficiency alone) in Figure 3.1. Typical steady states have a small amount of residual bandwidth to drain out transient queues and reach the steady state operation of near-zero queuing delay and high throughput.

3.4 Transient Response

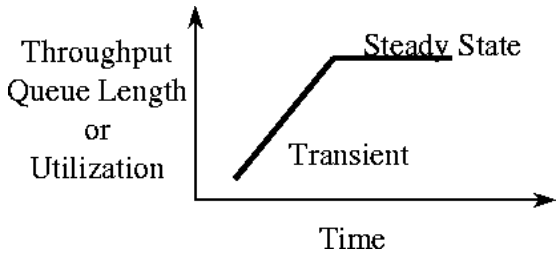


Figure 3.5: Transient vs Steady State Performance

In the real world, the transient response is almost as important as steady state performance. Jain and Routhier [47] point out that most real world traffic is bursty because most sources are transient. Transient response can be tested using transient sources which start after other sources have started and/or stop before the other sources have stopped. Good schemes must be able to respond rapidly to these load transients and achieve optimal performance in the steady state. The difference between steady state and transient performance is shown in figure 3.5.

The transient performance implies that the scheme should converge quickly from overloaded or underloaded states to the steady state, given that the sources can respond to the feedback. Typically, in linear control systems, the transient can either be short and sharp (i.e., resulting in large transient queues), or slow and smooth (underutilization and near-zero queues). On the other hand, if the scheme is optimized just for transients, the steady state may exhibit oscillations. We desire a scheme which optimizes both the transient and the steady state performance, i.e., quickly converges to a solid steady state from any initial conditions, and drains the queues produced in the transient phase rapidly. The "congestion avoidance" steady state is finally reached when both the rates and the queues stabilize.

3.5 Miscellaneous Goals

Robustness and Handling High Variation Workloads: Another complexity issue for the ABR service (and in general for data service classes in high speed integrated service networks) is the fact the available capacity for data traffic is variable. Older networks typically had constant capacity links dedicated for data transmission and the capacity was also lower. As a result, the congestion

control problem was also simpler. Specifically, the ABR service needs to provide good performance given that the variation in both the demand and capacity is high. Demand is usually variable because of the bursty nature of data traffic. Capacity varies because of the presence of higher priority CBR or VBR traffic classes. Further, the scheme must be robust to adapt to conditions like delayed or lost feedback.

Implementation Cost/Performance tradeoffs: The scheme should provide several tradeoff points for implementation incurring different costs. The basic version of the scheme should not require expensive implementation (for example, per-VC queuing as in the case of credit-based schemes). The scheme should be flexible enough to perform well for the target workload scenario, at an acceptable cost.

Scalability: The scheme should not be limited to a particular range of speed, distance, number of switches, or number of VCs. Typical parameters which have scalability implications are: the amount of buffers required, the buffer allocation, queuing and scheduling policy required, the number of switch algorithm operations required per-control cell, and the convergence time (time taken to reach a steady state) of the switch algorithm.

Implementation, Space and Time Complexity: The scheme should be simple to implement - it should not require measurements or logic which are expensive. Further, the amount of memory required for the scheme should be minimal. The best possibility is to have a constant space (or O(1) space complexity) algorithm which utilizes constant space irrespective of the number of VCs setup

or active. The Phantom algorithm [3] is an example of an O(1) space complexity algorithm. If the algorithm is not O(1) in space, it should utilize no more than a constant amount of space per connection (extra space in the VC table) to store per-VC parameters.

The next requirement is for the algorithm to execute a constant number of steps to calculate feedback, especially when an RM cell is being processed. Such an algorithm is said to have an O(1) time complexity. Some algorithms like those developed in this dissertation may do some simple O(N) operations in the background (like the averaging of measured quantities, clearing of bits etc). Such operations can also be efficiently compiled on parallel architectures to have a lower execution complexity like O(log N). However, the algorithm should not require complex implementations like per-VC queuing and scheduling.

3.6 ABR Switch Scheme Limitations

ABR switch algorithms are limited in the following respects:

- **Initial Cell Rate of Sources**: Sources negotiate the Initial Cell Rate (ICR) parameter from the switches along the path during connection setup. Switch algorithms attempt to allocate the available capacity among the currently active sources. In practice, though a large number of VCs are setup, only a few are active at any time. If a number of inactive VCs suddenly become active, and they send data at the negotiated ICR. This may cause a buffer overflow at the switch before the feedback reaches the sources, and the zero loss goal is not met. It is possible that the negotiated Cell Loss Ratio (CLR) may be violated under such conditions. This is an inherent tradeoff in ABR, but the probability

of such an occurrence is expected to be low. For high latency paths, the cell loss can be controlled using source end system parameters (open loop control) during the first round trip. ICR negotiation for ABR is a connection admission control (CAC) problem.

- **Time lag for feedback to be effective**: Observe that a switch does not give rate feedback with every cell. In particular, a feedback may be given in an RM cell traversing in the forward or reverse direction. Sometimes the switch experiences a load, but may not find RM cells to give the feedback. Again, the result is that the feedback is delayed. It is also possible that the load disappears when the RM cell actually arrives. In summary, there is a time lag between the switch experiencing a load, giving feedback, and experiencing the new load due to the feedback. There are three components which affect this time lag:

  Round trip time and Feedback delay: There is a non-zero delay between the switch giving feedback, the sources receiving the feedback, and the switch experiencing the new load.

  The *round trip time (RTT)* is the time taken by a cell to traverse the path from the source to the destination and back. It includes the transmission, propagation and the queuing delays. The sum of the transmission and propagation delays is called fixed round trip time (FRTT).

  The time between the switch giving the feedback and it experiencing the load due to the feedback is called *feedback delay (FD)*. In general, the feedback delay is less than or equal to the round trip time. The shortest feedback delay (SFD) is twice the propagation and transmission delay from

the source to the switch. The feedback delay equals RTT for sources sending data after idle periods, because that is when these sources get their first feedback for the new burst. The feedback delay equals SFD for sources which are already active, and new sources overload the switch.

Inter-RM Cell Time (IRCT): The switches can give feedback in every RM cell they see, and no faster. Hence, the time between successive RM cells determines the rate of feedback. This is true once there is a continuous flow of RM cells (the control loop is set up). The inter RM cell time (IRCT) is not constant, but a function of the source rate. Specifically, it is the inverse of the rate of RM cells, which in turn is a small fraction (1/32) of the source's rate. The IRCT is large when the source rates are small. The IRCT and the switch averaging interval (see the next item) is usually the dominant factor in local area networks (LANs) in determining the time lag for feedback to be effective (note that RTTs and SFDs are small in LANs). Further, in LANs, due to asynchrony sources with smaller IRCTs get feedback faster.

Switch Averaging Interval (AI): The switches usually measure certain quantities which are then used to calculate rate feedback. These quantities are called "metrics." For example, switches typically measure the ABR capacity to be shared among contending sources. Some switch algorithms may measure other quantities like the number of active ABR sources, the input rate and the individual rates of the sources. One important concern of the switch algorithm is to maintain the correlation of the measured quantities ("control") and the "feedback" given. Our algorithm, ERICA achieves this

by giving only one feedback per measurement (i.e., per averaging interval). As a result, the length of the averaging interval determines how often new feedback is given to sources. Shorter averaging intervals result in more feedback, but also more variation in feedback. Longer intervals impact the response time to load changes, but provides more stable feedback in the presence of asynchrony, and heterogeneous RTTs and FDs. The averaging interval length and the IRCT are the dominant factors in LANs (and for sources having short SFDs and RTTs).

In general, the time required for convergence due to a change in load, assuming no further changes in load is depends upon the RTTs of the newly active connections, the feedback delays of already active connections, the length of the averaging interval and the inter-RM Cell time of all connections.

- **ACR Retention** by sources: It is possible that a source gets a high allocation, becomes idle and uses the (now stale) high allocation later when the network conditions have changed. This is called ACR Retention. When multiple sources use their stale allocations simultaneously, buffers may overflow at the switch before feedback is effective. While there are some source policies which can control this, the problem is not eliminated. The switches can gradually decrease a source's allocation (down to ICR) if it detects inactivity, but such policies can reduce the average response time and throughput of bursty sources over any time scale. There are several solutions to this problem (described in chapter 7, each with a set of side effects.

- **Measurement Inaccuracies**: Measurement and the use of metrics is sometimes preferred over the use of parameters to characterize the system and calculate the feedback. This is because, measurement gives real data as opposed to assumed values due to parameters. However, measurement can introduce variance in the system because of inaccuracies during measurement. The more the number of metrics, the more the effect of variance. Variance can be reduced by taking averages of quantities rather than instantaneous values. Averages taken over an interval may still not capture certain conditions in the input workload. This may cause unnecessary queues or rate fluctuations and limit the accuracy with which the goals are achieved. Compensation for these measurement errors must be provided in the algorithm. Interestingly enough, such compensation requires the use of parameters. For example, the drain capacity may be parametrically increased when queues increase without control, due to measurement errors.

- **Limitations of Parametric Control**: The main problem with parameters is to find optimal sets of parameters for the different workload conditions. Even with optimal sets, it may not be possible to achieve optimal performance, because of the tradeoffs in the design of the parameters. For example, a parameter may control the maximum rate increase per feedback. This parameter limits the convergence time from underload. When the input traffic pattern is not known and the wrong parameter sets are chosen, the performance can degrade drastically. It is hence important to minimize the set of parameters, understand their effects and make the parameters easily settable, and design the scheme to provide acceptable performance even for slightly mistuned parameters.

CHAPTER 4

SURVEY OF ATM SWITCH CONGESTION CONTROL SCHEME PROPOSALS

In this chapter, we shall look at a number of ATM rate-based feedback congestion control scheme proposals. Prior to the development of rate-based mechanisms, there was a prolonged debate in the ATM forum between the hop-by-hop credit (or window) based framework and the end-to-end rate-based framework [52]. We will briefly summarize this debate and concentrate on the survey of rate-based switch algorithms. For each algorithm, we will identify the key contributions and present its drawbacks. This will lay a foundation for comparison with the OSU, ERICA and ERICA+ schemes developed in this dissertation. It should be noted that several of these schemes were designed after the ERICA scheme was developed, and therefore, may exhibit some overlap of concepts. This chapter uses the terminology developed in section 3.

4.1 Credit-Based Framework

The credit-based framework was proposed by Professor H. T. Kung, it was supported by Digital, BNR, FORE, Ascom-Timeplex, SMC, Brooktree, and Mitsubishi [64, 24]. The approach bears some similarity in concept to the sliding window-based

protocols used in data link control protocols. The framework consists of a per-link, per-VC window flow control. Each link consists of a sender node (which can be a source end system or a switch) and a receiver node (which can be a switch or a destination end system). Each node maintains a separate queue for each VC. The receiver monitors queue lengths of each VC and determines the number of cells that the sender can transmit on that VC. This number is called "credit." The sender transmits only as many cells as allowed by the credit.

If there is only one active VC, the credit must be large enough to allow the whole link to be full at all times. In other words:

$$\text{Credit} \geq \text{Link Cell Rate} \times \text{Link Round Trip Propagation Delay}$$

The link cell rate can be computed by dividing the link bandwidth in Mbps by the cell size in bits.

The scheme as described so far is called "Flow Controlled Virtual Circuit (FCVC)" scheme. There are two problems with this initial static version. First, if credits are lost, the sender will not be aware of it. Second, each VC needs to reserve the entire round trip worth of buffers even though the link is shared by many VCs. These problems were solved by introducing a credit resynchronization algorithm and an adaptive version of the scheme.

The credit resynchronization algorithm consists of both sender and receiver maintaining counts of cells sent and received for each VC and periodically exchanging these counts. The difference between the cells sent by the sender and those received by the receiver represents the number of cells lost on the link. The receiver reissues that many additional credits for that VC.

The adaptive FCVC algorithm [63] consists of giving each VC only a fraction of the round trip delay worth of buffer allocation. The fraction depends upon the rate at which the VC uses the credit. For highly active VCs, the fraction is larger while for less active VCs, the fraction is smaller. Inactive VCs get a small fixed credit. If a VC doesn't use its credits, its observed usage rate over a period is low and it gets smaller buffer allocation (and hence credits) in the next cycle. The adaptive FCVC reduces the buffer requirements considerably but also introduces a ramp-up time. If a VC becomes active, it may take some time before it can use the full capacity of the link even if there are no other users.

4.2 Rate-Based Approach

This approach, which was eventually adopted as the standard was proposed originally by Mike Hluchyj and was extensively modified later by representatives from twenty two different companies [25].

Original proposal consisted of a rate-based version of the DECbit scheme [46], which consists of end-to-end control using a single-bit feedback from the network. Initially, sources send data at an negotiated "Initial Cell rate." The data cells contain a bit called the EFCI bit in the header. The switches monitor their queue lengths and, if congested, set the EFCI bit in the cell headers. The destination monitors these indications for a periodic interval and sends an RM cell back to the source. The sources use an additive increase and multiplicative decrease algorithm to adjust their rates.

This is an example of a "bit-based" or "binary" feedback scheme. As we shall see later in this section, it is possible to give explicit rate feedback in the rate-based

framework. The complete framework has been treated in the chapter introducing ATM traffic management 2.

4.3 Binary Feedback Schemes

4.3.1 Key Techniques

Binary feedback schemes essentially use a single bit feedback. The initial binary feedback algorithm used a "negative polarity of feedback" in the sense that RM cells are sent only to decrease the source rate, and no RM cells are required to increase the rate. A "positive polarity," on the other hand, would require sending RM cells for increase but not on decrease. If RM cells are sent for both increase and decrease, the algorithm would be called "bipolar."

The problem with negative polarity is that if the RM cells are lost due to heavy congestion in the reverse path, the sources will keep increasing their load on the forward path and eventually overload it.

This problem was fixed in the next version by using positive polarity. The sources set EFCI on every cell except the nth cell. The destination will send an "increase" RM cell to source if they receive any cells with the EFCI off. The sources keep decreasing their rate until they receive a positive feedback. Since the sources decrease their rate proportional to the current rate, this scheme was called "proportional rate control algorithm (PRCA)."

PRCA was found to have a fairness problem. Given the same level of congestion at all switches, the VCs traveling more hops have a higher probability of having EFCI set than those traveling smaller number of hops. If p is the probability of EFCI being set on one hop, then the probability of it being set for an n-hop VC is $1-(1-p)^n$ or

np. Thus, long path VCs have fewer opportunities to increase and are beaten down more often than short path VCs. This was called the "beat-down problem [12]."

One solution to the beat down problem is the "selective feedback" [72] or intelligent marking [9] in which a congested switch takes into account the current rate of the VC in addition to its congestion level in deciding whether to set the EFCI in the cell. The switch computes a "fair share" and if congested it sets EFCI bits in cells belonging to only those VCs whose rates are above this fair share. The VCs with rates below fair share are not affected.

The RM cell also contains two bits called the "Congestion Indication" bit and the "No Increase" bit. Schemes which mark these bits are not necessarily classified as "binary schemes" since they may feedback more information than just one bit. Several ER-based schemes like CAPC2 and DMRCA (discussed later in this chapter) also use the CI and NI bit setting options.

4.3.2 Discussion

The attractive features about the binary schemes are:

- Simple to implement. Requires only a bit in the header. Feedback calculation is also typically simple. The multiple round trip times required for convergence is not a problem for local area networks because the round trip delay for these networks is in the order of microseconds.

- Cost effective option to introduce ABR in LANs.

- Typical bit-based schemes look only at the queue length as a metric for congestion. Though we point out later that the queue length is not an accurate

metric of congestion, using a single metric localizes the errors possible. Further, since the end-result of all errors is additional queues, taking the queue length as a metric is a safe method to avoid divergence, even when the demand and capacity are variable.

The drawbacks of this approach are:

- The bit-based feedback schemes were initially designed for low-speed networks. Since a bit gives only two pieces of information ("up" or "down"), the system may take several round trips to converge to stable values. That is, the transient convergence period is long. During this period, the network might either be underutilized, or queues might build up, which is a definite concern in high-speed networks.

- The bit-based feedback was originally designed for window-flow control where the maximum queue is simply the sum of all source windows. In rate control, if the sum of the source rates is larger than the capacity, the queues could grow to infinity unless the rates are changed [50, 48]. The transient convergence period determines what this worst case queue will be. The queue can be large (proportional to the steady state queue plus a term proportional to the transient convergence time) when a new source starts up after the system is in the steady state.

- The bit-based feedback was originally designed for connectionless networks where it is possible that packets from a source to a destination may take multiple paths. Hence, it is not a good idea to give authoritative feedback information based on partial knowledge. On the other hand, switches in connection-oriented

networks like ATM can have complete knowledge about a flow based upon measurement, and can give authoritative feedback.

- The steady state itself exhibits oscillatory behavior in terms of queue length and rate allocations. The reason for the behavior is that the control is based upon the queue length, a highly variable quantity in rate-based control. Further, when the queue length is zero, or beyond a threshold, the schemes essentially "guess" the allocations due to the unreliability of the metric.

- The technique requires several parameters to force convergence. The system is also quite sensitive to the parameter settings.

- The buffer requirement at switches is large, and increases linearly with the number of connections.

- The "beat-down" fairness problem needs to be solved in every implementation. This adds to the complexity at switches.

A theme we gather from the above observations is that the bit-based feedback and the technique of using queue length as a congestion indicator is a legacy from the window-based control schemes for low-speed, connectionless networks. The adaptation to rate-based, high-speed, connection-oriented networks like ATM has some advantages in terms of simplicity, and hence cost-effectiveness. But, the performance in the Wide Area scenario leaves a lot to be desired. This led to the introduction of explicit rate feedback schemes.

4.4 Explicit Rate Feedback Schemes

In July 1994, Charny, Clark and Jain [18] argued that the binary feedback was too slow for rate-based control in high-speed networks and that an explicit rate indication would not only be faster but would offer more flexibility to switch designers.

In addition to providing a solution to the problems of the bit-based feedback schemes described in the previous section, explicit rate schemes are attractive for other reasons. First, policing is straight forward. The entry switches can monitor the returning RM cells and use the rate directly in their policing algorithm. Second with fast convergence time, the system come to the optimal operating point quickly. Initial rate has less impact. Third, the schemes are robust against errors in or loss of RM cells. The next RM cell carrying "correct" feedback will bring the system to the correct operating point in a single step.

Further, one of the reasons for choosing the rate-based framework was that ABR could be used for applications other that just data applications - to provide a cost-effective alternative to applications that traditionally use higher priority classes. Typical applications are compressed video, which could tolerate variable quality. The explicit rate schemes could reduce the variation in the rates seen at the end-systems, higher throughput and a controlled delay through the network. Further, the video applications could directly use the rate values to tune their parameters as opposed to the credit value, which cannot be directly used without knowledge of the round trip delay.

In the following sections we survey several rate-based explicit feedback schemes. In each section, we will have a brief discussion of the key techniques used by the

scheme followed by a discussion of the contributions and drawbacks of the proposed scheme.

4.5 MIT Scheme

The explicit rate approach was substantiated with a scheme designed by Anna Charny during her master thesis work at the Massachusetts Institute of Technology (MIT) [18, 17].

4.5.1 Key Techniques

The MIT scheme consists of each source sending an control (or RM) cell every n th data cell. The RM cell contains the VC's current cell rate (CCR) and a "desired rate." The switches monitor all VC's rates and compute a "fair share." Any VC's whose desired rate is less than the fair share is granted the desired rate. If a VC's desired rate is more than the fair share, the desired rate field is reduced to the fair share and a "reduced bit" is set in the RM cell. The destination returns the RM cell back to the source, which then adjusts its rate to that indicated in the RM cell. If the reduced bit is clear, the source could demand a higher desired rate in the next RM cell. If the bit is set, the source uses the current rate as the desired rate in the next RM cell.

The switches maintain a list of all of its VCs and their last seen desired rates. All VCs whose desired rate is higher than the switch's fair share are considered "overloading VCs." Similarly, VCs with desired rate below the fair share are called "underloading VCs." The underloading VCs are bottlenecked at some other switch and, therefore, cannot use additional capacity at this switch even if available.

The capacity unused by the underloading VCs is divided equally among the overloading VCs. Thus, the fair share of the VCs is calculated as follows:

$$\text{Fair Share} = \frac{\text{Capacity} - \sum \text{Bandwidth of underloading VCs}}{\text{total number of VCs} - \text{Number of underloading VCs}}$$

It is possible that that after this calculation some VCs that were previously underloading with respect to the old fair share can become overloading with respect to the new fair share. In this case these VCs are re-marked as overloading and the fair share is recalculated.

Charny [17] has shown that two iterations are sufficient for this procedure to converge. Charny also showed that the MIT scheme achieves max-min optimality in $4k$ round trips, where k is the number of bottlenecks.

4.5.2 Discussion

The contributions of the MIT scheme were as follows:

- Help define the framework for explicit rate feedback mechanisms in the ATM ABR specifications
- Provided a reference iterative algorithm
- Max-min fairness is achieved because the underloading VCs see the same advertised rate
- The switch algorithm is essentially a rate calculation algorithm which is not concerned with the enforcement of the rates. The enforcement of rates may be carried out either at the edge of the network or at every network switch though queuing and scheduling policies. This algorithm gives the network designer the

flexibility of decoupling the enforcement and feedback calculation. This aspect has since become a standard feature in all schemes developed.

- The algorithm quickly adapts to dynamic changes in the network provided the declared values of the parameters "desired rate" etc are accurate. The algorithm is shown to be "self-stabilizing" in the sense that it recovers from any past errors, changes in the set of network users, individual session demands and session routes.

- The algorithm provides fast convergence to max-min rates (within $4k$ round trips, where k is the number of bottlenecks).

- Charny also shows that the algorithm is "well-behaved" in transience, i.e., given an upper bound on the round-trip delay, the actual transmission rates can be kept *feasible* throughout the transient stages of the algorithm operation while still providing reasonable throughput to all users. A feasible set of rate allocations ensures that a rate allocation is such that no link capacity is exceeded. The arguments assumed synchronization among sources, or a special source policy which forces synchronization in the asynchronous case.

The drawbacks of the scheme were:

- The computation of the fairshare requires order n operations, where n is the number of VCs. The space requirements of the scheme are also order n.

- The feedback procedure is unipolar, i.e., switches only reduce the rates of sources. As a result, the sources require an extra round trip for increase. This feature is addressed in the Precise Fair Share Computation option of the OSU

scheme which provides a bipolar feedback (i.e., switch can increase as well as decrease the desired rates).

- If the sources do not use their allocations, or temporarily go idle, there is no mechanism prescribed to detect this condition. Since the scheme relies on the declared values and does not measure the source rates, nor the offered load, it is possible that the offered load is very different from the sum of the desired rate values, leading to underutilization.

- There is no policy prescribed to drain queues built up during transient periods, or errors in feedback.

- The scheme as described is not compatible with the current ATM Forum standard, and requires minor realignment to be compatible.

This proposal was well received, and considered a baseline for other schemes to be compared with. The key exception was that the computation of fair share requires order n operations, where n is the number of VCs. The space requirements of the scheme are also order n. This set off a search for schemes which were O(1) both in time and space complexity. This led to the EPRCA, the OSU scheme proposals in September 1994, and later the CAPC2 proposal in late 1994. We continue our survey looking at these schemes. The OSU scheme and ERICA schemes will be treated in separate chapters of this dissertation.

4.6 EPRCA and APRC

The merger of PRCA with explicit rate scheme lead to the "Enhanced PRCA (EPRCA)" scheme at the end of July 1994 ATM Forum meeting [74].

4.6.1 Key Techniques

In EPRCA, the sources send data cells with EFCI set to 0. After every n data cells, they send an RM cell. The RM cells contain desired explicit rate (ER), current cell rate (CCR), and a congestion indication (CI) bit. The sources initialize the ER field to their peak cell rate (PCR) and set the CI bit to zero.

The destinations monitor the EFCI bits in data cells. If the last seen data cell had EFCI bit set, they mark the CI bit in the RM cell.

In addition to setting the explicit rate, the switches can also set the CI bit in the returning RM cells if their queue length is more than a certain threshold. Some versions of the EPRCA algorithm do not set the EFCI bits, and mark the CI and ER fields alone.

The scheme uses two threshold values QT and DQT on the queue length to detect congestion. When the queue length is below QT, all connections are allowed to increase their rate.

When the queue length exceeds QT, the switch is considered congested and performs *intelligent marking*. By intelligent marking, we mean that the switch selectively asks certain sources to increase their rates and certain sources to reduce their rates. In order to do this, the switch maintains the Mean ACR ($MACR$), and selectively reduces the rate of all connections with ACR larger than $MACR$. The switch may reduce the rates by setting the CI bit and/or by setting the ER field of an RM cell when CCR value exceeds $MACR \times DPF$ (DPF is the *Down Pressure Factor*). The DPF is introduced to include those VCs whose rate is very close to $MACR$. Typically DPF is 7/8. The CI bit setting forces the sources to decrease their rate as described in the source end system rules (see chapter 2.

If the port remains congested and the queue length exceeds DQT threshold, the switch is considered heavily congested and all connections have their rate reduced.

To avoid the O(N) computation of the advertised rate, the fairshare is approximated by $MACR$ using the running exponential weighted average, computed every time the switch receives an RM cells,as :

$MACR = MACR(1 - AV) + CCR * AV$

where AV is an averaging factor, typically equal to 1/16, allowing the implementation using addition and shift operations.

4.6.2 Discussion

The contributions of the EPRCA scheme are:

- Introduced a class of algorithms which operate with O(1) space and O(1) time requirements.
- EPRCA allows both binary-feedback switches and the explicit feedback switches on the path, since it bridged the gulf between PRCA and explicit rate schemes. This feature has been incorporated in the ATM Traffic Management standard.
- Uses the mean ACR as the threshold and allocates this rate to all unconstrained VCs. This technique converges to fair allocations when the mean ACR is a good estimate of the "fair share," i.e., the max-min advertised rate (computed by the MIT scheme).

The drawbacks of the EPRCA scheme are:

- If the mean ACR is not a good estimate of the "fair share," then the scheme can result in considerable unfairness [20].

- The exponential averaging of the rates may become biased towards the higher rates. For example, consider two sources running at 1000 Mbps and 1 Mbps. In any given interval, the first source will send 1000 times more control cells than the second source and so the exponentially weighted average is very likely to be 1000 Mbps regardless of the value of the weight used for computing the average.

  The problem is that the exponential averaging technique (which is similar to the arithmetic mean) is not the right way to average a set of ratios (like ACRs = number of cells/time) where the denominators are not equal [49]. We address this averaging issue in the design of ERICA later in this dissertation.

- The scheme uses queue length thresholds for congestion detection. As a result, it effectively "guesses" the rate allocations when the queue value is zero, or above the high threshold. We will argue later in this dissertation that the queue length does not provide full information about the congestion at the switch, and hence is not reliable as the primary metric for rate-based congestion control.

- The scheme uses a number of parameters whose values are typically set conservatively. This technique trades off transient response time (time required to reach the steady state after a change in network conditions). This means that the utilization of the bottlenecks will be lower on the average compared to aggressive allocation schemes. Further, when the network is constantly in the state when demand and capacity are variable (no steady state), the performance

of the scheme is unclear, and is expected to be lower because of the conservative parameter settings.

Researchers at University of California at Irvine (UCI) suggested a solution to the problems of EPRCA through a scheme they developed called "Adaptive Proportional Rate Control" [57]. Essentially, they suggested that the queue growth rate be used as the load indicator instead of the queue length. The change in the queue length is noted down after processing, say, K cells. The overload is indicated if the queue length increases.

However, this approach still suffers from the defect that the metric gives no information when the queue lengths are close to zero (underutilization). Basically, the problem is that the queue length information needs to be combined with the ABR capacity and ABR utilization to get a full picture of the congestion situation at the switch.

4.7 CAPC2

In October 1994, Barnhart from Hughes Systems proposed a scheme called "Congestion Avoidance using Proportional Control (CAPC)[10]."

4.7.1 Key Techniques

This scheme used some of the concepts developed in the OSU scheme and used a phase-locked loop style filter in the algorithm. In this scheme, as in OSU scheme (described later in this dissertation), the switches set a target utilization parameter slightly below 1. This is the ABR capacity utilization the scheme aims to achieve. As in this OSU scheme, the switches measure the input rate and load factor z (which

is the ratio of the input rate to the product of the ABR capacity and the target utilization). The load factor z is used as the primary congestion detection metric as opposed to using the queue length for that purpose. The scheme calculates a single "fairshare" using the load factor as follows.

During underload ($z < 1$), fair share is increased as follows:

$$\text{Fair share} = \text{Fair share} \times \text{Min}(ERU, 1 + (1 - z) * Rup)$$

Here, Rup is a slope parameter in the range 0.025 to 0.1. ERU is the maximum increase allowed and was set to 1.5.

During overload ($z > 1$), fair share is decreased as follows:

$$\text{Fair share} = \text{Fair share} \times \text{Max}(ERF, 1 - (z - 1) * Rdn)$$

Here, Rdn is a slope parameter in the range 0.2 to 0.8 and ERF is the minimum decrease required and was set to 0.5.

The fair share is the maximum rate that the switch will grant to any VC.

This method of using (1- z) (or a term proportional to unused capacity) for feedback calculation is also used by the Phantom [3] described later in this survey.

In addition to the load factor, the scheme also uses a queue threshold. Whenever the queue length is over this threshold, a congestion indication (CI) bit is set in all RM cells. This prevents all sources from increasing their rate and allows the queues to drain out.

4.7.2 Discussion

The CAPC scheme and its successor CAPC2 (which addressed some initialization issues) was proposed in late 1994, before many of the scheme proposals surveyed in this chapter. The contributions of the CAPC scheme include:

- An oscillation-free steady state performance. The frequency of oscillations is a function of $1 - z$, where z is the load factor. In steady state, $z = 1$, the frequency is zero, that is, the period of oscillations is infinite.

- Simple to implement.

- Uses the load factor as the primary metric, and does not use the CCR field.

- The single "fairshare" threshold is similar to the EPRCA concept. This allows the scheme to have an O(1) space complexity and easily converge to fairness under conditions of constant demand and capacity.

The drawbacks of the scheme include:

- The convergence time of the scheme is longer since it uses parameters whose values are chosen conservatively.

- Since the algorithm uses a binary indication bit in very congested states, it is prone to unfair behaviors [3].

4.8 Phantom

This scheme was developed by Afek, Masour and Ostfeld at the Tel-Aviv University [3]. An important design goal in this work is to develop a *constant space* congestion avoidance algorithm, while achieving max-min fairness, and good transient response.

4.8.1 Key Techniques

The key idea is to bound the rate of sessions that share a link by the amount of unused bandwidth on that link. The scheme uses the concept of a *Phantom* session

which shares the link equally as all other connections. The link allocates rates fairly among all sources including the phantom.

Specifically, the variable Δ is defined to be the unused link capacity, i.e.,

Link Capacity $-$

Σ(Rates of sessions that use the link).

It is measured as:

$(Numberofcellstransmittableonlink - Numberofcellsinput)/\tau$

where τ is a fixed time interval.

Observe that Δ can be greater than zero, when the actual queue at the link is non-zero.

The rate of sessions that are above Δ are reduced towards Δ and the rate of sessions that are below Δ may be increased. The mechanism reaches a steady state only when the unused capacity (Δ) is equal to the maximum rate of any session that crosses the link and all the sessions that are constrained by the link are at this rate. So, Δ is the "fairshare" value at each link.

For example [3], if three sessions share a 100 Mbps link, then in the steady state, each session receives 25 Mbps and the link utilization is 75 % ($\Delta = 25Mbps$). However, if two of the three sessions are restricted elsewhere to 10 Mbps each, the third sessions gets 40 Mbps ($\Delta = 40Mbps$).

The scheme addresses five important implementation aspects:

1. **Measuring Δ:** Naive measurement of Δ can be very noisy. The scheme uses exponential averaging to smooth out variance in Δ and accumulates it in a variable called "Maximum Allowed Cell Rate (MACR)":

$$MACR = max(MACR \cdot (1 - \alpha) + \Delta \cdot \alpha, MACR \cdot dec\_factor)$$

The lower bound is required to filter out variations caused by sudden capacity changes.

2. **Sensitivity to queue length:** The scheme recognizes the need to compensate for errors, and transient queues by looking at the absolute value of the queue length. The averaging parameter α is replaced by two parameters α_{inc}, when $\Delta > MACR$, and α_{dec}, when $\Delta \leq MACR$. Both these parameters vary depending upon the queue length.

3. **Utilization:** The utilization may be improved by restricting the bandwidth of connections by $utilization\_factor$ times $MACR$, instead of Δ.

4. **Variance consideration:** The problem with the utilization factor is that $MACR$ may exhibit large oscillations. The scheme therefore smoothes out the factors α_{inc} and α_{dec} based upon the variance in Δ. The algorithm used is similar to the TCP RTT estimation smoothing algorithm.

5. **Reducing maximum queue length:** The scheme also sets the NI bit based on another variable called $Fast\_MACR$ which tracks the variation of the capacity more closely.

4.8.2 Discussion

The contributions of the Phantom scheme are as follows:

- The idea of a phantom connection, combined with the utilization factor can bring the allocations close to max-min. The basic algorithm allocates rates

proportional to the the unused ABR capacity. The inclusion of the utilization factor brings the efficiency closer to the maximum possible.

- The algorithm developed is O(1) in both space and time requirements.
- In the basic scheme, the residual unused capacity to accommodate new sessions without queue buildup.
- Fairness is maintained because, over a period, all sources see the same "advertised rate" (MACR).
- The scheme explicitly addresses the issues in measurement, variance reduction and error compensation. The variance suppression is a necessity for the scheme since the phantom bandwidth, Δ is highly variant.
- The exponential averaging of measurements is valid (unlike the EPRCA algorithm where the technique is dubious) because they are made over fixed intervals. We note again that the arithmetic mean (or exponential averaging) is not the correct method for averaging ratios where the denominator is not constant [49].
- The scheme considers the issues of high bottleneck utilization combined with a systematic method to cope with queuing delays (due to transient queues).
- Fast implementations can be derived by replacing multiply operations by bit-shifting.

The drawbacks of the scheme are:

- The use of the utilization factor introduces higher degree of variation in load, and the possibility of sharp queue spikes. This necessitates complex variance reduction and queue control techniques within the algorithm, and introduces several extra parameters. The scheme may also require the sources to negotiate a lower value of the "Rate Increase Factor (RIF)" parameter to moderate the network-directed rate increases.

- The queue thresholding procedures may require a new set of parameter recommendations for Wide Area Networks. It is not clear whether the scheme will work in WANs without complex parameter changes.

4.9 UCSC Scheme

This scheme was proposed by researchers Kalampoukas and Varma at the University of California, Santa Cruz (UCSC), and K.K. Ramakrishnan of AT&T Research [58].

4.9.1 Key Techniques

The scheme cleverly approximates the MIT scheme with O(1) bookkeeping, and hence brings the computational complexity from O(N) to O(1). Intuitively, the scheme spreads the MIT O(N) iteration over successive RM cells. As a result, the convergence time and buffer requirements are traded off with computational complexity. The space requirements compared to the MIT scheme remain O(N) since the scheme maintains some per-VC state information. In special cases, optimization may be achieved by using shift operations instead of multiply/divide operations.

The key technique in the scheme is the following. On the lines of the MIT scheme, the scheme assumes that the source demands a certain rate, and the switch tries to satisfy that demand. In the scheme, VC_i "requests" bandwidth equal to $min(ER_i, CCR_i)$. We can consider this as the "demand" of VC_i. The same quantity can also be considered as the bandwidth "usage" of the VC. The scheme computes a "maximum bandwidth" value A_{max} depending upon the VC's current state. A_{max} is the *fairshare* which is given to the source as feedback. Next, we describe the states of the VCs and show how they are used to compute the bandwidth allocations.

Each VC_i can be in one of the following two states:

1. **Bottlenecked:** the switch cannot allocate the requested bandwidth to VC_i on the outgoing link, $A_{max} < min(ER_i, CCR_i)$. The set of bottlenecked connections is B. Intuitively, the bottlenecked connections are those that can use a higher rate allocation at the switch.

2. **Satisfied:** the switch can satisfy the request, $A_{max} \geq min(ER_i, CCR_i)$. The set of bottlenecked connections is S. Intuitively, the bottlenecked connections are those which cannot use even the current maximum bandwidth allocation A_{max}. In some sense, they are currently "saturated."

Typically, a given VC_i will be in different states (bottlenecked and satisfied) at different switches. Observe that connections can move from one state to another depending upon their demand and the available bandwidth. *Free bandwidth* is defined as the amount of bandwidth available as a result of the satisfied connections not claiming their equal share, B_{eq}. The computation of the maximum bandwidth allocation for a connection is done as follows.

First, the state changes of the connection are detected and variables updated:

- If $A_{max} < min(ER_i, CCR_i)$, VC_i is marked as "bottlenecked." Further, in this case, if VC_i was "satisfied" prior to the update, the free bandwidth, B_f is updated, and the number of bottlenecked connections, N_{bot} is incremented.

  Observe that the VC's allocation A_i is not updated since it is not used in the computation of A_{max} as long as it is bottlenecked.

- If $A_{max} \geq min(ER_i, CCR_i)$, VC_i is marked as "satisfied;" its allocation A_i is set to $min(ER_i, CCR_i)$; the free bandwidth, B_f is updated.

  Further, if VC_i was "bottlenecked" prior to the update, the number of bottlenecked connections, N_{bot}, is decremented.

The next step is the computation of the bandwidth allocation. If a connection, $VC_i \in B$, i.e., is currently *bottlenecked*, its maximum allocation (or fairshare, A_{max}) is calculated as:

$$A_{max} = B_{eq} + \frac{B_f}{N_{bot}}$$

On the other hand, if $VC_i \in S$, i.e., is currently satisfied, it is *treated as bottlenecked* and the maximum allocation (or fairshare, A_{max}) is calculated as:

$$A_{max} = B_{eq} + \frac{B_f + A_i - B_{eq}}{N_{bot} + 1}$$

In the preceding equation, observe that the bandwidth allocation of VC_i over and above the equal share $A_i - B_{eq}$ is also considered as part of the "free bandwidth". The use of $N_{bot} + 1$, in the denominator of the fraction shows that the source is considered a bottlenecked connection in the calculation. The purpose of this step is to ensure

the bandwidth allocations to *satisfied* connections as always less than or equal to the allocations to *bottlenecked* connections. The algorithm thus "claims" back any extra bandwidth previously allocated to the connection.

The explicit rate field in the RM cell is updated as:

$$ER_i = min(ER_i, A_{max})$$

4.9.2 Discussion

The authors classify their work as a "state-maintaining" algorithm since they maintain state information on a per-connection basis. They observe that "stateless" algorithms which do not maintain per-connection state may allocate rates such that there may be significant discrepancies between the sum of the ER values signaled to ABR connections and available link bandwidth.

While this observation is valid in general, an optimistic over-allocation can help increase network utilization, especially in cases when the ABR demand and capacity is variable. The arguable risk is that of queuing delays.

The contributions of the UCSC scheme are the following:

- O(1) emulation of MIT scheme concept
- Focus on scalability. If the VCs set up are always active, then the scheme has O(1) computational complexity with respect to the number of VCs.
- In the steady state, $min(ER_i, CCR_i$ gives the path bottleneck rate. This is because ER_i gives the downstream bottleneck rate, while CCR_i gives the upstream bottleneck rate. This observation is valid when the ER marking is done in the backward RM cells.

- The scheme requires no parameter settings.
- Performance analysis with fixed and variable ABR capacity.

The drawbacks of the scheme are:

- The scheme does not measure the load (aggregate input rate) at the switch. As a result, if a source is sending at a rate below its CCR, then the bottleneck will be underutilized.
- The scheme also does not observe the queuing delay at the switch. Errors in estimation of ABR capacity result in errors in feedback and eventually result in queues. Hence, there is a possibility of infinite queues if the queuing delay is not considered as a metric. However, such a mechanism may easily be developed on similar lines as the ERICA+ proposal studied later in the dissertation.
- The scheme assumes that the sum of the number of bottlenecked and satisfied connections is equal to the number of connections setup. The scheme does not measure the number of active connections. As a result, if a connection is setup, but remains idle for a while, the allocations to other connections remain low and may result in underutilization.
- The convergence time is slower since the scheme attempts never to over-allocate (conservative). This non-optimistic strategy may result in link underutilization of the sources are not always active, or cannot utilize their ER allocations [2].

4.10 DMRCA scheme

The *Dynamic Max Rate Control Algorithm (DMRCA)* scheme [20] was developed by Chiussi, Xia and Kumar at Lucent Technologies, in an attempt to improve the EPRCA scheme.

4.10.1 Key Techniques

DMRCA uses a rate marking threshold similar in concept to the MACR of EPRCA. However, the DMRCA threshold is a function of the *degree of congestion* at the switch and the *maximum rate of all active connections*. This rate threshold is used to estimate the maximum fairshare of any active connection on the link.

The authors observe that the EPRCA depends upon the *mean cell rate of all connections* which it uses as a rate marking threshold. If this mean is close to the fairshare of available bandwidth on the link, then EPRCA performs well. But, if the approximation does not hold, then EPRCA introduces considerable unfairness. For example, if some connections are bottlenecked in other switches, they may cause underestimation of the fairshare. Another case is when rates oscillate due to transient behaviors and/or interactions with multiple switches, leading to incorrect estimates of the actual rate of the connection.

The authors propose to use the maximum rate of all the active connections instead of the mean rate used by EPRCA. They observe that the maximum rate of all connections quickly rises to be above the desired "fairshare" (the maximum rate allocation for unconstrained connections at this switch). Further, this value can be made to converge to fairshare in the steady state.

However, certain problems need to be solved before this idea can be used effectively. First, *the maximum VC rate oscillates* excessively leading to transient instabilities in the behavior. This problem can be tackled by smoothing the maximum rate, filtering out the short-term variations. Second, in some situations, *the maximum rate does not converge rapidly* to the fairshare, again compromising fair behavior. The authors address this by using a reduction factor which is a function of the *degree of congestion* in the switch.

DMRCA uses two thresholds QT and DQT on the queue length for congestion detection. The switch also monitors the maximum rate MAX of all connections arriving at the switch, as well as the VC number of the corresponding connection, MAX_VC.

The algorithm smoothes excessive oscillations in MAX using exponential averaging to calculate an *adjusted maximum rate*, as:

$A\_MAX = (1 - Alpha) \times A\_MAX + Alpha \times MAX$

The averaging factor, $Alpha$ is typically 1/16. The implementation is as follows:

if($RMCell->CCR \geq Beta \times MAX$) {

$A\_MAX = (1 - Alpha) \times A\_MAX + Alpha \times RMCell->CCR$

}

This implementation avoids the need for measurement of MAX over a measurement interval. MAX increases when some VC other than MAX_VC observes that its rate is larger than MAX. MAX decreases when MAX_VC updates MAX based

on its CCR. Further MAX times out if it is not updated for a while (as in the case of bursty sources where some sources can become idle and start up with old allocations).

When the queue length exceeds the threshold QT, the switch considers itself congested and performs intelligent marking. The threshold used to perform intelligent marking is:

$Marking\ Threshold = A\_MAX \times Fn(QueueLength)$

where *Function(Queue Length)* is a discrete non-increasing function of the queue length.

The work also addresses how to tackle the case of connections with $MCR > 0$. For example, the fairness criterion "MCR plus Equal Share" is implemented by subtracting the MCR of the corresponding connection for the CCR of each RM cell and using the result as the algorithm. MCR is added back in order to set the ER field in RM cells. The fairness criterion "Maximum of MCR or Max-Min Share" is implemented by simply ignoring the forward RM cells whose CCR is equal to their MCR.

4.10.2 Discussion

The contributions of the scheme are:

- An enhanced EPRCA-like approach with better fairness and control of rate oscillations.
- Low implementation complexity. A chip implementation of the algorithm is available (the "Atlanta" chip of Lucent Technologies [76].
- The use of a single advertised rate threshold value for all VCs results in nearly equal allocations to unconstrained VCs, even in the presence of asynchrony.

- Use of exponentially averaged "Maximum VC rate" instead of "Mean VC rate," combined with an aggressive queue thresholding policy improves efficiency over EPRCA. The scheme is optimistic in the sense that even if there is a single unconstrained connection and the switch is not fully loaded, the allocated rates increase leading to high utilization.

The drawbacks of the scheme are as follows:

- The scheme measures neither the aggregate load (demand), the aggregate ABR capacity, nor the number of active sources at any point of time. This leads to inaccurate control when the input load does not equal the sum of the declared rates.

- The scheme depends heavily on the queue thresholding, and parameterized control to achieve efficiency. In other words, it does not explicitly try to match ABR demand with the ABR capacity, but indirectly controls it looking at the queue length. As we shall describe later in this thesis, the queue length alone is not a good metric for detecting congestion, and this approach may lead to oscillations especially when the ABR demand and capacity are both highly variable.

- The scheme uses CI bit setting in cases where ER setting becomes unreliable. This approach may result in unfairness especially when the load is variable.

- Another effect of parametric control is longer transient convergence times.

- The queue thresholding procedure requires a number of parameters to be set. These parameters are sensitive to the round-trip time and feedback delay. In

other words, a different set of parameters are required if round trip times change by an order of magnitude, with the link capacities being constant.

- The performance of the scheme in the presence of variable ABR demand and capacity is unclear. Also, the side effects (if any) of the resetting the MAX variable will become more clearer under such conditions.

- Arithmetic mean (or exponential averaging) is not the correct method for averaging ratios where the denominator is not constant [49]. Further, the running average assumes that the successive values averaged are close to each other. The technique cannot effectively average (or track) sequence of values which are uncorrelated.

4.11 FMMRA Scheme

The "Fast Max-Min Rate Allocation (FMMRA)" scheme [6] was developed by researchers Arulambalam, Chen, Ansari at NJIT and Bell Labs.

4.11.1 Key Techniques

The algorithm combines ideas from the ERICA scheme (described in this dissertation) and the UCSC scheme described in section 4.9. It is based on the measurement of available capacity and the exact calculation of fair rates, while not being sensitive to inaccuracies in CCR values.

It uses the concept of an advertised rate, γ, a rate which is given to unconstrained connections. The advertised rate is updated upon receipt of a BRM cell of a session, using its previous value, the change in the bottleneck bandwidth of the session and the change in the bottleneck status of the session. A connection which cannot use

the advertised rate is marked as a bottlenecked connection and its bandwidth usage is recorded. The ER field in the RM cell is read and marked in both directions to speed up the rate allocation process.

The scheme uses the load factor (similar to ERICA) and ER to compute an exponential running average of the maximum value of ER, ER_max:

$ER_{max} = (1 - \alpha)ER_{max} + \alpha max(ER, \frac{ER_{max}}{LoadFactor})$

This computation is done in the backward direction and is expected to reflect the advertised rate after considering the load. Based on the level of congestion, which is determined as a function of the queue length and the load factor, the ER field in the RM cell (both forward and backward) is updated according to:

$ER = min(ER, max(\gamma, (1 - \beta)ER_{max}))$

where β is a single bit value indicating that the connection is bottlenecked elsewhere.

The work also mentions approaches to update the ER field in order to control the queue growth. Specifically, if the queue length reaches a low threshold QT, and $LoadFactor > 1$, only the advertised rate is used in marking the ER field, i.e.,

$ER = min(ER, \gamma)$

The algorithm also has a mode for "severe congestion" $(Q > DQT)$ where ER_{max} is set to the advertised rate. This implies that even if some connections are idle, the non-idle connections are not given any extra bandwidth, allowing queues to drain.

4.11.2 Discussion

The contributions of the scheme are:

- A combination of several ideas in a scheme which achieves the essential goals of fast convergence to fair shares and control of queues.

- An O(1) approximation of the MIT scheme idea, combined with the tracking of load through the exponential averaging of the ER_{max} variable.

Some of the drawbacks of the scheme are:

- The calculation of feedback at the receipt of both the forward and backward RM cells increases the computation burden on the switch.

- The setting of ER in both directions may inhibit rate increase for one round trip time (when the backward direction feedback using the latest information cannot increase the rate because the forward direction had commanded a rate decrease).

- The use of exponential averaging of rates is not entirely correct because a) the rates are ratios and averaging of ratios should be done carefully [49], b) the successive values of rates used in the averaging may not be correlated. In general, Exponential averaging does not produce good results if the values averaged do not exhibit correlation.

4.12 HKUST Scheme

4.12.1 Key Techniques

This scheme was developed by researchers Tsang and Wong at the Hong Kong University of Science and Technology (HKUST). The scheme is a modification of the MIT scheme, which retains the O(N) computational complexity and marks the

ER in both the forward and backward directions. It assumes that the destination resets the ER field to the peak cell rate (PCR), which is not mandated by the traffic management standard. Since it derives from the properties of the MIT scheme, it is fair. The setting of feedback in both the forward and backward directions improves the response of the scheme compared to the MIT scheme. Another interesting aspect is that due to the bidirectional ER setting, and the resetting at the destination, the minimum of ER fields in the forward and backward directions gives the current bottleneck rate for that VC.

4.12.2 Discussion

Though given the above interesting aspects, the scheme has several drawbacks:

- It retains the O(N) complexity of the MIT scheme. Further, doing the ER calculation at the receipt of both the forward and backward RM cells increases the computation burden on the switch.
- The scheme does no load measurement, and as a result may not work if the sources are bottlenecked at rates below their allocations.
- The scheme does not measure the number of active VCs, and uses the (static) total number of VCs for the computation.
- The scheme is incompatible with the ATM Forum's Traffic Management 4.0 specification since it requires the ER to be reset by the destination.
- It is not clear how the scheme accounts for variable capacity, especially the handling of queues which build up during transient phases.

4.13 SP-EPRCA scheme

The SP-EPRCA scheme [15] was developed by Cavendish, Mascolo and Gerla at the University of California at Los Angeles.

4.13.1 Key Techniques

The key idea in SP-EPRCA is the use of a *proportional controller* with a *Smith Predictor (SP)* to compensate for the delay in the ABR feedback loop. Effectively, the dynamic control system with a delay in the feedback loop is converted into a simple first order dynamic system with a delay in cascade. Since, theoretically the delay is brought out of the feedback loop, it does not affect stability and the system should not have oscillations in the steady state.

The scheme aims to keep the queue occupancy under some desired value while achieving a fair distribution of rates. In the steady state, the scheme aims for the following relation between the rate stationary rate u_s, and the stationary queue length x_s of a VC:

$$u_s = \frac{X^o - x_s}{1/K + RTD}$$

K is the gain factor, a parameter of the Smith Predictor, X^o is the target queue length, and RTD is the round trip delay.

The scheme functions as follows. The switches send the available buffer space for cell storage for that particular connection back to the source (in one version of the scheme, the target queue length, X^o, can be fed back instead of using individual buffer allocations). Each source implements a Smith Predictor which requires the knowledge of the round trip delay and an estimate of the varying delay in the network.

The gain factor (K), a parameter of the smith predictor determines the rate of convergence to a steady state. There is also a tradeoff between the buffer space needed, maximum achievable throughput, and the maximum RTD estimation error supported. The queue implementation (FIFO or per-VC queuing) also has a significant impact on convergence. In the default case, the scheme requires a separate Smith Predictor for each VC. The conversion to the single predictor, and the implementation of the FIFO service at the switches requires additional complexity at the switches.

The challenge faced by the scheme designers was to estimate the network delays accurately. Errors in delay would cause the system to be of a higher order. Due to these constraints, the default implementation of the scheme requires per-VC queuing at the switches, and the rate computation to be done at the source end system. Another reason for this was that the round trip times of VCs (required for the smith predictors) can be estimated better at the sources rather than at all switches. Since the ATM Forum standard [32] does not specify rate computation at the source end system or provide hooks for measuring the round trip time at the source end system, the scheme is incompatible with the standards. Note also that the ATM Forum standard expects the switch to compute rates and feedback the rates and not the queue length.

One contribution of the scheme is in its mechanisms for estimating the round trip delays. The scheme uses two mechanisms for dealing with delays, acting in different time scales: **a)** a long time scale delay, keeping track of the variation of the round trip delay due to queuing at intermediate switches and **b)** a short time scale delay, which is called "virtual feedback." The latter mechanisms measures the variability of the

RTD and shuts off the source (for stability) until the RTD come back to reasonable levels.

4.13.2 Discussion

The contributions of the scheme are:

- Use of a control-theoretic approach to the ATM congestion control problem.
- Use of a Smith Predictor to remove the effect of delay from the control loop leading to a simple controller design.
- Techniques for estimating round trip delays and maintaining scheme stability.
- Proof of steady state and stability analysis of the controlled system
- Queues can be controlled to provide zero-loss.

The drawbacks of the scheme are as follows:

- The scheme is incompatible with the current ATM Forum standards, and cannot inter-operate with other schemes implemented in different switches.
- The default version requires the implementation of per-VC queuing at the switches and a separate smith predictor at every source - involving high implementation complexity.
- The transient performance of the scheme is dependent on the accuracy of RTD estimation and the gain factor, K. The latter parameter needs to be reduced to compensate for oscillatory behavior, which in turn affects the convergence time.
- The performance of the scheme in the presence of variable ABR demand and capacity is unclear.

4.14 Summary of Switch Congestion Control Schemes

We have observed in our preceding survey that different schemes have addressed different subsets of switch scheme goals listed in chapter 3. In this section, we summarize these goals and approaches and identify areas not addressed by these proposals.

If we sort the schemes by time, we find that early schemes addressed the basic problem of achieving max-min fairness with minimal complexity. We can see a transition from using purely bit-based ideas for ER-feedback to using purely ER-based ideas for the same purpose.

Early schemes used a number of concepts which have been based on the legacy of bit-based feedback design, which may not be best when explicit-rate feedback capability is available. For example, control of queuing delay is done typically through an threshold-based or hysteresis-based approach which is a legacy from bit-based feedback design. This approach does not work when there is high variance in queue fluctuations due to traffic variation. As discussed in a later chapter, using queue thresholds alone to detect congestion is a flawed technique especially when rate-based control is used.

Later schemes addressed the speed of convergence and the implementation complexity of the scheme, and faced a tradeoff between the two. The issue of measurement raised in this dissertation has been recognized in several contemporary schemes. Some of the schemes described in this section have been developed at the same time, or after the development of the OSU, ERICA and ERICA+ schemes. As a result, they share several features with the schemes we have described in this dissertation.

4.14.1 Common Drawbacks

Though the evolution of switch schemes has yielded increasingly efficient and fair algorithms, with reduction in implementation complexity, and a broader scope, many of these proposals suffer from a common set of drawbacks as listed in this section.

In general schemes, with a few notable exceptions, ***have not been comprehensive in design***, i.e., they either do not address all the goals of a switch scheme and/or make too many assumptions about measurement related aspects of the scheme.

For example, many schemes ***do not address the issue of how to measure the ABR demand***. The lack of information about the demand may lead to under-allocation of rates. Several schemes (like those which use the concept of Mean ACR (MACR)) approximate the average demand per connection. However, if the total demand (aggregate input rate) is not measured, the scheme could be consistently making estimation errors.

Other schemes ***do not monitor the activity of sources***, and may overlook a source becoming temporarily idle. If the idle source is considered while determining allocations for all other sources, the allocations for the other sources may be reduced.

In brief, measurement is necessary to track the current network state used by the scheme. Ideally, a scheme should measure every component of the network state it uses for its calculations.

Another issue is ***how to measure the scheme metrics when there is high variation in the traffic demand and available capacity***. Several metrics need to be observed over intervals of time and averaged over many such intervals to smooth out the effects of such variation. The length of the interval is a key factor in a tradeoff between quick response and accurate response. Implementation issues include

specifying where and when exactly the measurements should be made, and feedback should be given. Several schemes (with the notable exceptions of the Phantom and, to some extent, the DMRCA scheme) do not attempt to address these concerns.

Several switch algorithms ***require the source to restrict its rise by limiting the Rate Increase Factor (RIF) parameter*** to avoid oscillations. But, this affects the transient performance of the scheme. Other switch algorithms ***require setting multiple parameters, and may sometimes be sensitive to parameters***.

Another issue with respect to parameters is ***control-feedback correlation***. Switch algorithms use several control parameters (available capacity, source's rate, the aggregate input rate, the number of active sources etc) to calculate the feedback quantities. Typically, control parameters values are measured asynchronously with respect to when feedback is given. One important responsibility of the switch algorithm is to ensure that the feedback is correlated with the control. Lack of such correlation will lead to perpetual oscillations at best, and queue divergence and collapse at worst. Most schemes do not specify in detail how the correlation is maintained (especially when there is high variation in the network traffic).

Many schemes change from one policy to another for small changes in system state. This introduces ***discontinuities*** in the feedback rate calculation function. If the system state is oscillating around the places where discontinuity is introduced, the scheme would exhibit undesirable oscillations. However, the presence of discontinuities in the feedback function alone does not mean that the scheme is bad. If the number of discontinuities are many (like the use of several queuing thresholds) the scope for undesirable oscillations increases.

The ATM Traffic Management standard also mention ***the Use-it or Lose-it problem*** where sources may retain allocations and use it later when the allocations are invalid. The standard provides minimal support from the source end systems. The switch needs to be able to tolerate transient queuing, and recover quickly from such uncontrollable circumstances.

In this dissertation, we address all these issues and present the design, performance analysis of the switch scheme, and several other aspects of ABR traffic management.

CHAPTER 5

THE OHIO STATE UNIVERSITY (OSU) SCHEME

The OSU scheme was one of the first attempts to show the power of the explicit rate feedback method having O(1) execution complexity. It was developed as an alternative to the MIT scheme which had O(N) complexity. The EPRCA and APRC were some of the approaches proposed in parallel to the OSU scheme. It should be noted that the OSU scheme was developed at a time when the rate-based framework was being designed in the ATM Forum Traffic Management Group. As we describe the scheme, we shall also discuss the contributions of the scheme towards forming the standards.

5.1 The Scheme

The OSU scheme requires sources to monitor their load and send control cells *periodically* at intervals of T microseconds. These control cells contain source rate information. The switches monitor their own load and use it with the information provided by the control cells to compute a factor by which the source should go up or down. The destination simply returns the control cells to the source, which then adjusts its rate as instructed by the network. This section described the various components of the scheme.

5.1.1 Control-Cell Format

The control cell contains the following fields:

1. Transmitted Cell Rate (TCR): The TCR is the inverse of the minimum inter-cell transmission time and indicates instantaneous peak load input by the source.

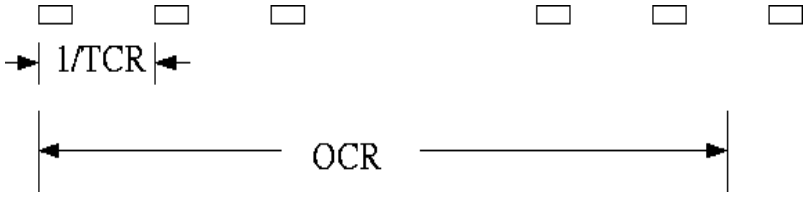


Figure 5.1: Transmitted cell rate (instantaneous) and Offered Average Cell Rate (average).

2. The Offered Average Cell Rate (OCR): For bursty sources which may not send a cell at every transmission opportunity, TCR is not a good indication of overall load. Therefore, the average *measured* load over T interval is indicated in the OCR field of the control cell. The inter-cell time is computed based on the transmitted cell rate. However, the source may be idle in between the bursts and so the average cell rate is different from the transmitted cell rate. This average is called the offered average cell rate and is also included in the cell. This distinction between TCR and OCR is shown in Figure 5.1. Notice that TCR is a control variable (like the knob on a faucet) while the OCR is a measured quantity (like a meter on a pipe). This analogy is shown in Figure 5.2.

3. Load Adjustment Factor (LAF): This field carries the feedback from the network. At the source, the LAF is initialized to zero. Switches on the path can

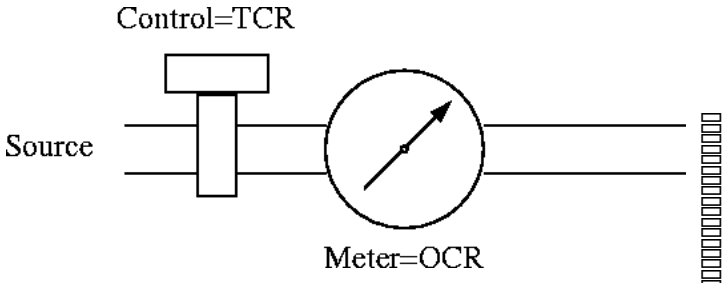


Figure 5.2: Transmitted cell rate (controlled) and Offered Average Cell Rate (measured).

only increase LAF. Increasing the LAF corresponds to decreasing the allowed source rate. Hence, successive switches only reduce the rate allowed to the source. Thus, the source receives the rate allowed by the bottleneck along the path

4. Averaging interval (AI): The OSU scheme primarily uses measured quantities instead of parameters for control. These quantities are measured at the source (eg., OCR) and the switch (eg., current load level z discussed in section 5.1.3). The measurements are done over intervals (called "averaging intervals") to smoothen out the variance in these quantities. To ensure correlation of the measured quantities at the switch and at the source, we require the source averaging intervals to be the maximum of the averaging interval of the switches along the path. This maximum value is returned in the AI field. The AI field is initialized to zero at the source.

5. The direction of feedback (backward/forward)

6. Timestamp containing the time at which the control cell was generated at the source

The last two fields are used in the backward congestion notification option described in Section 5.2.8 and need not be present if that option is not used.

5.1.2 The Source Algorithm

The source algorithm consists of three components:

1. How often to send control cells
2. How to measure the offered average cell rate
3. How to respond to the feedback received from the network

These three questions are answered in the next three subsections.

Control-Cell Sending Algorithm

The sources send a control cell into the network every T microseconds. The source initializes all the fields. The network reads only the OCR, LAF and AI fields and modifies only the LAF and AI fields. The TCR field is used by the source to calculate the new TCR as discussed in the next section.

LAF and AI are both initialized to zero as discussed earlier. The initialization of the OCR and TCR fields are discussed in the next section.

Measuring Offered Average Load

Unlike any other scheme proposed so far, each source also measures its own load. The measurement is done over the same averaging interval that is used for sending the control cells. The transmission cell rate (TCR), as defined, is the inverse of minimum inter-cell transmission time at the source. However, when the source is not always

active, the average rate of the source is different from the transmitted cell rate. This average is called the offered average cell rate and is also included in the cell.

Normally the OCR should be less than the TCR, except when the TCR has just been reduced. In such cases, the switch will actually see a load corresponding to the previous TCR and so the feedback will correspond to the previous TCR. The OCR, in such cases, is closer to the previous TCR. Putting the maximum of current TCR and OCR in the TCR field helps overcome unnecessary oscillations caused in such instances. In other words,

$$\text{TCR in Cell} \leftarrow \max\{\text{TCR, OCR}\}$$

During an idle interval, no control cells are sent. If the source measures the OCR to be zero, then one control cell is sent, subsequent control cells are sent only after the rate becomes non-zero.

Responding to Network Feedback

The control cells returned from the network contain a "load adjustment factor" along with the TCR. The current TCR may be different from that in the cell. The source computes a new TCR by dividing the TCR in the cell by the load adjustment factor in the cell:

$$\text{New TCR} \leftarrow \frac{\text{TCR in the Cell}}{\text{Load Adjustment Factor in the Cell}}$$

If the load adjustment factor is more than one, the network is asking the source to decrease. If the new TCR is less than the current TCR, the source sets its TCR to the new TCR value. However, if the new TCR is more than current TCR, the source is already operating below the network's requested rate and there is no need make any adjustments.

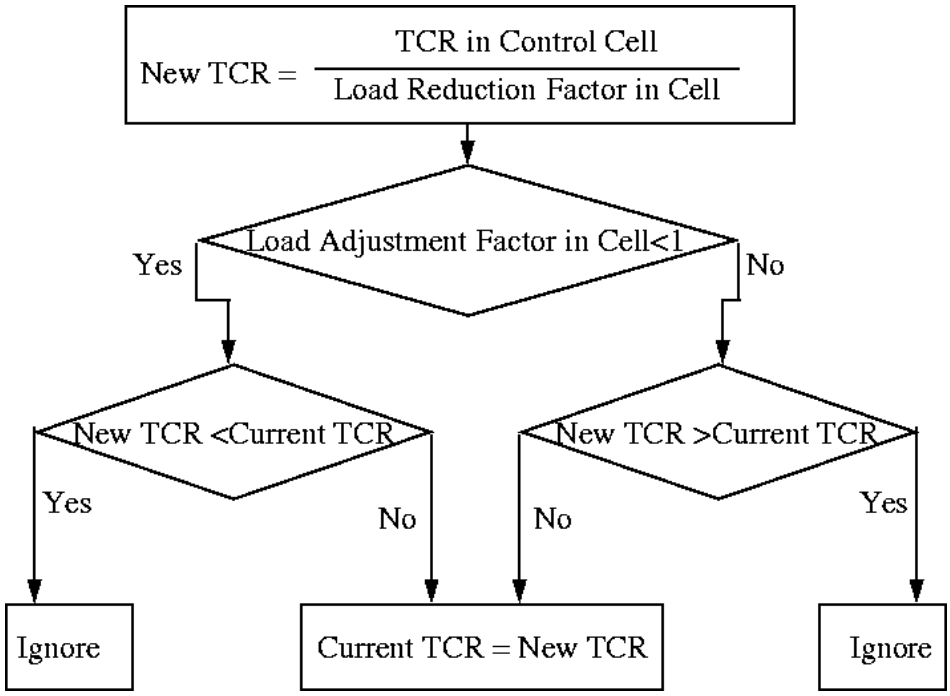


Figure 5.3: Flow chart for updating TCR

Similarly, if the load adjustment factor is less than one, the network is permitting the source to increase. If the current TCR is below the new TCR, the source increases its rate to the new value. However, if the current TCR is above the new TCR, the new value is ignored and no adjustment is done. Figure 5.3 presents a flow chart explaining the rate adjustment.

5.1.3 The Switch Algorithm

The switch algorithm consists of two parts: measuring the current load level periodically and calculating the feedback whenever a control cell is received. The feedback calculation consists of an algorithm to achieve efficiency and an algorithm

to achieve fairness. The measured value of the current load level is used to decide whether the efficiency or the fairness algorithm is used to calculate feedback.

Measuring The Current Load *z*

The switch measures its current load level, z , as the ratio of its "input rate" to its "target output rate". The input rate is measured by counting the number of cells *received* by the switch during a fixed averaging interval. The target output rate is set to a fraction (close to 100 %) of the link rate. This fraction, called Target Utilization (U), allows high utilization and low queues in steady state. The current load level z is used to detect congestion at the switch and determine an overload or underload condition.

$$\text{Target Output Cell Rate} = \frac{\text{Target Utilization (U)} \times \text{Link bandwidth in Mbps}}{\text{Cell size in bits}}$$

$$z = \frac{\text{Number of cells received during the averaging interval}}{\text{Target Output Cell Rate} \times \text{Averaging Interval}}$$

The switches on the path have averaging intervals to measure their current load levels (z). These averaging intervals are set locally by network managers. A single value of z is assumed to correspond to *one* OCR value of every source. If two control cells of a source with different OCRs are seen in a single interval (for one value of z), the above assumption is violated and conflicting feedbacks may be given to the source. So, when feedback is given to the sources the AI field is set to the maximum of the AI field in the cell and the switch averaging interval:

$$\text{AI in cell} \leftarrow \text{Max(AI in cell, switch averaging interval)}$$

Achieving Efficiency

Efficiency is achieved as follows:

$$\text{LAF in cell} \leftarrow \text{Max}(\text{LAF in cell}, z)$$

The idea is that if all sources divide their rates by LAF, the switch will have $z = 1$ in the next cycle. In the presence of other bottlenecks, this algorithm converges to $z = 1$. In fact it reaches a band $1 \pm \Delta$ quickly. This band is identified as an efficient operating region. However, it does not ensure fair allocation of available bandwidth among contending sources. When $z = 1$, sources may have an unfair distribution of rates.

Achieving Fairness

Our first goal is to achieve efficient operation. Once the network is operating close to the target utilization, we take steps to achieve fairness. The network manager declares a target utilization band (TUB), say, 90±9% or 81% to 99%. When the link utilization is in the TUB, the link is said to be operating efficiently. The TUB is henceforth expressed in the U(1±Δ) format, where U is the target utilization and Δ is the half-width of the TUB. For example, 90±9% is expressed as $90(1 \pm 0.1)\%$. Equivalently, the TUB is identified when the current load level z lies in the interval $1 \pm \Delta$.

We also need to count the number of active sources for our algorithm. The number of active sources can be counted in the same averaging interval as that of load measurement. One simple method is to mark a bit in the VC table whenever a cell from a VC is seen. The bits are counted at the end of each averaging interval and are cleared at the beginning of each interval. Alternatively a count variable could be

incremented when the bit is changed from zero to one. This count variable and the bits are cleared at the end of the interval.

Given the number of active sources, a fair share value is computed as follows:

$$\text{FairShare} = \frac{\text{Target Cell Rate}}{\text{Number of Active Sources}}$$

Underloading sources are sources that are using bandwidth less than the FairShare and overloading sources are those that are using more than the FairShare. To achieve fairness, we treat underloading and overloading sources differently. If the current load level is z, the underloading sources are treated as if the load level is $z/(1+\Delta)$ and the overloading sources are treated as if the load level is $z/(1-\Delta)$.

$$\text{If (OCR in cell} < \text{FairShare)} \;\; \text{LAF in cell} \leftarrow \text{Max(LAF in cell,} \;\; \frac{z}{(1+\Delta)})\}$$

$$\text{else LAF in cell} \leftarrow \text{Max(LAF in cell,} \;\; \frac{z}{(1-\Delta)})\}$$

We prove later in this chapter that this algorithm guarantees that the system, once in the TUB, remains in the TUB, and consistently moves towards fair operation. We note that all the switch steps are O(1) w.r.t. the number of VCs.

If Δ is small, as is usually the case, division by $1+\Delta$ is approximately equivalent to a multiplication by $1-\Delta$ and vice versa.

What Load Level Value to Use?

The OCR in the control cell is correlated to z when the control cell *enters* the switch queue. This is because the queue state at arrival more accurately reflects the effect of the TCR indicated in the control cell. The value of z may change before the control cell leaves the switch queue. The OCR in the cell at the time of leaving the queue is not necessarily co-related with z. As shown in Figure 5.4, the queue state at

the time of departure (instant marked "2" in the figure) depends upon the load that the source put after the control cell had left the source. This subsequent load may be very different from that indicated in the cell.

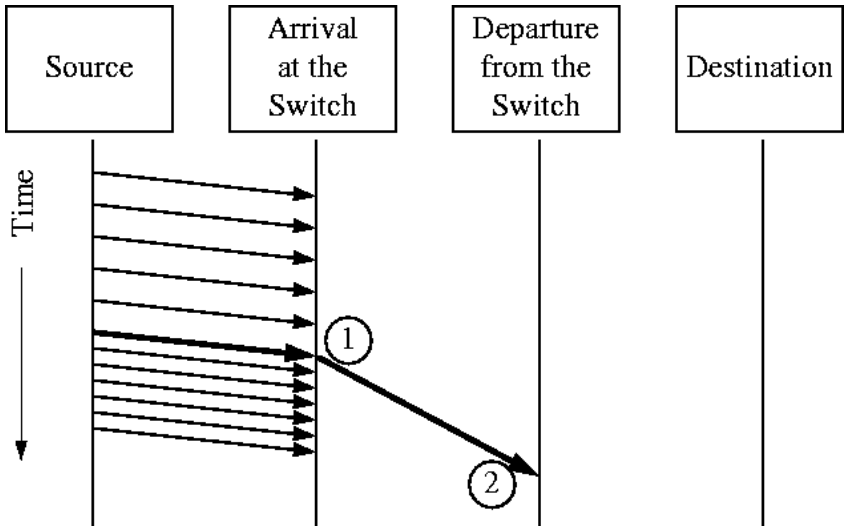


Figure 5.4: Correlation of Instantaneous Queue States to TCR

5.1.4 The Destination Algorithm

The destination simply returns all control cells back to the source.

5.1.5 Initialization Issues

When a source first starts, it may not have any idea of the averaging interval or what rate to use initially. There are two answers. First, since ATM networks are connection-oriented, the above information can be obtained during connection setup. For example, the averaging interval and the initial rate may be specified in the connection accept message. Second, it is possible to send a control cell (with TCR=OCR=0) and wait for it to return. This will give the averaging interval. Then

the source can pick any initial rate and start transmitting. It can use the averaging interval returned in the feedback to measure OCR, and at the end of the averaging interval send a control cell containing this OCR. When the control cell returns, it will have the information to change to the correct load level.

Since the averaging intervals depend upon the path, averaging interval may be known to the source host from other VCs going to the same destination host. Also, a network manager may hardcode the same averaging interval in all switches and hosts. We do not recommend this procedure since not all switches that a host may eventually use may be in the control of the network manager.

The initial transmission cell rate affects the network operation for only the first few (one or two) round trips. Therefore, it can be any value below (and including) the target cell rate of the link at the source. However, network managers may set any other initial rate to avoid startup impulses.

5.2 Key Features and Contributions of the OSU scheme

The OSU scheme was presented to the ATM Forum traffic management working group in its September and October 1994 meetings. It highlighted several new ideas that have now become common features of most such schemes developed since then. This includes applying the concept of congestion avoidance to rate-based algorithms and the use of input rate instead of queue length for congestion detection. The number of parameters is small and their effects are well understood.

5.2.1 Congestion Avoidance

The OSU scheme is a congestion *avoidance* scheme. As defined in [42], a congestion avoidance scheme is one that keeps the network at high throughput and low delay in

the steady state. The system operates at the *knee* of the throughput delay-curve as shown in Figure 3.1.

The OSU scheme keeps the steady state bottleneck link utilization in the target utilization band (TUB). The utilization is high and the oscillations are bounded by the TUB. Hence, in spite of oscillations in the TUB, the load on the switch is always less than one. So the switch queues are close to zero resulting in minimum delay to sources.

The target utilization and target utilzation band per-link parameters are set by the network manager based on the cost of the bandwidth, and the anticipated degree of variance in the network demand and capacity. The target utilization affects the rate at which the queues are drained during overload. A higher target utilization reduces unused capacity but increase the time to reach the efficient region after a disturbance. A lower target utilization may be necessary to cope with the effects of variance in capacity and demand due to the introduction of errors introduced in measurement as a result of variance. A wide TUB results in a faster progress towards fairness. In most cases, a TUB of $90\%(1 \pm 0.1)$ is a good choice. This gives a utilization in the range of 81% to 99%.

5.2.2 Parameters

The OSU scheme requires just three parameters: the switch averaging interval (AI) , the target link utilization (U) , and the half-width of the target utilization band (Δ).

The target utilization (U) and the TUB present a few tradeoffs. During overload (transients), U affects queue drain rate. Lower U increases drain rate during transients, but reduces utilization in steady state. Further, higher U also constrains the size of the TUB.

A narrow TUB slows down the convergence to fairness (since the formula depends on Δ) but has smaller oscillations in steady state. A wide TUB results in faster progress towards fairness, but has more oscillations in steady state. We find that a TUB of $90\%(1 \pm 0.1)$ used in our simulations is a good choice.

The switch averaging interval affects the stability of z. Shorter intervals cause more variation in the z and hence more oscillations. Larger intervals cause slow feedback and hence slow progress towards steady state.

The OSU scheme parameters can be set relatively independent of the target workload and network extent. Variance in measurement is the key error factor in the OSU scheme, and a larger interval is desirable to smooth the effect of such variance. Some schemes, on the other hand, are very sensitive to the workload and network diameter in their choice of parameter values. An easy way to identify such schemes is that they recommend different parameter values for different network configurations. For example, a switch parameter may be different for WAN configurations than in a LAN configuration. A switch generally has some VCs travelling short distances while others travelling long distances. While it is ok to classify a VC as a local or wide area VC, it is often not correct to classify a switch as a LAN switch or a WAN switch. In a nationwide internet consisting of local networks, all switches could be classified as WAN switches. Note that the problem becomes more difficult when the scheme uses many parameters, and/or the parameters are not independent of each other.

5.2.3 Use Measured Rather Than Declared Loads

Many schemes prior to OSU scheme, including the MIT scheme, used source declared rates for computing their allocation without taking into account the actual load at the switch. In the OSU scheme, we measure the current total load. All unused capacity is allocated to contending sources. We use the source's declared rate to compute a particular sources' allocation but use the switch's measured load to decide whether to increase or decrease. Thus, if the sources lie or if the source's information is out-of-date, our approach may not achieve fairness but it still achieves efficiency.

For example, suppose a personal computer connected to a 155 Mbps link is not be able to transmit more than 10 Mpbs because of its hardware/software limitation. The source declares a desired rate of 155 Mbps, but is granted 77.5 Mbps since there is another VC sharing the link going out from the switch. Now if the computer is unable to use any more than 10 Mbps, the remaining 67.5 Mbps is reserved for it and cannot be used by the second VC and the link bandwidth is wasted.

The technique of measuring the total load has become minimum required part of most switch algorithms. Of course, some switches may measure each individual source's cell rate rather than relying on the information in the RM cell

5.2.4 Congestion Detection: Input Rate vs Queue Length

Most congestion control schemes for packet networks in the past were window based. Most of these schemes use queue length as the indicator of congestion. Whenever the queue length (or its average) is more than a threshold, the link is considered congested. This is how initial rate-based scheme proposals were also being designed.

We argued that the queue length is not a good indicator of load when the control is rate-based.

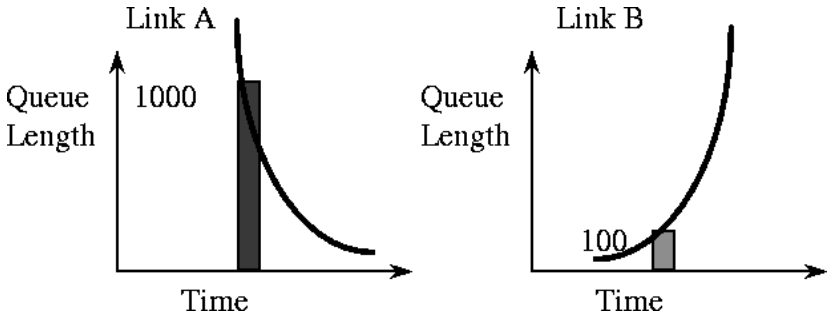


Figure 5.5: Congestion Detection Metric: Queue Length or Input Rate ?

As an example, consider two rate controlled queues as shown in figure 5.5. Suppose the first queue is only 100 cells long while the other is 1000 cells long. Without further information it is not possible to say which queue is overloaded. For example, if the first queue is growing at the rate of 1000 cells per second, it is overloaded while the second queue may be decreasing at a rate of 1000 cells per second and may actually be underloaded. Further, if the first queue can be processed at 622 Mbps, the queueing delay is much smaller than that of a 100 cell queue processed at 1.54 Mbps. This factor becomes important because the capacity available to ABR can be quite variable.

Another important reason for the choice of the input rate metric has to do with rate and window controls. For a detailed discussion of rate versus window, see Jain (1990) [48]. In particular, a window controls the queue length, while the rate controls the queue growth rate. Given a particular window size, the maximum queue length can be guaranteed to be below the window. Given an input rate to a queue, the queue growth rate can be guaranteed below the input rate but there is nothing that can be

said about the maximum queue length. Queue length gives no information about the difference between current input rate and the ideal rate.

With rate-based control, the input rate is a better indicator of congestion. If the input rate is lower than available capacity, the link is not congested even if the queue lengths are high because the queue will be decreasing. Similarly, if the input rate is higher than the available capacity, the system should start taking the steps to reduce congestion since the queue length will be increasing.

Monitoring input rates not only gives a good indication of load level, it also gives a precise indication of overload or underload. For example, if the input rate to a queue is 20 cells per second when the queue server can handle only 10 cells per second, we know that the queue overload factor is 2 and that the input rate should be decreased by a factor of 2. No such determination can be made based on instantaneous queue length. The input rate can hence be used as a metric to compute the new rate allocations. The use of input rates as a metric avoids the use of unnecessary parameters.

The OSU scheme uses the input rate to compute the overload level and adjust the source rates accordingly. Each switch counts the number of cells that it received on a link in a given period, computes the cell arrival rate and hence the overload factor using the known capacity (in cells per second) of the link. It tries to adjust the source rate by a factor equal to the overload level and thus attempts to bring it down to the correct level as soon as possible.

In the later ERICA+ work described in this dissertation, we use the ***queueing delay*** as a *secondary metric* for congestion detection with input rate being the primary metric.

5.2.5 Bipolar Feedback

A network can provide two kinds of feedback to the sources. Positive feedback tells the sources to increase their load. Negative feedback tells the sources to decrease their load. These are called two polarities of the feedback. Some schemes are bipolar in the sense that they use both positive and negative feedback. The OSU scheme uses both polarities. The DECbit scheme [46] is another example of a bipolar scheme.

Some schemes use only one polarity of feedback, say positive. Whenever, the sources receive the feedback, they increase the rate and when they don't receive any feedback, the network is assumed to be overloaded and the sources automatically decrease the rate without any explicit instruction from the network. Such schemes send feedback only when the network is underloaded and avoid sending feedback during overload. The PRCA scheme [26] is an example of a unipolar scheme with positive polarity only.

Unipolar schemes with negative polarity are similarly possible. Early versions of PRCA used negative polarity in the sense that the sources increased the rate continuously unless instructed by to network to decrease. The slow start scheme used in TCP/IP is also an example of unipolar scheme with negative polarity although in this case the feedback (packet loss) is an implicit feedback (no bits or control packets are sent to the source).

The MIT scheme is unipolar with only negative feedback to the source. The switches can only reduce the rate and not increase it. For increase, the source has to send another control cell with a higher desired rate. Thus, increases are delayed resulting in reduced efficiency.

The key problem with some unipolar schemes is that the load is changed continuously – often on every cell. This may not be desirable for some workloads, such as compressed video traffic. Every adjustment in rate requires the application to adjust its parameters. Bipolar schemes can avoid the unnecessary adjustments by providing explicit instructions to the sources only when a load change is required.

One reason for prefering unipolar feedback in some cases is that the number of feedback messages is reduced. However, this is not always true. For example, the MIT and OSU schemes have the same data cell to control cells ratio. In the MIT scheme, a second control cell has to be sent to determine the increase amount during underload. This is avoided in the OSU scheme by using a bipolar feedback.

Since current ATM specifications allow the switches to increase or decrease the rate of a source, all ATM switch implementations are expected to be bipolar.

5.2.6 Count the Number of Active Sources

The OSU scheme introduced the concept of averaging interval and active sources. Most of the virtual circuits (VCs) in an ATM network are generally idle. Its the number of active VCs rather than the total number of VCs that is meaningful. We compute use the number of active VCs to compute fairshare. As discussed in section 5.10, if the measured value is wrong (which is possible if the averaging interval is short), fairness may be affected.

Other schemes like EPRCA attempt to achieve fairness without measuring the number of active sources. The technique they use is to advertise a single rate to all sources and parametrically increase or decrease the advertized rate.

5.2.7 Order 1 Operation

The MIT scheme uses an iterative procedure to calculate the feedback rate. Further it requires the switches to remember the rates for all VCs and. Therefore, its computation and storage complexity is of the order of n, O(n). This makes it somewhat undesirable for large switches that may have thousands of VCs going through it at any one time. The basic OSU scheme does not need all the rates at the same time and has a computational complexity of O(1).

5.2.8 Backward Congestion Notifications Cannot Be Used to Increase

One problem with end-to-end feedback schemes is that it may take long time for the feedback to reach the source. This is particularly true if the flow of RM cells has not been established in both directions. In such cases, switches can optionally generate their own RM cell and send it directly back to the source.

The OSU scheme research showed that indiscriminate use of BECNs can cause problems. For example, consider the case shown in Figure 5.6. The source is sending at 155 Mbps and sends a RM cell. The switch happens to be underloaded at that time and so lets the first RM cell (C1) go unchanged. By the time the second RM cell (C2) arrives, the switch is loaded by a factor of 2 and sends a BECN to the source to come down to 77.5 Mbps. A little later C1 returns asking the source to change to 155 Mbps. The RM cells are received out of order rendering the BECN ineffective. To ensure correct operation of the BECN option, we established a set of rules. These rules are described later in Section 5.3.3. The first two of the six rules described there are now part of the Traffic Management specifications.

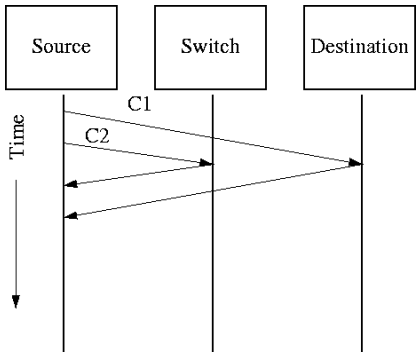


Figure 5.6: Space time diagram showing out-of-order feedback with BECN

5.3 Extensions of The OSU Scheme

5.3.1 Aggressive Fairness Option

In the basic OSU scheme, when a link is outside the TUB, all input rates are adjusted simply by the load level. For example, if the load is 200%, all sources will be asked to halve their rates regardless of their relative magnitude. This is because our goal is to get into the efficient operation region as soon as possible without worrying about fairness. The fairness is achieved after the link is in the TUB.

Alternatively, we could attempt to take steps towards fairness by taking into account the current rate of the source even outside the TUB. However, one has to be careful. For example, when a link is underloaded there is no point in preventing a source from increasing simply because it is using more than its fair share. We cannot be sure that underloading sources can use the extra bandwidth and if we don't give it to an overloading (over the fair share) source, the extra bandwidth may go unused.

The aggressive fairness option is based on a number of considerations. These considerations or heuristics improve fairness while improving efficiency. However,

these heuristics do not guarantee convergence to fair operation. We will hence use them outside the TUB, and the TUB algorithm inside the TUB.

The considerations for increase are:

1. When a link is underloaded, all its users will be asked to increase. No one will be asked to decrease.

2. The amount of increase can be different for different sources and can depend upon their relative usage of the link.

3. The maximum allowed adjustment factor should be less than or equal to the current load level. For example, if the current load level is 50%, no source can be allowed to increase by more than a factor of 2 (which is equivalent to a load adjustment factor of 0.5).

4. The load adjustment factor should be a continuous function of the input rate. Any discontinuities will cause undesirable oscillations and impulses. For example, suppose there is a discontinuity in the curve when the input rate is 50 Mbps. Sources transmitting 50-δ Mbps (for a small δ) will get very different feedback than those transmitting at 50+δ Mbps.

5. The load adjustment factor should be a monotonically non-decreasing function of the input rate. Again, this prevents undesirable oscillations. For example, suppose the function is not monotonic but has a peak at 50 Mbps. The sources transmitting at 50+δ Mbps will be asked to increase more than those at 50 Mbps.

The corresponding considerations for overload are similar to the above.

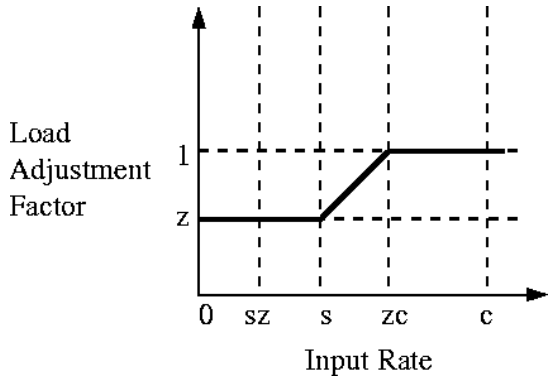
As noted, these heuristics do not guarantee convergence to fairness. To guarantee fairness in the TUB, we violate all of these heuristics except monotonicity.

A sample pair of increase and decrease functions that satisfy the above criteria are shown in Figure 5.7. The load adjustment factor is shown as a function of the input rate. To explain this graph, let us first consider the increase function shown in Figure 5.7(a). If current load level is z, and the fair share is s, all sources with input rates below zs are asked to increase by z. Those between zs and z are asked to increase by an amount between z and 1.

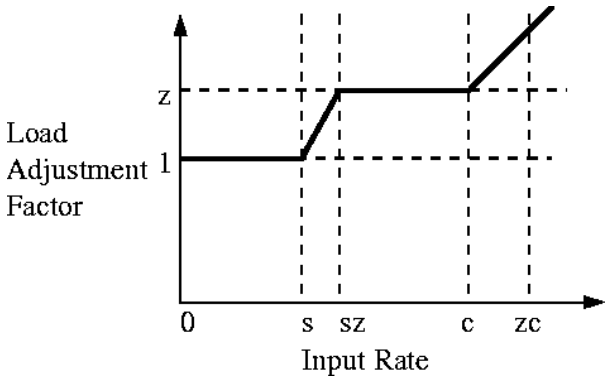
Figure 5.7(b) shows the corresponding decrease function to be used when the load level z is greater than 1. The underloading sources (input rate $x <$ fair share) are not decreased. Those between s and zs are decreased by a linearly increasing factor between 1 and z. Those with rates between zs and c are decreased by the load level z. Those above c are decreased even more. Notice that when the load level z is 1, that is, the system is operating exactly at capacity, both the increase and decrease functions are identical (a horizontal line at load reduction factor of 1). This is important and ensures that the load adjustment factor is a continuous function of z. In designing the above function we used linear functions. However, this is not necessary. Any increasing function in place of sloping linear segments can be used. The linear functions are easy to compute and provide the continuity property that we seek.

The detailed pseudo code of aggressive fairness option is given in appendix B.

Figure 5.8 shows the simulation results for the transient configuration with the aggressive fairness option.



(a) Multi-line Increase function

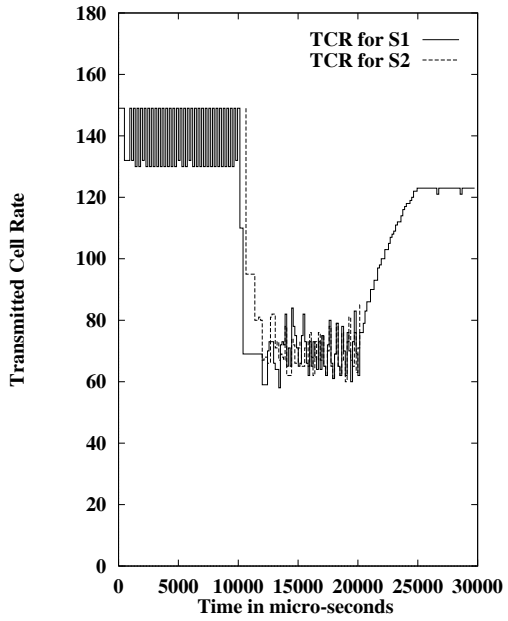


(b) Multi-line Decrease function

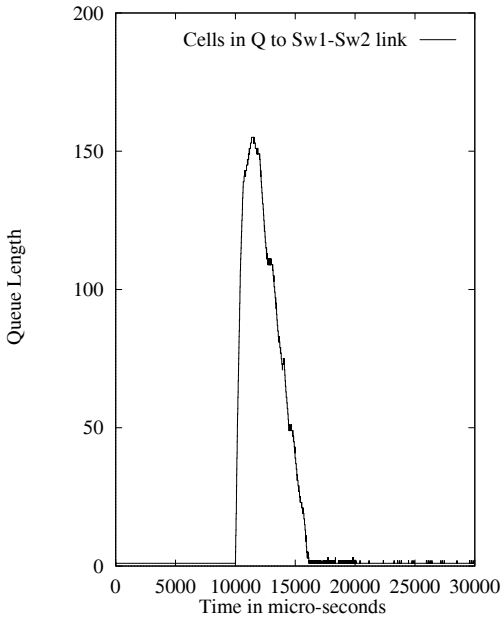
Figure 5.7: Multi-line Increase and Decrease Functions

5.3.2 Precise Fair Share Computation Option

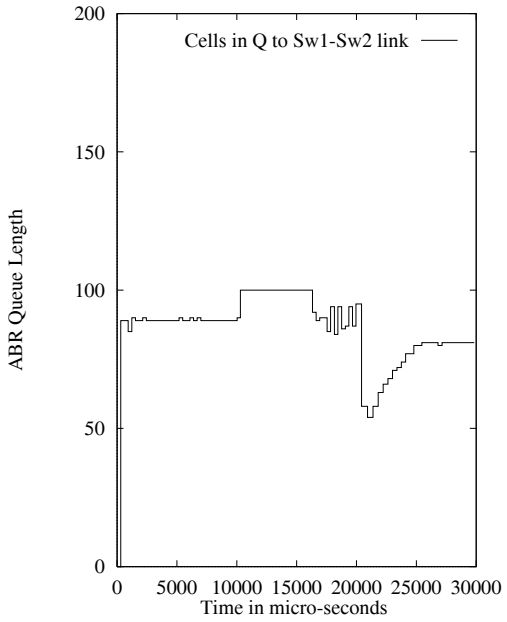
Given the actual rates of all active sources, we can exactly calculate the fair share using the MIT algorithm [18, 17] (MIT scheme uses desired rates). Thus, instead of using only the number of active VCs, we could use the OCRs of various sources to compute the fair share. This option yields a performance much better than that possible with MIT scheme because of the following features that are absent in the MIT scheme:



(a) Transmitted Cell Rates



(b) Queue Lengths



(c) Link Utilization

Figure 5.8: Simulation results for the experiment with transients and Multi-line fairness option

1. Provide a bipolar feedback. The switches can increase as well decrease the rate in the RM cell. This avoids the extra round trip required for increase in the MIT scheme.

2. Measure the offered average cell rate at the source and use it also to compute the fair share. Using measured value is better than using desired rates.

The detailed pseudo code of precise fair share computation is given in appendix B.

5.3.3 BECN Option

For long-delay paths, backward explicit congestion notifications (BECNs) may help reduce the feedback delay. Experiments with BECNs showed that, BECNs may cause problems unless handled carefully. In particular, we established the following rules for correct operation of the BECN option with OSU scheme:

1. The BECN should be sent only when a switch is overloaded **AND** the switch wants to decrease the rate below that obtained using the LAF field of the RM cell. There is no need to send BECN if the switch is underloaded.

2. The RM cell contains a bit called "BECN bit." This bit is initialized to zero at the source and is set by the congested switch in the BECN cell. The cells that complete the entire path before returning to the source are called forward explicit congestion notification (FECN) cells. They have the bit cleared.

3. All RM cells complete a round-trip. The switch which wants to send a BECN waits until it receives an RM cell, makes two copies of it and sends one copy in the forward direction. The other, called the "BECN cell," is sent back to the source.

4. The RM cell contains a timestamp field which is initialized by the source to the time when the RM cell was generated. The timestamp is ignored everywhere except at the source.

5. The source remembers the timestamp of the last BECN or FECN cell that it has acted upon in a variable called "Time already acted (Taa)." If the timestamp in an returned RM (BECN or FECN) cell is less than Taa, the cell is ignored. This rule helps avoid out-of-order RM cells.

6. If the timestamp of an RM cell received at the source is equal to or greater than Taa, the variable New TCR is computed as in section 5.1.2. In addition, if the BECN bit is set, we ignore the feedback if it directs a rate increase :

   IF BECN_bit AND (TCR < New TCR) THEN Ignore

   The rate increase has to wait until the corresponding FECN cell returns. BECN is therefore useful only for decrease on long feedback paths.

The ATM forum has adopted the first two of the above rules. The RM cells as specified in the ATM Forum Traffic Management specifications do not contain the timestamps and the last three rules are not relevant to them. These are specific to the OSU scheme. The detailed pseudo code of BECN option is given in appendix B.

One obvious disadvantage of the BECN scheme is that the number of control cells that sent back to the source are increased. Also, since BECN does not have any significant effect in the LAN environment, we recommend its use only in large WANs. This problem was recognized by the ATM Forum which limited the number of BECN cells sent by a switch to 10 cells/sec per-connection.

A complete layered view of various components of the OSU scheme is shown in Figure 5.9. The minimum that we need for correct operation is the fairness algorithm. The aggressive fairness option allows fairness to be achieved faster. The precise fair share computation option allows both fairness and efficiency to be achieved quickly but requires the switches to use all declared OCRs in computing the fair share. The BECN option helps reduce the feedback delay in large WAN cases. As shown in Figure 5.9, these options can be used individually or in a layered manner.

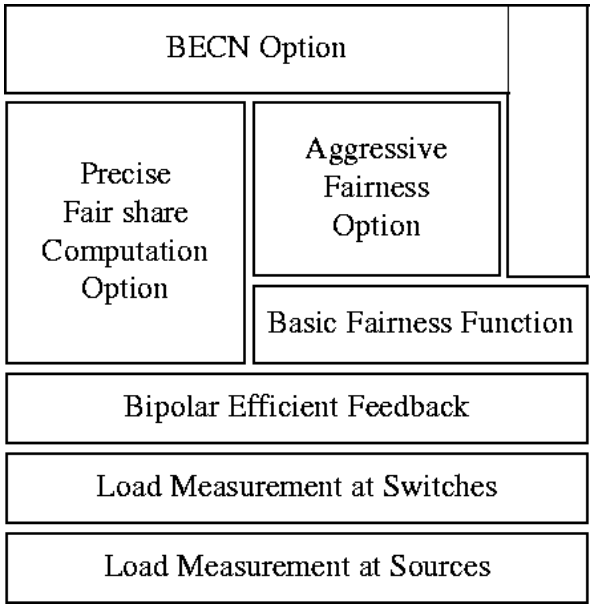


Figure 5.9: A layered view of various components and options of the OSU scheme

5.4 Other Simple Variants of the OSU Scheme

Some variations that do not materially change the performance of the OSU scheme are:

1. The source offered average cell rate is measured at the entry switch rather than at the source. This option may be preferable for policing and for operation in public network environments, where a sources' measurements cannot be trusted.

2. The offered average cell rate of a VC is measured at every switch. This is unnecessary since the average rate of a VC should not change from switch to switch. This may be used only if the VC crosses many ATM networks under different administrative domains.

3. Use multiplicative load adjustment factors instead of divisors. In OSU scheme, divisors are used for rates. However, for the inter-cell transmission time, the same factor is used as a multiplier.

4. Use dynamic averaging intervals. The averaging interval at the switch and the source are kept constant in the OSU scheme. It is possible to use regeneration intervals as the averaging interval as was done in the DECbit scheme [46]. However, our experience with DECbit scheme was that implementors didn't like the the regeneration interval and queue length averaging because of the number of instructions required in the packet forwarding path.

5. Use cell counts rather than cell rates. Since the averaging interval is constant, the cell rates are proportional to the counts.

5.5 Simulation Results

In this section, we present simulation results for several configurations. These configurations have been specially chosen to test a particular aspect of the scheme. In general, we prefer to use simple configurations that test various aspects of the

scheme. Simple configurations not only save time but also are more instructive in finding problems than complex configurations.

The configurations are presented later in this section in the order in which we use them repeatedly during design phase. For each design alternative, we always start with the simplest configuration and move to the next only if the alternative works satisfactorily for the simpler configurations.

5.5.1 Default Parameter Values

Unless specified otherwise, we assume all links are 1 km long running at 155 Mbps. The infinite source model is used for traffic initially. The burst traffic is considered in Section 5.7. The averaging interval of 300 μs and a target utilization band of 90(1$\pm$ 0.1)% is used.

5.5.2 Single Source

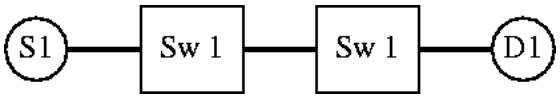
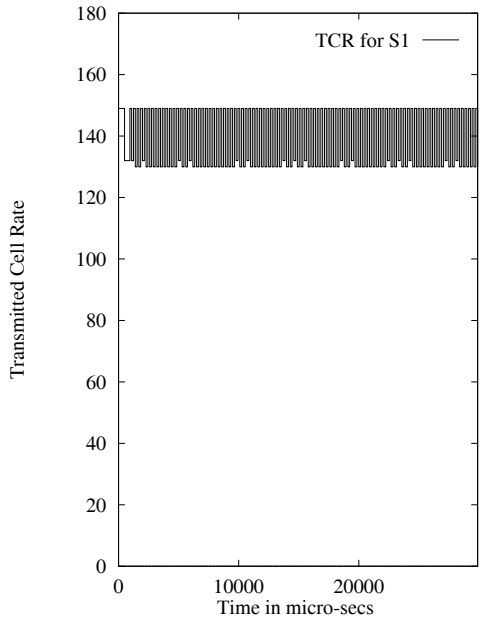
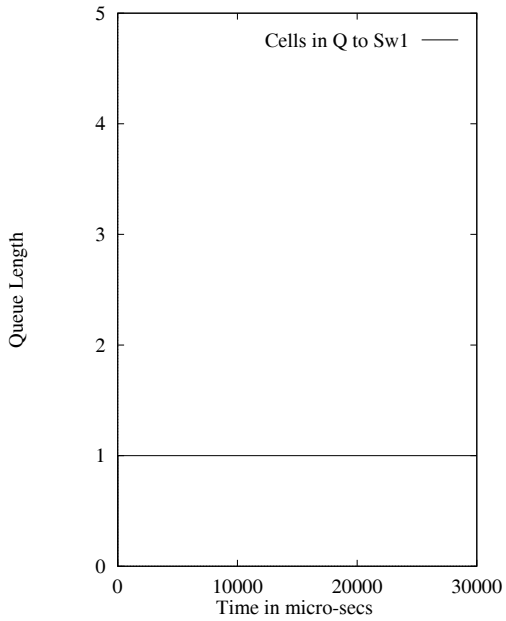


Figure 5.10: Single source configuration

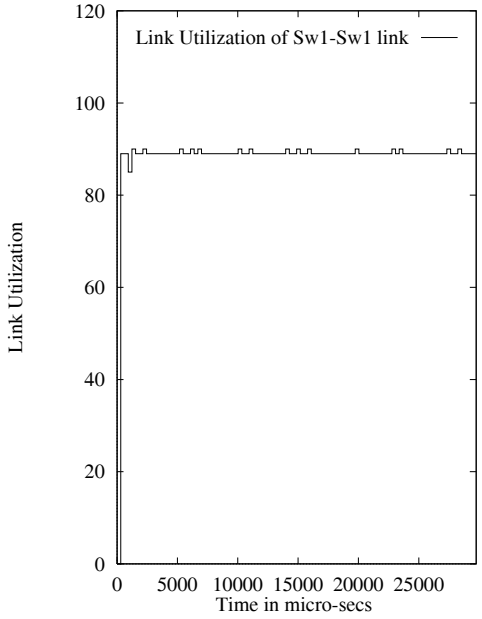
This configuration shown in Figure 5.10 consists of one VC passing through two switches connected via a link. This configuration was helpful in quickly discarding many alternatives. Figure 5.11 shows plots for TCR, link utilization, and queue length at the bottleneck link. Notice that there are no oscillations.



(a) Transmitted Cell Rate



(b) Queue Lengths



(c) Link Utilization

Figure 5.11: Simulation results for the single source configuration

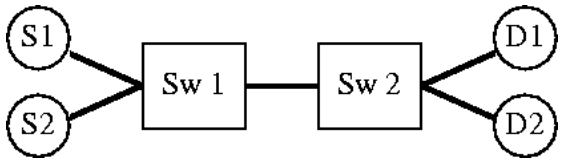


Figure 5.12: Two-source configuration

5.5.3 Two Sources

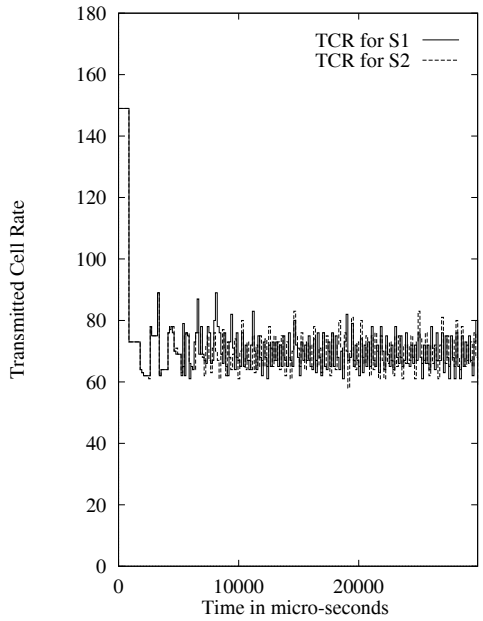
This configuration helps study the fairness. It is similar to the single source configuration except that now there are two sources as shown in Figure 5.12. Figure 5.13 shows the configuration and plots for TCR, link utilization, and queue length at the bottleneck link. Notice that both sources converge to the same level.

5.5.4 Three Sources

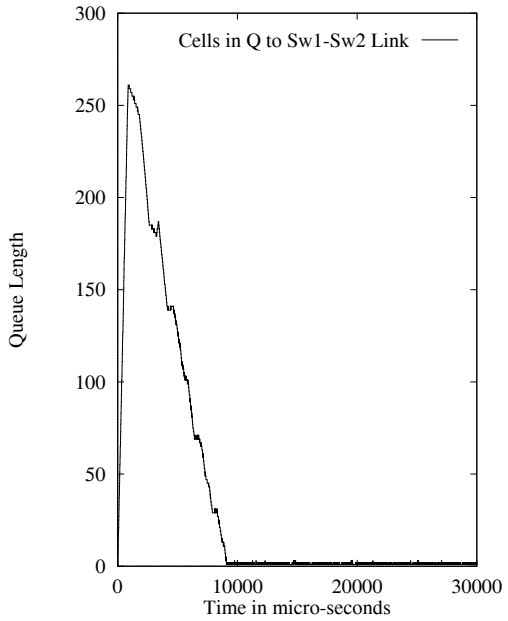
As shown in Figure 5.14, this is a simple configuration with one link being shared by three sources. The purpose of this configuration is to check what will happen if the load is such that the link is operating efficiently but not fairly. The starting rates of the three sources are specifically set to values that add up to the target cell rate for the bottleneck link. Figure 5.15 shows the simulation results for this configuration.

5.5.5 Transient Sources

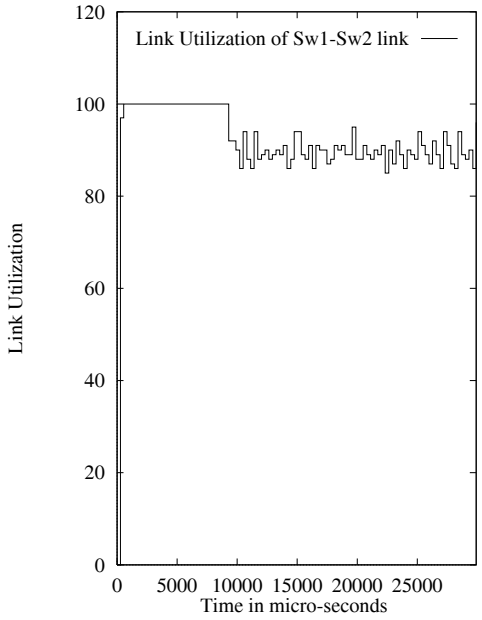
In order to study the effect of new sources coming in the network, we modified the two-source simulation such that the second source comes on after one third of the simulation run and goes off at two third of the total simulation time. The speed at which the TCRs of the two sources decrease and increase to the efficient region can be seen from Figure 5.16.



(a) Transmitted Cell Rates



(b) Queue Lengths



(c) Link Utilization

Figure 5.13: Simulation results for the two-source configuration

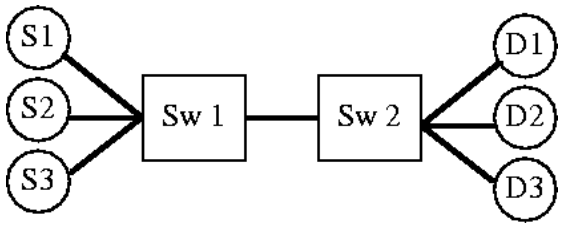


Figure 5.14: Three-source configuration

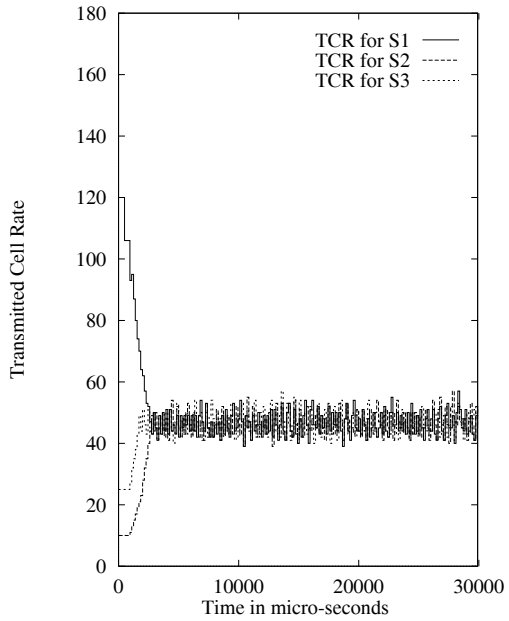
5.5.6 Parking Lot

This configuration is popular for studying fairness. The configuration and its name was derived from theatre parking lots, which consist of several parking areas connected via a single exit path. At the end of the show, congestion occurs as cars exiting from each parking area try to join the main exit stream.

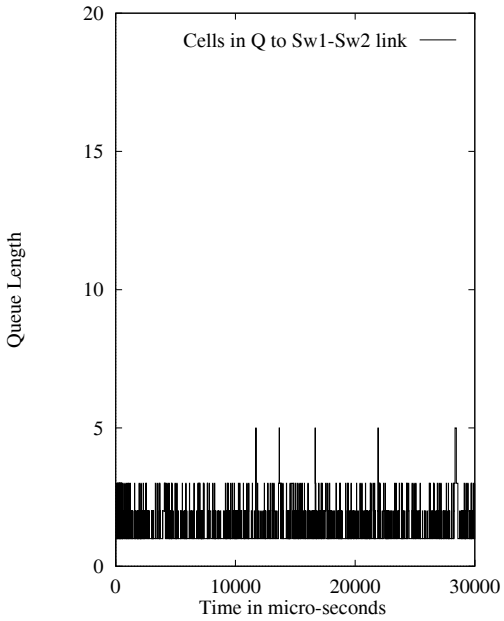
For computer networks, an n-stage parking lot configuration consists of n switches connected in a series. There are n VCs. The first VC starts from the first switch and goes to the end. For the remaining ith VC starts at the $i-1$th switch. A 3-switch parking lot configuration is shown in Figure 5.17. The simlation results are shown in Figure 5.18. Notice that all VCs receive the same throughput without any fair queueing.

5.5.7 Upstream Bottleneck

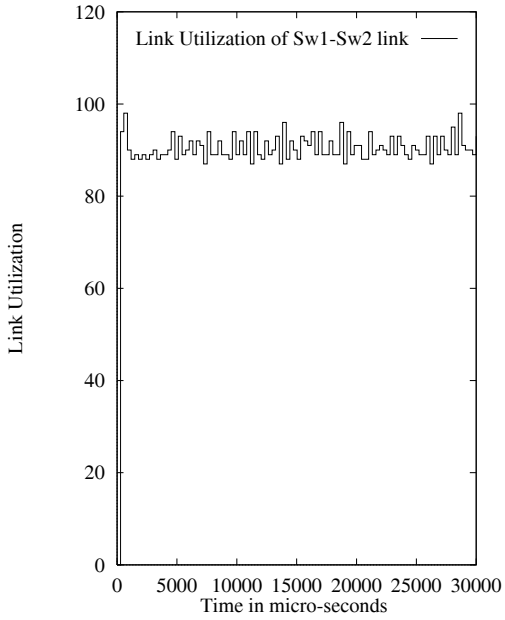
This configuration consists of four VCs and three switches as shown in Figure 5.19. The second link is shared by VC2 and VC4. However, because of the first link, VC2 is limited to a throughput of 1/3 the link rate. VC4 should, therefore, get 2/3 of the second link. This configuration is helpful in checking if the scheme will allocate all unused capacity to those source that can use it. Figure 5.20 show the simulation



(a) Transmitted Cell Rates

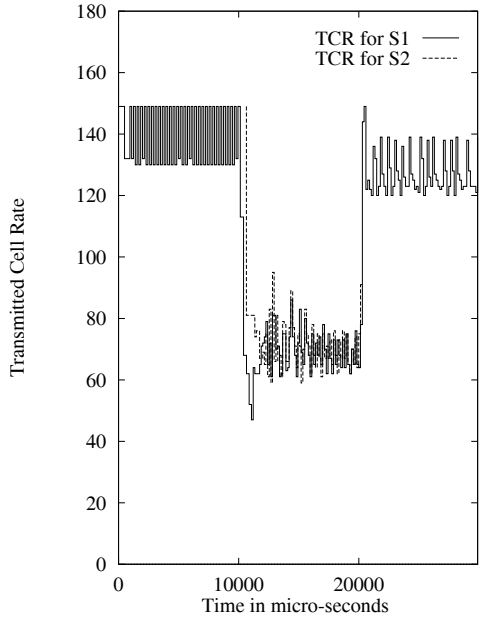


(b) Queue Lengths

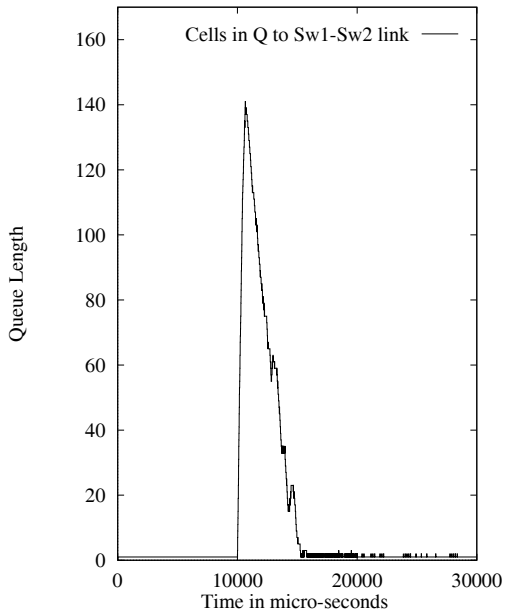


(c) Link Utilization

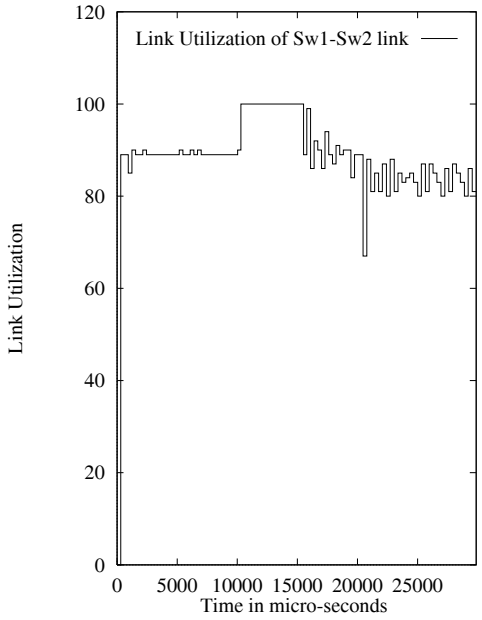
Figure 5.15: Simulation results for the three-source configuration



(a) Transmitted Cell Rates



(b) Queue Lengths



(c) Link Utilization

Figure 5.16: Simulation results for the transient experiment

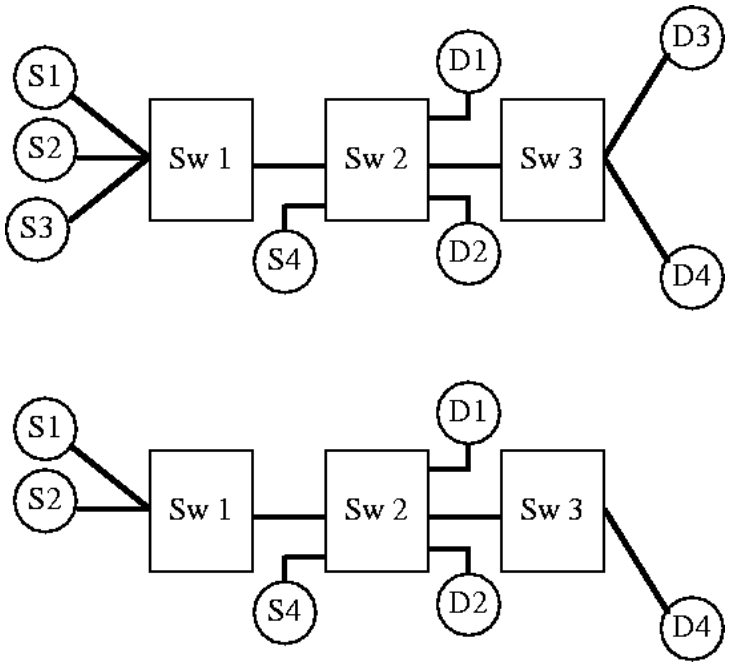
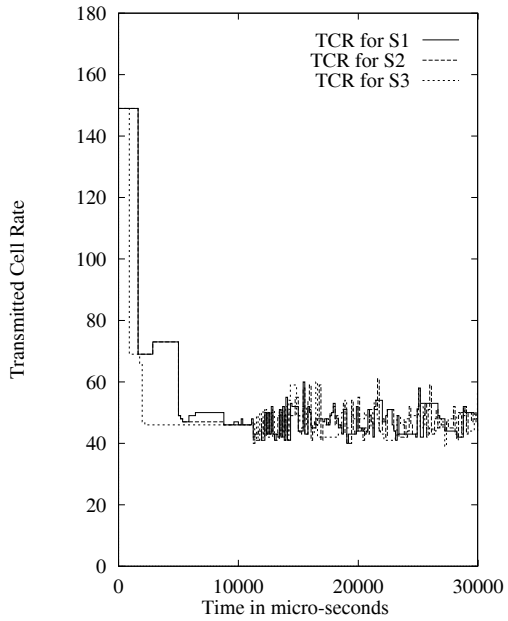


Figure 5.17: The parking lot fairness problem. All users should get the same throughput regardless of the parking area used.

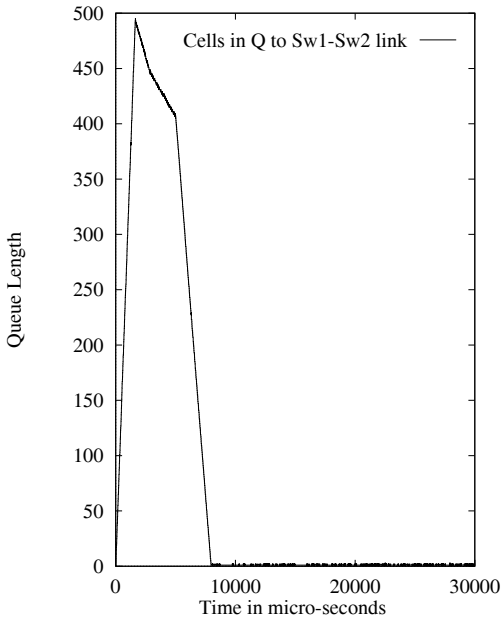
results for this configuration. In particular, the TCR for VC2 and VC4 are shown. Notice that VC4 does get the remaining bandwidth.

5.6 Results for WAN Configuration

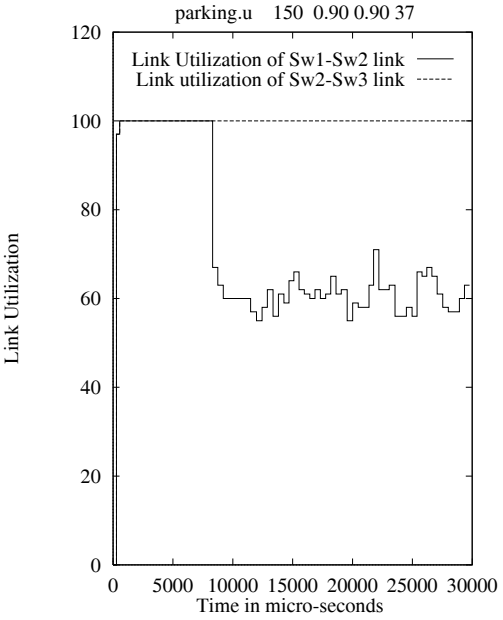
The results presented so far assumed link lengths of 1 km. The scheme works equally well for longer links. We have simulated all configurations with 1000 km links as well. Figures 5.21 shows the simulation results for two sources WAN configuration with transient.



(a) Transmitted Cell Rates



(b) Queue Lengths



(c) Link Utilization

Figure 5.18: Simulation results for the parking lot configuration

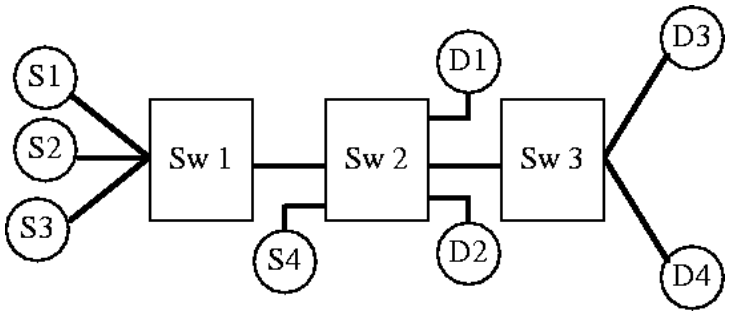
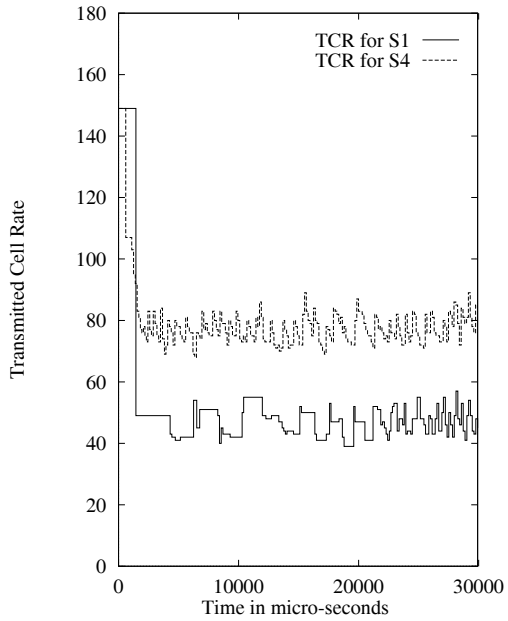


Figure 5.19: Network configuration with upstream bottleneck.

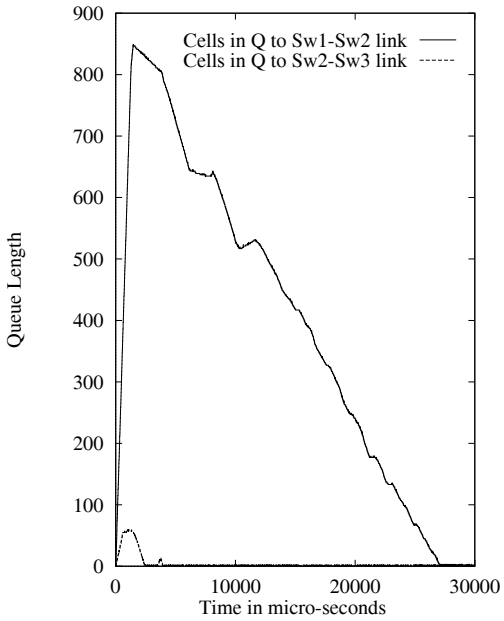
5.7 Results with Packet Train Workload

The most commonly used traffic pattern in congestion simulations is the so called "infinite source model." In this model, all sources have cells to send at all times. It is a good starting configuration because, after all, we are comparing schemes for overload and if a scheme does not work for infinite source it is not a good congestion scheme. In other words, satisfactory operation with infinite source model is necessary. However, it is not sufficient. We have found that many schemes work for infinite source models but fail to operate satisfactorily if the sources are bursty, which is usually the case.

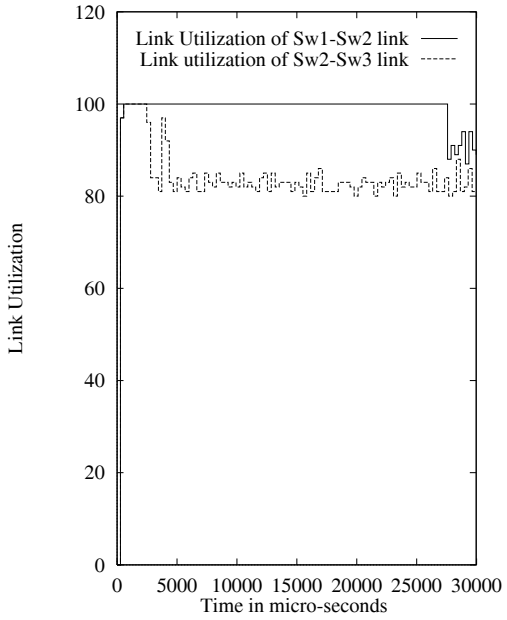
In developing the OSU scheme, we used a packet train model to simulate bursty traffic [47]. A packet train is basically a "burst" of k cells (probably consisting of segments of an application PDU) sent instantaneously by the host system to the adapter. In real systems, the burst is transfered to the adapter at the system bus rate which is very high and so simulating instantaneous transfers is justified. The adapter outputs all its cells at the link rate or at the rate specified by the network in case of rate feedback schemes. If the bursts are far apart, the resulting traffic on the link will look like trains of packets with a gap between trains.



(a) Transmitted Cell Rates

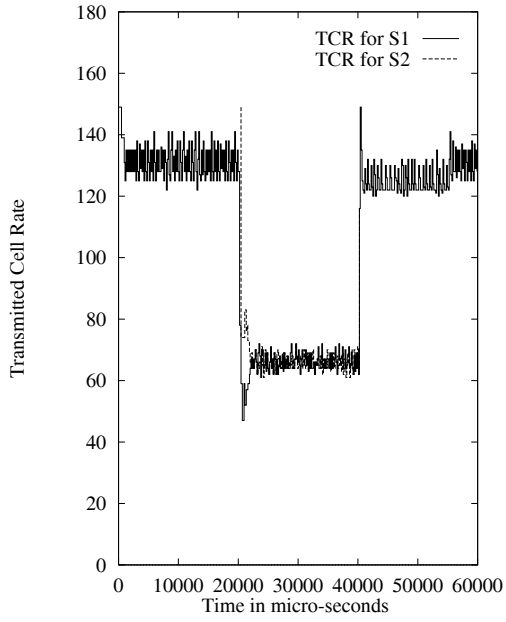


(b) Queue Lengths

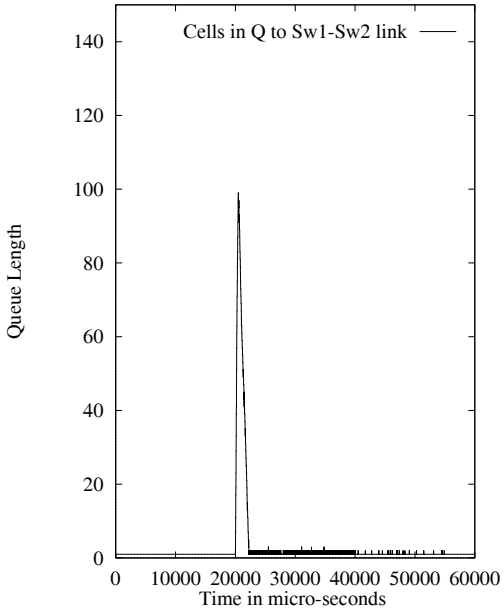


(c) Link Utilization

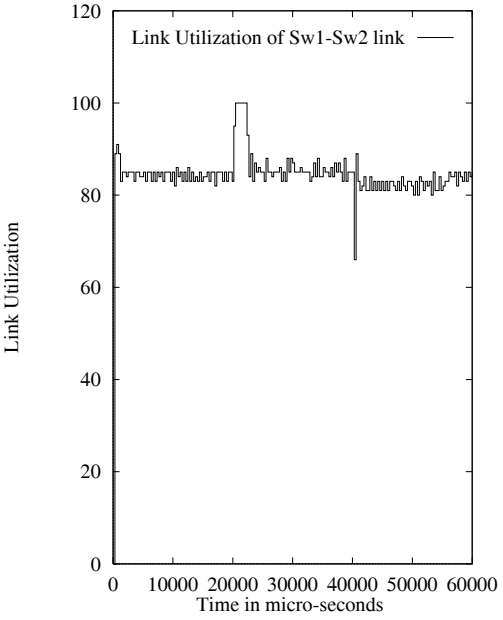
Figure 5.20: Simulation results for the upstream bottleneck configuration



(a) Transmitted Cell Rates



(b) Queue Lengths



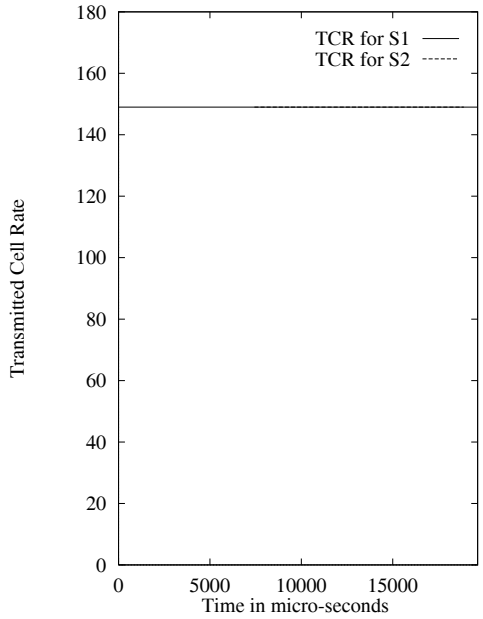
(c) Link Utilization

Figure 5.21: Simulation results for the transient configuration with 1000 km inter-switch links

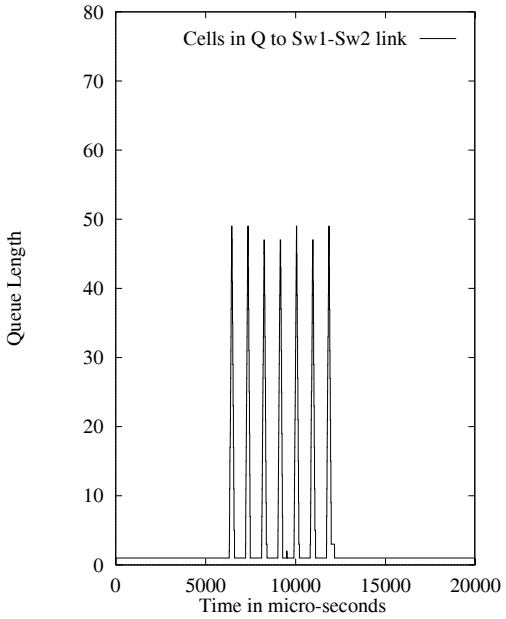
The key question in simulating the train workload is what happens when the adapter queue is full? Does the source keep putting more bursts into the queue or stops putting new bursts until permitted. We resolve this question by classifying the application as continuous media (video, etc) or interruptible media (data files). In a real system, continuous media cannot be interrupted and the cells will be dropped by the adapter when the network permitted rate is low. With interruptible media, the host stops generating new PDUs until permitted to do so by the adapter. We are simulating only interruptible packet trains for ABR traffic.

For interruptible packet trains, the intertrain gap is governed by a statistical distribution such as exponential. We use a constant interval so that we can clearly see the effect of the interval. In particular, we use one-third duty cycle, that is, the time taken to transmit the burst at the link rate is one-third of the inter-burst time. In this case, unless there are three or more VCs, the sources can not saturate the link and interesting effects are seen with some schemes. In real networks, the duty-cycle is very small of the order of 0.01; the inter-burst time may be of the order of minutes and the burst transmission time is generally a fraction of a second. To simulate overloads with such sources would require hundreds of VCs. That is why we selected a duty cycle of 1/3. This allows us to study both underload and overload with a reasonable number of VCs. We used a burst of 50 cells to keep the simulation times reasonable.

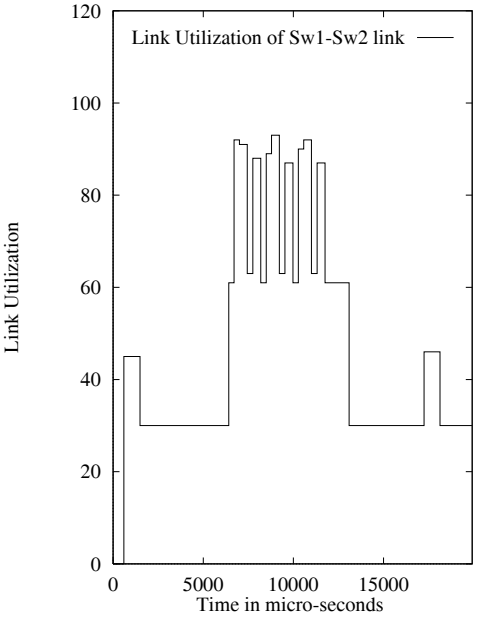
Figures 5.22 and 5.23 show simulation results for the transient and the upstream bottleneck configurations using the packet train model.



(a) Transmitted Cell Rates

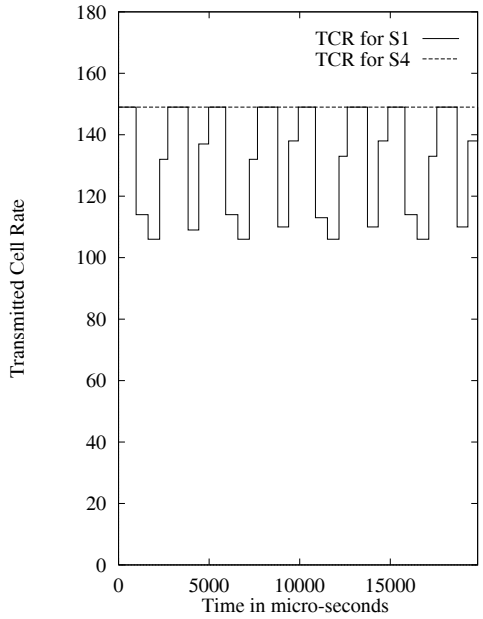


(b) Queue Lengths

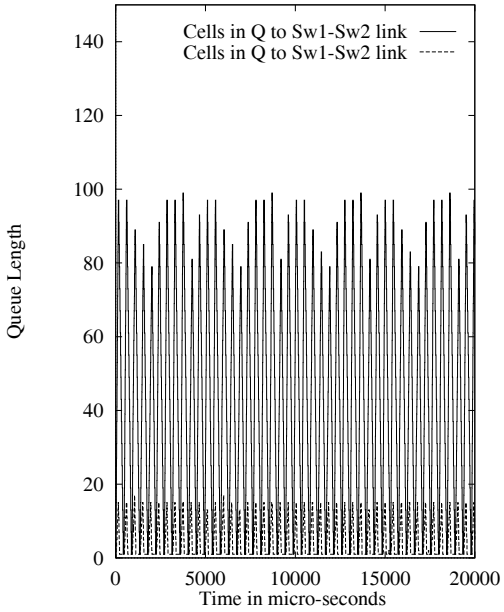


(c) Link Utilization

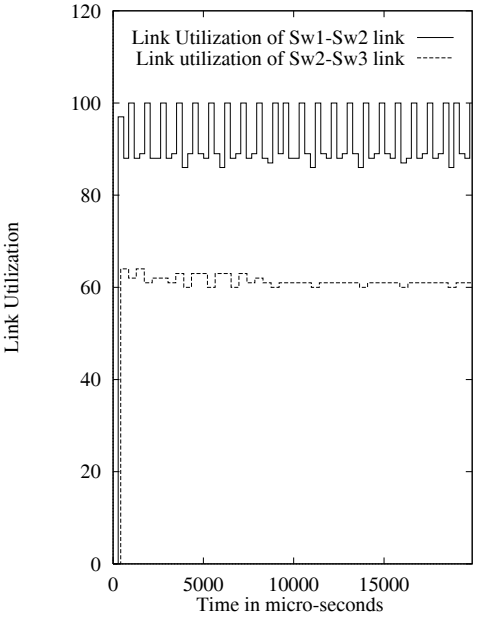
Figure 5.22: Simulation results for the transient configuration with packet train workload.



(a) Transmitted Cell Rates



(b) Queue Lengths



(c) Link Utilization

Figure 5.23: Simulation results for the upstream bottleneck configuration with packet train workload.

5.8 Proof: Fairness Algorithm Improves Fairness

In this section we analytically prove two claims about the simple fairness (TUB) algorithm:

C1. Once inside TUB, the fairness algorithm keeps the link in TUB.

C2. With the fairness algorithm, the link converges towards fair operation.

Our proof methodology is similar to that used in Chiu and Jain (1989)[19], where it was proven that multiplicative decrease and additive increase are necessary and sufficient for achieving efficiency and fairness for the DECbit scheme.

Consider two sources sharing a link of unit bandwidth. Let

x = Input rate of source 1
y = input rate of source 2
z = Load level of the link = $x + y$
U = Target utilization
Δ = Half-width of the target utilization band
s = Fair share rate = U/2

When $x + y = U$, the link is operating efficiently. This is shown graphically by the straight line marked "Efficiency line" in Figure 5.24(a). When $x = y$, the resource allocation is fair. This represents the straight line marked "Fairness line" in the figure. The ideal goal of the load adjustment algorithm is to bring the resource allocations from any point in the two dimensional space to the point marked "Goal" at the intersection of the efficiency and fairness line.

When the network is operating in a region close to the efficiency line, we consider the network to be operating efficiently. This region is bounded by the lines corresponding to $x + y = U(1 - \Delta)$ and $x + y = U(1 + \Delta)$ are in Figure 5.24(a). The quadrangular region bounded by these two lines and the x and y axes is the efficient operation zone also called the target utilization band (TUB). The TUB is described

by the four conditions: $x > 0$ and $y > 0$ and $U(1+\Delta) \geq x + y \geq U(1-\Delta)$ Observe that x and y are strictly greater than zero. The case of $x = 0$ or $y = 0$ reduces the number of sources to one.

Similarly, when the network is operating in a region close to the fairness line, we consider the network to be operating fairly. This region is bounded by the lines corresponding to $y = x(1-\Delta)/(1+\Delta)$ and $y = x(1+\Delta)/(1-\Delta)$. The quadrangular region bounded by these two lines in side the TUB is called the fairness region. This is shown in Figure 5.24(b). Mathematically, the conditions defining the fairness region are:

$$\frac{(1+\Delta)}{(1-\Delta)}x \geq y \geq \frac{(1-\Delta)}{(1+\Delta)}x \tag{5.1}$$

$$U(1+\Delta) \geq x + y \geq U(1-\Delta) \tag{5.2}$$

The fair share s is $U/2$. Recall that the TUB algorithm sets the load adjustment factor (LAF) as follows:

IF $(x < s)$ THEN LAF $= \frac{z}{1+\Delta}$ ELSE LAF $= \frac{z}{1-\Delta}$

The rate x is divided by the LAF at the source to give the new rate x'. In other words,

$x' = x\frac{1+\Delta}{z}$ if $x < s$ and $x\frac{1-\Delta}{z}$ otherwise.

5.8.1 Proof of Claim C1

To prove claim C1, we introduce the lines $x = s$ and $y = s$ and divide the TUB into four non-overlapping regions as shown in Figure 5.25(a). These regions correspond to the following inequalities:

Region 1: $s > x > 0$ and $y \geq s$ and $U(1+\Delta) \geq x + y \geq U(1-\Delta)$

Region 2: $y \geq s$ and $x \geq s$ and $U(1+\Delta) \geq x+y$

Region 3: $s > y > 0$ and $x \geq s$ and $U(1+\Delta) \geq x+y \geq U(1-\Delta)$

Region 4: $y < s$ and $x < s$ and $x+y \geq U(1-\Delta)$

In general, triangular regions are described by three inequalities, quandrangular regions by four inequalities and so on.

Proof for Region 1

Consider a point (x, y) in the quadrangular region 1. It satisfies the conditions: $x > 0$ and $y \geq s$ and $U(1+\Delta) \geq x+y \geq U(1-\Delta)$. The link is operating at a load level z given by:

$z = \frac{x+y}{U}$ or $y = Uz - x$

Since (x, y) is in the TUB, we have: $(1+\Delta) \geq z \geq (1-\Delta)$. According to the TUB algorithm, given that $x < s = U/2$ and $y \geq s = U/2$, the system will move the two sources from the point (x, y) to the point $(x', y') = (\frac{x(1+\Delta)}{z}, \frac{y(1-\Delta)}{z})$.

$$x' + y' = \frac{x(1+\Delta) + y(1-\Delta)}{z} \tag{5.3}$$

$$= U(1+\Delta) - \frac{2x\Delta}{z} \tag{5.4}$$

$$= U(1-\Delta) + \frac{2\Delta}{z}y \tag{5.5}$$

$$\tag{5.6}$$

The quantity on the left hand side of the above equation is the new total load. Since the last terms of equations 5.4 and 5.5 are both positive quantities, the new total load is below $U(1+\Delta)$ and above $U(1-\Delta)$. In other words, the new point is in TUB. This proves that claim C1 holds for all points in region 1.

Proof for Region 2

Points in the triangular region 2 satisfy the conditions: $y \geq s$, $x \geq s$, and $x + y \leq U(1 + \Delta)$

In this region, both x and y are greater than or equal to the fair share $s = U/2$. Therefore, the new point is given by : $(x', y') = (\frac{x(1-\Delta)}{z}, \frac{y(1-\Delta)}{z})$.Hence,

$$x' + y' = \frac{x(1-\Delta) + y(1-\Delta)}{z} = \frac{(x+y)(1-\Delta)}{z} = \frac{Uz(1-\Delta)}{z} = U(1-\Delta)$$

This indicates that the new point is on the lower line of the TUB (which is a part of the TUB) This proves claim C1 for all points in region 2.

The proof of claim C1 for regions 3 and 4 is similar to that of regions 1 and 2, respectively.

5.8.2 Proof of Claim C2

We show convergence to the fairness region (claim C2) as follows. Any point in the fairness region remains in the fairness region. Further, any point (x, y) in the TUB but not in the fairness region moves towards the fairness region at every step. Consider the line L joining the point (x, y) to the origin $(0, 0)$ as shown in Figure 5.25(a). As the angle between this line and the fairness line $(x = y)$ decreases, the operation becomes fairer. We show that in regions outside the fairness zone, the angle between the line L and the fairness line either decreases or remains the same. If the angle remains the same, the point moves to a region where the angle will decrease in the subsequent step.

We introduce four more lines to Figure 5.25(a). These lines correspond to $y = (1 + \Delta)x$, $y = (1 - \Delta)x$, $y = \frac{(1-\Delta)}{(1+\Delta)}x$ and $y = \frac{(1+\Delta)}{(1-\Delta)}x$. This results in the TUB

being divided into eight non-overlapping regions as shown in Figure 5.25(b). The new regions are described by the conditions:

Region 1a: $s > x > 0$ and $y \geq s$ and $U(1+\Delta) \geq x+y \geq U(1-\Delta)$ and $y > (1+\Delta)x$

Region 1b: $s > x$ and $(1+\Delta)x \geq y \geq s$

Region 2: $y \geq s$ and $x \geq s$ and $U(1+\Delta) \geq x+y$

Region 3a: $s > y > 0$ and $x \geq s$ and $U(1+\Delta) \geq x+y \geq U(1-\Delta)$ and $y < (1-\Delta)x$

Region 3b: $s > y \geq (1-\Delta)x$ and $x \geq s$

Region 4a: $y < s$ and $x < s$ and $x+y \geq U(1-\Delta)$ and $y \leq \frac{(1+\Delta)}{(1-\Delta)}x$ and $y \geq \frac{(1-\Delta)}{(1+\Delta)}x$

Region 4b: $y < s$ and $x+y \geq U(1-\Delta)$ and $y > \frac{(1+\Delta)}{(1-\Delta)}x$

Region 4c: $x < s$ and $x+y \geq U(1-\Delta)$ and $y < \frac{(1-\Delta)}{(1+\Delta)}x$

The regions 1a and 1b are subdivisions of region 1 in Figure 5.25(a). Similarly, regions 3a and 3b are subdivisions of region 3, and regions 4a, 4b, and 4c are subdivisions of region 4 in Figure 5.25(a) respectively. Observe that regions 1b, 2, 3b and 4a are completely contained in the fairness region.

Proof for Region 1a

Hexagonal region 1a is defined by the conditions: $s > x > 0$ and $y \geq s$ and $U(1+\Delta) \geq x+y \geq U(1-\Delta)$ and $y > (1+\Delta)x$. The new point is given by: $(x', y') = (\frac{x(1+\Delta)}{z}, \frac{y(1-\Delta)}{z})$. Hence,

$$\frac{y'}{x'} = \frac{y}{x} \times \frac{1-\Delta}{1+\Delta} \tag{5.7}$$

Since Δ is a positive non-zero quantity, the above relation implies:

$$\frac{y'}{x'} < \frac{y}{x} \tag{5.8}$$

Further since y/x is greater than $1 + \Delta$, equation 5.7 also implies:

$$\frac{y'}{x'} > (1 - \Delta) \tag{5.9}$$

Equation 5.8 says that the slope of the line joining the origin to new point (x', y') is lower than that of he line joining the origin to (x, y). While equation 5.9 says that the new point does not overshoot the fairness region. This proves Claim C2 for all points in region 1a.

Proof for Region 1b

Triangular region 1b is defined by the conditions: $s > x$ and $(1 + \Delta)x \geq y \geq s$. Observe that region 1b is completely enclosed in the fairness region because it also satisfies the conditions 5.1 and 5.2 defining the fairness region.

To prove claim C2, we show that the new point given by $(x', y') = (\frac{x(1+\Delta)}{z}, \frac{y(1-\Delta)}{z})$ remains in the fairness region.

Since (x, y) satisfies the conditions $1 < y/x \leq (1 + \Delta)$, we have:

$$\frac{1 - \Delta}{1 + \Delta} < \frac{y'}{x'} \leq (1 - \Delta) \tag{5.10}$$

Condition 5.10 ensures that the new point remains in the fairness region defined by conditions 5.1 and 5.2.

This proves Claim C2 for all points in region 1b.

Proof of claim C2 for region 3a and 3b is similar to that of regions 1a and 1b, respectively.

Proof for Region 2

Triangular region 2 is defined by the conditions: $y \geq s$ and $x \geq s$ and $x + y \leq U(1 + \Delta)$. This region is completely enclosed in the fairness region. The new point is given by:

$$x' = \frac{x(1-\Delta)}{z} \text{ and } y' = \frac{y(1-\Delta)}{z}$$

Observe that:

$$\frac{y'}{x'} = \frac{y}{x} \text{ and } x' + y' = \frac{(x+y)(1-\Delta)}{z} = U(1-\Delta)$$

That is, the new point is at the intersection of the line joining the origin and the old point and the lower boundary of the TUB. This intersection is in the fairness region. This proves Claim C2 for all points in region 2.

Proof for Region 4

Triangular region 4 is defined by the conditions: $y < s$ and $x < s$ and $x + y \geq U(1 - \Delta)$. The new point is given by:

$$x' = \frac{x(1+\Delta)}{z} \text{ and } y' = \frac{y(1+\Delta)}{z}$$

Observe that:

$$\frac{y'}{x'} = \frac{y}{x} \text{ and } x' + y' = \frac{(x+y)(1+\Delta)}{z} = U(1+\Delta)$$

That is, the new point is at the intersection of the line joining the origin and the old point and the upper boundary of the TUB.

As shown in Figure 5.25(b), region 4 consists of 3 parts: 4a, 4b, and 4c. All points in region 4a are inside the fairness region and remain so after the application of the TUB algorithm. All points in region 4b move to region 1a where subsequent

applications of TUB algorithm will move them towards the fairness region. Similarly, all points in region 4c move to region 3a and subsequently move towards the fairness region.

This proves claim C2 for region 4.

5.8.3 Proof for Asynchronous Feedback Conditions

We note that our proof has assumed the following conditions:

- Feedback is given to sources instantaneously.
- Feedback is given to sources synchronously.
- There are no input load changes (like new sources coming on) during the period of convergence
- The analysis is for the bottleneck link (link with the highest utilization).
- The link is shared by unconstrained sources (which can utilize the rate allocations).

It may be possible to relax one or more of these assumptions. However, we have not verified all possibilities. In particular, the assumption of synchronous feedback can be relaxed as shown next.

In the previous proof, we assumed that the operating point moves from (x, y) to (x', y'). However, if only one of the sources is given feedback, the new operating point could be (x, y') or (x', y). This is called asynchronous feedback.

The analysis procedure is similar to the one shown in the previous sections. For example, consider region 1 of Figure 5.25(a). If we move from (x, y) to (x, y'), we

have:

$$y' = \frac{y(1-\Delta)}{z}$$

and

$$\begin{aligned} x + y' &= \frac{xz + y(1-\Delta)}{z} && (5.11) \\ &= U(1-\Delta) + \frac{x\{z-(1-\Delta)\}}{z} && (5.12) \\ &= U(1+\Delta) - \frac{x\{(1+\Delta)-z\} + 2y\Delta}{z} && (5.13) \\ & && (5.14) \end{aligned}$$

Since, the last terms of equations 5.12 and 5.13 are both positive, the new point is still in the TUB. This proves Claim C1.

Further, we have:

$$\frac{y'}{x} = \frac{y}{x}(1-\Delta)$$

Therefore,

$$\frac{y'}{x} < \frac{y}{x} \text{ and } \frac{y'}{x} \geq (1-\Delta)$$

That is, the slope of the line joining the operating point to the origin decreases but does not overshoot the fairness region.

Note that when $z = 1-\Delta$, $y' = y$. That is, the operating point does not change. Thus, the points on the lower boundary of the TUB ($x + y = U(1-\Delta)$) do not move, and hence the fairness for these points does not improve in this step. It will change only in the next step when the operating point moves from (x, y') to (x', y').

The proof for the case (x', y) is similar. This completes the proof of C1 and C2 for region 1. The proof for region 3 is similar.

5.9 Current Traffic Management Specifications vs OSU Scheme

In the previous sections, we have mentioned several features of the OSU scheme that have either been adopted in the standard or have been commonly implemented. In this section, we describe two features that were not adopted.

In the OSU scheme, the sources send RM cells every T microseconds. This is the time-based approach. A count-based alternative is to send RM cells after every n data cells. We argued that the time-based approach is more general. It provides the same feedback delay for all link speeds and source rates.

The ATM forum has adopted the count-based approach mainly because it guarantees that the overhead caused by RM cells will be a fixed percentage (100/n)% of the total load on the network.

The disadvantage with the count-based approach is that if there are many low-rate sources, it will take a long time to control them since the inter-RM cell times will be large. The time-based approach uses a fixed bandwidth per active source for RM cell overhead. For many active sources, this could be excessive.

The RM cells in the OSU scheme contain an averaging interval field. The network manager sets the averaging interval parameter for each switch. The maximum of the averaging interval along a path is returned in the RM cell. This is the interval that the source uses to send the RM cells. With the count-based approach, this field is not required.

Another major difference is the indication of rate. The OSU scheme requires sources to present both average and peak rates (along with the averaging interval) in the RM cell. The standard requires only one rate.

The OSU scheme is, therefore, incompatible with the ATM forum's current traffic management standards. Although, it cannot be used directly, most of its features and results can be ported to design compatible schemes. We have upgraded the ideas from the OSU scheme to create the Explicit Rate Indication for Congestion Avoidance (ERICA) scheme [45]. The ERICA scheme which is described later in this dissertation, is also mentioned in the ATM Traffic Management 4.0 standards as a sample switch algorithm.

5.10 Limitations and Summary of the OSU Scheme

This chapter describes an explicit rate based congestion avoidance scheme for ATM networks. The scheme was developed as the ATM Forum traffic management specifications were being developed. While the strengths of the OSU scheme are its choice of congestion indicator, metric, small number of parameters, and O(1) complexity, its limitations are slow convergence for complex configurations, and slight sensitivity to the averaging interval parameter. The following statements apply to the basic OSU scheme.

Our proof in section 5.8 is applicable to the bottleneck link (link with the highest utilization) which is shared by unconstrained sources (which can use any given allocation). It assumes that feedback is given to sources instantaneously and synchronously. In the general case, where these assumptions do not hold, the system may take longer to converge to the fair and efficient operating point. If the perturbations to the system (due to VBR, asynchronous feedback, multiple bottlenecks, or rapid changes in source load pattern) are of a time scale smaller than this convergence

time, the system may be unstable. This statement is true for the convergence of *any* switch algorithm.

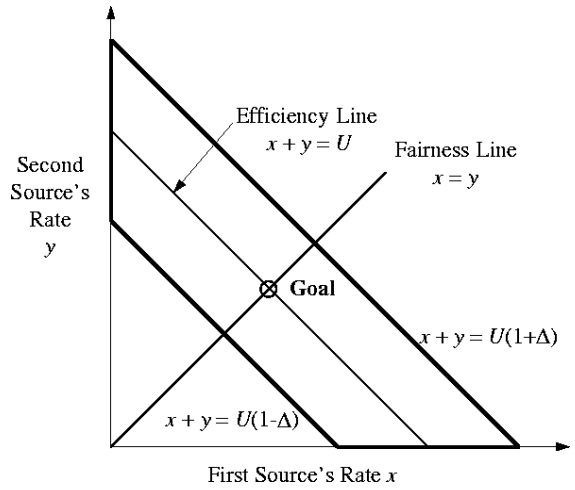
Further, since the scheme is measurement-based, it is slightly sensitive to the averaging interval in the switch. For example, if the number of sources is underestimated, the scheme will attempt to converge to a higher fairshare value and keep moving in and out of the TUB. Note that even then, the bottleneck is maintained at a high utilization level and the excess capacity is used to drain out queues. The number of sources is never overestimated; hence our scheme always achieves efficiency. The second quantity measured in the averaging interval is the current load level, z. If the system is actually overloaded, then the overload is measured correctly in z. However, if the system is underloaded, the averaging interval may not be long enough to exactly measure the underload. In such a case, z may be underestimated, and the system may initially move to an overload region before converging.

Although the scheme itself is no longer strictly compatible with the specifications, many of the results obtained during this research have affected the direction of the specifications. Many features of the scheme are now being commonly used in many switch implementations. A patent on the inventions of this scheme is also pending [56].

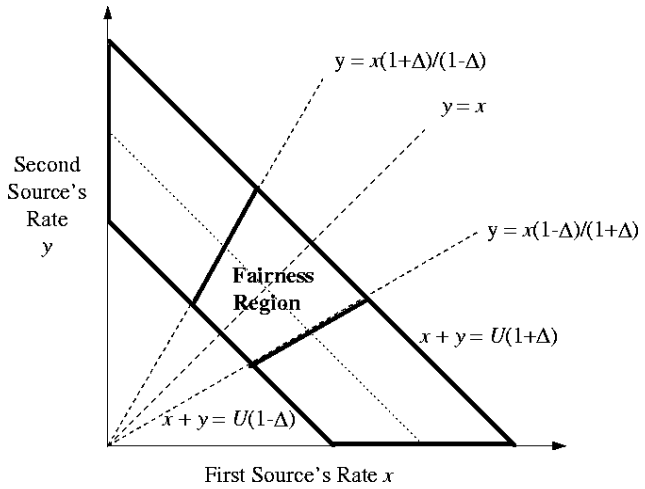
Three different options that further improve the performance over the basic scheme are also described. These allow the fairness to be achieved quickly, oscillations to be minimized, and feedback delay to be reduced.

As stated in the previous section, we have developed a new ATM standards compatible algorithm called ERICA. ERICA and its extensions use a new set of algorithms. These algorithms achieve fast convergence and robustness for complex

workloads, where input load and capacity may fluctuate arbitrarily. This will be the subject of our future chapters.

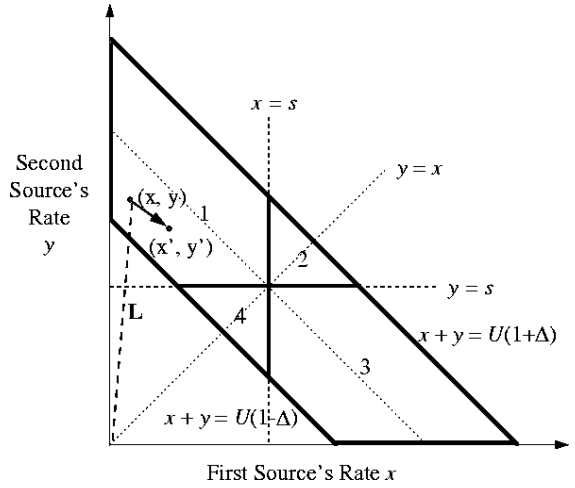


(a) Ideal Fairness Goal

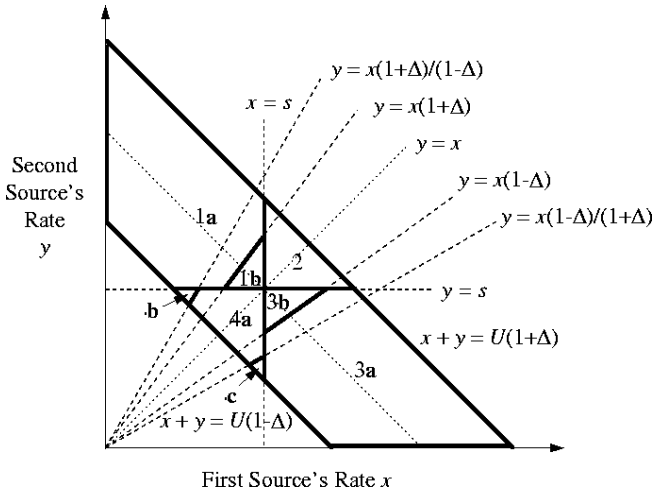


(b) The Fairness Region

Figure 5.24: A geometric representation of efficiency and fairness for a link shared by two sources



(a) Regions used to prove Claim C1



(b) Regions used to prove Claim C2

Figure 5.25: Subregions of the TUB used to prove Claims C1 and C2

CHAPTER 6

THE ERICA AND ERICA+ SCHEMES

The ERICA scheme is built upon the ideas of the OSU scheme (described in chapter 5. The key limitations of the OSU scheme were the incompatibility with current ATM Forum Traffic Management 4.0 standards [32], and the long time taken to converge to steady state (transient response) from arbitrary initial conditions in complex configurations.

The ERICA and ERICA+ schemes overcome the limitations of the OSU scheme, while keeping the attractive features. Further, they are optimistic algorithms which allocate rates to optimize for both the transient performance, as well as the steady state performance. Since real networks are in a transient state most of the time (sources starting and stopping, ABR capacity varying constantly), we believe that a scheme deployed in real-world switches need to perform well under both transient and steady state conditions.

This chapter is organized as follows. Section 6.1 describes the basic ERICA algorithm. Modifications of this basic algorithm are then presented one by one. The simulation results and performance evaluation are described in section 6.22, while the pseudocode for the algorithm can be found in appendix C.

6.1 The Basic ERICA Algorithm

The switch periodically monitors the load on each link and determines a load factor, z, the available capacity, and the number of currently active virtual connections or VCs (N). The load factor is calculated as the ratio of the measured input rate at the port to the target capacity of the output link.

$$z \leftarrow \frac{\text{ABR Input Rate}}{\text{ABR Capacity}}$$

where ABR Capacity$\leftarrow$Target Utilization (U) $\times$ Link Bandwidth.

The Input Rate is measured over an interval called the switch averaging interval. The above steps are executed at the end of the switch averaging interval.

Target utilization (U) is a parameter which is set to a fraction (close to, but less than 100 %) of the available capacity. Typical values of target utilization are 0.9 and 0.95.

The load factor, z, is an indicator of the congestion level of the link. High overload values are undesirable because they indicate excessive congestion; so are low overload values which indicate link underutilization. The optimal operating point is at an overload value equal to one. The goal of the switch is to maintain the network at unit overload.

The fair share of each VC, $FairShare$, is also computed as follows:

$$\text{FairShare} \leftarrow \frac{\text{ABR Capacity}}{\text{Number of Active Sources}}$$

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

$$\text{VCShare} \leftarrow \frac{CCR}{z}$$

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

$$\text{ER Calculated} \leftarrow \text{Max (FairShare, VCShare)}$$

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

The calculated ER value cannot be greater than the ABR Capacity which has been measured earlier. Hence, we have:

$$\text{ER Calculated} \leftarrow \text{Min (ER Calculated, ABR Capacity)}$$

To ensure that the bottleneck ER reaches the source, each switch computes the minimum of the ER it has calculated as above and the ER value in the RM cell. This value is inserted in the ER field of the RM cell:

ER in RM Cell$\leftarrow$Min(ER in RM cell, ER Calculated).

A flow chart of the basic algorithm is presented in figure C.1 (see appendix C). The flow chart shows steps to be taken on three possible events: at the end of an averaging interval, on receiving a cell (data or RM), and on receving a backward RM cell. These steps have been numbered for reference in further modifications of the basic scheme.

6.2 Achieving Max-Min Fairness

Assuming that the measurements do not suffer from high variance, the above algorithm is sufficient to converge to efficient operation in all cases and to the max-min fair allocations in most cases. The convergence from transient conditions to the desired operating point is rapid, often taking less than a round trip time.

However, we have discovered cases in which the basic algorithm does not converge to max-min fair allocations. This happens if all of the following three conditions are met:

1. The load factor z becomes one

2. There are some sources which are bottlenecked elsewhere upstream

3. CCR for all remaining sources is greater than the $FairShare$

If this happens, then the system remains in its current state, because the term CCR/z is greater than $FairShare$ for the non-bottlenecked sources. This final state may or may not be fair in the max-min sense.

To achieve max-min fairness, the basic ERICA algorithm is extended by remembering the highest allocation made during one averaging interval and ensuring that

all eligible sources can also get this high allocation. To do this, we add a variable $MaxAllocPrevious$ which stores the maximum allocation given in the previous interval, and another variable $MaxAllocCurrent$ which accumulates the maximum allocation given during the current switch averaging interval. The step 9 of the basic algorithm is replaced by the flow chart shown in figure C.2 (see appendix C).

Basically, for $z > 1 + \delta$, where δ is a small fraction, we use the basic ERICA algorithm and allocate the source Max (FairShare, VCShare). But, for $z \leq 1 + \delta$, we attempt to make all the rate allocations equal. We calculate the ER as Max (FairShare, VCShare, MaxAllocPrevious).

The key point is that the $VCShare$ is only used to achieve efficiency. The fairness can be achieved only by giving the contending sources equal rates. Our solution attempts to give the sources equal allocations during underload and then divide the (equal) CCRs by the same z during the subsequent overload to bring them to their max-min fair shares. The system is considered to be in a state of overload when its load factor, z, is greater than $1 + \delta$. The aim of introducing the quantity δ is to force the allocation of equal rates when the overload is fluctuating around unity, thus avoiding unnecessary rate oscillations. The next subsection examines one further modification to the ERICA algorithm.

6.3 Fairshare First to Avoid Transient Overloads

The inter-RM cell time determines how frequently a source receives feedback. It is also a factor in determining the transient response time when load conditions change. With the basic ERICA scheme, it is possible that a source which receives feedback first can keep getting rate increase indications, purely because it sends more RM cells

before competing sources can receive feedback. This results in unnecessary spikes (sudden increases) in rates and queues with the basic ERICA scheme.

The problem arises when the Backward RM (BRM) cells from different sources arrive asynchronously at the switch. Consider a LAN configuration of two sources (A and B), initially sending at low rates. When the BRM arrives, the switch calculates the feedback for the current overload. Without loss of generality, assume that the BRM of source A is encountered before that of source B. Now it is possible that the BRM changes the rate of source A and the new overload due to the higher rate of A is experienced at the switch before the BRM from the source B reaches the switch. The transient overload experienced at the switch may still be below unity, and the ACR of source A is increased further (BRMs for source A are available since source A sends more RM cells at higher rates). This effect is observed as an undesired spike in the ACR graphs and sudden queue spikes when the source B gets its fair share.

This problem can be solved by incorporating the following change to the ERICA algorithm. When the calculated ER is greater than the fair share value, and the source is increasing from a CCR below $FairShare$, we limit its increase to $FairShare$. Alternatively, the switch could decide not to give new feedback to this source for one measurement interval. The following computation is added to the switch algorithm.

After "ER Calculated" is computed:

IF ((CCR $<$ FairShare) AND (ER Calculated $\geq$ FairShare)) THEN

ER Calculated $\leftarrow$FairShare

We can also disable feedback to this source for one measurement interval.

"ER in RM Cell" is then computed as before.

6.4 Forward CCR Used for Reverse Direction Feedback

Earlier schemes [43] provided their feedback to the RM cells going in the forward direction. This ensured that the CCR in the RM cell was correlated to the load level measured by the switch during that interval. However, the time taken by the forward going RM cell to travel back to the source was long and this slowed down the response of the system.

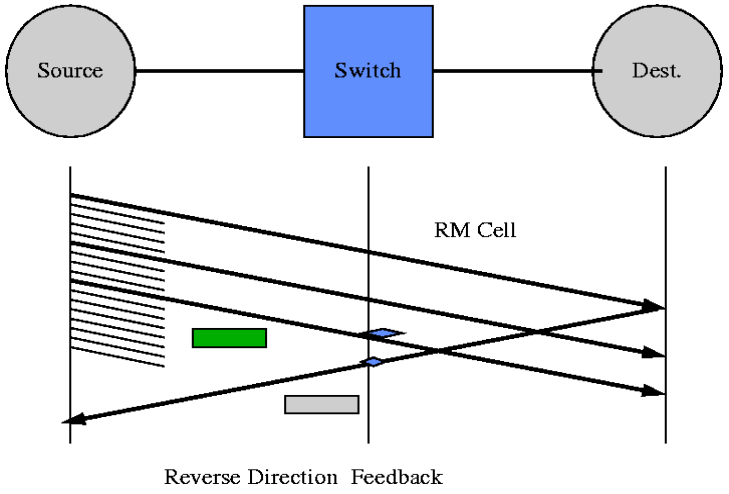


Figure 6.1: Reverse direction feedback

Switches can indicate their feedback to the sources in the reverse path of the RM cell. The backward going RM (BRM) cell takes less time to reach the source than the forward going RM (FRM) cell which has to reach the destination first. Thus, the system responds faster to changes in the load level. However, the CCR carried by the BRM cell no longer reflects the load level in the system. To maintain the most current CCR value, the switch copies the CCR field from FRM cells, and uses this information to compute the ER value to be inserted in the BRM cells. This ensures

that the latest CCR information is used in the ER calculation and that the feedback path is as short as possible. Figure 6.1 shows that the first RM cell carries (in its backward path), the feedback calculated from the information in the most recent FRM cell. The CCR table update and read operations still preserve the O(1) time complexity of the algorithm.

6.5 Single Feedback in a Switch Interval

The switch measures the overload, the number of active sources and the ABR capacity periodically (at the end of every switch averaging interval). The source also sends RM cells periodically. These RM cells may contain different rates in their CCR fields. If the switch encounters more than one RM cell from the same VC during the same switch interval, then it uses the same value of overload for computing feedback in both cases. For example, if two RM cells from the same VC carried different CCR values, then the feedback in one of them will not accurately reflect the overload. As a result, the switch feedback will be erroneous and may result in unwanted rate oscillations. The switch thus needs to give only one feedback value per VC in a single switch interval.

The above example illustrates a fundamental principle in control theory, which says that the system is unstable when the control is faster than feedback. But the system is unresponsive if the control is slower than feedback. Ideally, the control rate should be matched to the feedback rate. In our system, the delay between successive feedbacks should not be greater than the delay between successive measurements (controls).

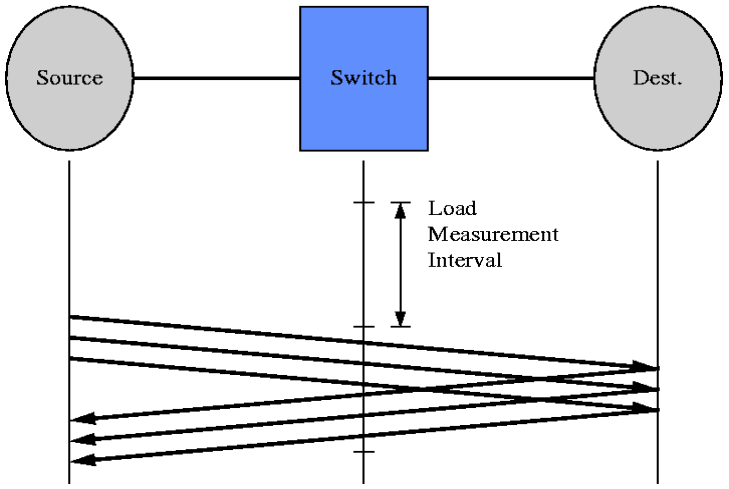


Figure 6.2: Independence of source and switch intervals

The switch provides only one feedback value during each switch interval irrespective of the number of RM cells it encounters. The switch calculates the ER only once per interval, and the ER value obtained is stored. It inserts the same ER value in all the RM cells it sees during this interval. In figure 6.2, the switch interval is greater than the RM cell distance. The ER calculated in the interval marked *Load Measurement Interval* is maintained in a table and set in all the RM cells passing through the switch during the next interval.

6.6 Per-VC CCR Measurement Option

The CCR of a source is obtained from the CCR field of the forward going RM cell. The latest CCR value is used in the ERICA computation. It is assumed that the CCR is correlated with the load factor measured. When the CCR is low, the frequency of forward RM cells becomes very low. Hence, the switch may not have a new CCR estimate though a number of averaging intervals have elapsed. Moreover,

the CCR value may not be an accurate measure of the rate of the VC if the VC is bottlenecked at the source, and is not able to use its ACR allocation. Note that if a VC is bottlenecked on another link, the CCR is set to the bottleneck allocation within one round-trip.

A possible solution to the problems of inaccurate CCR estimates is to measure the CCR of every VC during the same averaging interval as the load factor. This requires the switch to count the number of cells received per VC during every averaging interval and update the estimate as follows:

At the *end of an switch averaging interval:*

```
FOR ALL VCs DO
    CCR[VC] ←NumberOfCells[VC]/IntervalLength
    NumberOfCells[VC] ←0
END
```

When a *cell is received:*

```
NumberOfCells[VC] ←NumberOfCells[VC] + 1
```

Initialization:

```
FOR ALL VCs DO NumberOfCells[VC] ←0
```

When an *FRM cell is received,* do not copy CCR field from FRM into CCR[VC].

Note that using this method, the switch ignores the CCR field of the RM cell. The per-VC CCR computation can have a maximum error of (one cell/averaging interval) in the rate estimate. Hence the error is minimized if the averaging interval is larger.

The effect of the per VC CCR measurement can be explained as follows. The basic ERICA uses the formula: ER Calculated←Max (FairShare, VCShare).

The measured CCR estimate is always less than or equal to the estimate obtained from the RM cell CCR field. If the other quantities remain constant, the term "VCShare" decreases. Thus the ER calculated will decrease whenever the first term dominates. This change results in a more conservative feedback, and hence shorter queues at the switches.

6.7 ABR Operation with VBR and CBR in the Background

The discussion so far assumed that the entire link was being shared by ABR sources. Normally, ATM links will be used by constant bit rate (CBR) and variable bit rate (VBR) traffic along with ABR traffic. In fact, CBR and VBR have a higher priority. Only the capacity left unused by VBR and CBR is given out to ABR sources. For such links, we need to measure the CBR and VBR usage along with the input rate. The ABR capacity is then calculated as follows:

$$\text{ABR Capacity} \leftarrow \text{Target Utilization} \times \text{Link Bandwidth} - \text{VBR Usage} - \text{CBR Usage}$$

The rest of ERICA algorithm remains unchanged. Notice that the target utilization is applied to the entire link bandwidth and not the the left over capacity. That is,

$$\text{ABR Capacity} \neq \text{Target Utilization} \times \{\text{Link Bandwidth} - \text{VBR Usage} - \text{CBR Usage}\}$$

There are two implications of this choice. First, (1-Target Utilization) ×(Link Bandwidth) is available to drain the queues, which is much more than what would be available otherwise. Second, the sum of VBR and CBR usage must be less than

(Target Utilization)×(Link Bandwidth). Thus, the VBR and CBR allocation should be limited to below the target utilization.

6.8 Bi-directional Counting of Bursty Sources

A bursty source sends data in bursts during its active periods, and remains idle during other periods. It is possible that the BRM cell of a bursty source could be traveling in the reverse direction, but no cells of this source are traveling in the forward direction. A possible enhancement to the counting algorithm is to also count a source as active whenever a BRM of this source is encountered in the reverse direction. We refer to this as the "bidirectional counting of active VCs".

One problem with this technique is that the reverse queues may be small and the feedback may be given before the $FairShare$ is updated, taking into consideration the existence of the new source. Hence, when feedback is given, we check to see if the source has been counted in the earlier interval and if the $FairShare$ has been updated based upon the existence of the source. If the source had not been counted, we update the number of active sources and the $FairShare$ before giving the feedback. This option is called "the immediate fairshare update option" in the flow chart of figure C.3 (see appendix C).

We could also reset the CCR of such a source to zero after updating the $FairShare$ value, so that the source is not allocated more than the $FairShare$ value. The motivation behind this strategy is that the source may be idle, but its CCR is unchanged because no new FRMs are encountered. When the per-VC CCR measurement is used, this option is not necessary, because the switch measures the CCRs periodically. The setting of CCR to zero is a conservative strategy which avoids large queues due to

bursty or ACR retaining sources. A drawback of this strategy is that in certain configurations, the link may not be fully utilized if the entire traffic is bursty. This is because all the bursty sources are asked to send at $FairShare$, which may not be the optimal value if some sources are bottlenecked elsewhere. This option can also be enabled and disabled based upon a certain queue threshold.

6.9 Averaging of the Number of Sources

Another technique to overcome the problem of underestimating the number of active sources is to use exponential averaging to decay the contribution of each VC to the number of active sources count. The main motivation behind this idea is that if a source is inactive during the current interval, but was recently active, it should still contribute to the number of active sources. This is because this source might be sending its data in bursts, and just happened to be idle during the current interval.

Flow charts of figures C.4 and C.5 show this technique (see appendix C).

The $DecayFactor$ used in decaying the contribution of each VC is a value between zero and one, and is usually selected to be a large fraction, say 0.9. The larger the value of the $DecayFactor$, the larger the contribution of the sources active in prior intervals, and the less sensitive the scheme is to measurement errors. Setting the $DecayFactor$ to a smaller fraction makes the scheme adapt faster to sources which become idle, but makes the scheme more sensitive to the averaging interval length.

6.10 Boundary Cases

Two boundary conditions are introduced in the calculations at the end of the averaging interval. First, the estimated number of active sources should never be less

ABR Capacity	Input Rate	Overload	Fairshare	CCR/Overload	Feedback
Zero	Non-zero	Infinity	Zero	Zero	Zero
Non-zero	Zero	Infinity	C/N	Zero	C/N
Non-zero	Non-zero	I/C	C/N	CCR×C/I	Max (CCR×C/I, C/N)
Zero	Zero	Infinity	Zero	Zero	Zero

Table 6.1: Boundary Cases

than one. If the calculated number of sources is less than one, the variable is set to one. Second, the load factor becomes infinity when the ABR capacity is measured to be zero, and the load factor becomes zero when the input rate is measured to be zero. The corresponding allocations are described in Table 6.1.

6.11 Averaging of the Load Factor

In cases where no input cells are seen in an interval, or when the ABR capacity changes suddenly (possibly due to a VBR source going away), the overload measured in successive intervals may be considerably different. This leads to considerably different feedbacks in successive intervals. An optional enhancement to smoothen this variance is by averaging the load factor. This effectively increases the length of the averaging interval over which the load factor is measured.

One way to accomplish this is shown in the flow chart of figure C.6 (see appendix C).

The method described above has the following drawbacks. First, the average is reset everytime z becomes infinity. The entire history accumulated in the average prior to the interval where the load is to be infinity is lost.

For example, suppose the overload is measured in successive intervals as: 2, 1, Infinity, 3, Infinity, 0.5. The method previously described forgets the history in the fourth interval, and restarts at the new value 3. Similarly in the sixth interval, it restarts at the value 0.5. Note that this introduces dependencies between the boundary cases and the average value of the load factor.

The second problem with this method is that the exponential average does not give a good indication of the average value of quantities which are not additive. In our case, the load factor is not an additive quantity. However, the number of ABR cells received or output is additive.

The load factor is a ratio of the input rate and the ABR capacity. The correct way to average a ratio is to find the ratio of the average (or the sum) of the numerators and divide it by the average (or the sum) of the denominators. That is, the average of $x_1/y_1, x_2/y_2, \ldots, x_n/y_n$ is $(x_1 + x_2 + \ldots + x_n)/(y_1 + y_2 + \ldots + y_n)$.

To average load factor, we need to average the input rate (numerator) and the ABR capacity (denominator) separately. However, the input rate and the ABR capacity are themselves ratios of cells over time. The input rate is the ratio of number of cells input and the averaging interval. If the input rates are $x_1/T_1, x_2/T_2, \ldots, x_n/T_n$, the average input rate is $((x_1 + x_2 + \ldots + x_n)/n)/((T_1 + T_2 + \ldots + T_n)/n)$. Here, x_i's are the number of ABR cells input in averaging interval i of length T_i. Similarly the average ABR capacity is $((y_1 + y_2 + \ldots + y_n)/n)/((T_1 + T_2 + \ldots + T_n)/n)$, where y_i's are the maximum number of ABR cells that can be output in averaging interval i of length T_i.

The load factor is the ratio of these two averages. Observe that each of the quantities added is not a ratio, but a number. Exponential averaging is an extension

of arithmetic averaging used above. Averages such as $(x_1 + x_2 + \ldots x_n)/n$ can be replaced by the exponential average of the variable x_i.

The flow chart of figure C.7 describes this averaging method.

Observe that the load factor thus calculated is never zero or infinity unless the input rate or ABR capacity are always zero. If the input rate or the ABR capacity is measured to be zero in any particular interval, the boundary cases for overload are not invoked. The load level increases or decreases to finite values.

6.12 Time and Count Based Averaging

The load factor, available ABR capacity and the number of active sources need to be measured periodically. There is a need for an interval at the end of which the switch renews these quantities for each output port. The length of this interval determines the accuracy and the variation of the measured quantities. As mentioned before, longer intervals provide lower variation but result in slower updating of information. Alternatively, shorter intervals allow fast response but introduce greater variation in the response. This section proposes alternative intervals for averaging the quantities.

The averaging interval can be set as the time required to receive a fixed number of ABR cells (M) at the switch in the forward direction. While this definition is sufficient to correctly measure the load factor and the ABR capacity at the switch, it is not sufficient to measure the number of active VCs (N) or the CCR per VC accurately. This is because the quantities N and CCR depend upon the fact that at least one cell from the VC is encountered in the averaging interval. Moreover, when the rates are low, the time to receive M cells may be large. Hence the feedback in the reverse direction may be delayed.

An alternative way of averaging the quantities is by a fixed time interval, T. This ensures that any source sending at a rate greater than (one cell/T) will be encountered in the averaging interval. This interval is independent of the number of sources, but is dependent upon the minimum rate of the source. In addition to this, if the aggregate input rate is low, the fixed-time interval is smaller than the fixed-cells interval. However, when there is an overload, the fixed-cells interval provides faster response.

One way of combining these two kinds of intervals is to use the minimum of the fixed-cell interval and the fixed-time interval. This combination ensures quick response for both overload and underload conditions. But it still suffers from the disadvantages of a fixed-cell interval, where N and per-VC CCR cannot be measured accurately [62].

Another strategy for overcoming this limitation is to measure N and per-VC CCR over a fixed-time interval, and the capacity and load factor over the minimum of the fixed-cell and fixed-time interval. The time intervals can be different as long as some correlation exists between the quantities measured over the different intervals. Typically, the intervals to measure CCR and N would be larger to get more stable estimates.

6.13 Selection of ERICA Parameters

Most congestion control schemes provide the network administrator with a number of parameters that can be set to adapt the behavior of the schemes to their needs. A good scheme must provide a small number of parameters that offer the desired

level of control. These parameters should be relatively insensitive to minor changes in network characteristics.

ERICA provides a few parameters which are easy to set because the tradeoffs between their values are well understood. Our simulation results have shown that slight mistuning of parameters does not significantly degrade the performance of the scheme. Two parameters are provided: the target Utilization (U) and the switch measurement interval.

6.13.1 Target Utilization *U*

The target utilization determines the link utilization during steady state conditions. If the input rate is greater than Target Utilization $\times$ Link Capacity, then the switch asks sources to decrease their rates to bring the total input rate to the desired fraction. If queues are present in the switch due to transient overloads, then $(1 - U) \times$ Link Capacity is used to drain the queues.

Excessively high values of target utilization are undesirable because they lead to long queues and packet loss, while low target utilization values lead to link underutilization. The effectiveness of the value of target utilization depends on the feedback delay of the network. Transient overloads can potentially result in longer queues for networks with longer feedback delays. Due to this, smaller target utilization values are more desirable for networks with long propagation delays.

Our simulation results have determined that ideal values of target utilization are 0.95 and 0.9 for LANs and WANs respectively. Smaller values improve the performance of the scheme when the traffic is expected to be highly bursty.

6.13.2 Switch Averaging Interval *AI*

The switch averaging or measurement interval determines the accuracy of feedback. This interval is used to measure the load level, link capacity and the number of active VCs for an outgoing link. The length of the measurement interval establishes a tradeoff between accuracy and steady state performance. This tradeoff has been briefly discussed in section 6.5.

ERICA measures the required quantities over an averaging interval and uses the measured quantities to calculate the feedback in the next averaging interval. Averaging helps smooth out the variation in the measurements. However, the length of the averaging interval limits the amount of variation which can be eliminated. It also determines how quickly the feedback can be given to the sources, because ERICA gives at most one feedback per source per averaging interval. Longer intervals produce better averages, but slow down the rate of feedback. Shorter intervals may result in more variation in measurements, and may consistently underestimate the measured quantities.

The load factor and available capacity are random variables whose variance depends on the length of the averaging interval. In practice, the interval required to measure the number of active sources is sufficient for the measurement of the load factor and available capacity. Both of these averaged quantities are fairly accurate, with an error margin of (one cell/averaging interval). Setting the target utilization below 100% helps drain queues due to errors in measurement of all the quantities. Whenever the scheme faces tradeoffs due to high errors in measurement, the degree of freedom is to reduce the target utilization parameter, sacrificing some steady state utilization for convergence.

6.14 ERICA+: Queue Length as a Secondary Metric

ERICA+ is a further modification of ERICA. In this and the following section, we describe the goals, target operating point, the algorithm, and parameter settings for ERICA+.

ERICA depends upon the measurement of metrics such as the overload factor and the number of active ABR sources. If there is a high error in the measurement, and the target utilization is set to very high values, ERICA may diverge, i.e., the queues may become unbounded, and the capacity allocated to drain the queues becomes insufficient. The solution in such cases is to set the target utilization to a smaller value, allowing more bandwidth to drain queues. However, steady state utilization (utilization when there is no overload) is reduced because it depends upon the target utilization parameter.

A simple enhancement to ERICA is to have a queue threshold, and reduce the target utilization if the queue exceeds the threshold. Once the target utilization is low, the queues are drained out quickly. Hence, this enhancement maintains high utilization when the queues are small, and drains out queues quickly when they become large. Essentially, we are using the queue length as a secondary metric (input rate is the primary metric).

In ERICA, we have not considered the queue length or queue delay as a possible metric. In fact, we rejected it because it gives no indication of the correct rates of the sources. In ERICA+, we maintain that the correct rate assignments depend upon the aggregate input rate, rather than the queue length. However, we recognize two facts about queues: a) non-zero queues imply 100% utilization, and, b) a system with very long queues is far away from the intended operating point. Hence, in ERICA+,

if the input rates are low and the queues are long, we recognize the need to reserve more capacity to drain the queues and allocate rates conservatively till the queues are controlled. Further, keeping in line with the design principles of ERICA, we use continuous functions of the queue length, rather than discontinuous functions. Since feedback to sources is likely to be regular (as long as queues remain), the allocations due to a continuous function in successive averaging intervals track the behavior of the queue and reflect it in the rate allocations.

6.15 ERICA+: 100% Utilization and Quick Drain of Queues

ERICA achieves high utilization in the steady state, but utilization is limited by the target utilization parameter. For expensive links, it is desirable to keep the steady state utilization at 100%. This is because a link being able to service 5% more cells can translate into 5% more revenue. The way to get 100% utilization in steady state, and quick draining of queues is to vary the target ABR rate dynamically. During steady state, the target ABR rate is 100% while it is lower during transient overloads. Higher overloads result in even lower target rates (thereby draining the queues faster). In other words:

Target rate = function (queue length, link rate, VBR rate)

The "function" above has to be a decreasing function of the queue length.

Note that ERICA has a fixed target utilization, which means that the drain rate is independent of the queue size.

6.16 ERICA+: Maintain a "Pocket" of Queues

The ABR capacity varies dynamically, due to the presence of higher priority classes (CBR and VBR). Hence, if the higher priority classes are absent for a short interval (which may be smaller than the feedback delay), the remaining capacity is not utilized. In such situations, it is useful to have a "pocket" full of ABR cells which use the available capacity while the RM cells are taking the "good news" to the sources and asking them to increase their rates.

One way to achieve this effect is to control the queues to a "target queue length." In the steady state, the link is 100% utilized, and the queue length is equal to the target queue length, which is the "pocket" of queues we desire. If the queue length falls below this value, the sources are encouraged to increase their rate and vice versa. In other words:

Target rate = function (queue length, target queue length, link rate, VBR rate)

6.17 ERICA+: Scalability to Various Link Speeds

The above function is not scalable to various link speeds because the queue length measured in cells translates to different drain times for different transmission speeds. For example, a queue length of 5 at a T1 link may be considered large while a queue length of 50 at an OC-3 link may be considered small. This point is significant due to the varying nature of ABR capacity, especially in the presence of VBR sources.

To achieve scalability, we need to measure all queue lengths in units of time rather than cells. However, the queue is the only directly measurable quantity at the switch. The queueing delay is then estimated using the measured ABR capacity value. The

above function for target rate becomes:

Target rate = function (queue delay, target queue delay, link rate, VBR rate)

In the following sections, we define and describe a sample function to calculate the target rate.

6.18 ERICA+: Target Operating Point

ERICA+ uses a new target operating point which is in the middle of the "knee" and the "cliff," as shown in figure 3.2. The new target operating point has 100% utilization and a fixed non-zero queueing delay. This point differs from the *knee* point (congestion avoidance: 100% throughput, minimum delay) in that it has a fixed non-zero delay goal. This is due to non-zero queueing delay at the operating point. Note that the utilization remains 100% as long as the queue is non-zero. The utilization remains at 100% even if there are short transient underloads in the input load, or the output capacity increases (appearing as an underload in the input load).

We note that non-zero queue values in steady state imply that the system is in an unstable equilibrium. Queues grow immediately during transient overloads. In contrast, ERICA could allow small load increases (5 to 10%) without queue length increases.

The challenge of ERICA+ is to maintain the unstable equilibrium of non-zero queues and 100% utilization. Specifically, when the queueing delay drops below the target value, $T0$, ERICA+ increases allocation of VCs to reach the optimum delay. Similarly, when the queueing delay increases beyond $T0$, the allocation to VCs is reduced and the additional capacity is used for queue drain in the next cycle. When the queueing delay is $T0$, 100% of the ABR capacity is allocated to the VCs. ERICA+,

hence, introduces a new parameter, $T0$, in place of the target utilization parameter of ERICA.

6.19 The ERICA+ Scheme

As previously mentioned, the ERICA+ scheme is a modification of the ERICA scheme. In addition to the suggested scheduling method between VBR and ABR classes, the following are the changes to ERICA.

1. The link utilization is no longer targeted at a constant *Target Utilization* as in ERICA. Instead, the total ABR capacity is measured given the link capacity and the VBR bandwidth used in that interval: $Total\ ABR\ Capacity + VBR\ Capacity = Link\ Capacity$

2. The target ABR capacity is a fraction of the total ABR capacity

$$Target\ ABR\ Capacity \leftarrow f(T_q) \times Total\ ABR\ Capacity$$

This function must satisfy the following constraints:

1. It must have a value greater than or equal to 1 when the queueing delay, T_q is 0 (zero queues). This allows the queues to increase and T_q can go up to $T0$, the threshold value. A simple choice is to keep the value equal to one. The queue increases due to the slight errors in measurement. Another alternative is to have a linear function, with a small slope. Note that, we should not use an aggressive increase function. Since queueing delay is a highly variant quantity, a small variation in delay values may cause large changes in rate allocations, and hence lead to instability.

2. It must have a value less than 1 when the queueing delay, T_q is greater than $T0$. This forces the queues to decrease and T_q can go down to $T0$. Since queue increases are due to traffic bursts, a more aggressive control policy is required for this case compared to the former case where we project a higher capacity than available. Since we project a lower capacity than what is available, the remaining capacity is used to drain the queues.

3. If the queues grow unboundedly, then we would like the function to go to zero. Since zero, or very low, ABR capacity is unacceptable, we place a cutoff on the capacity allocated to queue drain. The cutoff is characterized by a parameter, called the queue drain limit factor (QDLF). A value of 0.5 for the QDLF parameter is sufficient in practice.

4. When the queueing delay, T_q is $T0$ we want f $(T_q) = 1$.

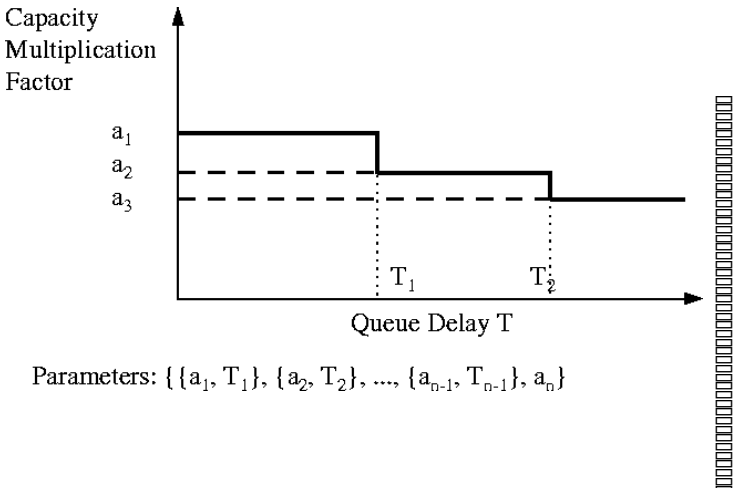


Figure 6.3: Step functions for ERICA+

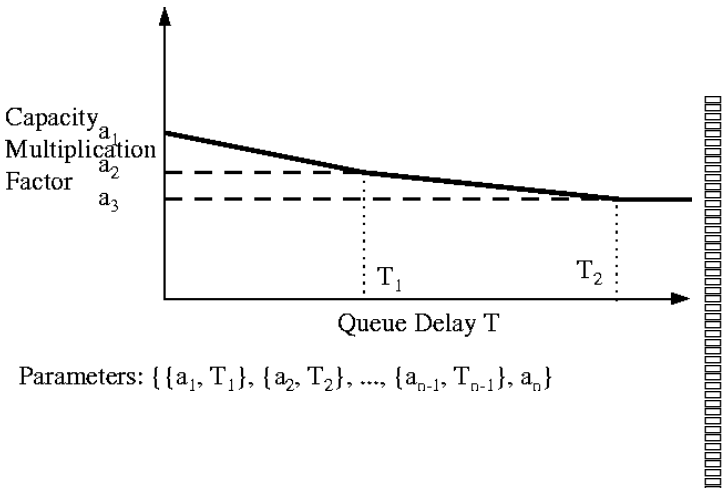


Figure 6.4: Linear functions for ERICA+

A step function which reduces the capacity in steps (down to the cutoff value) as the queueing delay exceeds thresholds is a possible choice. This is shown in figure 6.3. Linear segments as shown in figure 6.4 can be used in place of step functions. Hysteresis thresholds (figure 6.5) can be used in place of using a single threshold to increase and decrease the capacity. Hysteresis implies that we use one threshold to increase the capacity and another to decrease the capacity. However, these functions require the use of multiple thresholds (multiple parameters). Further, the thresholds are points of discontinuity, i.e., the feedback given to the source will be very different if the system is on the opposite sides of the threshold. Since queueing delay is a highly variant quantity, the thresholds and experience is required to choose these different parameters.

However, it is possible to have a function with just 2 parameters, one for the two ranges: (0, Q0) and (Q0, infinity) respectively. The rectangular hyperbolic and the negative exponential functions are good choices to provide the aggressive control

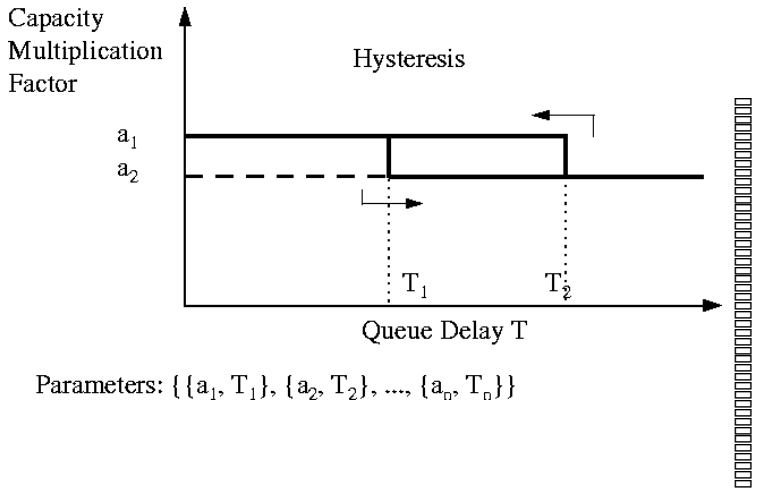


Figure 6.5: Hysteresis functions for ERICA+

required when the queues grow. We choose the former which is the simpler of the two.

Since the portion $T < T0$ requires milder control, we can have a different hyperbola for that region. This requires an extra parameter for this region. The queue control scheme uses a time (queueing delay) as a threshold value. Hence, depending upon the available capacity at the moment, this value $T0$ translates into a queue length Q0, as follows:

$$Q0 = Total\ ABR\ Capacity \times T0$$

In the following discussion, we will refer to Q0 and queues alone, but Q0 is a variable dependent upon available capacity. The fixed parameter is $T0$. The queue control function, as shown in figure 6.6, is:

$$f(T_q) = \frac{a \times Q0}{(a-1) \times q + Q0} \quad for\ q > Q0$$

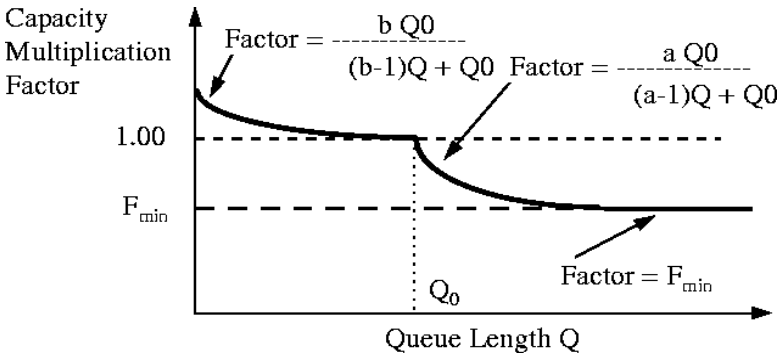


Figure 6.6: The queue control function in ERICA+

and

$$f(T_q) = \frac{b \times Q0}{(b-1) \times q + Q0} \ \ for\ 0 \leq q \leq Q0$$

Note that $f(T_q)$ is a number between 1 and 0 in the range Q0 to infinity and between b and 1 in the range 0 to Q0. Both curves intersect at Q0, where the value is 1. These are simple rectangular hyperbolas which assume a value 1 at Q0. This function is lower bounded by the queue drain limit factor (QDLF):

$$f(T_q) = Max(QDLF, \frac{a \times Q0}{(a-1) \times q + Q0}) \ \ for\ q > Q0$$

6.20 Effect of Variation on ERICA+

ERICA+ calculates the target ABR capacity, which is the product of $f(T_q)$ and the ABR capacity. Both these quantities are variant quantities (random variables), and the product of two random variables (say, A and B) results in a random variable which has more variance than either A or B. Feedback becomes less reliable as the variance increases.

For example, overload depends upon the ABR capacity and is used in the formula to achieve max-min fairness. Since the ERICA+ algorithm changes the ABR capacity depending upon the queue lengths, this formula needs to tolerate minor changes in load factor. In fact, the formula applies hysteresis to eliminate the variation due to the load factor. Since techniques like hysteresis and averaging can tolerate only a small amount of variation, we need to reduce the variance in the target ABR capacity.

We examine the ABR capacity term first. ABR capacity is estimated over the averaging interval of ERICA. A simple estimation process can entail counting the VBR cells sent, calculating the VBR capacity, and subtracting it from the link capacity. This process may have an error of one VBR cell divided by the averaging interval length. The error can be minimized by choosing longer averaging intervals.

However, the measured ABR capacity has less variance than instantaneous queue lengths. This is because averages of samples have less variance than the samples themselves, and ABR capacity is averaged over an interval, whereas queue length is not. The quantity $Q0 = T0 \times ABRCapacity$ has the same variance as that of the measured ABR capacity.

We now examine the function, $f(T_q)$. This function is bounded below by $QDLF$ and above by b. Hence, its values lie in the range (QDLF,b) or, in practice, in the range (0.5, 1.05). Further, it has variance because it depends upon the queue length, q and the quantity $Q0$. Since the function includes a ratio of $Q0$ and q, it has higher variance than both quantities.

One way to reduce the variance is to use an averaged value of queue length (q), instead of the instantaneous queue length. A simple average is the mean of the queue lengths at the beginning and the end of a measurement interval. This is sufficient for

small averaging intervals. If the averaging interval is long, a better average can be obtained by sampling the queue lengths during the interval and taking the average of the samples. Sampling of queues can be done in the background.

Another way to reduce variation is to specify a constant $Q0$. This can be specified instead of specifying $T0$ if a target delay in the range of $[\frac{Q0}{MinimumABRcapacity}, \frac{Q0}{MaximumABRcapacity}]$ is acceptable.

6.21 Selection of ERICA+ Parameters

The queue control function in ERICA+ has four parameters: $T0$, a, b, and $QDLF$. In this section, we explain how to choose values for the parameters and discuss techniques to reduce variation in the output of the function.

The function $f(T_q)$ has three segments: (1) a hyperbola characterized by the parameter b (called the b-hyperbola) between queueing delay of zero and $T0$, (2) an a-hyperbola from a queueing delay of $T0$ till $f(T_q)$ equals $QDLF$, (3) $QDLF$. Hence, the range of the function $f(T_q)$ is $[QDLF, b]$.

6.21.1 Parameters a and b

a and b are the intercepts of the a-hyperbola and b-hyperbola, i.e., the value of $f(T_q)$ when $q = 0$. b determines how much excess capacity would be allocated when the queueing delay is zero. a and b also determine the slope of the hyperbola, or, in other words, the rate at which $f(T_q)$ drops as a function of queueing delay. Larger values of a and b make the scheme very sensitive to the queueing delay, whereas, smaller values increase the time required to reach the desired operating point.

The parameter b is typically smaller than a. b determines the amount of over-allocation required to reach the target delay $T0$ quickly in the steady state. Any

small over-allocation above 100% of ABR capacity is sufficient for this purpose. The parameter a primarily determines how quickly the function $f(T_q)$ drops as a function of queueing delay. a should not be very different from b because, this can result in widely different allocations when the delay slightly differs from $T0$. At the same time, a should be high enough control the queues quickly.

Through simulation, we found that the values 1.15 and 1.05 for a and b respectively work well for all the workloads we have experimented with. Hence, at zero queues, we over-allocate up to 5% excess capacity to get the queues up to $Q0$. Higher values of b would allow sources to overload to a higher extent. This can aggravate transient overloads and result in higher queue spikes. Using a value of 1 for b is also acceptable, but the "pocket" of queues builds up very slowly in this case. A value of 1 for b is preferable when the variance is high. Further, these parameters values for a and b are relatively independent of $T0$ or $QDLF$. Given these values for a and b, the function depends primarily on the choice of $T0$ and $QDLF$ as discussed below.

6.21.2 Target Queueing Delay $T0$

When the function $f(T_q)$ is one of the two hyperbolas, its slope ($\frac{df}{dq}$) is inversely proportional to the parameter $T0$. For a constant value of a, larger $T0$ reduces the slope of the function, and hence its effectiveness. The queueing delay required to reduce the ABR capacity by a fixed fraction is directly proportional to $T0$. It is also directly proportional to the ABR capacity. Hence, if the ABR capacity is high (as is the case in OC-3 and higher speed networks), the queues need to build up to a large value before the drain capacity is sufficient. Hence, the maximum value of $T0$ depends upon and how fast the transient queues need to be cleared.

The maximum value of $T0$ also depends on the buffer size at the switch, and must be set to allow the control of the queues before the buffer limit is reached. One strategy is to keep the buffer size at least the sum of the feedback delay and $8 \times T0$ (assuming a = 1.15 and $QDLF = 0.5$, and ABR capacity is constant, and other factors like measurement interval length are negligible). One feedback delay is enough for the feedback to reach the sources and $8 \times T0$ is enough for the function to reach $QDLF$. For other values of $QDLF$, the recommended buffer size is:

$$\frac{(a - QDLF) \times T0}{[(a-1) \times QDLF]}$$

The maximum value of $T0$ can be calculated reversing the above formula, given the buffer size.

$$T0 = \frac{[(a-1) \times QDLF]}{(a - QDLF)}$$

A minimum value of $T0$ is also desired for stable operation. If $T0$ is very small, the function $f(T_q)$ can traverse the range $[QDLF, b]$ in a time $\frac{(a-QDLF)\times T0}{[(a-1)\times QDLF]}$, assuming that capacity is constant over this period of time. This time can be shorter than the feedback delay, and lead to undesired oscillations in rates and queues. This is because the function changes from b to $QDLF$ before feedback is effective. Such a behavior is undesired because, the scheme now is very sensitive to the changes in queue length. Recall that queue length is only a secondary metric, i.e., we want the input rate and not the queue length to be the primary metric of congestion. Further, the minimum $T0$ is at least the "pocket" of queues desired. For WANs, $T0$ is at least $\frac{[(a-1)\times QDLF]}{(a-QDLF)}$ of the feedback delay, which is 1/8, assuming $a = 1.15$, $QDLF = 0.5$. For LANs, we set $T0$ to at least one feedback delay, to reduce the sensitivity of the ABR capacity to small queue lengths. In cases of high variation and measurement errors, the "pocket"

of queues may not be achievable. High throughput is the goal in this case, and $T0$ should be set close to the minimum value to allow queues to be quickly drained.

6.21.3 Queue Drain Limit Factor $QDLF$

$QDLF$ ensures that there is enough capacity to drain out the transient queues. We recommend a value of 0.5 for WAN switches and 0.8 for LAN switches.

WAN switches need to have greater drain capacity because of the longer feedback delays of its VCs and consequently longer response times to transient overloads. If the fluctuations in load or capacity are of a time-scale much smaller than the feedback delay, the rate allocations using a high target rate may not be sufficient. Transient queues may build up in such cases unless there is sufficient capacity allocated to drain the queues. An example of such high variation workload is TCP traffic combined with a VBR load which has an ON-OFF period of 1 ms, whereas the feedback delay is 10 ms.

However, for LAN switches which can receive feedback rapidly, and $T0$ is small, the function can move quickly through the range $[QDLF, b]$. Given these conditions, a large drain capacity is not required, since large queues never build up. For such configurations, $QDLF$ can have higher values like 0.8.

Since the $QDLF$ parameter defines the lower bound of the function $f(T_q)$, we should ensure that this value is reached only for large queue values. This can be achieved by choosing small values for a, or large values for $T0$. Since large values of $T0$ reduce the effectiveness of the function $f(T_q)$, the parameter a is chosen small. This is another factor in the choice of a. It turns out that the recommended value for a (1.15) is small enough for the $QDLF$ values recommended.

6.22 Performance Evaluation of the ERICA and ERICA+ Schemes

In this section, we shall describe the methodical performance evaluation of the ERICA scheme, and provides simple benchmarks to test the performance of different ATM switch algorithms. We use the principles discussed in chapter 3 to test ERICA for various configurations, source models and background traffic patterns.

We present the set of experiments conducted to ensure that ERICA meets all the requirements of switch algorithms. In the cases where the original algorithm failed to meet the requirements and an enhancement to the algorithm was deemed necessary, the performance of the basic algorithm is compared to the performance of the enhanced algorithm, and a discussion of why the enhancement was needed is presented.We prefer to use simple configurations when applicable because they are more instructive in finding problems [49]. The results are presented in the form of four graphs for each configuration:

1. Graph of allowed cell rate (ACR) in Mbps over time for each source
2. Graph of ABR queue lengths in cells over time at each switch
3. Graph of link utilization (as a percentage) over time for each link
4. Graph of number of cells received at the destination over time for each destination

We will examine the efficiency and delay requirements, the fairness of the scheme, its transient and steady state performance, and finally its adaptation to variable capacity and various source traffic models. The experiments will also be selected

such that they have varying distances and number of connections to examine the scalability requirement.

6.22.1 Parameter Settings

Throughout our experiments, the following parameter values are used:

1. All links have a bandwidth of 155.52 Mbps.

2. All LAN links are 1 Km long and all WAN links are 1000 Km long.

3. All VCs are bidirectional.

4. The source parameter Rate Increase Factor (RIF) is set to one, to allow immediate use of the full Explicit Rate indicated in the returning RM cells at the source.

5. The source parameter Transient Buffer Exposure (TBE) is set to large values to prevent rate decreases due to the triggering of the source open-loop congestion control mechanism. This was done to isolate the rate reductions due to the switch congestion control from the rate reductions due to TBE.

6. The switch target utilization parameter was set at 95% for LAN simulations and at 90% for WAN simulations.

7. The switch averaging interval was set to the minimum of the time to receive 50 cells and 1 ms for LAN simulations, and to the minimum of the time to receive 100 cells and 1 ms for WAN simulations.

8. The ERICA+ parameters are set as follows. The parameters a and b (intercepts of the two hyperbolas in the queueing delay function) are set to 1.15 and 1.05

respectively. The target delay $T0$ is set at 100 microseconds for LAN simulations and 500 microseconds for WAN simulations, and the queue drain limit factor ($QDLF$) is set at 0.5.

9. All sources, including VBR sources are deterministic, i.e., their start/stop times and their transmission rates are known. The bursty traffic sources send data in bursts, where each burst starts after a request has been received from the client.

6.22.2 Efficiency

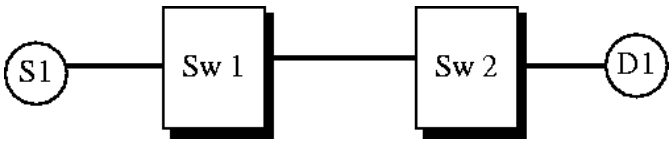


Figure 6.7: One source configuration

The very first test to verify efficient operation is to use a single source configuration as shown in figure 6.7. A scheme that does not work for this simple configuration is not worth further analysis. The source is active over the entire simulation period. Figure 6.11 illustrates that ERICA achieves the required efficiency, since the source rate rises to almost fully utilize the link. Observe that there are no rate oscillations in the steady state, and that utilization is at the target utilization goal (95%).

The same configuration has also been simulated to examine the efficiency of ERICA+. As seen in figure 6.12, the source rate rises to fully utilize the link (100%

utilization) with no oscillations and minimal queues. Please note that the simulation graphs, though introduced periodically, are at the end of the chapter.

6.22.3 Minimal Delay and Queue Lengths

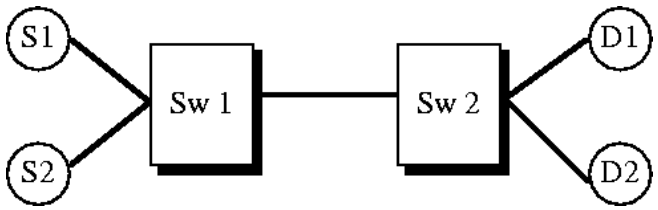


Figure 6.8: Two source configuration

To test for minimal delays and short queue lengths, we use a multiple source configuration. The simplest such configuration is the two source configuration, where two sources share a link as illustrated in figure 6.8. Each source must converge to almost half of the link rate (1/2 × Target Utilization), which is the max-min optimal allocation.

Figure 6.13 shows that the convergence is fast, the queue lengths are small (hence, the delay is minimal) and steady state performance is good. For ERICA+, the two sources rapidly converge to their optimal rates as seen in figure 6.14, and the queue length rises to reach the limit corresponding to its target delay parameter (100 microseconds corresponds to approximately 30 cells at 155.52 Mbps). There is a slight rate oscillation seen in figure 6.14(a) to allow the queues to reach the target value, but the steady state has no rate oscillations and 100% link utilization (figure 6.14(c)).

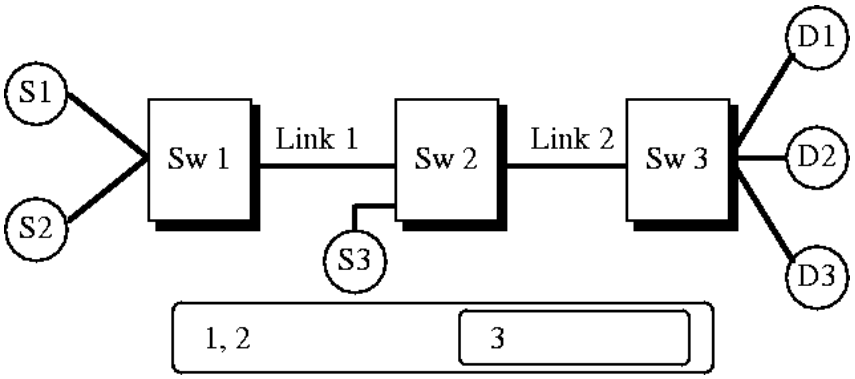


Figure 6.9: Parking lot configuration

6.22.4 Fairness

Two configurations are used for studying the fairness of the scheme: the parking lot configuration and the upstream configuration. The parking lot configuration and its name were derived from theatre parking lots, which consist of several parking areas connected via a single exit path. At the end of the show, congestion occurs as cars exiting from each parking area try to join the main exit stream.

For computer networks, an n-stage parking lot configuration consists of n switches connected in series. There are n VCs. The first VC starts from the first switch and goes through all the remaining switches. For the remaining VCs, the i^{th} VC starts at the $i-1^{st}$ switch. The link between the last two switches is the bottleneck link. The max-min allocation for each VC is $1/n$ of the bandwidth. A 3-switch parking lot configuration is shown in figure 6.9. Figure 6.15 illustrates that ERICA achieves the desired max-min allocation.

Although the parking lot configuration had been believed to test fairness, we discovered that it is not sufficient to demonstrate max-min fairness, and a more

stringent test is required. We had observed that the original ERICA algorithm does not converge to max-min fairness in certain situations. Such situations arise when the ERICA algorithm is executed in a state where some of the sources cannot fully utilize their allocated bandwidth on a link (for example, because they are bottlenecked on another link), and the rest of the sources contending for bandwidth have unequal CCR values, which are greater than the fair share value (first term in the maximum formula). The ERICA algorithm does not converge to max-min fairness in these situations because, after z converges to one, the second term in the maximum formula becomes $CCR_i/1 = CCR_i$, and the first term is constant. The maximum of the two terms for the contending sources is the second term, because there are sources that are not fully utilizing their allocated bandwidth. Hence, the sources do not change their rates.

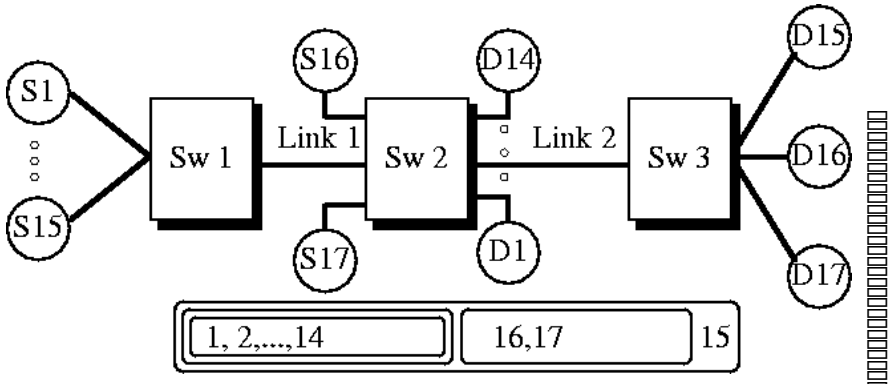


Figure 6.10: Upstream Configuration

An example of this situation can be illustrated by an upstream configuration (see figure 6.10). The upstream configuration consists of three switches and 17 VCs. The second link is shared by VC_{15}, VC_{16}, and VC_{17}. Because there are 15 VCs on the

first link, VC_{15} is limited to a throughput of less than 1/15 the link rate. VC_{16} and VC_{17} should, therefore, each converge to a little less than 7/15 of the second link rate. The configuration is called an upstream configuration because the bottleneck link is the first link (upstream link). A WAN configuration (1000 Km links) is used in this situation to illustrate the scalability of ERICA to long distances.

Figure 6.17 shows that the original ERICA algorithm was unfair in this situation, and figure 6.18 shows that ERICA, after the modification discussed in section 6.2, is fair. As seen in figure 6.18, the modified algorithm converges to max-min allocations. Regardless of the initial load factor value, after a short transient period, all sources contending for bandwidth are allocated equal rates, and the two curves in figure 6.18(b) (number of cells received at the destination) have the same slope (compare with figure 6.17(b)). The transient response is slightly worse than the original ERICA algorithm due to the temporary over-allocation needed to equalize the shares, but the steady state performance is as good as with the original ERICA algorithm.

6.22.5 Transient and Steady State Performance

To test the transient response of the system, we use a modified two source configuration. The configuration is similar to the two source configuration because two sources share the same link, but one of the sources is only active from 10 ms to 20 ms while the other source is active throughout. Besides illustrating the transient response of the system, this configuration also illustrates the effect of the "fairshare first" algorithm discussed in section 6.3. That algorithm (see section 6.3) prevents a low rate VC to rise above $FairShare$. This VC takes an extra round trip compared to the basic ERICA because it first comes to $FairShare$ before rising further. The

switch can use the extra round trip to give feedback to all the sources, measure a new load factor and reduce overloading sources. The modification reduces the maximum queues in transient situations.

Figures 6.19 and 6.20 illustrate the effect of the "fairshare first" modification on a transient configuration in a LAN. Figure 6.19 shows the transient performance of ERICA without the "fairshare first" modification, and figure 6.20 illustrates how ERICA with the "fairshare first" modification avoids transient overloads. It is clear that ERICA exhibits good transient response characteristics to changing load, and the modification mitigates sudden overloads, constraining the queue length when the second source starts transmission. The figure also shows that the steady state performance of the scheme is excellent, as there are minimal oscillations in the rates of the sources.

6.22.6 Adaptation to Variable ABR Capacity

Constant Bit Rate (CBR) and Variable Bit Rate (VBR) service classes have a higher priority than the ABR service. In cases of VBR traffic, the ABR capacity becomes a variable quantity.

The two source configuration in a wide area network is used to demonstrate the behavior of ERICA in the presence of VBR sources. A deterministic VBR source is used whose peak rate is 124.42 Mbps (80% of the link capacity). Figure 6.21 illustrates the behavior of ERICA on a WAN where the VBR source was active for alternating periods of 1 ms with 1 ms inactive periods in between (high frequency VBR), while figure 6.22 shows the performance with VBR on/off periods of 20 ms (low frequency VBR).

From the figures, it is clear that ERICA rapidly detects the change in the available ABR capacity and gives the appropriate feedback to the sources. When the VBR source is active, the ABR sources rapidly reduce their rates (figures 6.21(a) and 6.22(a)). The utilization is generally high; the utilization drops reflect the time taken for the feedback to reach the sources: the feedback delay (figures 6.21(c) and 6.22(c)). The spikes in the queue lengths seen in figures 6.21(b) and 6.22(b) also reflect the feedback delay, but the queues are rapidly drained. Observe that the number of cells received in both cases (figures 6.21(d) and 6.22(d)) is approximately equal, which shows that the performance is approximately the same. The throughput can be calculated from the graphs showing the number of cells received at the destination. It is clear that the throughput is high, indicating a high link utilization.

Figure 6.23 illustrates how ERICA+ adapts to high frequency VBR in the background, and figure 6.24 shows its performance with low frequency VBR. ERICA+ adapts rapidly to the changing background traffic, recomputing the available bandwidth and the rate allocations. The link utilization is higher than that with ERICA, and the queue lengths are constrained. The target queue goal is never reached due to the high variation, but the utilization goal is partially reached.

6.22.7 Adaptation to Various Source Traffic Models

In all the previous experiments, the ABR sources are assumed to be persistent sources, which means that they always have data to send, and can utilize their full capacity at all times. It is essential to examine the performance of the scheme with bursty sources which alternate between active periods when they utilize their full capacity, and idle periods when they do not send any data. In addition, situations

where sources are bottlenecked are also of particular interest. The scheme should be able to rapidly react to the overload that can arise if the bottlenecked sources suddenly start utilizing their full capacity.

6.22.8 Bursty Traffic

Figures 6.25 through 6.29 illustrate the performance of ERICA in a wide area network two source configuration where one of sources is a persistent (greedy or infinite) source, while the other connection is a request-response type connection. The request-response connection consists of a source sending a request (of size 16 cells), and the destination responding with a burst of data. Two different burst sizes are used in our simulations: small bursts are 128 cells, and large bursts are 6144 cells. Upon the receipt of the response at the source, and after a certain period of time, the source sends another request for data, and the cycle is repeated (see chapter 7). The figures show the performance of the reverse (response) connection where a burst of data is sent in response to every request.

Figure 6.25 illustrates the performance of ERICA with small response burst sizes, figure 6.26 shows the effect of medium burst sizes, while figure 6.27 illustrates the effect of large burst sizes. As seen in the figures, ERICA can adapt to small and medium bursts of data, and the queue lengths are constrained. However, with a target utilization of 90%, ERICA does not have enough capacity to drain large bursts of data from the switch queues before the next burst is received. This problem can be solved by using smaller values for the target utilization parameter.

Figure 6.28 shows that bi-directional counting of the number of active sources (as discussed in section 6.8) limits the queue sizes for large bursts. This is because it

counts the bursty source as active if its RM cells are traveling in the reverse direction, even though it might not be sending any data in the forward direction during its idle periods. This situation is called the "out-of-phase" effect, and is also a common problem with TCP sources. The problem affects the load measurement, as well as the measurement of the number of active VCs. As seen in figure6.28(b), the queue lengths are constrained, and the problem seen in figure 6.27(b) has been solved, even for a target utilization of 90%.

Another method to limit the queue sizes in this case is by averaging the number of active sources as discussed in section6.9. As previously explained, we should account for the presence of a source, even though it might be currently idle. The effect of averaging the value of the number of active sources is illustrated in figure 6.29. The bidirectional counting option is not used in this case. Figure 6.29(b) shows that the queue length is constrained.

Figure6.30, 6.31 and 6.32 illustrates the performance of ERICA+ for the same configuration without the averaging the number of sources option. The figure illustrates the effect of large burst sizes. It is clear that ERICA+ can adapt to bursty traffic better than ERICA, because it accounts for the time to drain the queues when estimating the available capacity. Even with large burst sizes, the queues built up when the bursty source is active can drain before the next burst arrives at the switch. The bidirectional counting and the averaging of number of sources options are not necessary in this case.

In cases of many sources running TCP on ABR in the presence of high frequency VBR background, ERICA+ sometimes fails to drain the queues when the averaging interval parameter is set to very small values. This phenomenon is explained in

depth in chapter 8. Averaging the number of sources and averaging the load factor as explained earlier in this chapter can also alleviate this problem.

6.22.9 ACR Retention

ACR retention is the problem which occurs when sources are not able to fully use their rate allocations. For example, the input to the ATM end-system can be steady, but have a rate lower than its ABR allocation (allowed cell rate). Another example is an end-system which supports multiple VCs (to possibly different destinations) on a single outgoing link. A VC may not be able to use its ACR allocation because the outgoing link is running at capacity. In such situations, the switches reallocate the unused capacity to the other sources which are unconstrained. However, if the ACR retaining sources suddenly use their allocations, a potential overload situation exists.

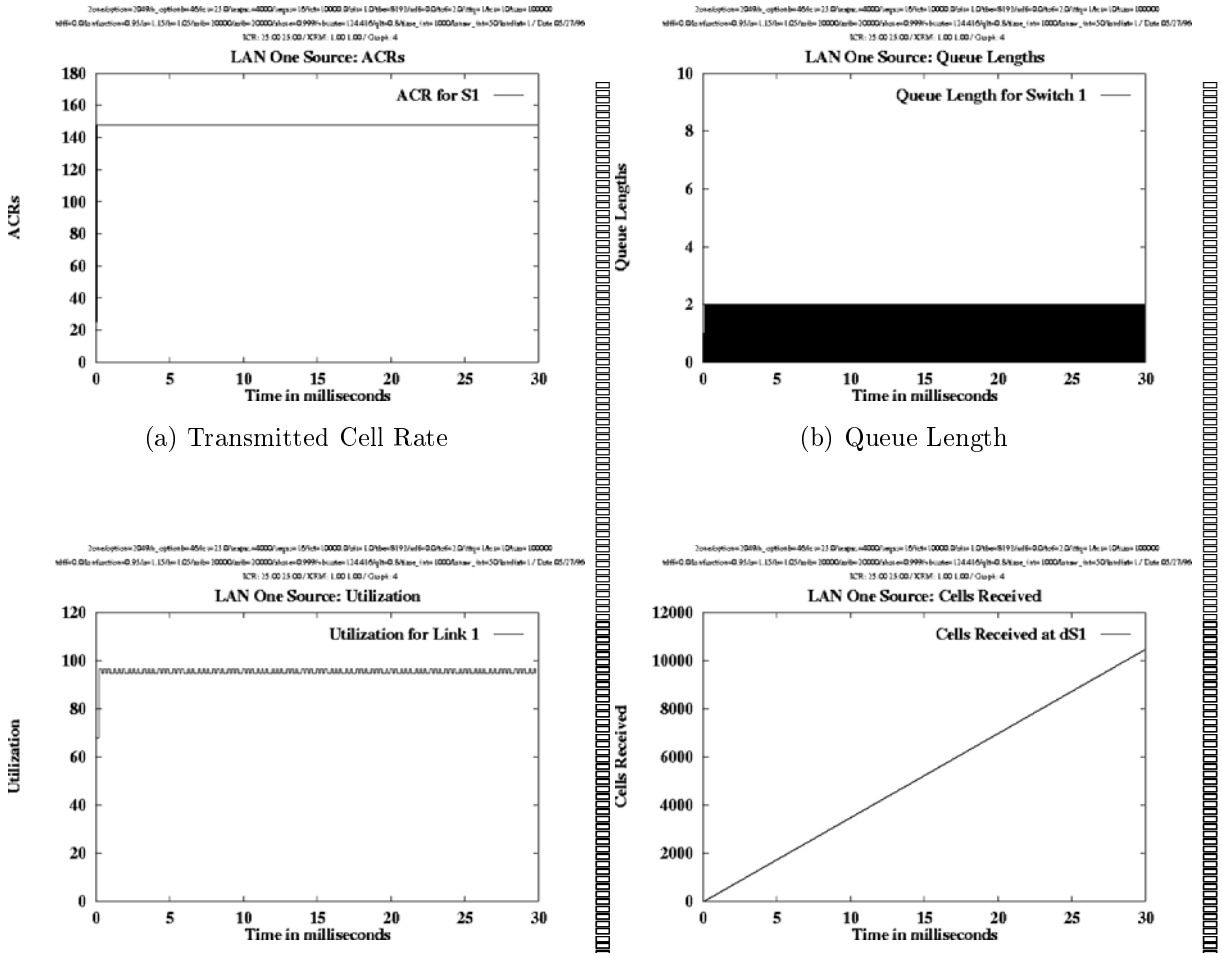
Figure 6.33 illustrates the performance of ERICA when there are ten VCs sharing a link. This larger number of connections has been selected to demonstrate the scalability of ERICA to more VCs, as well as to aggravate the problem of ACR retention. Initially, the ten sources are retaining their ACRs, and each cannot send at a rate of more than 10 Mbps. After 100 ms, all the sources suddenly start sending at their full allocations. ERICA rapidly detects the overload and gives the appropriate feedback asking sources to decrease their rates. All the ten sources stabilize at their optimal rates after that.

Figure 6.34 shows how the per-VC CCR measurement option can mitigate the overload situation arising when all the ACR retaining sources start transmission at their full capacities. The per-VC CCR measurement results in more conservative initial allocations, and hence smaller queues in this case.

6.23 Summary of the ERICA and ERICA+ Schemes

This chapter has examined the ERICA and ERICA+ schemes, explicit rate indication schemes for congestion avoidance in ATM networks, and explained several extensions and enhancements of the scheme. The scheme entails that the switches periodically monitor their load on each link and determine a load factor, the available capacity, and the number of currently active virtual channels. This information is used to calculate a fair and efficient allocation of the available bandwidth to all contending sources. The algorithm exhibits a fast transient response, and achives high utilization and short delays, in addition to adapting to high variation in the capacity and demand.

Based on the discussion of requirements of switch algorithms in chapter 3 we have examined how each of these requirements can be tested. Using these techniques, we presented a comprehensive performance evaluation of the ERICA switch algorithm and demonstrated the effect of several features and options of the algorithm. We have examined the efficiency and delay requirements, the fairness of the scheme, its transient and steady state performance, its scalability, and its adaptation to variable capacity and various source traffic models. Simulation results have illustrated that the algorithm gives optimal allocations, and rapidly adapts to load and capacity changes. The performance of the algorithm was examined for various configurations, source models and background traffic patterns.



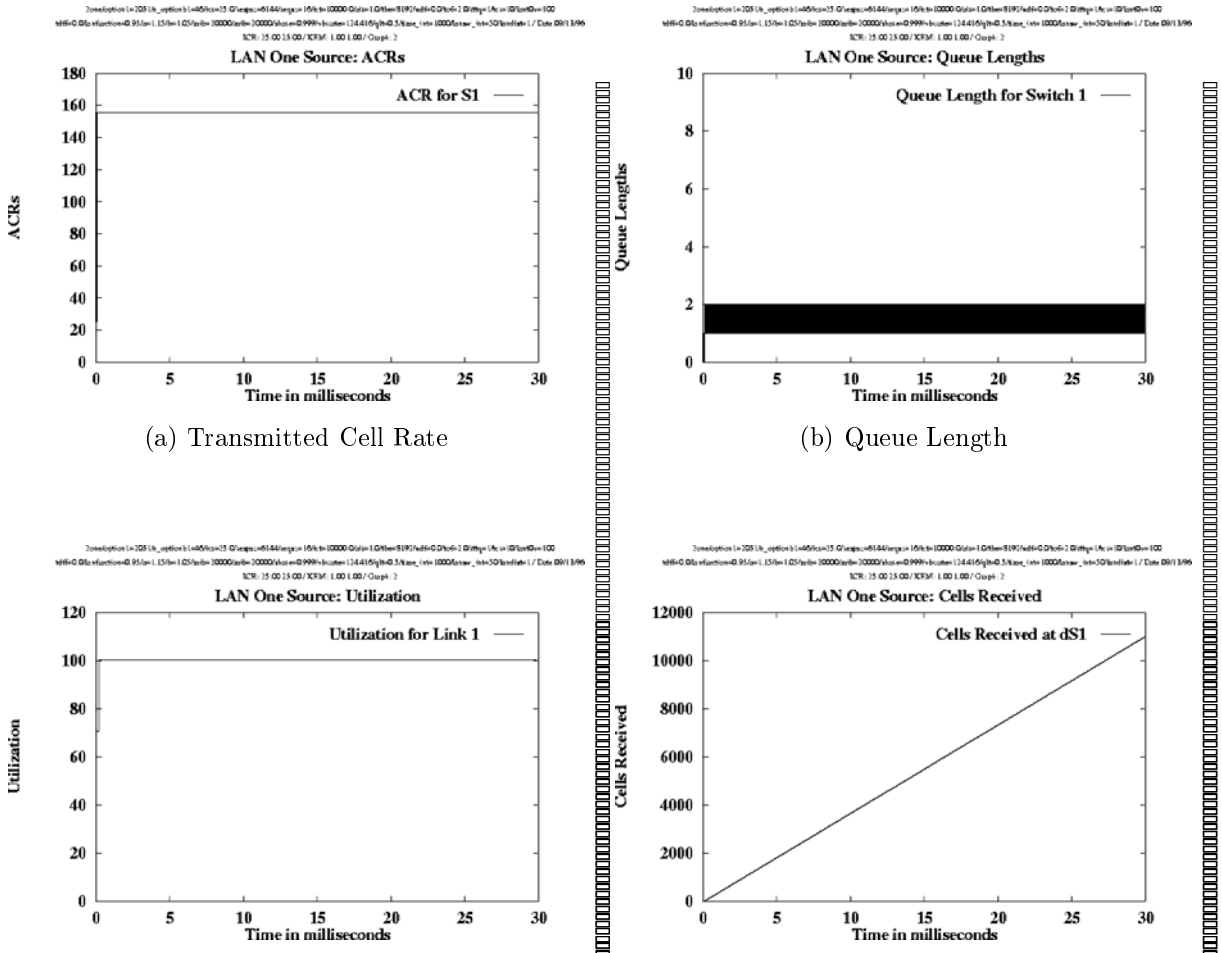
(a) Transmitted Cell Rate

(b) Queue Length

(c) Link Utilization

(d) Cells Received

Figure 6.11: Results for a one source configuration in a LAN (ERICA)



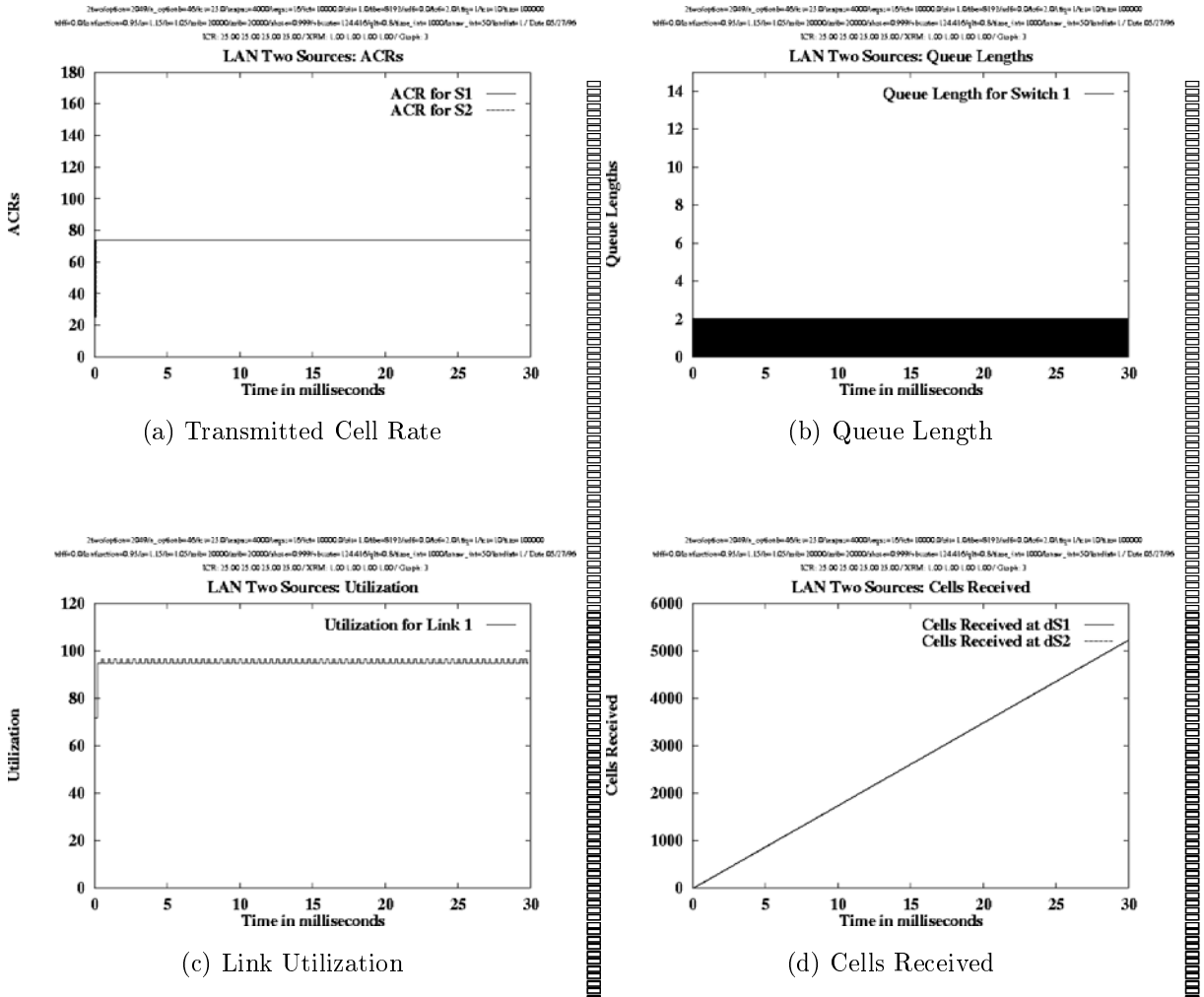
(a) Transmitted Cell Rate

(b) Queue Length

(c) Link Utilization

(d) Cells Received

Figure 6.12: Results for a one source configuration in a LAN (ERICA+)



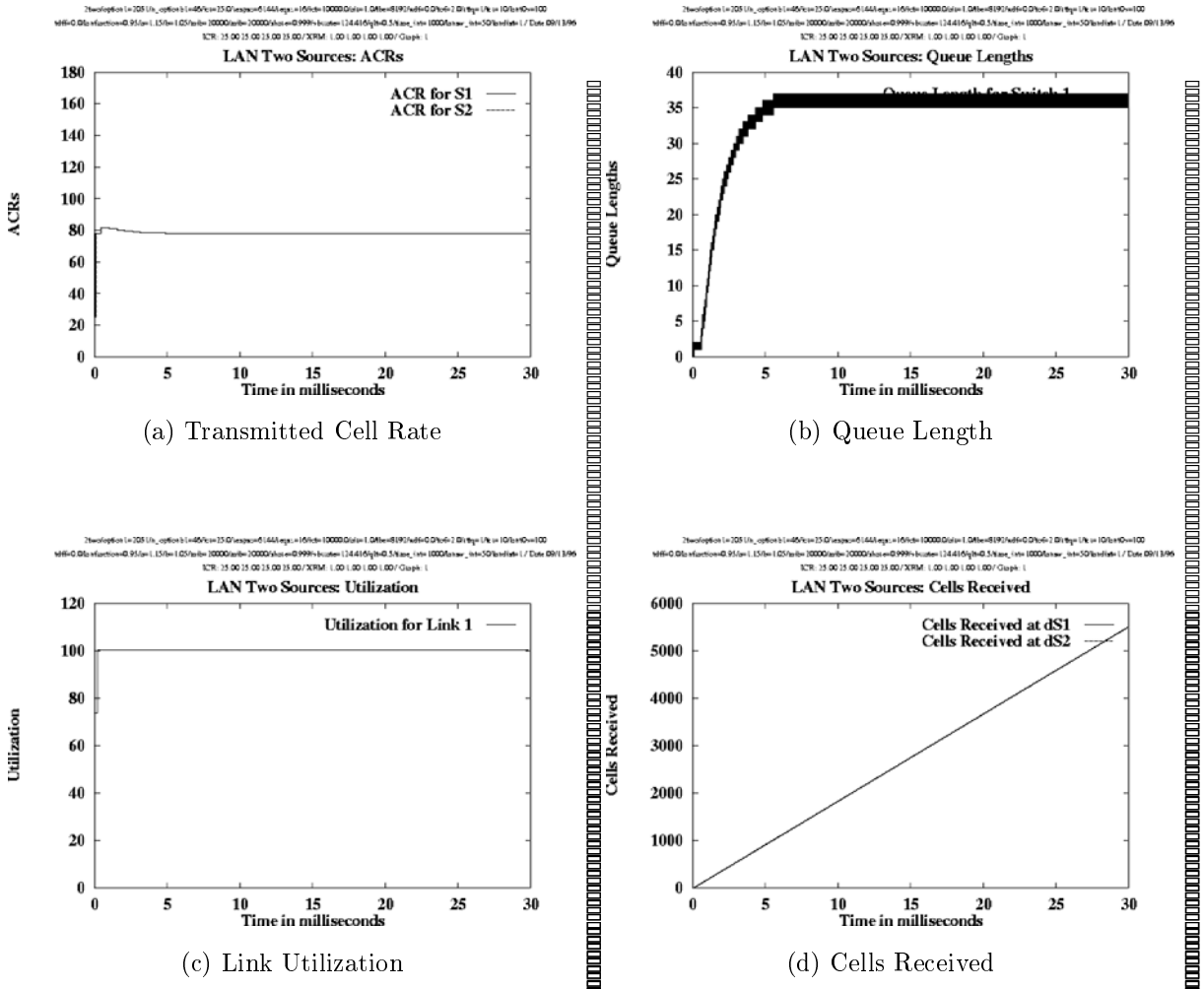
(a) Transmitted Cell Rate

(b) Queue Length

(c) Link Utilization

(d) Cells Received

Figure 6.13: Results for a two sources configuration in a LAN (ERICA)



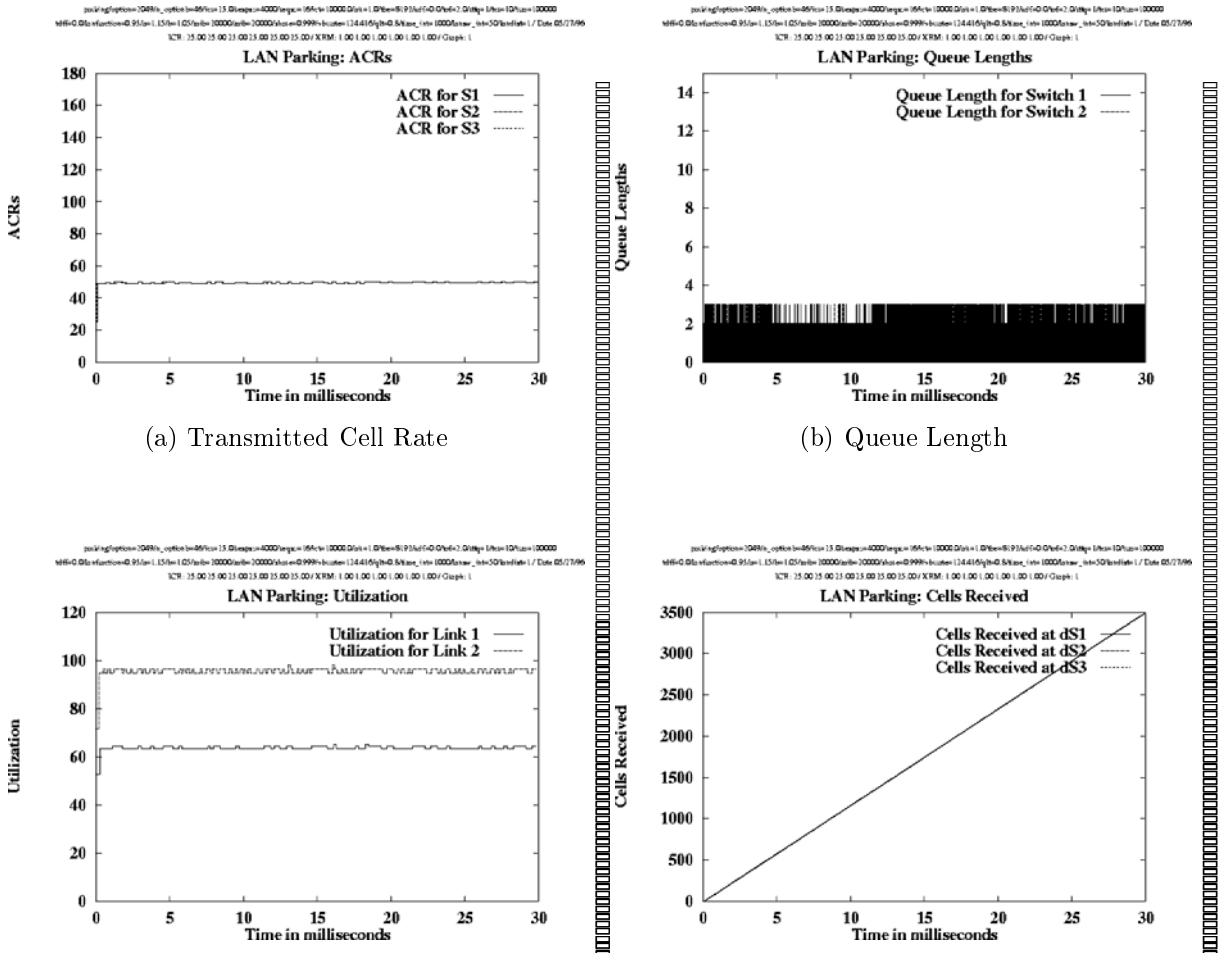
(a) Transmitted Cell Rate

(b) Queue Length

(c) Link Utilization

(d) Cells Received

Figure 6.14: Results for a two sources configuration in a LAN (ERICA+)



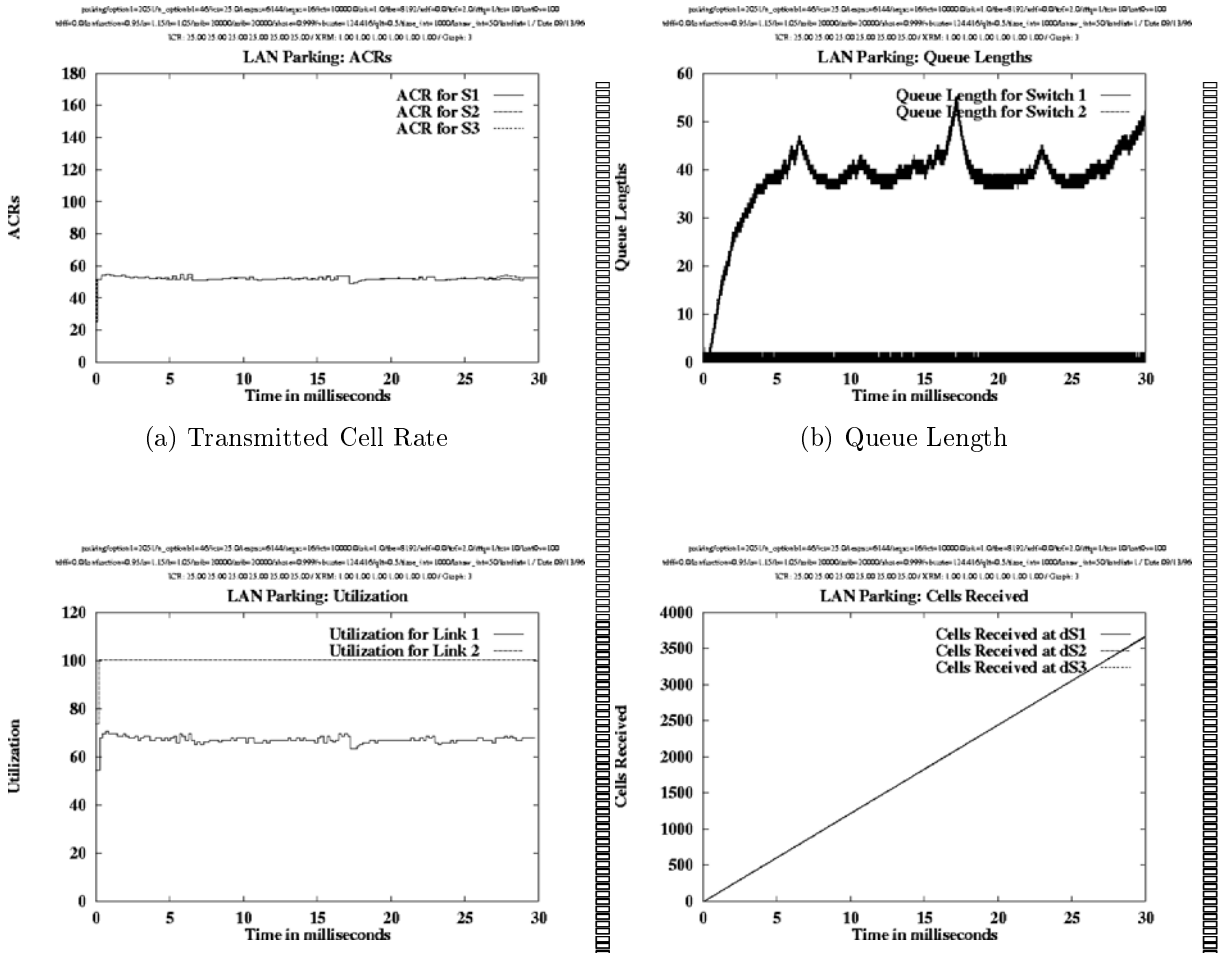
(a) Transmitted Cell Rate

(b) Queue Length

(c) Link Utilization

(d) Cells Received

Figure 6.15: Results for a parking lot configuration in a LAN (ERICA)



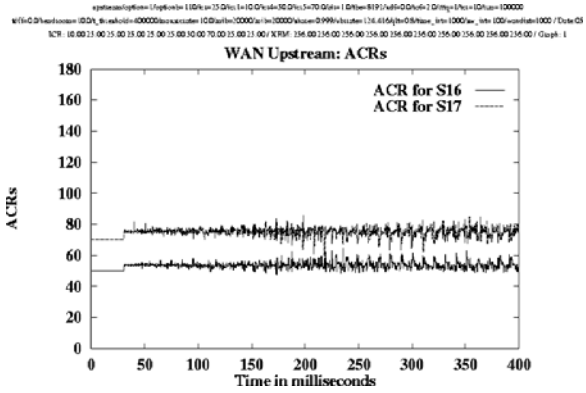
(a) Transmitted Cell Rate

(b) Queue Length

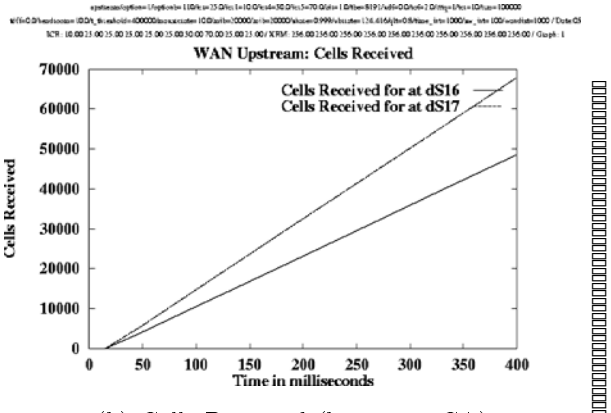
(c) Link Utilization

(d) Cells Received

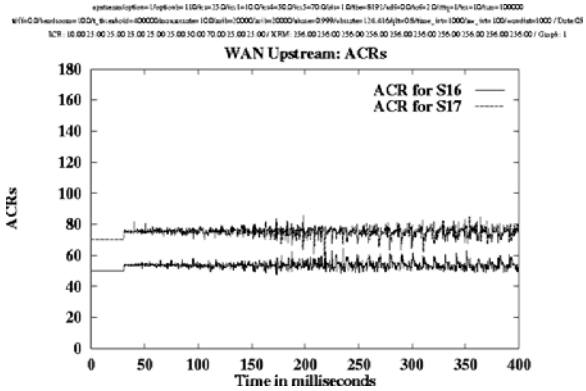
Figure 6.16: Results for a parking lot configuration in a LAN (ERICA+)



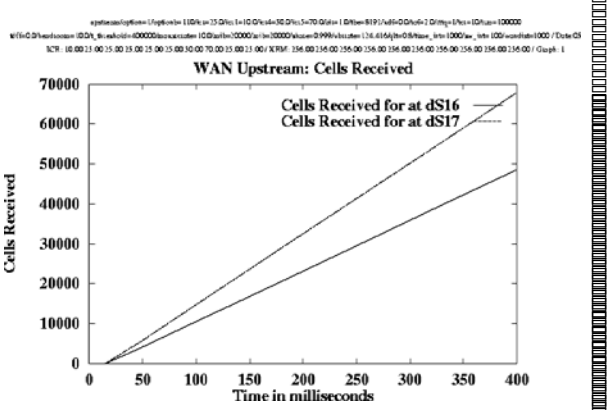
(a) Transmitted Cell Rate (basic ERICA)



(b) Cells Received (basic ERICA)

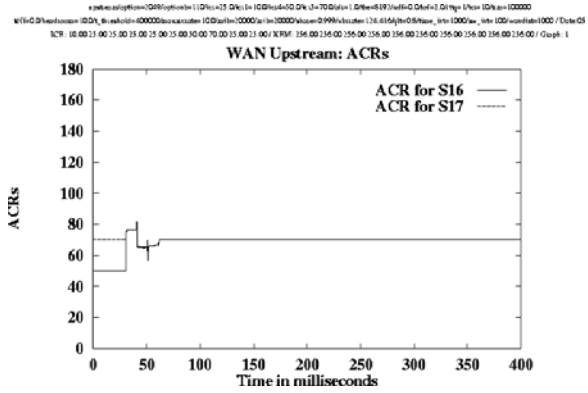


(c) Transmitted Cell Rate

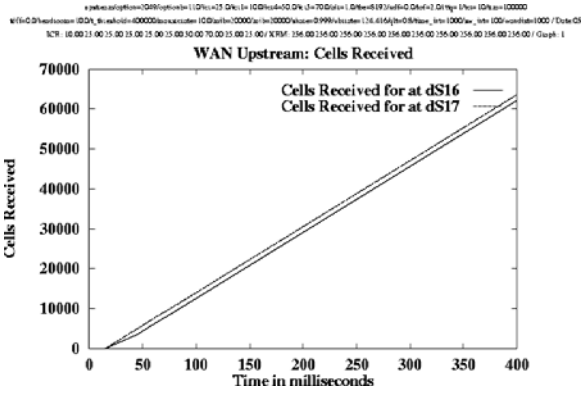


(d) Cells Received

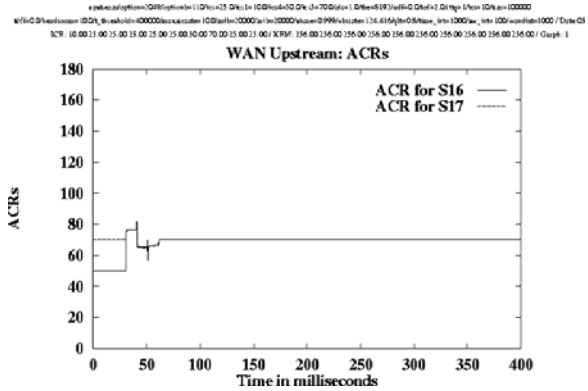
Figure 6.17: Results for an upstream configuration in a WAN (ERICA without the max-min fairness enhancement)



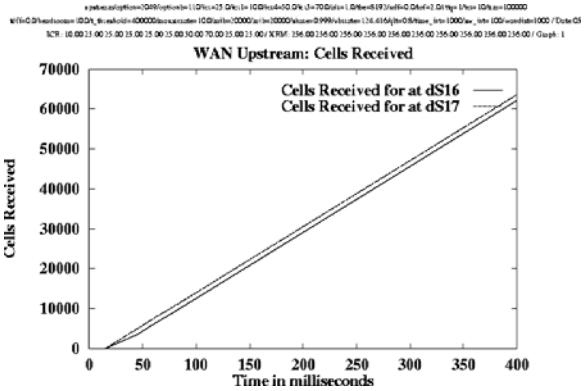
(a) Transmitted Cell Rate (basic ERICA)



(b) Cells Received (basic ERICA)

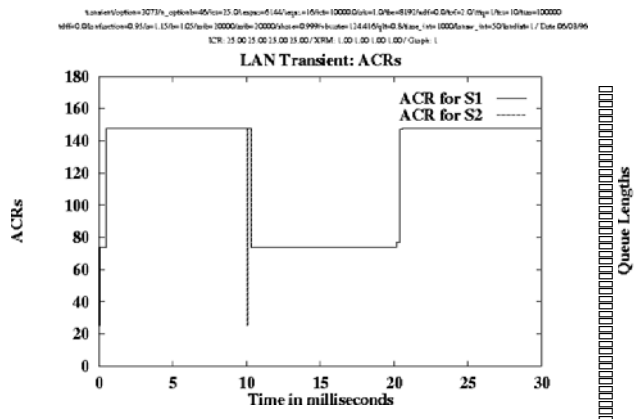


(c) Transmitted Cell Rate

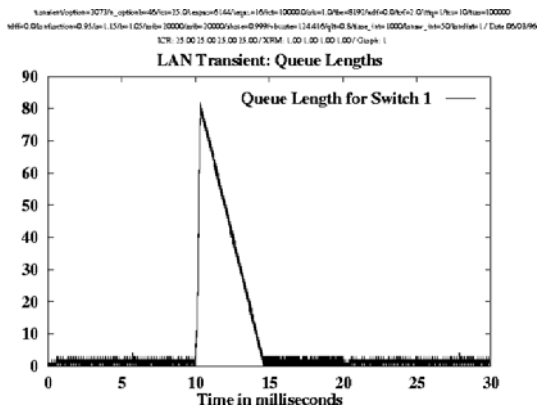


(d) Cells Received

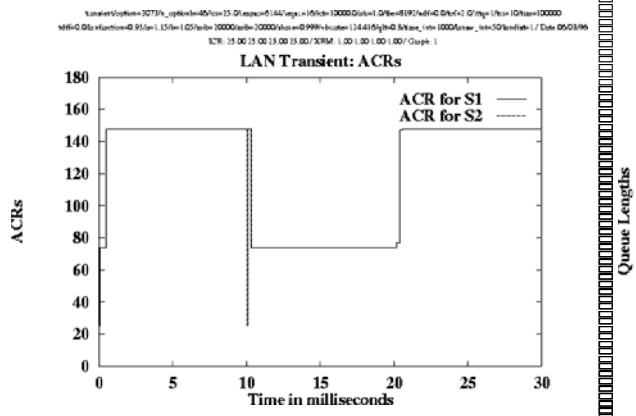
Figure 6.18: Results for an upstream configuration in a WAN (ERICA with the max-min fairness enhancement)



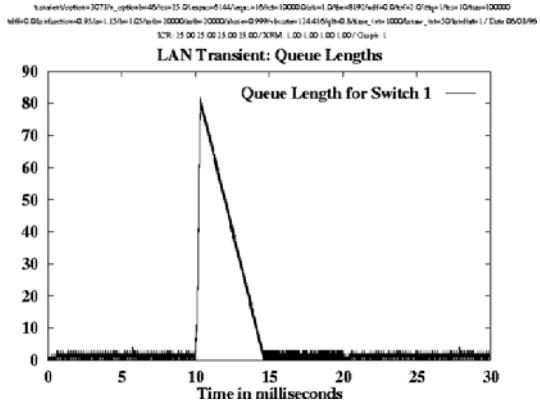
(a) Transmitted Cell Rate (basic ERICA)



(b) Queue Length (basic ERICA)

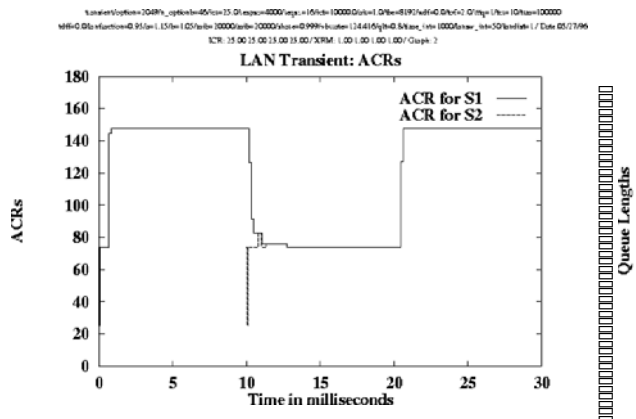


(c) Transmitted Cell Rate

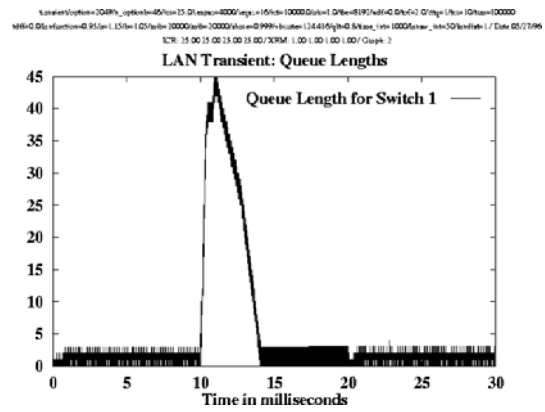


(d) Queue Length

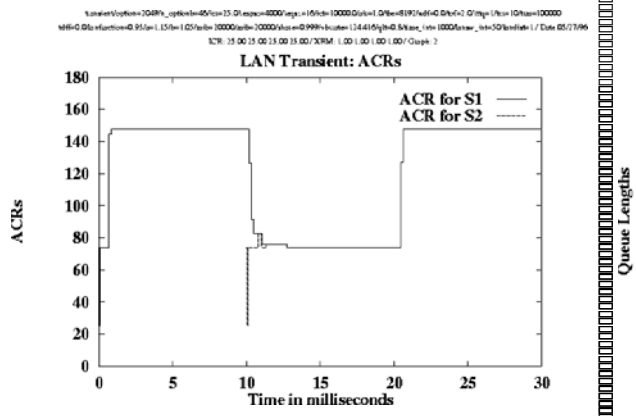
Figure 6.19: Results for a transient source configuration in a LAN (ERICA without the "fairshare first" enhancement)



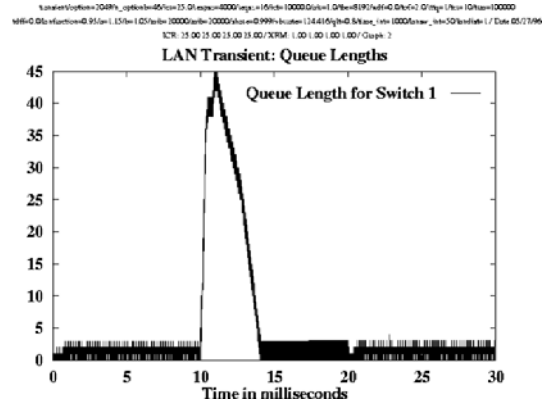
(a) Transmitted Cell Rate (basic ERICA)



(b) Queue Length (basic ERICA)

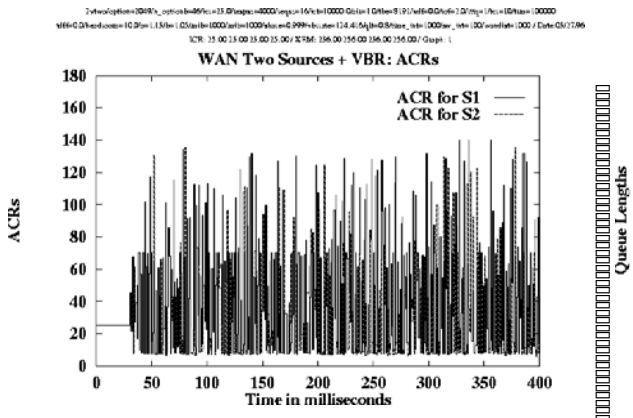


(c) Transmitted Cell Rate

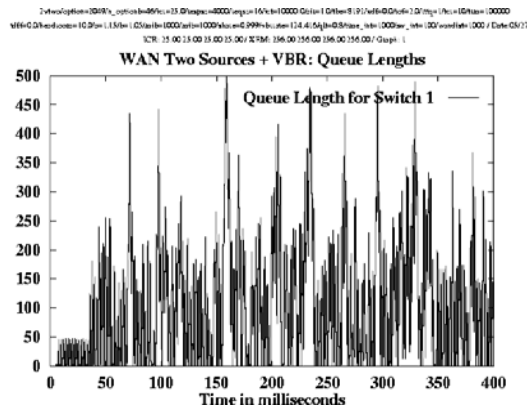


(d) Queue Length

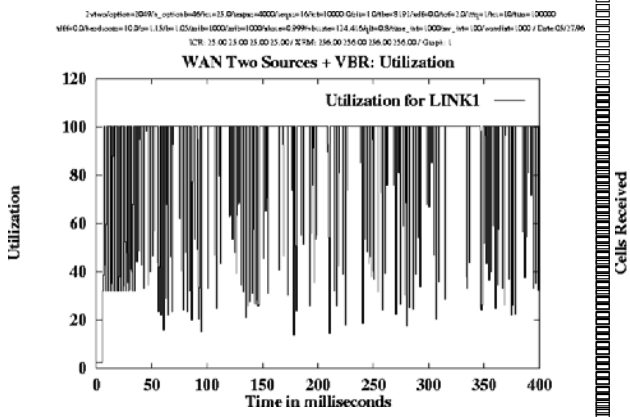
Figure 6.20: Results for a transient source configuration in a LAN (ERICA with the "fairshare first" enhancement)



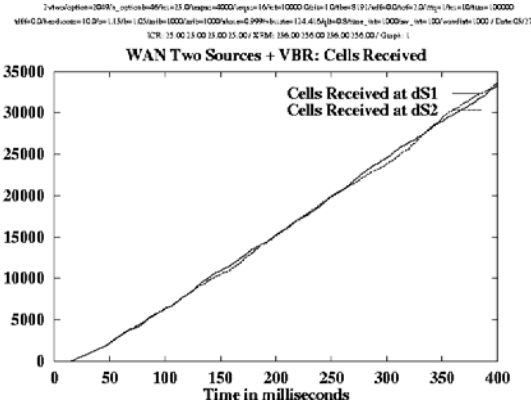
(a) Transmitted Cell Rate



(b) Queue Length

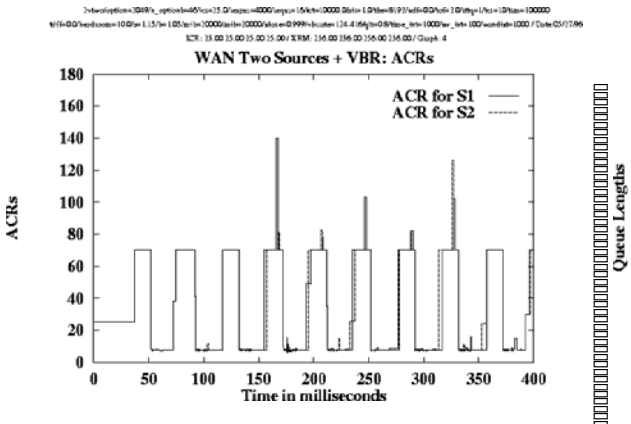


(c) Link Utilization

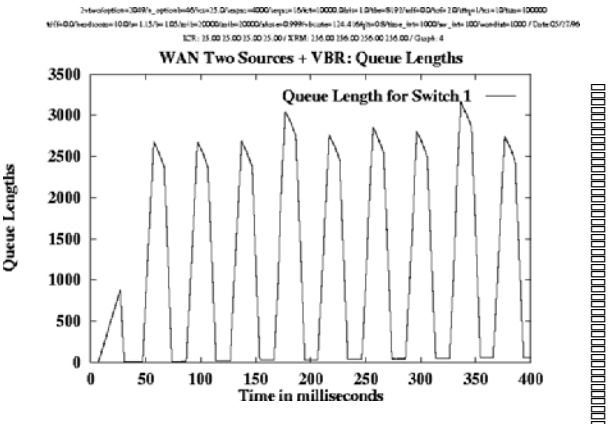


(d) Cells Received

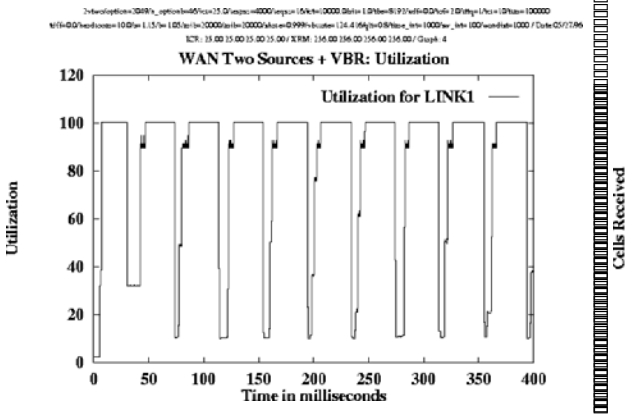
Figure 6.21: Results for a two source configuration with (1 ms on/1 ms off) VBR background in a WAN (ERICA)



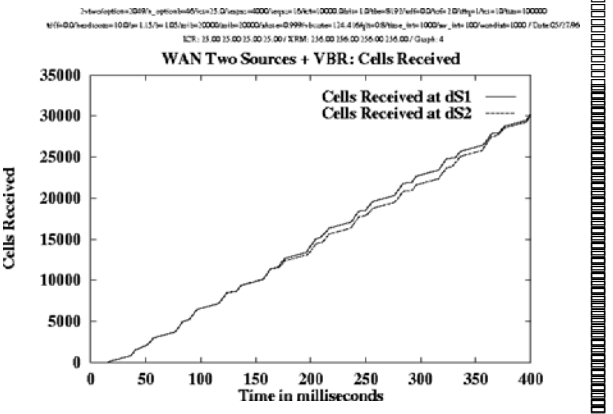
(a) Transmitted Cell Rate



(b) Queue Length

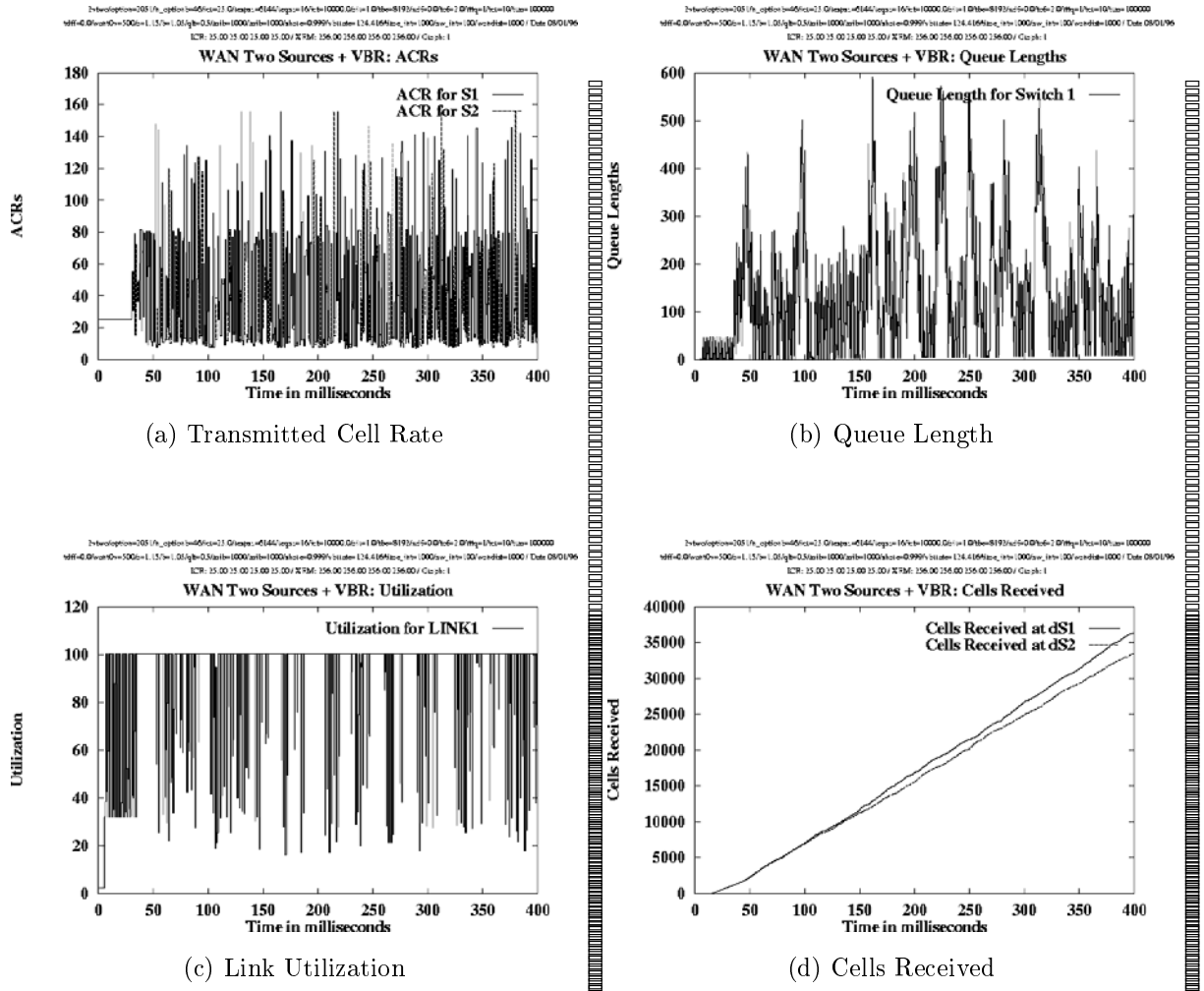


(c) Link Utilization



(d) Cells Received

Figure 6.22: Results for a two source configuration with (20 ms on/20 ms off) VBR background in a WAN (ERICA)



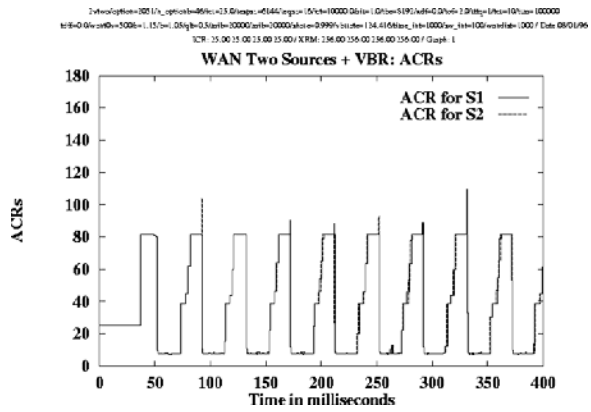
(a) Transmitted Cell Rate

(b) Queue Length

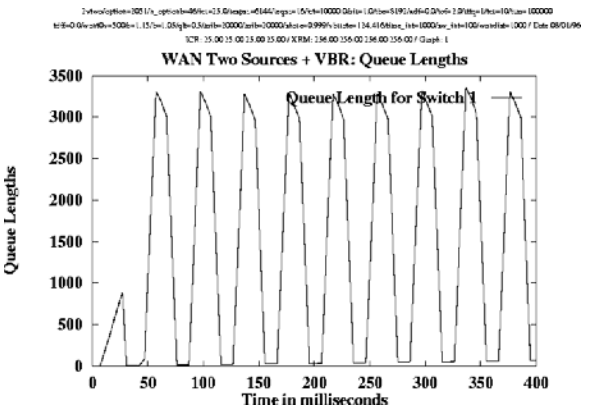
(c) Link Utilization

(d) Cells Received

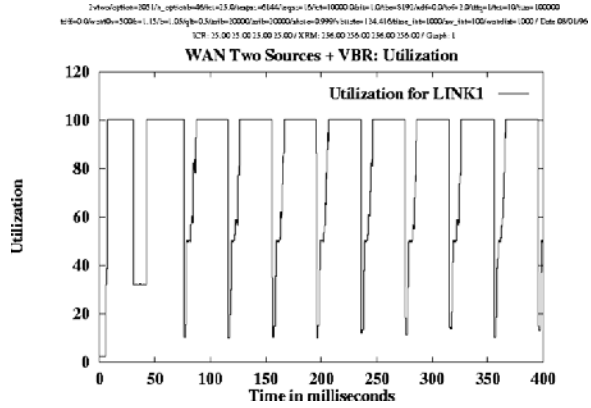
Figure 6.23: Results for a two source configuration with (1 ms on/1 ms off) VBR background in a WAN (ERICA+)



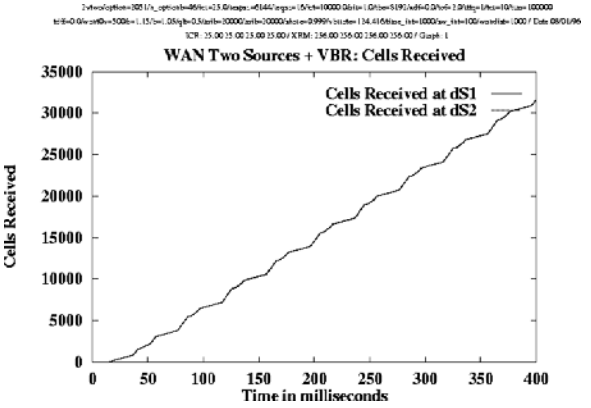
(a) Transmitted Cell Rate



(b) Queue Length

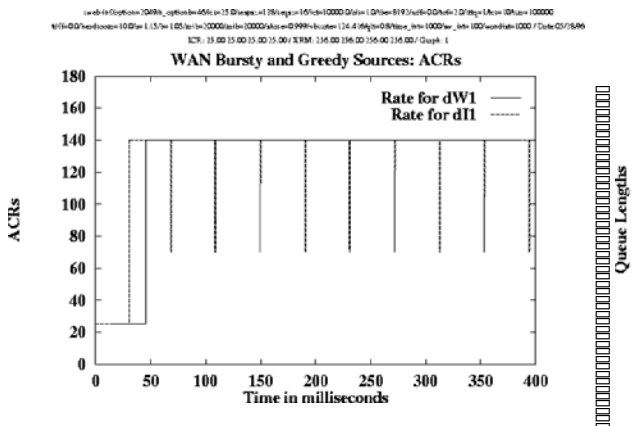


(c) Link Utilization

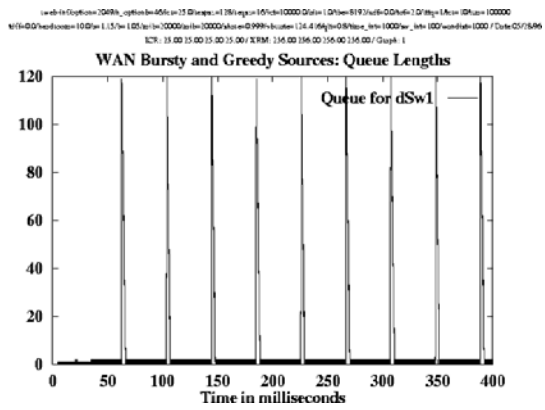


(d) Cells Received

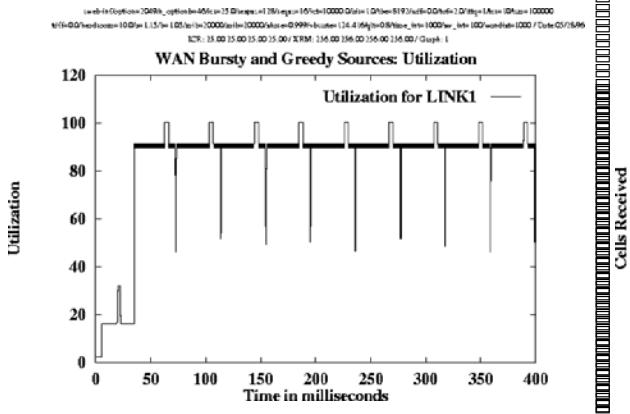
Figure 6.24: Results for a two source configuration with (20 ms on/20 ms off) VBR background in a WAN (ERICA+)



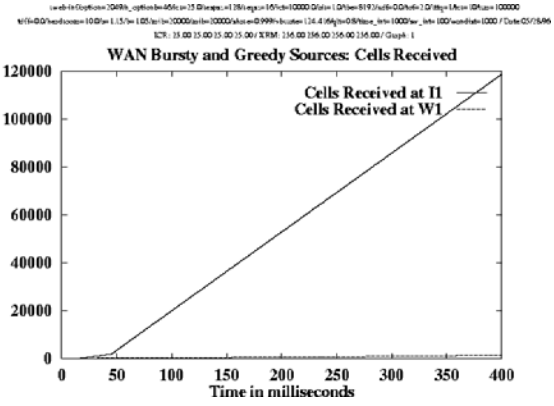
(a) Transmitted Cell Rate



(b) Queue Length

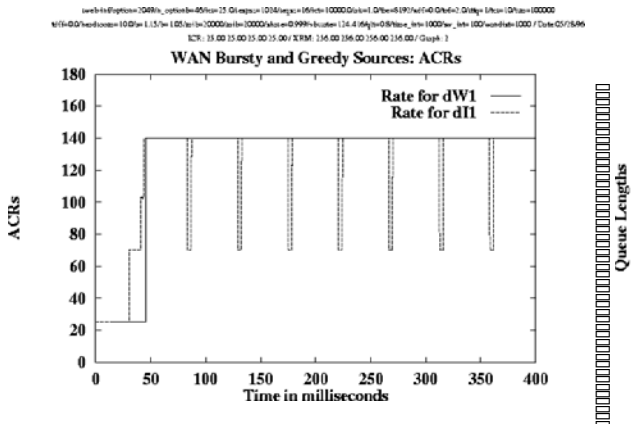


(c) Link Utilization

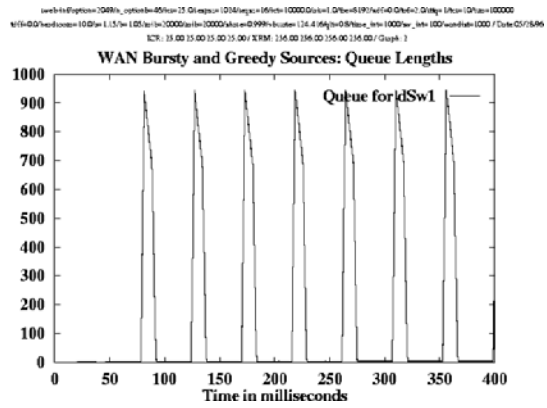


(d) Cells Received

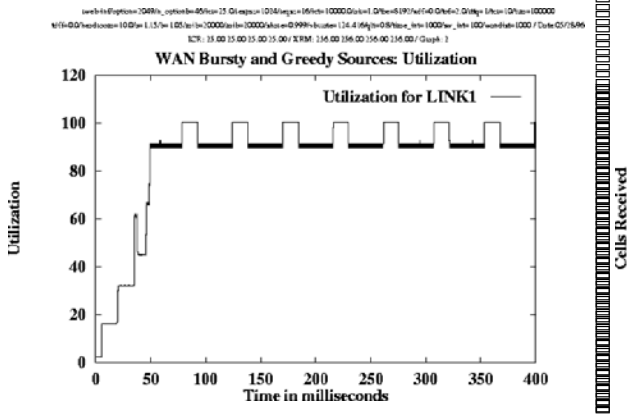
Figure 6.25: Results for one persistent source and one bursty source (small bursts) in a WAN (ERICA)



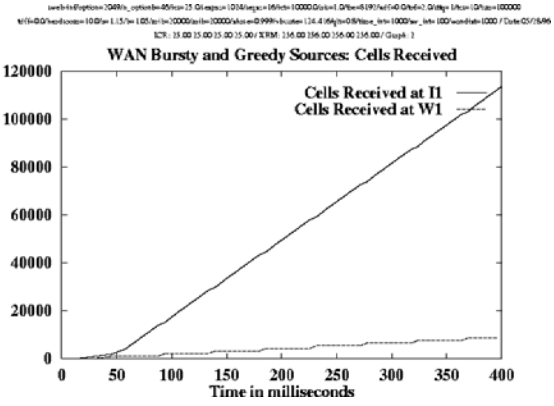
(a) Transmitted Cell Rate



(b) Queue Length

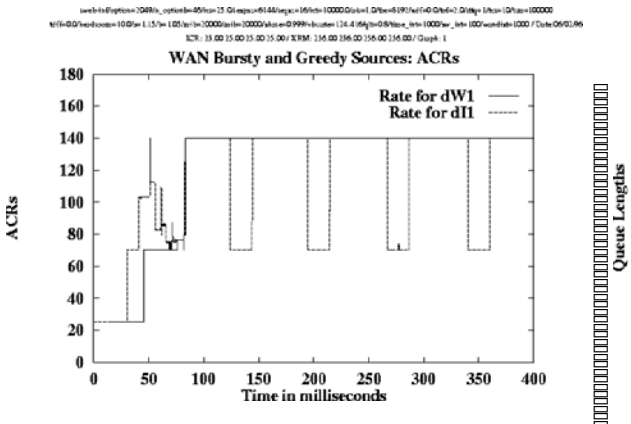


(c) Link Utilization

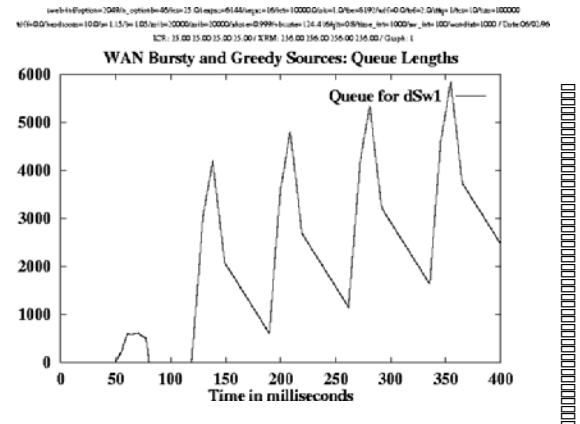


(d) Cells Received

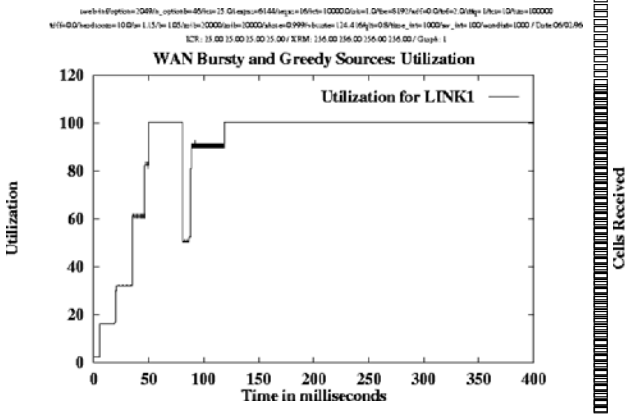
Figure 6.26: Results for one persistent source and one bursty source (medium bursts) in a WAN (ERICA)



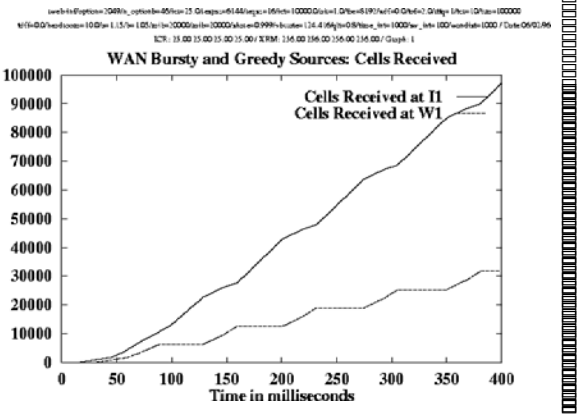
(a) Transmitted Cell Rate



(b) Queue Length

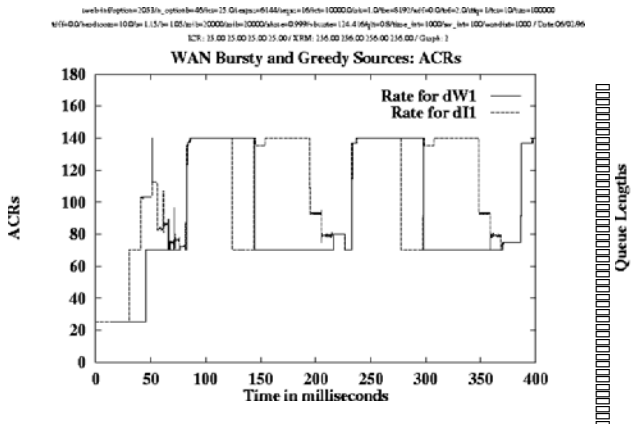


(c) Link Utilization

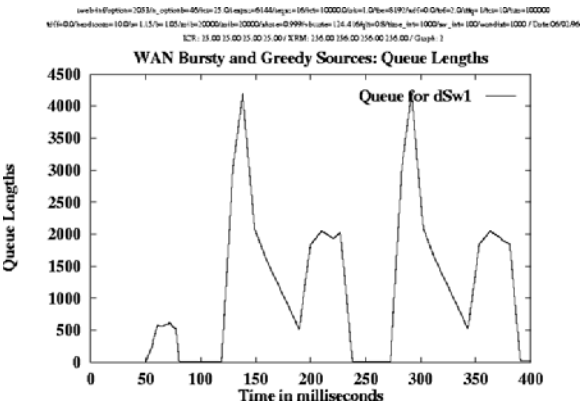


(d) Cells Received

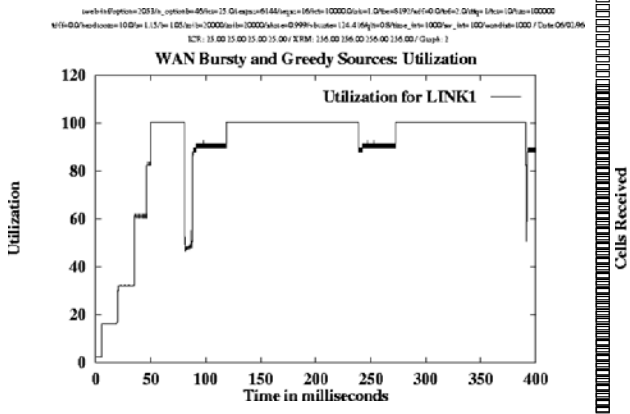
Figure 6.27: Results for one persistent source and one bursty source (large bursts) in a WAN (ERICA)



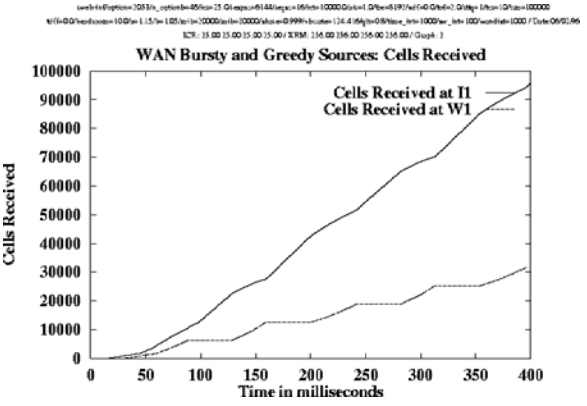
(a) Transmitted Cell Rate



(b) Queue Length

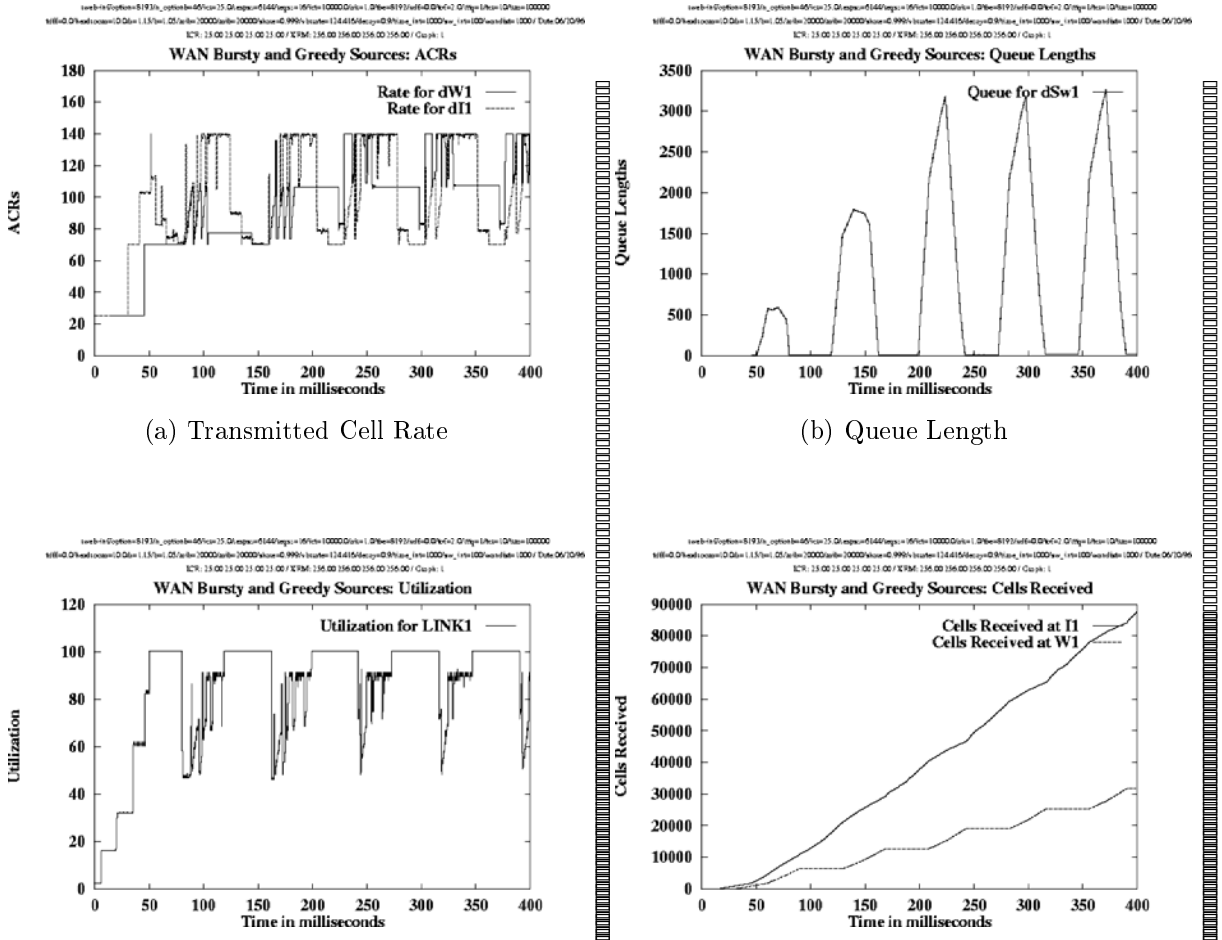


(c) Link Utilization



(d) Cells Received

Figure 6.28: Results for one persistent source and one bursty source (large bursts) in a WAN (ERICA with bidirectional counting)



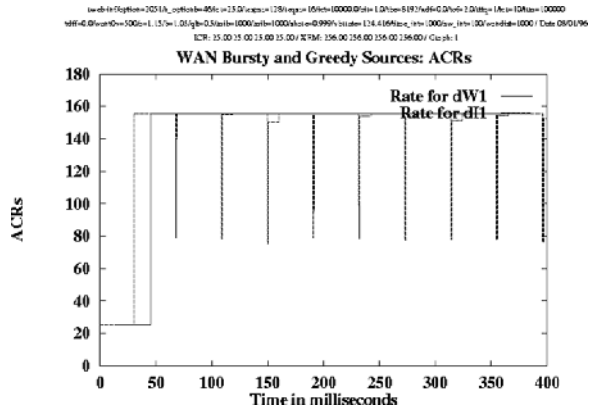
(a) Transmitted Cell Rate

(b) Queue Length

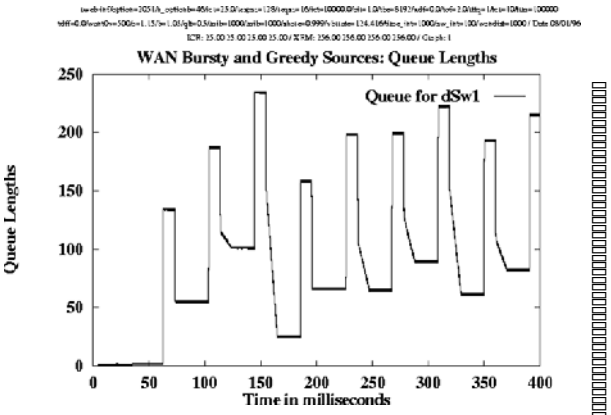
(c) Link Utilization

(d) Cells Received

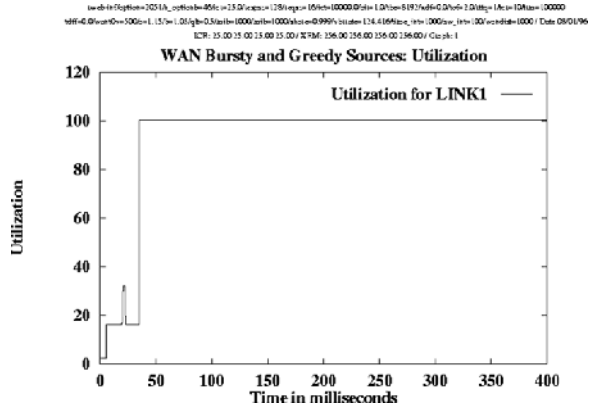
Figure 6.29: Results for one persistent source and one bursty source (large bursts) in a WAN (ERICA with averaging of number of sources)



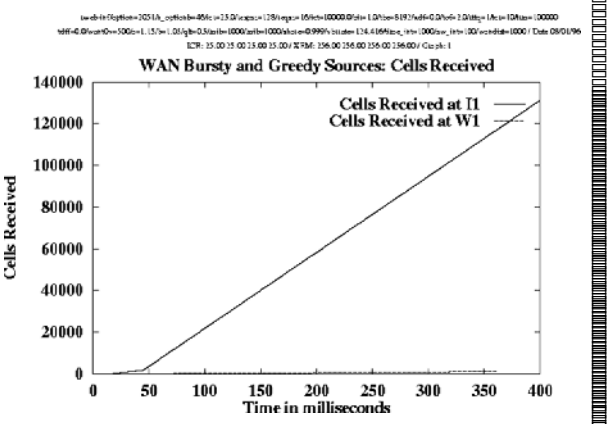
(a) Transmitted Cell Rate



(b) Queue Length

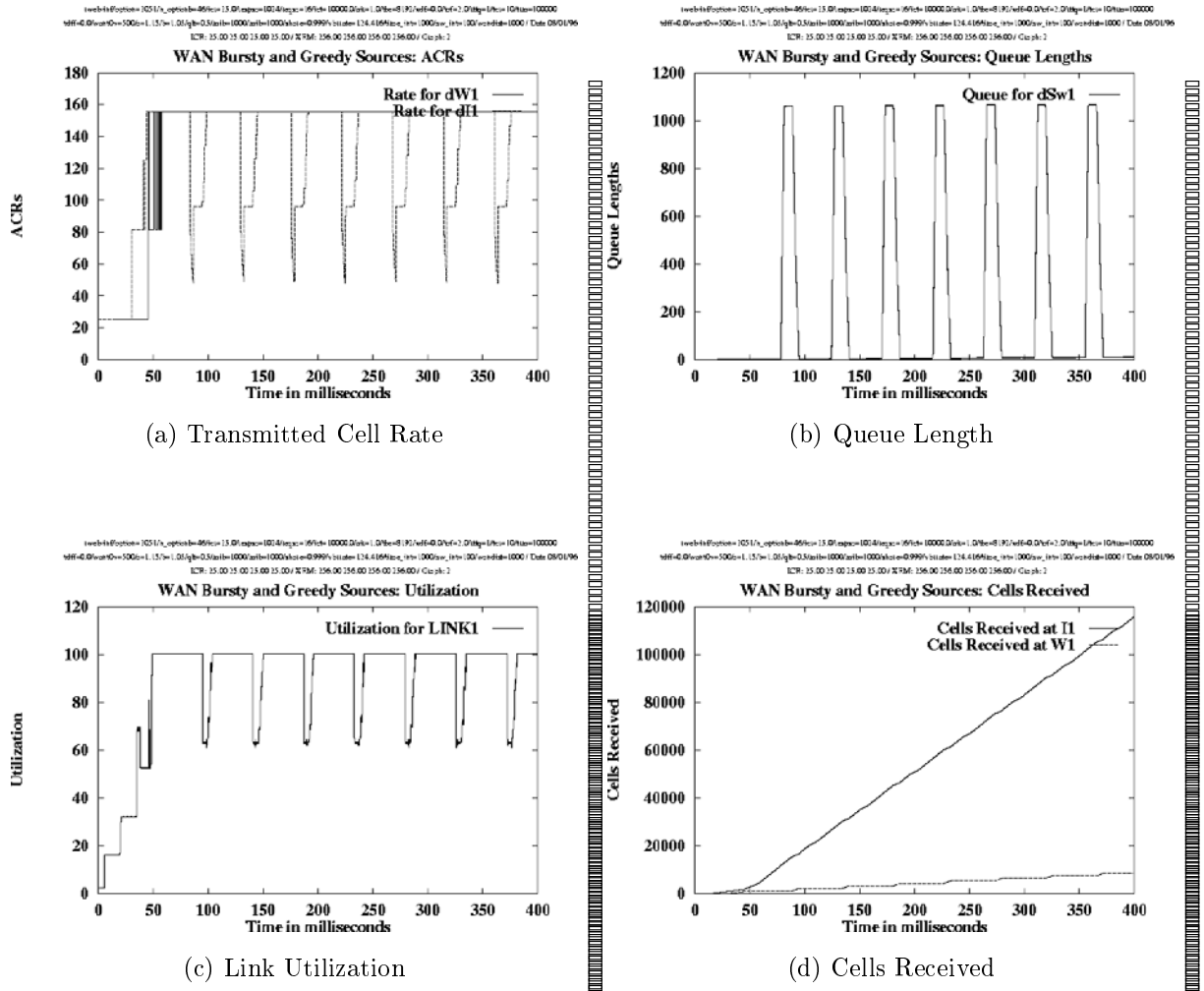


(c) Link Utilization



(d) Cells Received

Figure 6.30: Results for one persistent source and one bursty source (small bursts) in a WAN (ERICA+)



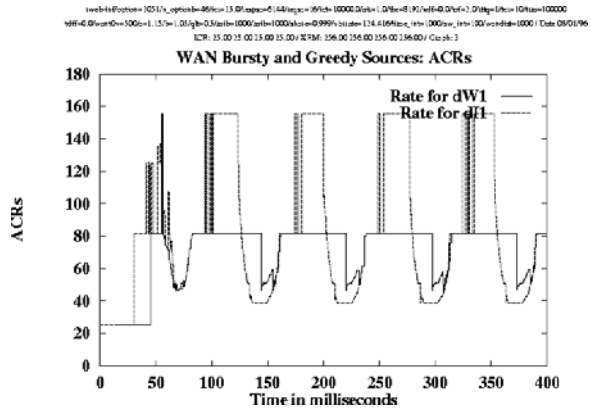
(a) Transmitted Cell Rate

(b) Queue Length

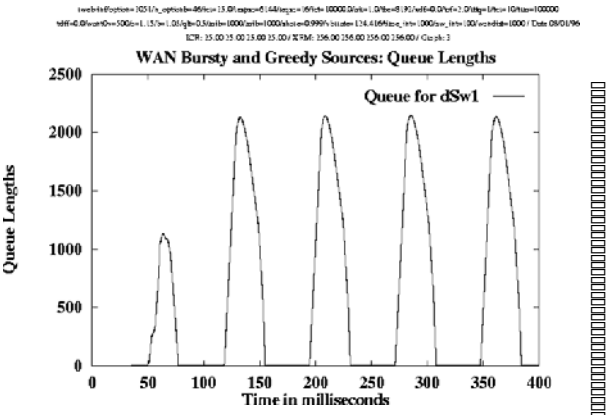
(c) Link Utilization

(d) Cells Received

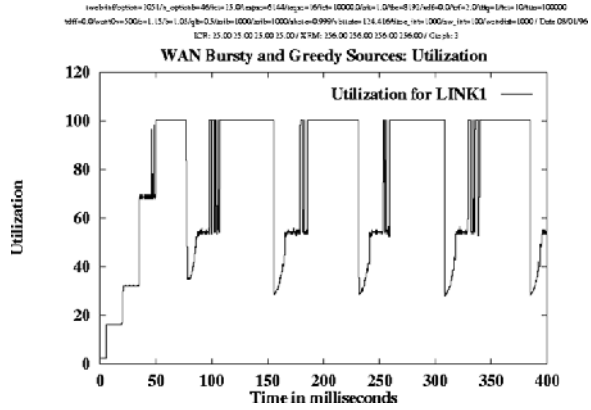
Figure 6.31: Results for one persistent source and one bursty source (medium bursts) in a WAN (ERICA+)



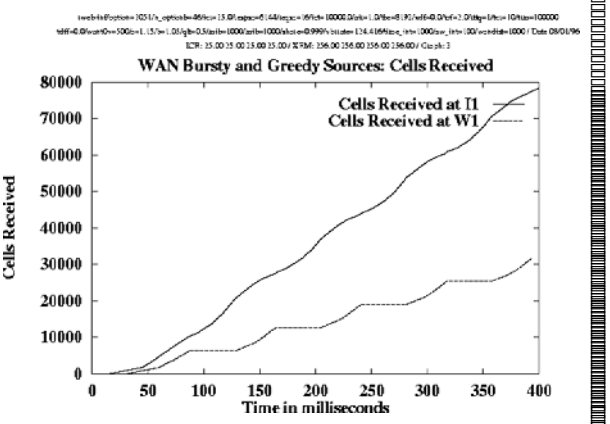
(a) Transmitted Cell Rate



(b) Queue Length

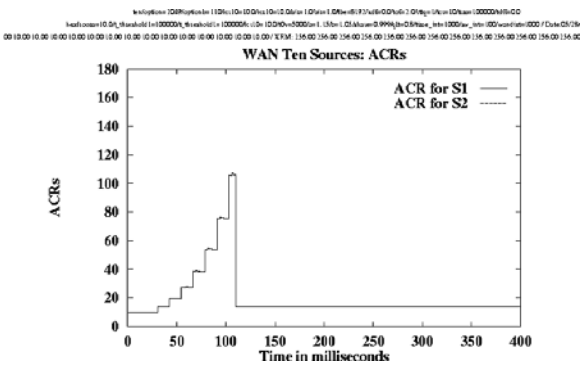


(c) Link Utilization

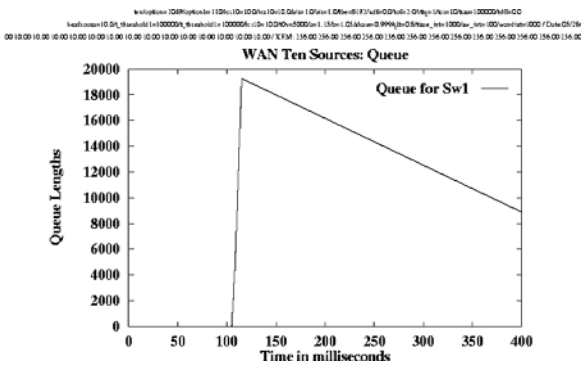


(d) Cells Received

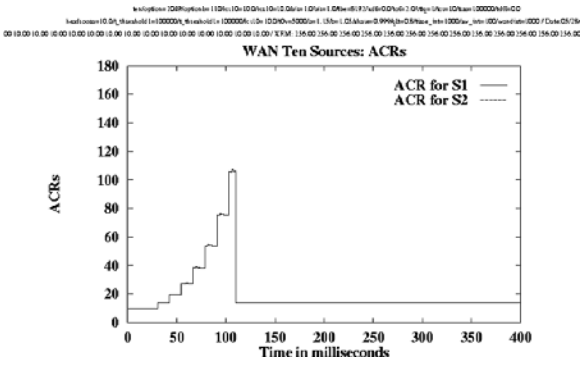
Figure 6.32: Results for one persistent source and one bursty source (large bursts) in a WAN (ERICA+)



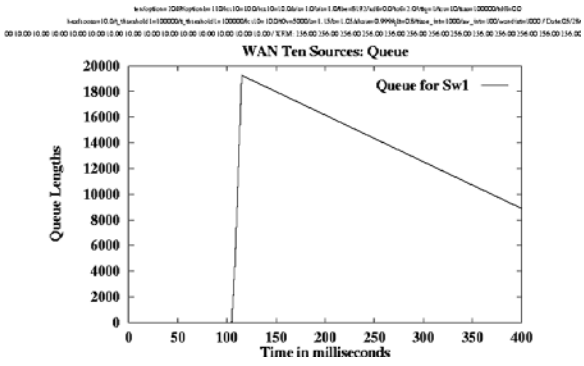
(a) Transmitted Cell Rate (basic ERICA)



(b) Queue Length (basic ERICA)

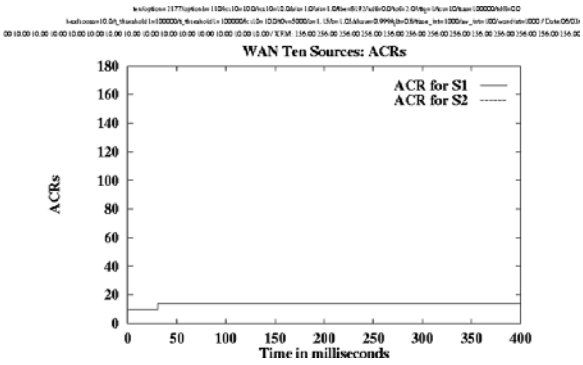


(c) Transmitted Cell Rate

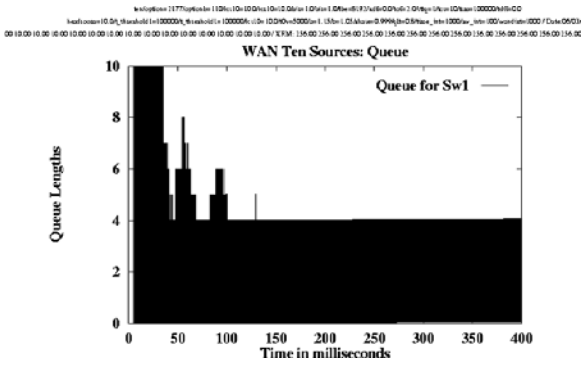


(d) Queue Length

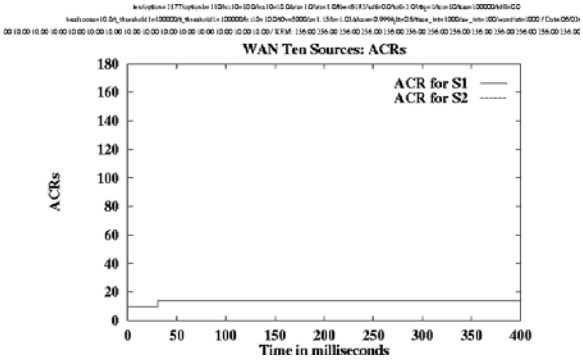
Figure 6.33: Results for ten ACR retaining sources in a WAN (ERICA)



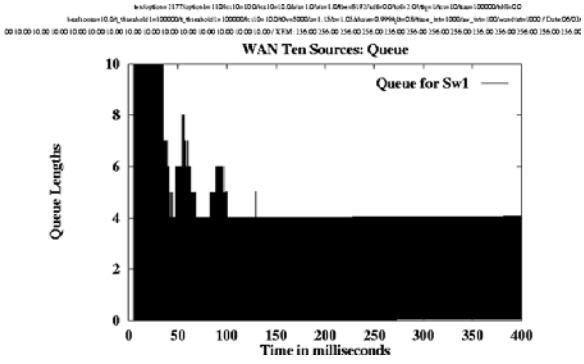
(a) Transmitted Cell Rate (basic ERICA)



(b) Queue Length (basic ERICA)



(c) Transmitted Cell Rate



(d) Queue Length

Figure 6.34: Results for ten ACR retaining sources in a WAN (ERICA with per-VC CCR measurement)

CHAPTER 7

SOURCE RULE DESIGN FOR THE ABR SERVICE

In the earlier chapters of this dissertation we have examined switch scheme design in detail, including our proposals (OSU, ERICA and ERICA+). In this chapter, we will examine the design of source rules in the ATM Traffic Management framework. The source rules, as described in Chapter 2, determine the scheduling of data and bidirectional RM cells, the policies at the source when feedback is disrupted, or when the source does not use the allocated rate, and the response to binary and explicit rate feedback.

This dissertation work has helped design some of the source rules of the international standard (esp. SES Rules 5, 9, 11, and 13) and we shall examine the details later in this chapter. These rules can be broadly be considered as providing some open-loop functionality in ABR. SES Rules 5 and 13 deal with the problem of sources not using their allocated rates - the related policies are popularly called the Use-it-or-Lose-it policies. In a related work [53, 55], and in the introductory chapter 2, we consider parameter related issues in SES Rule 6. In Rule 9, we developed the "rescheduling" option to allow low rate sources to immediately use higher allocations. We helped develop SES Rule 11 which deals with the issue of low rate sources and out-of-rate

RM cells. In this chapter, we devote one section to the topic of Use-it-or-Lose-it policies, and another to the topic of low rate sources.

7.1 Use-It-or-Lose-It Policies

The ABR framework is predominantly closed-loop, i.e., sources normally change their rates in response to network feedback. Another form of control is open-loop control where sources change their rates independent of network feedback. Open-loop control can complement closed-loop control when the network delays are large compared to application traffic chunks, or when network feedback is temporarily disrupted. It is also useful to control applications which are bursty or source bottlenecked. Bursty application traffic alternates between active periods (application has data to send) and idle periods (application has no data to send). Source-bottlenecked applications cannot sustain a data rate as high as the network allocated rate. The ATM Forum debated on the issue of using open-loop control to reduce rate allocations of sources which do not use them. The proposed solutions, popularly known as the Use-It-or-Lose-It (UILI) policies, have had significant impact on the ABR service capabilities. In this section, we discuss and evaluate these policies, and their implications on the ABR service.

This section is subdivided as follows. Section 7.1.1 discusses the issues in the design of UILI policies. We then discuss early UILI proposals in section 7.1.2. We identify the problems with the early proposals in section 7.1.3 and present the final set of proposals which were debated in the ATM Forum in section 7.1.6. We then evaluate the performance of various alternatives in Section 7.1.12 and summarize the implications of UILI on ABR in section 7.1.13.

7.1.1 Issues in Use-It-or-Lose-It

When some VCs' present bursty or source-bottlenecked traffic, the network may experience underload even after rate allocation. It then allocates higher rates to all VCs without first taking back the unused allocations. As a result, the underloading sources retain their high allocations without using them. When these sources suddenly use their allocations, they overload the network.This problem is called "ACR Retention." A related problem is "ACR Promotion" where a source intentionally refrains from using its allocation aiming to get higher allocations in later cycles. The effect of ACR Retention/Promotion is shown in Figure 7.1. In the figure, before time t_0 the source rate is much smaller than its ACR allocation. The ACR allocation remains constant. At time t_0, the source rate rises to ACR and the network queues correspondingly rise. These problems were first identified by Barnhart [7].

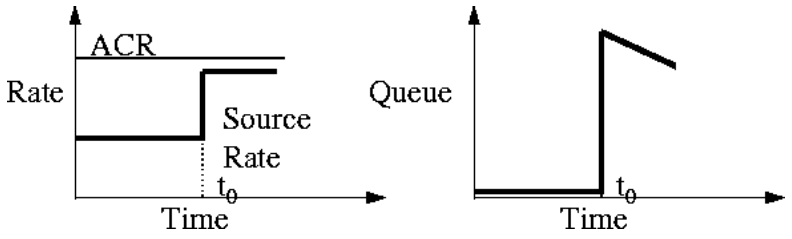


Figure 7.1: Effect of ACR Retention/Promotion

A solution to this problem is to detect an idle or source-bottlenecked source and reduce its rate allocation before it can overload the network. But this has an important side effect on bursty sources. If the rates are reduced after every idle period and the active periods are short, the aggregate throughput experienced by the source is

low. This tradeoff was discovered and studied carefully in the ATM Forum. The solutions proposed are popularly known as the Use-It-or-Lose-It (UILI) policies, referring to the fact that the source's ACR is reduced (lost) if it is not used.

7.1.2 Early UILI Proposals

The UILI function can be implemented at the SES (source-based) or at the switch (switch-based) or at both places. The early UILI proposals were all source-based. In these proposals, the test for ACR retention is done when an RM cell is being sent by the source. If ACR retention is detected, the source's ACR is immediately reduced using a rate reduction algorithm. Further, to prevent network feedback from overriding the ACR reduction, some proposals ignore the next feedback from the switch (if the feedback requests a rate increase). Over the February, April, May and June 1995 meetings of the ATM Forum, several UILI proposals were considered. The proposals differ in how the ACR retention is detected (additive or multiplicative metric), and in the algorithm used to reduce ACR.

In February 1995, Barnhart proposed a formula which reduced ACR as a function of the time since the last RM cell was sent or rate decrease was last done:

$$\text{ACRn} = \text{ACRo}(1 - \text{T} \times \text{ACRo}/\text{RDF})$$

ACRn is the new ACR and ACRo is the old ACR. The time 'T' in the formula is the time which has transpired since the last *backward* RM cell was received or since the last ACR decrease. RDF is the rate decrease factor which is normally used to calculate the new rate for single-bit feedback. However, it is reused in the reduction formula to avoid choosing a new parameter. ACR retention is detected when the

source has sent out k RM cells (k is the Time out Factor (TOF) parameter) but does not hear from the network or has not decreased its rate during the same period.

In April 1995, several flaws with this proposal were corrected. Further, the ACR decrease function was found to be too aggressive and was changed to a harmonic function:

$$1/\text{ACRn} = 1/\text{ACRo} + \text{T}/\text{RDF}$$

The time 'T' in the function is now the time which has transpired since the last *forward* RM cell was sent. In the May and June 1995 meetings several other side effects were identified and corrected. For example, it was felt that the decrease function should not reduce the ACR below the negotiated Initial Cell Rate (ICR), because the source is allowed to start at that rate after an idle period. Kenney [61] observed that the harmonic ACR reduction formula was difficult to implement and proposed a linear reduction formula, which was similar to, but less aggressive than the February proposal:

$$\text{ACRn} = \text{ACRo}(1 - \text{T} \times \text{TDF})$$

'TDF' is a new parameter called "Timeout Decrease Factor". Incorporating these changes, the ABR SES (source) specification in August 1995 read as follows:

*"5. Before sending a forward in-rate RM-cell, if the time T that has elapsed since the last in-rate forward RM-cell was sent is greater than TOF*Nrm cell intervals of (1/ACR), and if ACR > ICR, then:*

*a) ACR shall be reduced by at least ACR * T * TDF, unless that reduction would result in a rate below ICR, in which case ACR shall be set to ICR, and TDF is equal to TDFF/RDF times the smallest power of 2 greater or equal to PCR, TDFF =* $\{ 0, 2^i, 2^j, 2^l \}$ *(2 bits), where the values of the integers* $i, j,$ *and* l *are to be determined*

in the specification.

b) ACR shall not be increased upon reception of the next backward RM-cell."

The above UILI rule will also be interchangeably called "rule 5" henceforth, referring to the rule number in the ABR SES specification. The two parts are called "rule 5a" and "rule 5b" respectively.

7.1.3 Problems and Side Effects of Early Proposals

In August 1995, Anna Charny et al [16] pointed out certain undesirable side effects in the above proposal. In particular, sources experience performance degradation in the transient phase when they increase from low ACR to high ACR. As a result, the links may be underutilized for a long period of time.

7.1.4 Worst Case Performance

The worst case occurs when ICR is small and the source rises to a high rate from a very low rate, and when the backward RM cell (BRM) is received just before a forward RM cell (FRM) is sent. The BRM carries the network feedback and asks the source to increase its rate to a value greater than TOF $\times$ (old rate). When the FRM is sent, the measured source rate S is close to the earlier low rate. This results in triggering UILI and the reduction of ACR by ACR $\times$ T $\times$ TDF. Now ACR is large and T is also large since it depends on the earlier low rate. Hence, ACR is reduced by a large amount upto ICR. Since ICR again is a small value, the cycle repeats when the BRM is received just before a FRM is sent. As a result, a source starting from a low ICR may never send at a rate higher than ICR.

7.1.5 Bursty and RPC Traffic Performance

Charny et al [16] also observed that bursty traffic having low ICR experienced a long-term performance degradation due to UILI resulting in large ACR fluctuations. Further, rule 5b prevents the increase of the source rate even though the network may have bandwidth available. In such bursty traffic configurations, it was found that rule 5a without rule 5b yielded better performance than both the parts together. However there was no way to selectively turn off rule 5b. Hence, it was decided to introduce a PNI (Prohibit No Increase) bit which when set turns off rule 5b selectively. Note that this also allows us to turn off rule 5 completely if TDF is also set to zero.

The performance degradation due to remote procedure call (RPC) ping-pong type traffic was independently observed by Bennet et al [11]. These authors pointed out that such applications may not want their rates to be decreased or reset to ICR after every idle period. They also suggested that UILI be performed by the switch and the source-based UILI be left optional.

We note that these side effects of rule 5 are not seen when the source is in the steady state (with source rate approximately equal to ACR) or in the transient phase when the source is decreasing from a high ACR to a low ACR. The main problem seemed to be due to the fact that the decrease function was proportional to T resulting in large ACR decreases after an ACR increase, leading to ramp-up delays.

Another problem which emerged was that some parameters like RDF and ICR were being used in multiple rules. Hence, choosing optimal values for these parameters became difficult due to their various side effects. These problems were addressed in the new set of proposals in December 1995 when the issue was voted upon to arrive at a final decision.

7.1.6 December 1995 Proposals

There were three main proposals in December 1995: the time-based proposal [8, 66, 68], our count-based proposal [44], and the switch-based proposal [73]. The time-based and the count-based proposals were later combined into one joint proposal. The ATM Forum voted between the switch-based proposal and the joint source-based proposal.

7.1.7 Unresolved UILI Issues

The following were the unresolved issues in UILI in December 1995. Essentially, a UILI proposal which works for both source-bottlenecked and bursty sources was desired.

- How to avoid UILI from affecting the normal rate increase (ramp up) of sources ?
- How long should the switch feedback be ignored after an ACR adjustment ?
- How to ensure good throughput and response time for bursty sources having small, medium and large active periods, when the idle periods are small, medium or large ?
- The floor of the August 1995 UILI ACR reduction function is ICR. If the source rate, S, is larger than ICR, the ACR may be reduced below the source rate down to ICR. We want a reduction function which does not decrease the ACR below the source's rate, S.
- "Headroom" measures how much the ACR is greater than the source rate, S, when it is declared as not an ACR retaining source. Should the headroom be

multiplicative (ACR $\leq$ TOF $\times$ S) or additive (ACR $\leq$ S + headroom) ? Is a separate headroom parameter necessary (to avoid depending on ICR) ?

- Can UILI be done effectively in the switch ?
- Under what circumstances is UILI unnecessary or harmful ?

7.1.8 Count-Based UILI Proposal

The count-based UILI proposal [44] was made by us. It solved a large subset of the above problems and presented results of an extensive study on bursty traffic behavior.

Count vs Time

First, the count-based proposal removes the dependency of the ACR reduction function on the time factor, T, which is the time since the last FRM is sent. The reduction formula suggested is:

$$\text{ACR} = \text{ACR} - \text{ACR} \times \text{TDF}$$

The proposal is called "count-based" because a constant ACR decrease is achieved by triggering UILI n times. On the other hand, the time-based UILI decreases the ACR proportional to the time factor, T.

Multiplicative vs Additive Headroom

The count-based proposal uses an additive headroom for ACR detection (ACR $\leq$ S + headroom). Recall that if the ACR of the source is within the headroom, UILI is not triggered. The problem with multiplicative headroom (ACR $\leq$ TOF $\times$ S) used in the August 1995 proposal is that depending upon the value of S it results in a large

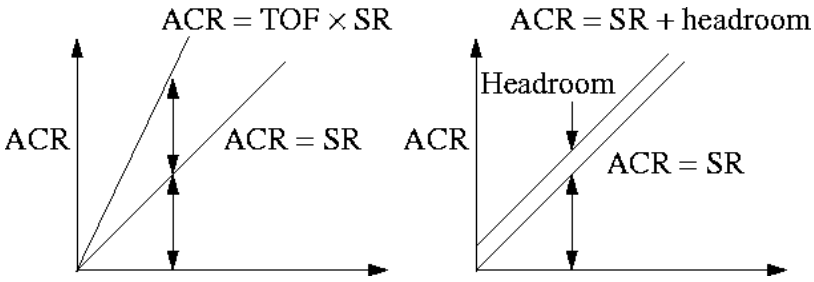


Figure 7.2: Multiplicative vs Additive Headroom

difference between ACR and source rate, S. A large difference (ACR - S) results in large network queues when the source suddenly uses its ACR.

The additive headroom allows only a constant difference (ACR − S) regardless of the source rate, S. The queue growth is hence bounded by a constant: $(ACR - S)\times$ Feedback Delay × Number of Sources. Hence, the additive headroom provides better network protection than the multiplicative headroom. The difference between the multiplicative and additive headroom is shown in Figure 7.2. Further, the latter is easier to implement since fewer multiply operations are required.

Floor of the ACR Reduction Function

We also observed that the floor of the August 1995 UILI ACR reduction function is ICR and independent of the source rate, S. This is problematic because if S is larger than ICR, the ACR may be reduced below the source rate down to ICR. Therefore, we use a different floor function (S + headroom) which ensures that the ACR does not decrease below S or the headroom. This floor function ensures that if the headroom equals the ICR, the ACR is guaranteed not to decreased below ICR.

Normal Rate Increase (Ramp Up)

The August 1995 proposal inhibited the ACR ramp up from a low rate because it triggered UILI immediately after the rate increase. Further, the amount of decrease could be large as explained in section 7.1.4.

Though our proposal does trigger UILI after ramp up from a low rate, it only reduces ACR by a step $\Delta = \text{ACR} \times \text{TDF}$. The next BRM cell brings the rate back to the ACR value before the decrease. If TDF is small, UILI is no longer triggered. For larger values of TDF, UILI may still be triggered multiple times. But, our new floor function ensures that the source rate consistently increases by at least the "headroom" value and eventually UILI is no longer triggered.

The count-based proposal also demonstrates a technique which avoids all oscillations due to normal rate increase. The UILI test is disabled *exactly once* after a normal rate increase. This allows the source rate to stabilize to the new (high) rate before the next UILI test, and thus UILI is not unnecessarily triggered. We use a bit called the PR5 ("Prohibit Rule 5") bit which is enabled whenever there is a normal rate increase. The bit is cleared otherwise.

This technique also has one important side effect. Consider a source which is using its ACR allocation but suddenly becomes idle. Using the RM cells remaining in the network, the network may request a rate increase during the idle period. According to the above technique, the UILI test is disabled exactly once when the source becomes active again. Now observe that the first FRM cell opportunity after an idle period is the only opportunity for the source to reduce its ACR using UILI. This is because the memory of the prior idle period is lost when the next FRM is sent. As a result, UILI

is never triggered. However, the PR5 technique is not necessary and can be disabled if TDF is chosen to be small.

Action on BRM

We observed that the ACR reduction function alone is not enough to ensure that that ACR retention is eliminated. For example, the August 1995 proposal requires that if the immediately next BRM feedback, after an UILI ACR reduction, requests a rate increase, and the PNI bit is not set, the BRM feedback is ignored. However, subsequent feedbacks may undo the ACR reduction and the problem of ACR retention still persists.

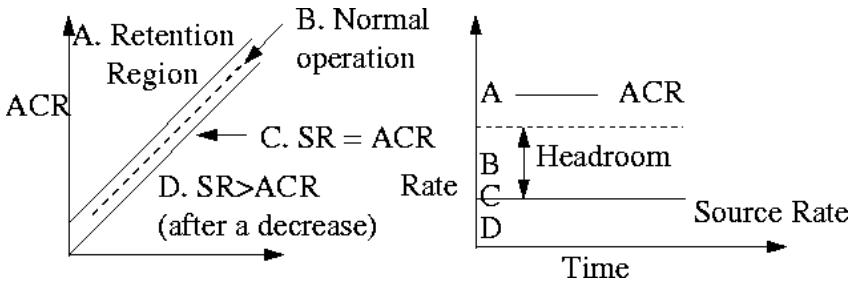


Figure 7.3: Regions of Operation

The count-based proposal ignores the BRM feedback as long as the source does not use its ACR allocation. The proposal uses the headroom area as a hysterisis zone in which network feedback to increase ACR is ignored. The proposal defines four regions of operation A, B, C, and D, as shown in Figure 7.3. Region A is called the ACR retention region. In this region, $ACR > SR + Headroom$, and UILI is triggered unless the PR5 bit (if used) is set. Region B is the headroom area. In this region, $ACR \leq SR + Headroom$, but $ACR > SR$. In this region BRM feedback requesting

Table 7.1: BRM Actions In The Different Regions Of Count-Based UILI

Region	Trigger UILI	Increase On BRM	Decrease On BRM
A	Yes unless PR5	No	Yes
B	No	No	Yes
C	No	Yes	Yes
D	No	Yes	Yes

increase is ignored. Region C has the source rate equal to ACR. Region D has source rate greater than ACR. Region D is touched briefly when the ACR decreases and the measured source rate is a mixture of the old and new ACRs. In regions C and D, the source obeys the feedback of the network to increase or decrease its ACR. In these regions, the source is not ACR retaining because its source rate is at least equal to its current ACR allocation. The actions in various regions are shown in Table 7.1. Note that there is no need for the PNI parameter, since UILI can be disabled by simply setting the parameter TDF to zero.

Parameter Selection

The count-based proposal has two parameters: "headroom" and "TDF". We recommended a separate "headroom" parameter is to avoid overloading the ICR parameter. This allows the ICR parameter to be set to a high value based on short-term congestion information. The headroom parameter can be set to a more conservative value. It controls how much the sources can lie about their rates at any time and determines how many cells the switch receives at once. However, as discussed in the

simulation results of bursty sources (Section 7.1.12), very small headroom is not desirable. A value of 10 Mbps is recommended. This allows LANE traffic to go at full Ethernet speed. Smaller values can be used for WANs.

The parameter TDF determines the speed of convergence to the desired UILI goals (region B in Figure 7.3). Hence, it determines the duration for which the network is susceptible to load due to sources suddenly using their ACRs. Larger values of TDF give faster convergence. However, a low value is preferred for bursty sources as discussed in Section 7.1.12, and TDF set to zero disables UILI. A value of 1/8 or 1/16 is recommended.

Pseudo Code For the Count-Based Proposal

In the pseudo code for the count-based proposal given below, the variable 'ACR_ok' indicates that the source has used its allocated ACR, and is allowed to increase its rate as directed by network feedback. The variable 'PR5' when set conveys the fact that the network has just directed an increase. 'SR' is a temporary variable and is not stored between successive execution of the code. Further, the proposal requested a separate parameter 'headroom' instead of using ICR in the UILI formula.

- **At FRM Send event:**

  SR = Nrm/T;

  ACR_ok = ((ACR $\leq$ SR) OR (TDF == 0.0));

  IF (PR5 == FALSE)

  IF (ACR > SR + headroom)

  ACR = Max(SR + headroom, ACR $\times$ (1.0 $-$ TDF));

ENDIF

ELSE PR5 = FALSE;

- **At BRM Receive event:**

  IF (NI = 0 AND ACR_ok)

  IF (ACR < ER) PR5 = TRUE ELSE PR5 = FALSE;

  ACR = Min(ACR + AIR × PCR, PCR);

  ENDIF

  ACR = Min(ACR, ER);

  ACR = Max(ACR, MCR);

- **Initialization**

  ACR_ok = True;

  PR5 = False;

Note that the comparison (ACR $\leq$ SR) may always yield false due to the fact that cells may be scheduled only at certain fixed slots. There is typically a minimum granularity Δ which dictates the cell scheduler at the source. To account for this scheduler, the comparison may be replaced by (ACR $\leq$ SR + Δ).

7.1.9 Time-Based UILI Proposal

The time-based UILI proposal has a ACR reduction function which depends upon the time T since the last FRM was sent. While this aspect is similar to the August 1995 UILI proposal, the other changes suggested are:

1. The time-based proposal also independently observes the problem with using ICR as the floor of the reduction function (as discussed in Section 7.1.8). The proposal suggests two possible floor values:

   a) ACR_{max} = Max(ICR, TOF $\times$ SR)

   b) ACR_{max} = ICR + SR

2. IF (ACR > ACR_{max})

$$ACR_{new} = \text{Max}(\text{ ACR} \times (1 - \text{T/Tc}), ACR_{max}\ );$$

   The recommended value for Tc is Max(ADDF $\times$ FRTT, TBE/PCR), where ADDF has a default value of 2. FRTT is the Fixed Round Trip Time measured at connection setup.

The ACR reduction formula decreases ACR depending upon how long the idle period is compared to the round-trip time. A performance comparison of the count-based and the time-based alternatives is presented in Section 7.1.12.

7.1.10 Joint Source-Based UILI Proposal

The time-based and count-based camps agreed on a consensus, which we refer to as the "joint source-based proposal." The proposal uses the count-based reduction function and a constant value for TDF. It uses the new floor of the reduction function and the additive headroom. However, ICR is used in the UILI function instead of the proposed "headroom" parameter. The hysterisis region (region B in Figure 7.3) suggested by the count-based proposal is not used. Rule 5b remains the same as the August 1995 proposal, and PR5 is not used since TDF is set to a small value (1/16), the count-based reduction formula is used.

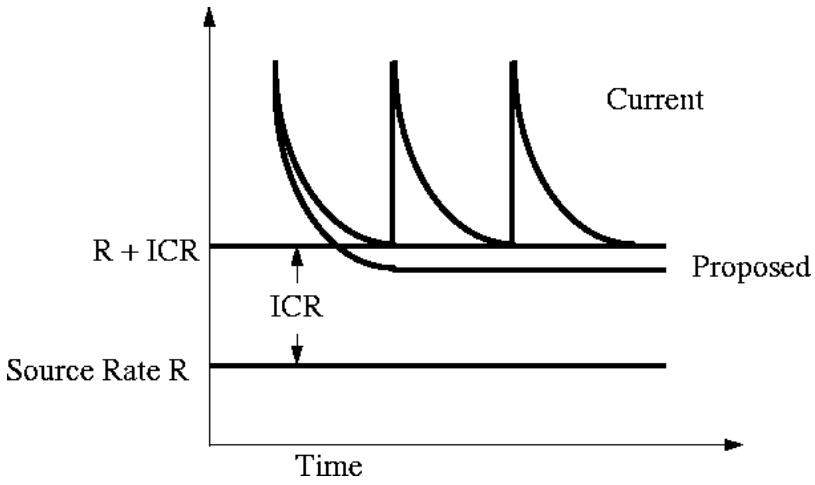


Figure 7.4: Joint Source-Based UILI Proposal vs Count-Based Proposal

The effect of removing the hysterisis region in the joint proposal is shown in Figure 7.4. In the joint proposal, the source will ignore one ER feedback after reducing the ACR to within the desired threshold. However, it may increase its rate-based upon ER feedback henceforth. The source thus re-enters the danger zone of ACR retention. In the count-based proposal, a source which reaches the desired operating zone (ACR <= SR + ICR), it remains in this region until the source actually uses its ACR allocation.

7.1.11 Switch-Based Proposal

AT&T [73] argued that the UILI function can be implemented in the switches on the following lines:

- Estimate rate of a connection and derive a smoothed average. This requires per-VC accounting at the switches.

- The switch maintains a local allocation for the VC based on the max-min fair allocation and the rate the VC claims to go at, i.e., its CCR.

- Use an "aging" function at the switch which allocates a rate to the VC based on the the ratio of the CCR and the actual rate-estimate. Basically, this function widthdraws the allocations from ACR retaining sources.

  A suggested aging function was ($e^{\alpha u} - e^{\alpha \delta}$) where, u is the ratio of the expected rate and the actual rate, and, α and δ are parameters. The function has the property that the larger the difference between the CCR and the estimated actual rate, the greater the reduction factor. Essentially, the switch allocates conservatively to sources which it knows are not using their allocations.

A switch-based policy with no support from the source faces problems in handling sources which go idle because idle sources do not send RM cells. The switch may take away the allocation of an idle source after a timeout, but there is no way to convey this information to the idle source, since there are no RM cells from the source. Therefore, the switch-based UILI proposal suggests a simple timeout mechanism at the source which reduces the rate of the source to ICR after a timeout (parameter ATDF) of the order of 500 ms. Note that idle sources which become active before the timeout expires may still overload the network. The proposal does not implement UILI for such sources.

7.1.12 Simulation Results

In this section, we study the tradeoffs in the UILI design through simulation results. We look at both source-bottlenecked and bursty source configurations and present sample simulation results for the following five UILI alternatives:

1. No UILI

2. August 1995 UILI proposal

3. Baseline Rule 5 (enhanced August 1995) proposal, where the time-based reduction formula is replaced by the count-based formula, and an additive headroom (equal to ICR) is used in place of the multiplicative headroom.

4. The count-based UILI proposal

5. The time-based UILI proposal

A complete set of simulation results may be found in reference [60].

Source Bottlenecked Configuration

The configuration is a network consisting of five ABR sources (Figure 7.5) going through two switches to corresponding destinations. All simulation results use ERICA switch algorithm. All links are 155 Mbps and 1000 km long. All VCs are bidirectional, that is, D1, D2, through D5 are also sending traffic to S1, S2 through S5. Some important ABR SES parameter values are given below. The values have been chosen to allow us to study UILI without the effect of other SES rules.

PCR = 155.52 Mbps, MCR = 0 Mbps, ICR = 155.52 Mbps, 1 Mbps

RIF (AIR) = 1, Nrm = 32, Mrm = 2, RDF = 1/512

Crm = Min{TBE/Nrm, PCR $\times$ FRTT/Nrm}

TOF = 2, Trm = 100 ms, FRTT = 30 ms, TCR = 10 cells/sec

TBE = 4096 (Rule 6 effectively disabled), CDF (XDF) = 0.5

TDF = {0, 0.125} : {0 $\Rightarrow$ No rule 5, 0.125 for all versions of rule 5}

PNI = {0, 1} : {1 ⇒ No rule 5b, 0 ⇒ Rule 5b for August 1995 and Baseline UILI}

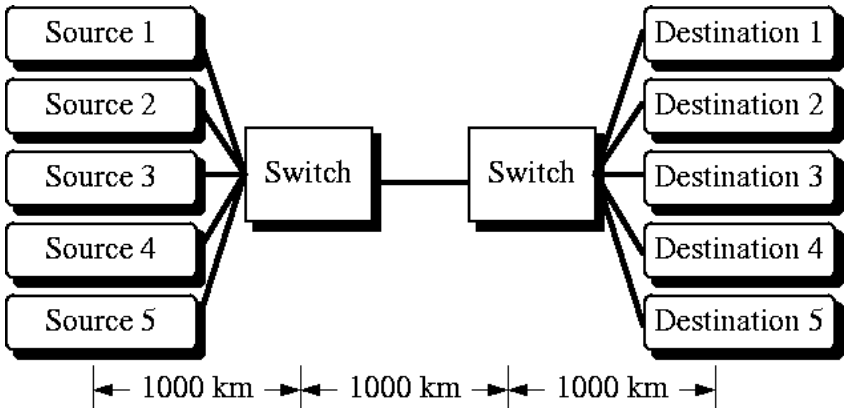
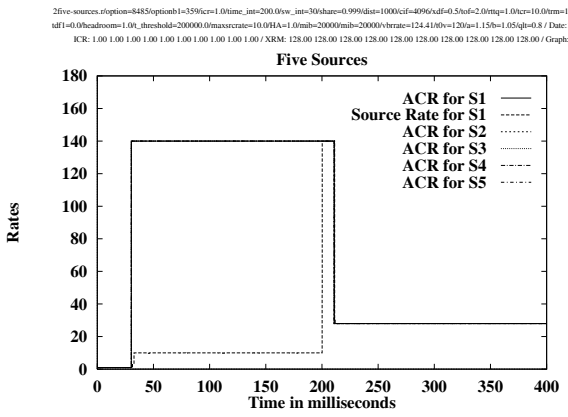


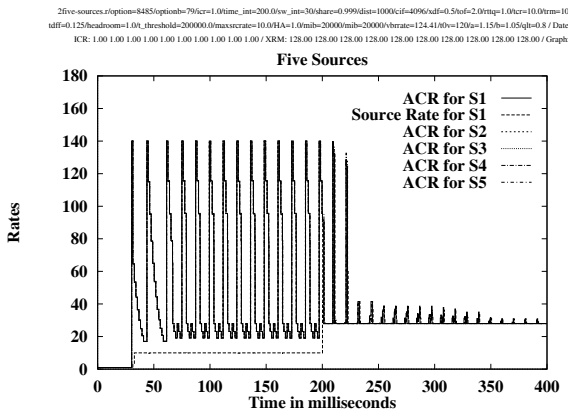
Figure 7.5: Five Sources Configuration

The simulation is run for 400 ms. For the first half of the simulation (200 ms), all the VCs are source-bottlenecked at 10 Mbps. After t=200 ms, all sources are able to use their allocated rates.

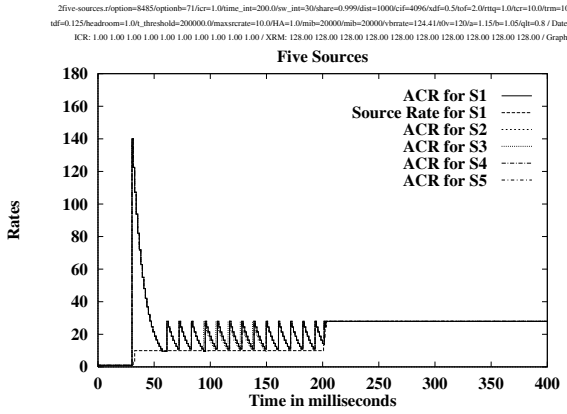
Figure 7.6 shows the ACR, and the actual source rates for the five UILI alternatives studied. There are six lines in each graph consisting of five ACR values and one actual source rate. Since all five sources are identical, the curves lie on the top of each other. With no UILI implemented (figure 7.6(a)) the ACR is initially much larger than the actual source rate. At 200 ms, the source rate jumps to the ACR and results in network overload. Figure 7.6(b) shows oscillatory behavior of the August 1995 proposal due to the wrong floor of the ACR reduction function. The Baseline UILI reaches the goal. However it oscillates between the goal and the network feedback. The count-based UILI converges quickly to the goal and does not have oscillations



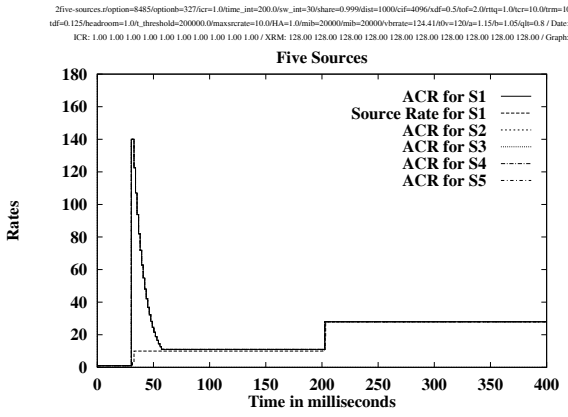
(a) No UILI



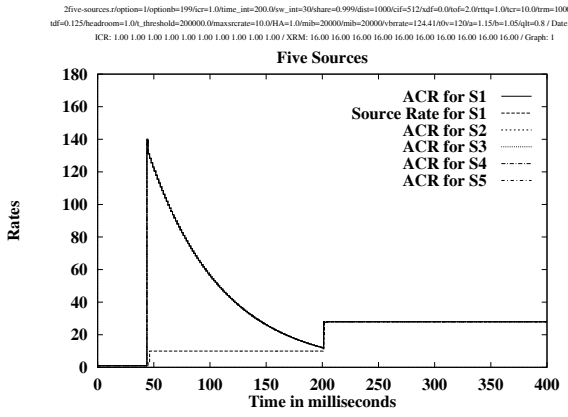
(b) Aug 1995 UILI



(c) Baseline UILI



(d) Count-Based UILI



(e) Time-Based UILI

Figure 7.6: Five Source Configuration: Rates, Low ICR = 1.0 Mbps, Headroom = 1 Mbps, MaxSrcRate = 10 Mbps for 200 ms

after reaching the goal. The time-based UILI converges very slowly to the goal. Had the sources started using their ACR allocations earlier (than 200ms), it would have resulted in network overload.

Bursty Sources

Recall that bursty sources have active periods when they send data at the allocated rate and idle periods when they do not have data to send. From the point of view of the bursty application, the following two measures are of interest (figure 7.7):

- *Burst response time* is the time taken to transmit the burst.
- *Effective throughput* is the average transmission rate of the burst.

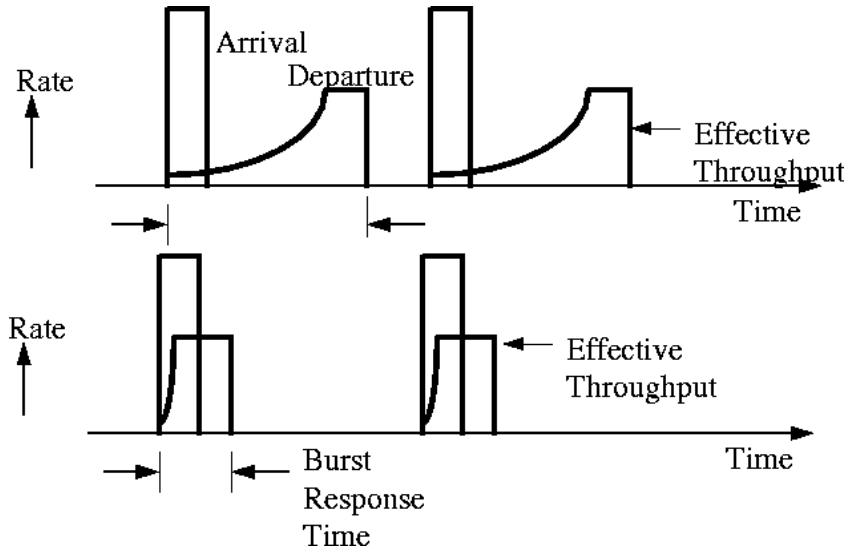


Figure 7.7: Burst Response Time vs Effective Throughput

Figure 7.7 shows the arrival and departure of a burst at an end system. The top part of the figure shows a burst which takes a long time to be transmitted and the

bottom part shows one which is transmitted quickly. In the former case, the burst response time is short and effective throughput is higher, and vice versa for the latter case. Note that the effective throughput is related to the size of the burst and the burst response time.

Observe that the UILI goals conflict with the above bursty traffic performance goals. When UILI works, ACR is effectively reduced and a bursty source keeps restarting from low rates after every idle period. This results in a high burst response time which implies reduced performance. We study the effect of the UILI policy for different lengths of the active period: short (burst size is smaller than Nrm), medium (burst time smaller than round trip time (RTT), but burst size larger than Nrm) and large (burst time larger than RTT). Handling the network queues is usually not a problem for short or medium bursts. But it does become important when larger bursts active periods are used. The next section describes a model to generate short, medium and long bursts.

Closed-Loop Bursty Traffic Model

We define a new "closed-loop" bursty traffic model as shown in Figure 7.8. The model consists of cycles of request-response traffic. In each cycle the source sends a set of requests and receives a set of responses from the destination. The next cycle begins after all the responses of the previous cycle have been received and an inter-cycle time has elapsed. There is a gap between successive requests called the inter-request time. The request contains a bunch of cells sent back-to-back by the application at rate PCR and the adapter controls the output rate to ACR.

The model as presented above may roughly represent World Wide Web traffic, transaction-oriented traffic, or client-server traffic. The model is "closed-loop" in the

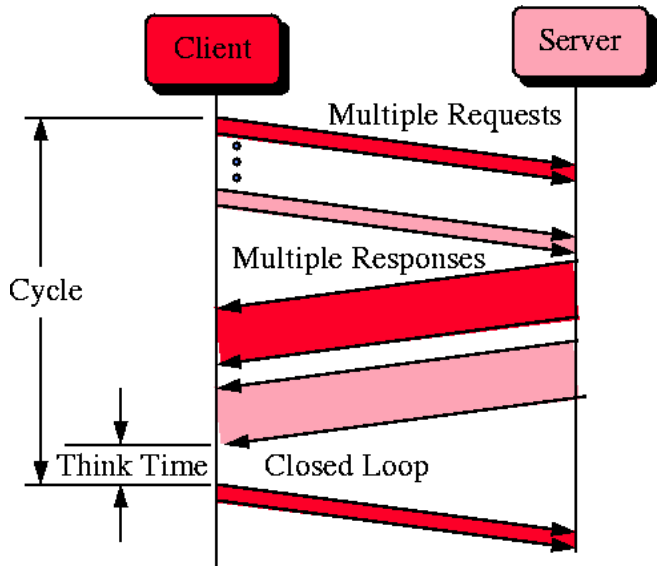


Figure 7.8: Closed-Loop Bursty Traffic Model

sense that the rate at which cycles (and hence requests) are generated depends upon the responsivity of the network. If the network is congested the response take longer time to come back and the sources do not generate new requests until the previous ones have been responded to. In an "open-loop" traffic model like the packet-train model [47], bursts are generated at a fixed rate regardless of the congestion in the network.

Note that the time between two sets of requests (called a cycle time) is at least the sum of the time to transmit requests, the round-trip time and the inter-cycle time. Thus the idle time between two sets of requests is always greater than the round-trip time. All the RM cells from the previous set of requests return to the source before the new set of requests are sent. When a new burst starts there are no RM cells of the source in the network (ignoring second-order effects).

In our simulations, a cycle consists of one request from the client and one response from the server. We use a small response burst size (16 cells), and vary the request burst size.

Single-Client Configuration and Parameter Values

The configuration we use is called the single-client configuration (Figure 7.9). It consists of a single client which communicates with the server, via a VC which traverses a bottleneck link. An infinite source is used in the background to ensure that the network is always loaded, and any sudden bursts of traffic manifest as queues. All the links run at 155 Mbps.

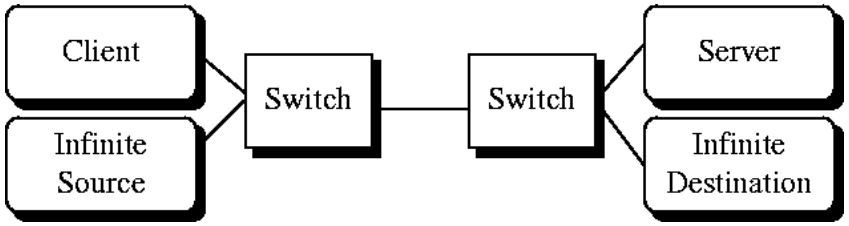


Figure 7.9: Client-Server Configuration With Infinite Source Background

The response size is kept constant at 16 cells. The request size can be 16, 256 or 8192 for small, meduim or large bursts respectively. The inter-cycle time is chosen to be 1ms. All links are 500km long. The other source parameters are chosen to maximize ACR and disable the effects of other source rules:

ICR = 10 Mbps, TDF = 1/8, TCR = 10 cells/sec

TRM = 100 ms, TBE = 512, CDF = 0 to disable SES Rule 6.

The switch uses the ERICA algorithm to calculate rate feedback. The ERICA algorithm uses two key parameters: target utilization and averaging interval length.

The algorithm measures the load and number of active sources over successive averaging intervals and tries to achieve a link utilization equal to the target. The averaging intervals end either after the specified length or after a specified number of cells have been received, whichever happens first. In the simulations reported here, the target utilization is set at 90%, and the averaging interval length defaults to 100 ABR input cells or 1 ms, represented as the tuple (1 ms, 100 cells).

In the following sections, we pictorially describe the simulation results; a full set of graphs may be found in reference [60].

Small Bursts

Small bursts are seen in LANE traffic. For example, the ethernet MTU, 1518 bytes is smaller than 32 (Nrm) cells. Since small bursts are smaller than Nrm cells, no RM cells are transmitted during certain bursts. As a result, no SES rules are triggered during these bursts. In other words, the entire burst is transmitted at one rate. However, when RM cells are finally transmitted, UILI is triggered which brings down the ACR to ICR. The source rate, S, is nearly zero due to the short burst time and long idle time. Hence, ICR + S is approximately equal to ICR.

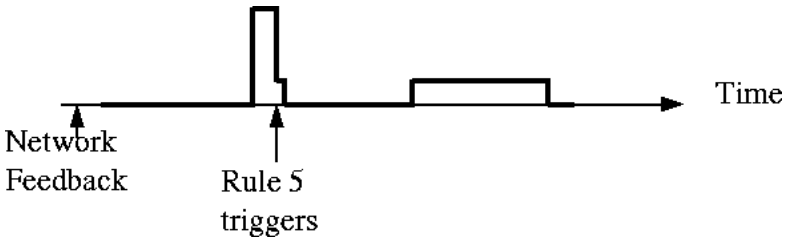


Figure 7.10: Effect of UILI on Small Bursts

Figure 7.10 shows the effect of UILI on the source rate of small bursts. The network feedback first arrives when the source is idle, asking it to increase its ACR. The source uses its ACR to almost send the full burst. The first RM cell sent reduces its source rate back to ICR. The source rate goes back to zero when the source is idle. Now, the time-based and count-based proposals differ in the way they respond to subsequent network feedback.

In the time-based proposal, the feedback brought by the next RM cell is ignored because of rule 5b. Now there is no RM cell of the source in the network and at least two bursts are sent at ICR before the next RM cell is sent which results in an ACR increase. Note that the sending of this second RM cell does not decrease the ACR further because ACR is already at ICR. Therefore, on the average one out of every three bursts is sent at a higher rate.

In the count-based proposal, the rate-increase feedbacks are always ignored because the system is in region B (Figure 7.3). The ACR slowly reduces to ICR and then remains at ICR. Over the long term, all short bursts are sent out at ICR only. This can be improved by using a leaky bucket or GCRA [32] type burst tolerance mechanism where small bursts can be sent at link rate irrespective of ACR or ICR. Other alternatives include choosing a small TDF or a larger ICR. An ICR of 10 Mbps allows LANE traffic (the source of small bursts) to go through at full speed. On the other hand, since the burst is very short, there is not a significant time difference in transmitting the burst at ACR and transmitting it at ICR (assuming ICR is not very small). In such a case, the emphasis then shifts to supporting medium bursts and large bursts efficiently.

Medium Bursts

Medium bursts are expected in ATM backbone traffic or in native mode ATM applications. Medium bursts contain more than Nrm (32) cells, but the active time is shorter than the round trip time. Though multiple RM cells are sent in a single burst, the network feedback for the burst arrives only after the burst has already been transmitted.

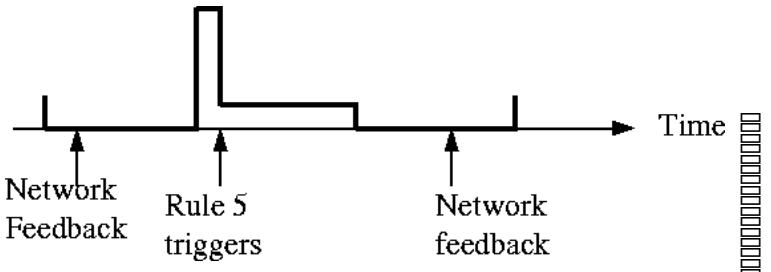


Figure 7.11: Effect of UILI on Medium Bursts

As shown in Figure 7.11, the UILI mechanism triggers once when the first RM cell is sent. In the time-based proposal, the amount of decrease is proportional to the idle time prior to the burst, while in the count-based UILI, the decrease is a constant amount. In the time-based proposal, if the idle time is large, almost the entire burst may be transmitted at ICR. Since, the count-based proposal sends the burst almost at ACR $\times$ (1 $-$ TDF), it provides better burst response. Accordingly, simulation results in reference [60] show that the average source rate experienced by the bursts is higher for the count-based option (120 Mbps) compared to the time-based option (68 Mbps).

Large Bursts

Large bursts are expected to be seen in backbone ATM links. Large bursts have a burst time larger than the round trip time. The network feedback returns to the source before the burst completes transmission.

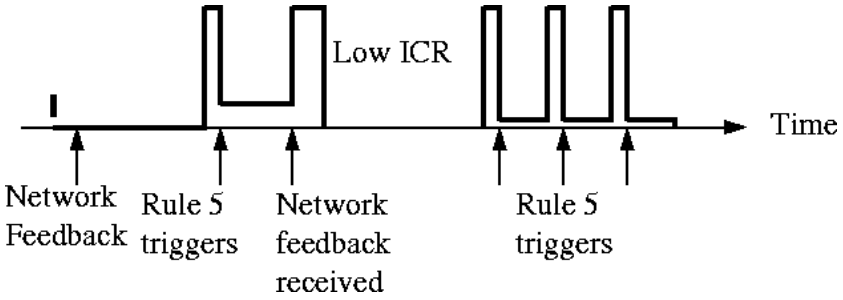


Figure 7.12: Effect of UILI on Large Bursts

Figure 7.12 shows the behavior of large bursts with the August 1995 proposal. When the burst starts, UILI triggers when the first RM cell is sent and brings the rate to ICR. Some part of the burst is transmitted at ICR. When network feedback is received, the ACR increases to the network directed value. If ICR is not very low there are no further oscillations and normal increase is not hampered. However if ICR is very low UILI is triggered after the ACR increase bringing the rate down to ACR again. The cycle is repeated and UILI triggers multiple times during the transmission of the burst resulting in low effective throughput and high burst response time.

The time-based UILI avoids the multiple triggering of UILI. It triggers once when the burst starts, and reduces the ACR proportional to the idle time. The count-based UILI also triggers once, and reduces the ACR by a constant value. Since the burst size is large, for large idle times (> RTT), the network protection provided by the

count-based technique may be insufficient. However under such conditions a different SES rule (rule 6) can provide the required network protection.

7.1.13 Summary of UILI Alternatives and ATM Forum Decision

The ATM Forum debated considerably over the UILI issue in December 1995 before putting the issue to vote. The summary of the arguments were the following:

The UILI policy can be implemented in switches or in NICs (sources) or both. The advantage of switch-only implementation is that NICs are simpler. The advantage of NIC implementation is that switches can be more aggressive in their bandwidth allocation without worrying about long-term implications of any one allocation. Without source-based UILI, the switches have to provision buffers to allow for overallocation of bandwidth.

Finally, the ATM Forum decided not to standardize an elaborate source-based UILI policy. A simple timeout is mandated for the source, where sources keep their rate allocations until a timeout (parameter ATDF, of the order of 500 ms) expires. After the timeout expires, ACR is reduced to ICR. The burden of implementing UILI is on the switches. However, NIC manufacturers can optionally implement a source-based UILI policy. The Informative appendix I.8 of the ATM Traffic Management 4.0 specification [32] briefly describes some source-based policies including the joint source-based proposal. The purpose of this paper has been to describe and evaluate the performance of various options.

7.2 Issues with Low Rate Sources

Normally, the RM cells sent by the source or turned around by the destination are counted in the source's rate (ACR). However, SES Rule 11 of the traffic managemenet specification allows sources to send RM cells "out-of-rate" i.e., these RM cells are not counted towards the rate of the source. Out-of-rate RM cells are tagged with the CLP bit being set to one at the point of origination. The rate of such traffic is limited by the parameter TCR, which is 10 cells/s in all standard vectors. The text of the rule does not mandate the implementation of the out-of-rate RM cell policy.

We noted that if sources can set their ACRs to zero and switches can give a zero ACR feedback, then the out-of-rate mechanism is the only means to get out of the ACR = 0 situation. Further, for unidirectional VCs the reverse direction has no data to send and may be initialized with a rate of zero. Under such conditions, the out of rate mechanism is required to either obtain a non-zero rate for sending BRM cells, or for sending the BRM cell itself as an out-of-rate cell. In general, the out-of-rate mechanism can be selectively used to improve transient performance.

Another related issue for low-rate sources is the source scheduling policy. Consider the event trace (from an actual simulation) shown in figure 7.13. Prior to the first event (at 42.2 ms), the bottleneck link capacity had been grabbed by VBR traffic, leading to ACR = 0 allocation from the switch. At 42.2 ms, the out-of-rate RM policy triggers and an RM cell is sent, and the next opportunity to send is scheduled at 142.2 ms (due to the fact that ACR is currently zero, and the TCR value is 10 cells/s). Now new non-zero feedback is received from 43.2 ms (the switch uses the RM cells remaining in the network, including the out-of-rate RM cell). However, since the next send opportunity was scheduled at 142.2 ms, the rate allocations are ineffective

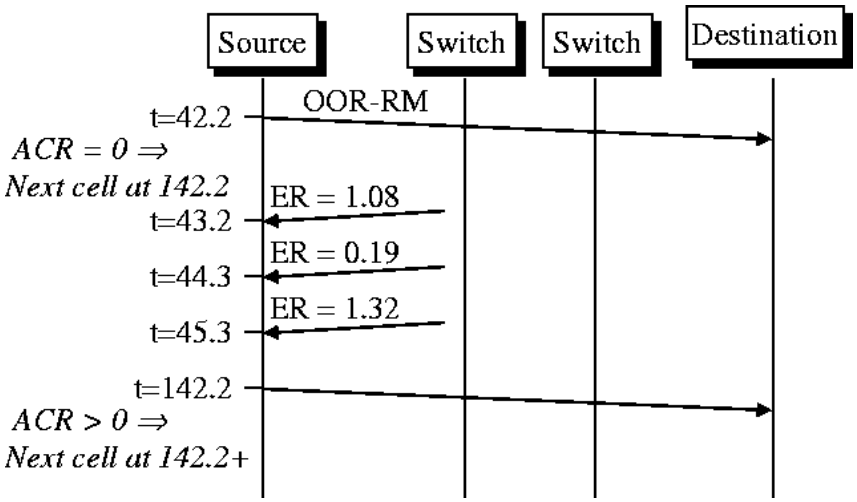


Figure 7.13: An event trace illustrating need for a rescheduling mechanism

till 142.2 ms, resulting in an unnecessary idle interval. In other words, the source once stopped is unable to use bandwidth for 100 ms even if it becomes available. The bandwidth is left unused not because there are no RM cells, but because the network feedback is ignored.

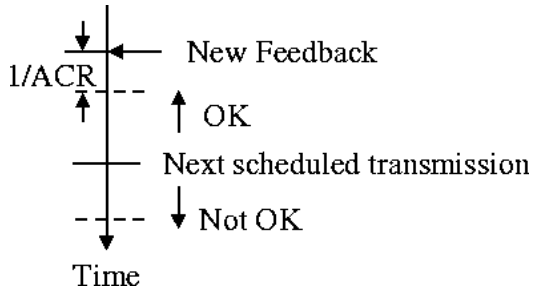


Figure 7.14: The Rescheduling Mechanism

To tackle this situation, we proposed that the sources may optionally "reschedule" their cell transmission opportunities based upon feedback. Specifically, the source

may reschedule upon receipt of new feedback if the next cell transmission opportunity calculated with the new ACR is earlier than the one currently scheduled. In terms of pseudo code:

IF(time_to_send > (now + 1/acr)

THEN time_to_send $\leftarrow$ (now + 1/acr)

This mechanism is also illustrated in figure 7.14.

7.3 Summary of Source Rule Design Issues

As explained in chapter 2, source and destination end system rules are important in complimenting the switch feedback calculation mechanisms. Specifically, the source rules provide "open-loop" control which is effective in cases when the source starts sending data after idle periods, and/or when the switch feedback to the sources is disrupted. Further, the sources have to consider the scheduling of RM and data cells, especially in the case of low rate traffic. This dissertation work has addressed the standardization aspects of Use-it-or-Lose-It policies (the issue arises when sources start sending data after idle periods), and that of low rate sources. Specifically, this work has helped design some of the source rules of the international standard (SES Rules 5, 9, 11, and 13).

CHAPTER 8

SUPPORTING INTERNET APPLICATIONS OVER THE ATM-ABR SERVICE

With the proliference of multimedia traffic over the Internet, it seems natural to move over to ATM technology which has been designed specifically to support integration of data, voice, and video applications. While multimedia applications are still in the development stage, most of the traffic on the Internet today is data traffic in the sense that they are bursty and relatively delay insensitive. It is, therefore, natural to ask how the current applications will perform over the ATM technology.

Although ATM technology has been designed to provide an end-to-end transport level service and so, strictly speaking, there is no need to have TCP or IP if the entire path from source to destination is an ATM path. However, in the forseeable future, this scenario is going to be rare. A more common scenario would be where only part of the path is ATM. In this case, TCP is needed to provide the end-to-end transport functions (like flow control, retransmission, ordered delivery) and ATM networks are used simply as "bit pipes" or "bitways."

Since the Available Bit Rate (ABR) and the Unspecified Bit Rate (UBR) service classes have been developed specifically to support data applications, it is important to investigate the performance of dominant internet applications like file transfer and

world wide web (which use TCP/IP) running over ABR and UBR. In this dissertation, we concentrate on the performance of TCP/IP over ATM-ABR. ATM-UBR performance has been examined in several recent studies [29, 28, 67, 35]. The aforementioned TCP studies also compare UBR performance with ABR using either EFCI switches [28] or explicit rate (ER) switches in local area network (LAN) topologies. Since LANs have short feedback loops, some properties of the ABR control mechanisms may not be clearly observed in LAN configurations. In this chapter, we provide a more detailed study of the dynamics and performance of TCP over ABR.

In the UBR service class, the only degree of freedom to control traffic is through scheduling, buffer allocation and cell drop policies. ABR has additional degrees of freedom in terms of switch schemes and source parameters. The ABR service requires network switches to constantly monitor their load and feed the information back to the sources, which in turn dynamically adjust their input into the network. The Transport Control Protocol (TCP), at the same time, uses packet loss in the subnetwork as an implicit feedback indicating network congestion and reduces its data load on the network. This mechanism is called the "Slow Start" congestion avoidance mechanism [40]. There is currently a debate in the networking community about the need for ABR service particularly in light of TCP's built-in congestion control facilities. We address some of these issues in this chapter.

We first study the dynamics of TCP traffic over ATM, the effect of cell loss, and the interaction of TCP with the ABR congestion control mechanisms. We find that TCP performs best when it does not experience packet loss. For the ABR service, we quantify the amount of buffering required at the ATM switches to avoid TCP packet loss. Specifically, we find that ABR is scalable over TCP in the sense that it requires

buffering which does not depend upon the number of connections. The amount of buffering depends upon factors such as the switch congestion control scheme used, and the maximum round trip time (RTT) of all virtual circuits (VCs) through the link. On the other hand, the UBR service is not scalable in the sense that it requires buffering proportional to the sum of the TCP receiver windows of all sources.

The above observations are true for applications like file transfer which have persistant demand characteristics. We verify that the requirements hold even in the presence of highly VBR background traffic (including multiplexed MPEG-2 video traffic). However, when TCP applications are bursty (i.e., have active and idle periods), it is possible that the network is overloaded by a burst of data from a number of TCP sources simultaneously. While there can be little guarantees under such pathological workloads, we find that our observations about buffer requirements hold for a large number of World Web Web (real-life bursty) applications running over TCP.

8.1 TCP control mechanisms

TCP is one of the few transport protocols that has its own congestion control mechanisms. The key TCP congestion mechanism is the so called "Slow start." TCP connections use an end-to-end flow control window to limit the number of packets that the source sends. The sender window is the minimum of the receiver window (Wrcvr) and a congestion window variable (CWND).

Whenever a TCP connection loses a packet, the source does not receive an acknowledgment and it times out. The source remembers the congestion window (CWND) value at which it lost the packet by setting a threshold variable SSTHRESH

at half the window. More precisely, SSTHRESH is set to max{2, min{CWND/2, Wrcvr}} and CWND is set to one.

The source then retransmits the lost packet and increases its CWND by one every time a packet is acknowledged. We call this phase the "exponential increase phase" since the window when plotted as a function of time increases exponentially. This continues until the window is equal to SSTHRESH. After that, the window w is increased by $1/w$ for every packet that is acked. This is called the "linear increase phase" since the window graph as a function of time is approximately a straight line. Note that although the congestion window may increase beyond the advertised receiver window, the source window is limited by that value. when packet losses occur, the retransmission algorithm may retransmit all the packets starting from the lost packet. That is, TCP uses a go-back-N retransmission policy. The typical changes in the source window plotted against time are shown in Figure 8.1.

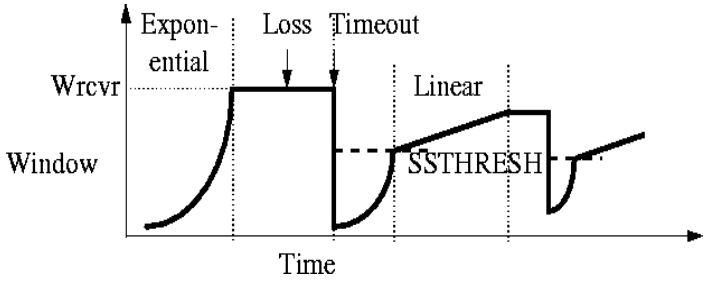


Figure 8.1: TCP Window vs Time using Slow Start

When there is a bursty loss due to congestion, time is lost due to timeouts and the receiver may receive duplicate packets as a result of the go-back-N retransmission strategy. This is illustrated in Figure 8.2. Packets 1 and 2 are lost but packets 3 and

4 make it to the destination are are stored there. After the timeout, the source sets its window to 1 and retransmits packet 1. When that packet is acknowledged, the source increases its window to 2 and sends packets 2 and 3. As soon as the destination receives packet 2, it delivers all packets upto 4 to the application and sends an ack (asking for packet 5) to the source. The 2nd copy of packet 3, which arrives a bit later is discarded at the destination since it is a duplicate.

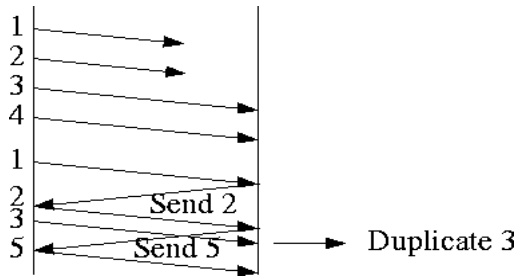


Figure 8.2: Timeout and Duplicate Packets in Slow Start

8.2 Closed Loop vs Open Loop Control Revisited

The ABR service provides flow control at the ATM level itself. When there is a steady flow of RM cells in the forward and reverse directions, there is a steady flow of feedback from the network. In this state, we say that the ABR control loop has been established and the source rates are primarily controlled by the network feedback (closed-loop control). The network feedback is effective after a time delay. The time delay required for the new feedback to take effect is the sum of the time taken for an RM cell to reach the source from the switch and the time for a cell (sent at the new rate) to reach the switch from the source. This time delay is called the "feedback delay."

When the source transmits data after an idle period, there is no reliable feedback from the network. For one round trip time (time taken by a cell to travel from the source to the destination and back), the source rates are primarily controlled by the ABR source end system rules (open-loop control). The open-loop control is replaced by the closed-loop control once the control loop is established. When the traffic on ABR is "bursty" i.e., the traffic consists of busy and idle periods, open-loop control may be exercised at the beginning of every active period (burst). Hence, the source rules assume considerable importance in ABR flow control.

8.3 Nature of TCP Traffic at the ATM Layer

Data which uses TCP is controlled first by the TCP "slow start" procedure before it appears as traffic to the ATM layer. Suppose we have a large file transfer running on top of TCP. When the file transfer begins, TCP sets its congestion window (CWND) to one. The congestion window increases exponentially with time. Specifically, the window increases by one for every ack received. Over any round trip time (RTT), the congestion window doubles in size. From the switch's point of view, there are two packets input in the next cycle for every packet transmitted in the current cycle (a cycle at a bottleneck is defined as the largest round trip time of any VC going through the bottleneck). In other words, the load (measured over a cycle) at most doubles every cycle. In other words, ***initially, the TCP load increases exponentially.***

Though the application on top of TCP is a persistant application (file-transfer), as shown in Figure 8.3, *the TCP traffic as seen at the ATM layer is bursty (i.e., has active and idle periods).* Initially, there is a short active period (the first packet is sent) followed by a long idle period (nearly one round-trip time, waiting for an

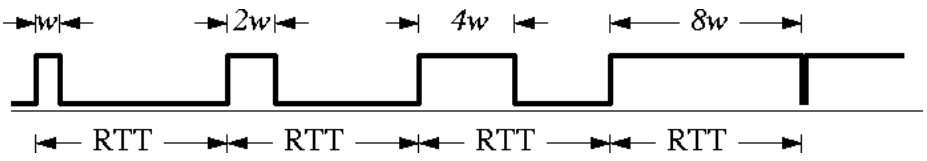


Figure 8.3: At the ATM layer, the TCP traffic results in bursts. The burst size doubles every round trip until the traffic becomes continuous.

ACK). The length of the active period doubles every round-trip time and the idle period reduces correspondingly. Finally, the active period occupies the entire round-trip time and there is no idle period. After this point, the TCP traffic appears as an infinite (or persistant) traffic stream at the ATM layer. Note that the total TCP load still keeps increasing unless the sources are controlled. This is because, for every packet transmitted, some TCP source window increases by one which results in the transmission of two packets in the next cycle. However, since the total number of packets transmitted in a cycle is limited by the delay-bandwidth product, ***the TCP window increases linearly after the bottleneck is fully loaded.*** Note that the maximum load, assuming sufficient bottleneck capacity, is the sum of all the TCP receiver windows, each sent at link rate.

When sufficient load is not experienced at the ABR switches, the switch algorithms typically allocate high rates to the sources. This is likely to be the case when a new TCP connection starts sending data. The file transfer data is bottlenecked by the TCP congestion window size and not by the ABR source rate. In this state, we say that the TCP sources are window-limited.

The TCP active periods double every round trip time and eventually load the switches and appear as infinite traffic at the ATM layer. The switches now give

feedback asking sources to reduce their rates. The TCP congestion window is now large and is increasing. Hence, it will send data at rate greater than the source's sending rate. The file transfer data is bottlenecked by the ABR source rate and not by the TCP congestion window size. In this state, we say that the TCP sources are ***rate-limited***. Observe that UBR cannot rate-limit TCP sources and would need to buffer the entire TCP load inside the network.

The ABR queues at the switches start increasing when the TCP idle times are not sufficient to clear the queues built up during the TCP active times. The queues may increase until the ABR source rates converge to optimum values. Once the TCP sources are rate-limited and the rates converge to optimum values, the lengths of the ABR queues at the switch will start decreasing. The queues now move over to the source end-system (outside the ATM network). Several proprietary techniques can be used to control the TCP queues at the edge of the ATM network [59], and we cover some of the possibilities later in this chapter.

The remaining part of the chapter is organized as follows. We first examine the performance of TCP under lossy conditions in section 8.4. We then study the interaction of the TCP and ABR congestion control algorithms and the justification and assumptions for buffer requirements for zero loss in section 8.14. Next, we look at the effect of variation in capacity (VBR backgrounds) on the buffer requirements. We discuss switch algorithm issues and our solutions to handle variation in demand and capacity in section 8.16. We then develop a model of multiplexed MPEG-2 sources over VBR, and study its effect on TCP sources running over ABR in section 8.17. Finally, we look at related work (an extension of this dissertation work), where the

issues and effect of bursty applications like World Wide Web running on TCP is examined.

8.4 TCP Performance With Cell Loss

Cell loss will occur in the network if the ATM switches do not have sufficient buffers to accomodate this queue buildup. In this section, we will show simulations to demonstrate the problem of cell loss on TCP performance and identify the factors which affect the performance under such conditions. Specifically, we show that TCP achieves peak throughput over ABR without the necessity of very large buffers (we quantify this requirement in section 8.14). Then we limit the buffer size based on the Transient Buffer Exposure (TBE) ABR SES parameter and the number of TCP sources. Though the TBE parameter was initially intended to allow some control over buffer allocation, we find that it is ineffective in preventing cell loss. We then study the effect of cell loss on TCP level packet throughput, and the various parameters affecting the performance (buffer size, number of sources, TCP timer granularity parameter, cell drop policy etc).

Specifically, when cell loss does occur, the cell loss ratio (CLR) metric, which quantifies cell loss, is a poor indicator of loss in TCP throughput. This is because TCP loses time (through timeouts) rather than cells (cell loss). Smaller TCP timer granularity (which controls timeout durations) can help improve throughput. Due to fragmentation, a single cell loss results in a packet loss. This further obscures the meaning of the CLR metric. If the ABR rates do not converge to optimum values before the cell loss occurs, the effect of the switch congestion scheme may be

dominated by factors such as the drop policy and TCP timer granularity. Intelligent drop policies can help improve the throughput slightly.

8.5 Source Model and TCP Options

We use an infinite source model at the application layer running on top of TCP. This implies that TCP always has a packet to send as long as its window will permit it. Other parameters values used are:

TCP maximum segment size MSS=512 bytes

IP MTU size = 9180 bytes (no IP segmentation)

TCP timer granularity = 100 ms

Delay-ack timer=0 (disabled)

Packet processing time at the destination=0

We implemented the window scaling option so that the throughput is not limited by path length. Without the window scaling option, the maximum window size is 2^{16} bytes or 64 kB. We use a window of 16 $\times$ 64 kB or 1024 kB. The network consists of three links of 1000 km each and therefore has a one-way delay of 15 ms (or 291 kB at 155 Mbps).

In our simulations, we have not used "fast retransmit and recovery" used in popular TCP Reno implementation. In a related work [36, 37] (TCP over UBR), we study the effect of these algorithms in detail. Briefly, these algorithms have been designed to improve TCP performance when a single (isolated) segment is lost (due to errors). However, in high bandwidth links, network congestion results in several dropped segments (a burst loss). In this case, these algorithms are not able to recover

from the loss and they trigger the TCP timeout and the slow start algorithm leading to possibly worse performance.

8.6 ABR Source End System and ERICA Parameters

The source end system parameters of ABR are selected to maximize the responsiveness and throughput. The values of source parameters are:

TBE = 128, 512

ICR = 10 Mbps

ADTF = 0.5 sec

CDF (XDF) = 0.5, CRM (Xrm) = TBE/Nrm

PCR = 155.52 Mbps, MCR = 0, RIF (AIR) = 1

Nrm = 32, Mrm = 2, RDF = 1/512,

Trm = 100 ms, TCR = 10 c/s

The ERICA switch algorithm parameters are chosen as follows. The target utilization parameter is chosen to be 90%. The overload and ABR capacity are measured at the switch over an interval of 100 cells or 1 ms (whichever is smaller). The buffer size at the bottleneck link is sized as TBE $\times$ n $\times$ 1, 2, or 4, where n is the number of ABR sources.

8.7 The n Source + VBR Configuration

Figure 8.4 illustrates the general configuration we analyze, which we call "the n Source + VBR configuration." This configuration has a single bottleneck link

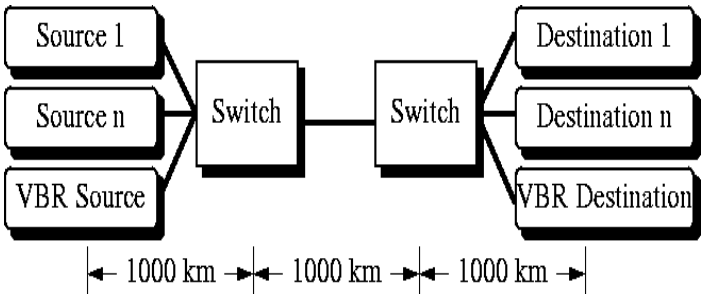


Figure 8.4: n Source + VBR Configuration

between two switches. The link capacity is shared by n ABR sources and possibly a VBR source. All links run at 155 Mbps and are 1000 km long.

The VBR background is optional. When present, it is an ON-OFF source with a 100 ms ON time and 100 ms OFF time. The VBR starts at t = 2 ms to avoid certain initialization problems. The maximum amplitude of the VBR source is 124.41 Mbps (80% of link rate). This is deliberately set below the ERICA target utilization of 90%. By doing so, we always leaves at least 10% for ABR. This avoids scheduling issues. We may safely assume that VBR is given priority at the link, i.e, if there is a VBR cell, it will be scheduled for output on the link before any waiting ABR cells are scheduled. Also, since ABR bandwidth is always non-zero, the ABR sources are never allocated zero rates. We, thus, avoid the need for out-of-rate RM cells, which are required if an ABR source is allocated an ACR of zero and cannot send any data cells.

All traffic is unidirectional. A large (infinite) file transfer application runs on top of TCP for the TCP sources. We experiment with 2 values of n = 2 and 5. The buffer size at the bottleneck link is sized as TBE $\times n\times$ {1, 2, or 4}.

8.8 Performance Metrics

We measure throughput of each source and cell loss ratio. Also, we can plot a number of variables as a function of time that help explain the behavior of the system. These include TCP sequence numbers at the source, congestion window (CWND), ACR of each source, link utilization, and queue length.

We define TCP throughput as the number of bytes delivered to the destination application in the total time. This is sometimes referred to as goodput by other authors. Cell Loss Ratio (CLR) is measured as the ratio of the number of cells dropped to the number of cells sent during the simulation.

The following equation should hold for the aggregate metrics of the simulation:

Number of bytes sent = Bytes sent once

+ Bytes retransmitted

= Bytes delivered to application

+ Data bytes dropped at the switch + Bytes in the path

+ Partial packet bytes dropped at the destination AAL5

+ Duplicate packet bytes dropped at the destination TCP

The places where cells or packets are dropped are illustrated in Figure 8.5.

8.9 Peak TCP Throughput

In order to measure the best possible throughput of TCP over ABR, we first present the results of a case with infinite buffers and fixed ABR capacity. With finite buffers or variable ABR capacity, it is possible that some cells are lost, which may

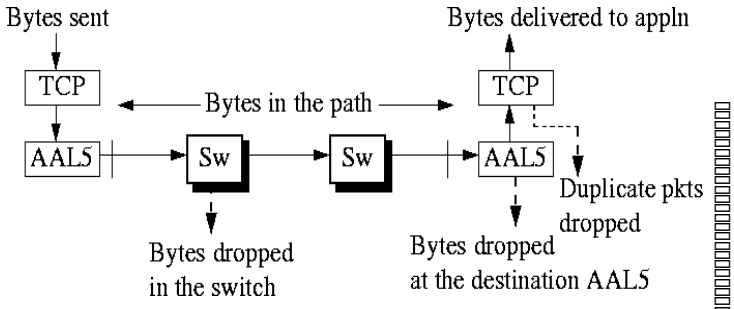


Figure 8.5: Cell/Packet Drop Points on a TCP/ATM connection

result in unnecessary timeouts and retransmissions leading to reduced throughput. Fixed ABR capacity is achieved by not having any VBR source in this case.

We simulate the configuration with $n = 2$, buffer size = 4096 and TBE = 512. In this case, no cells are lost, the CLR is zero and the throughput is 103.32 Mbps. This is the maximum TCP throughput with two sources in this configuration. It can be approximately verified as follows:

Throughput = 155 Mbps

$\times$ 0.9 for ERICA Target Utilization

$\times$ 48/53 for ATM payload

$\times$ 512/568 for protocol headers

(20 TCP + 20 IP + 8 RFC1577 + 8 AAL5 = 56 bytes)

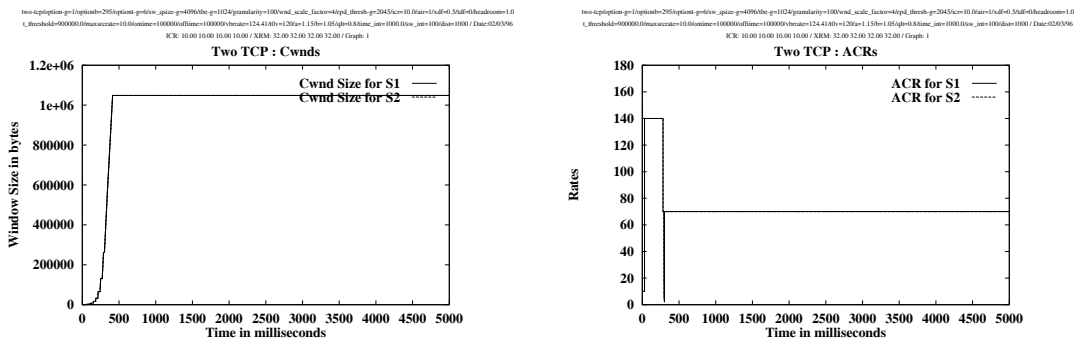
$\times$ 31/32 for ABR RM cell overhead

$\times$ a fraction (0.9) to account for the TCP startup time

$\simeq$ 103.32 Mbps

Figure 8.6 shows graphs of window size, sequence numbers, and ACR for the two sources. Note that the curves for the two sources completely overlap indicating that

the performance is fair. Also, the sources use the entire ACR allocated to them. In other words,*the TCP sources are rate-limited and not window-limited.* Note that given sufficient time, the ABR switch algorithm can control the rates of the VCs carrying TCP traffic. We shall quantify this time and corresponding buffer requirements in section 8.14 later in this chapter.



(a) Congestion Window (b) ACR

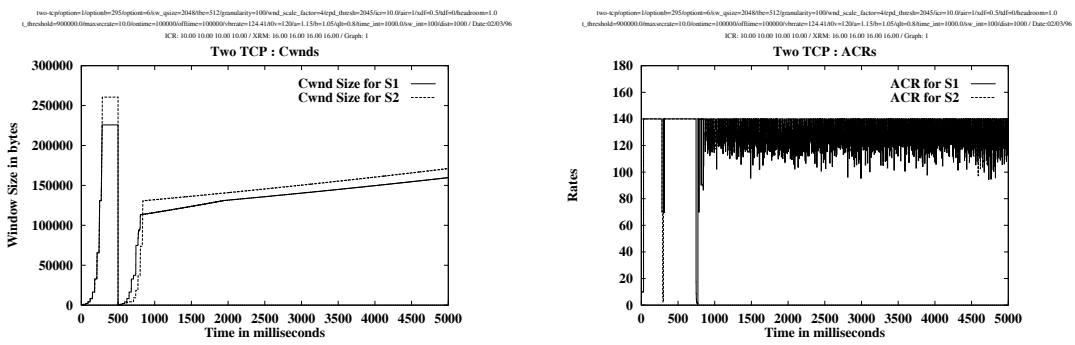
Figure 8.6: Two TCP Source Configuration, Buffer=4096 cells, TBE=1024

8.10 Effect of Finite Buffers

We now investigate the effect of smaller buffers, keeping the ABR capacity fixed. The buffer size is set to the product of TBE (512), the number of sources (2), and a safety factor (2), i.e., $2048 = 512 \times 2 \times 2$. The remaining configuration is the same as in Section 8.9 i.e., $n = 2$, TBE $= 512$ and fixed ABR capacity (no VBR source). Since the buffers are smaller, it is possible that they might overflow before the ABR control loop is set up. We expect some cell loss and reduced throughput due to timeout retransmission.

We observe that there is a drastic reduction of TCP throughput which is not proportional to the increase in CLR. The throughput drops by 36% while the CLR is only 0.18%.

Figure 8.7 shows graphs of window size, sequence numbers, and ACR for the two sources. Figure 8.7(a) shows that there is one loss around t=200 ms. No acks are received for the next 300 ms and therefore, the window remains constant and finally drops to 1 at t=500 ms. The packets are retransmitted and window rises exponentially upto the half of the value before the drop. Subsequently, the window rise linearly. Note that the linear rise is very slow. The source window is much below its maximum. In other words, the sources are window limited. The congestion windows of both sources are approximately equal, and so the operation is fair. However, the throughput in this experiment is only 64% of the maximum throughput. The measured cell loss ratio in this case was only 0.18%. Note that the *CLR and throughput loss are one order of magnitude apart.*



(a) Congestion Window (b) ACR

Figure 8.7: Two TCP Source Configuration, Buffer=2048 cells, TBE=512

Figure 8.7(b) shows the rates (ACRs) allocated to the two sources. Notice that the curves for the two sources and are at the maximum possible value (90% of the link rate) and so the sources have a large ACR. The reason for throughput being less than maximum possible is not the sources' ACRs but their windows. That is,*the sources are not rate-limited but are window-limited.* Also, notice that the two curves overlap. This shows that the ABR rate allocation mechanism is fair.

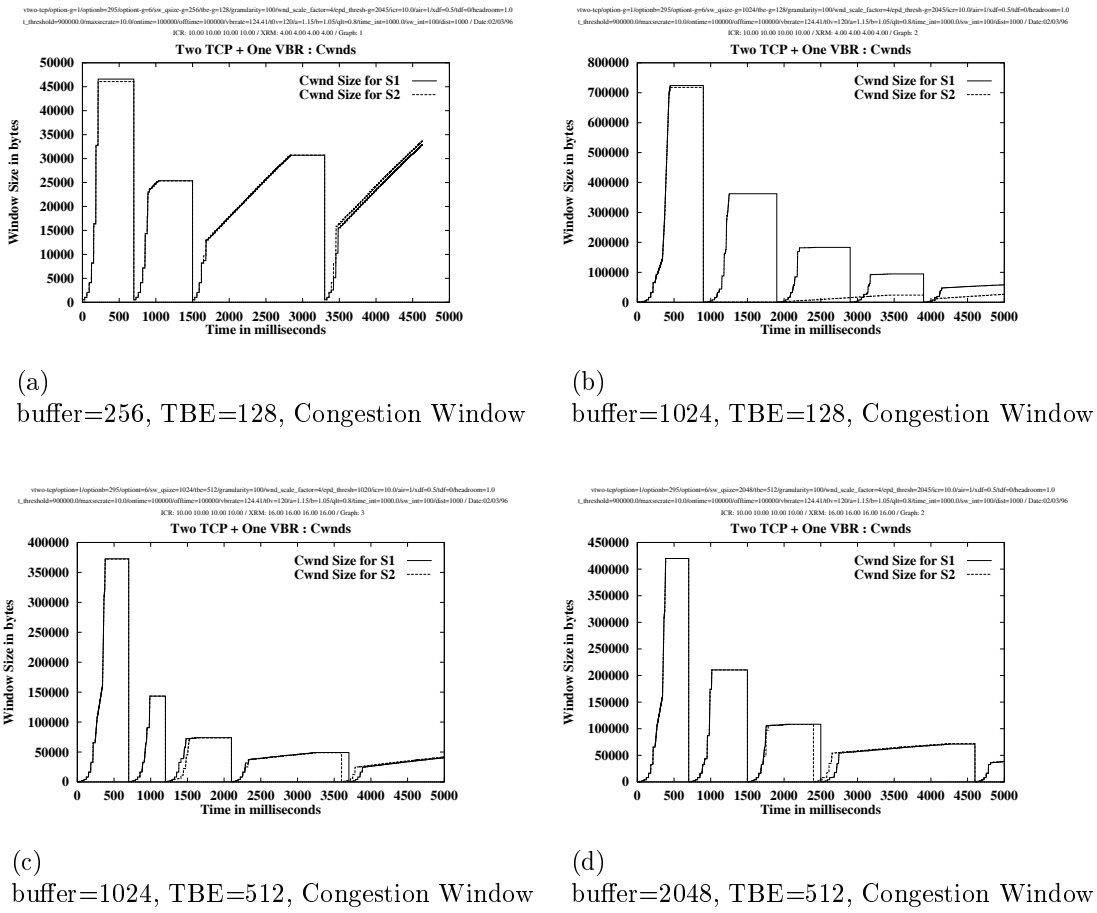
The main reason for the large drop in throughput is that cells (packets) are dropped. Each cell loss results in a significant loss of time and throughput. In this case, this happens before the ABR control loop is set up (open-loop period). The TBE in this case was 512. For two sources, one would assume that having 1024 buffers in the switch would be sufficient. But this case shows that cells are lost even when there are twice as many (2048) buffers in the switch. Thus, *TBE is not a good mechanism to control or allocate buffers.* This observation was also made in our earlier work on non-TCP bursty traffic [53, 27].

8.11 Effect of Finite Buffers and Varying ABR Capacity

Next we studied the effect of varying ABR capacity. For this purpose, we introduce a VBR traffic in the background. We conducted several experiments with two and five ABR sources. Since VBR takes up 40% of the link bandwidth, we expect the maximum ABR throughput to be 60% of the case without VBR.

Figures 8.8 and 8.9 show the window graphs for the two- and five-source source configurations, respectively. Four different TBE and buffer size combinations are used. The graphs clearly show the instants when cells are lost and the TCP windows

adjusted. The ACR and sequence number graphs have not been included here since there is not much new information in them.



(a)
buffer=256, TBE=128, Congestion Window

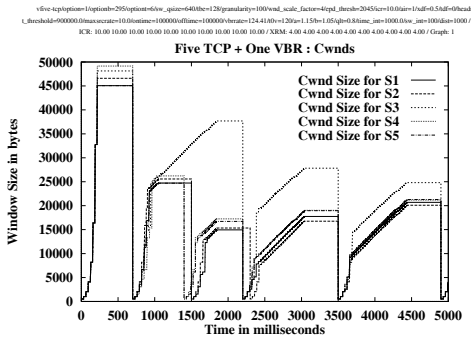
(b)
buffer=1024, TBE=128, Congestion Window

(c)
buffer=1024, TBE=512, Congestion Window

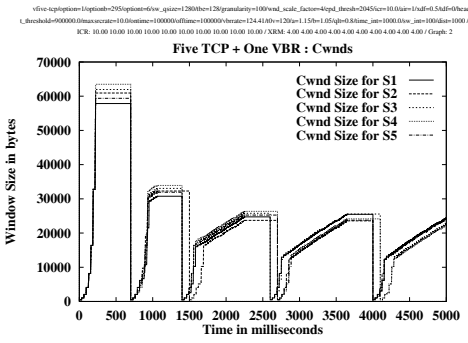
(d)
buffer=2048, TBE=512, Congestion Window

Figure 8.8: Two TCP + One VBR Configuration, TBE vs Buffer

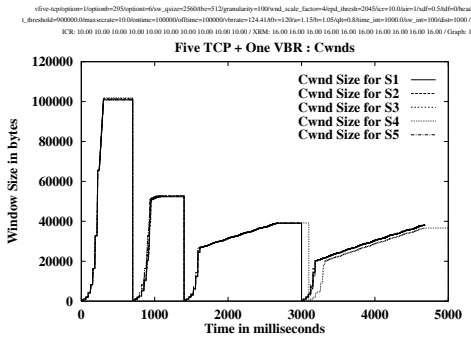
The simulation results are summarized in Table 8.1 and are discussed in the following subsection. The first column is the configuration used. The second and third columns show the TBE and the buffer sizes used. T1 through T5 are the throughput values for sources 1 through 5. We also show the total ABR throughput. It is helpful



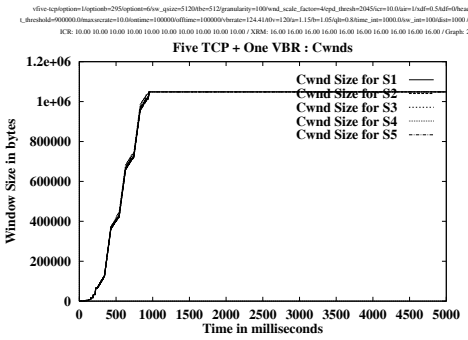
(a)
buffer=640, TBE=128, Congestion Window



(b)
buffer=1280, TBE=128, Congestion Window



(c)
buffer=2560, TBE=512, Congestion Window



(d)
buffer=5120, TBE=512, Congestion Window

Figure 8.9: Five TCP + One VBR Configuration, TBE vs Buffer

to express it as a percentage of maximum possible ABR throughput (58.4 Mbps). The last column shows the CLR.

From this table we can make the following conclusions:

1. CLR vs Throughput: Table 8.1 shows that that *CLR is small and has high variance*. CLR does not reflect TCP performance since higher CLR does not necessarily mean lower TCP throughput. The effect of cell loss depends not upon the number of cells lost but upon the the number of timeouts. If a large

Number of Sources	TBE	Buffer	Throughput T1	T2	T3	T4	T5	Total	%	CLR
2 A + V	128	256	3.1	3.1				6.2	10.6	1.2
2 A + V	128	1024	10.5	4.1				14.6	24.9	2.0
2 A + V	512	1024	5.7	5.9				11.6	19.8	2.7
2 A + V	512	2048	8.0	8.0				16.0	27.4	1.0
5 A + V	128	640	1.5	1.4	3.0	1.6	1.6	9.1	15.6	4.8
5 A + V	128	1280	2.7	2.4	2.6	2.5	2.6	12.8	21.8	1.0
5 A + V	512	2560	4.0	4.0	4.0	3.9	4.1	19.9	34.1	0.3
5 A + V	512	5720	11.7	11.8	11.6	11.8	11.6	58.4	100.0	0.0

Table 8.1: Simulation Results: Summary

number of cells are lost but there is only one timeout, the throughput degradation may not be that severe. On the other hand, even if a few cells are lost but the losses happen far apart triggering multiple timeouts, the throughput will be severely degraded. Hence, the cell level metric *CLR is not a good indicator of the TCP level performance.*

2. Effect of Buffering: *Larger buffers always give higher TCP throughput* for our infinite TCP applications.

   We study the effect of buffers on latency in section 8.14. Briefly, since the ABR control can drain out the queues after the initial transient, the average latency should be low. The effect of new TCP sources starting up does not increase the latency significantly since they start at a window of one MSS, irrespective of their ICRs.

The effect of large buffers on CLR is mixed. With large buffering, windows can be large and if the a loss occurs at a large window, CLR can be high. On the other hand, if the loss occurs at a low window, CLR can be low.

3. Effect of Multiple Sources: *As the number of sources is increased, generally the total throughput increases.* This is because, these TCP sources are generally window limited and five sources with small windows pump more data than two sources with small windows.

8.12 Observations on Tail Drop

In this section we report an interesting phenomenon due to tail drop and propose a simple fix. In AAL5, sources mark the last cell of each message by End-of-Message (EOM) bit. If the EOM cell is dropped at the switch, the retransmitted packet gets merged with previous partial packet at the destination. The merged packet fails the CRC test and is dropped at the destination by AAL5. The source will have to retransmit two packets.

After the first retransmission, the SSTHRESH is set to half the previous window size and the window is set to one. When the second retransmission occurs, the window is one and hence SSTHRESH is set to 2 (the minimum value). The window remains at one. TCP henceforth increases the window linearly resulting in low throughput for this source. Since the EOM cells of the other TCP sources may not have been dropped, they do not experience this phenomenon and get high throughput.

The disparity in throughput results in unfairness among sources as shown in Figure 8.10. Figure 8.8(b) shows a simulation where this unfairness is seen. In this figure, source S2 loses cells at 400 ms and 1300 ms. The corresponding timeout and

retransmissions occur at 900 ms and 1900 ms. The merging of the packets at the AAL5 occurs at 1300 ms. After the second timeout, the window of S2 increases linearly from one. Since source S1 does not experience this phenomenon, it gets higher throughput.

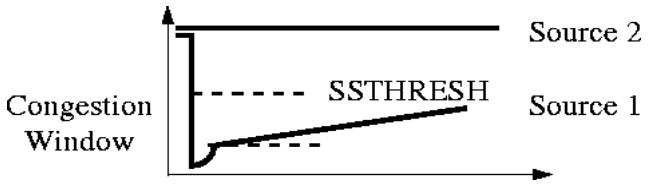


Figure 8.10: Unfairness due to TailDrop

A simple fix is what we call Intelligent Tail Drop. This policy sets a threshold a few cells before the buffer limit. Once the threshold is crossed, the switch drops all cells *except the first EOM cell*. The EOM cell will reach the destination and result in the dropping of the first packet and *merging of packets is avoided in the destination AAL5*. Whenever any cells are dropped, the switch should ensure that the next EOM cell is transmitted. This prevents the back-to-back retransmissions and *improves fairness*. Since this policy only enhances tail drop, it can still be used in conjunction with other drop policies like Early Packet Discard (EPD) [75]. A similar policy for partial packet discard is described in Reference [5]. Drop policies assume importance for the UBR service class and have been studied in a related work [36].

8.13 Summary of TCP/IP performance over ABR under lossy conditions

We have studied the effect of running TCP/IP traffic with ABR. The main results of the study are:

1. TCP achieves maximum throughput when there are enough buffers at the switches. We will quantify this requirement, and argue the case for scalability in the next section.

2. When maximum throughput is achieved, the TCP sources are rate-limited by ABR rather than window-limited by TCP.

3. When the number of buffers is smaller, there can be a large reduction in throughput even though CLR is very small.

4. The reduction in throughput is due to loss of time during timeouts (large timer granularity), and transmission of duplicate packets which are dropped at the destination.

5. When throughput is reduced, the TCP sources are window-limited by TCP rather than rate-limited by ABR.

6. Switch buffers should not be dimensioned based on the ABR Source parameter TBE. Dimensioning should be based upon the performance of the switch algorithm, and the round trip time, as discussed in the next section.

7. When ABR capacity is varied, CLR exhibits high variance and is not related to TCP throughput. In general, CLR is not a good indicator of TCP level performance.

8. Larger buffers increase TCP throughput.

9. Larger number of window-limited sources increase TCP throughput. This is because, the sum of the windows is larger when there are more sources.

10. Even when the buffers are small, dropping of EOM cells should be avoided. This avoids merging of packets at the destination AAL5 and improves fairness. When sufficient buffers are provided for ABR, the drop policy assumes only a minor importance, unlike its role in the UBR service.

8.14 Buffering Requirements for TCP over ABR

In this section, we analyze the buffer requirement at switches for TCP over the ATM-ABR service. We show by a combination of emperical and analytical studies that the buffer requirement for TCP over ABR for zero loss transmission is:
($a \times$ RTT + $b \times$ Averaging Interval Length + $c \times$ feedback delay) $\times$ link bandwidth, for low values of the coefficients
This requirement is heavily dependent on the switch algorithm. With the ERICA+ algorithm, typical conservative values of the coefficients are $(a = 3, b = 1, c = 1)$.

The formula is a linear relation on three key factors:

Round trip time (RTT): Twice the delay through the ABR network or segment (delimited by VS/VD switch(es)).

Averaging Interval Length: A quantity which captures the measurement aspects of a switch congestion control algorithm. Typical measured quantities are: ABR capacity, average queue length, ABR input rate, number of active sources, and VC's rate.

Feedback delay: Twice the delay from the bottleneck to the ABR source (or virtual source). Feedback delay is the minimum time for switch feedback to be effective.

Note that the formula does not depend upon the number of TCP sources. This fact implies that ABR can support TCP (data) applications in a scalable fashion. The buffer requirement is also an indication of the maximum delay through the network. Note that this is a worst case requirement and the average delay is much smaller due the congestion avoidance mechanisms at the ATM layer. As a result, ABR is a better fit for scalable support of interactive applications which involve data large transfers (like web-based downloading etc).

8.14.1 Assumptions

In the above formula, we have assumed that the traffic using TCP is a persistant kind of traffic (like a large file transfer). Note that it is possible for TCP to keep its window open for a while and not send data. In the worst case, if a number of TCP sources keep increasing their TCP windows slowly (during underload), and then synchronize to send data, the queue seen at the switch is the sum of the TCP windows [80].

Variation in ABR demand and capacity affects the feedback given by the switch algorithm. If the switch algorithm is highly sensitive to variation, the switch queues may never be bounded since, on the average, the rates are never controlled. The buffer requirement above assumes that the switch algorithm can tolerate variation in ABR capacity and demand. We discuss this issue further in section 8.16.

Also, in the above formula, we are assuming that the product of the number of active TCP sources times the maximum segment size (MSS) is small compared to the

buffer requirement derived. We are also assuming that the applications running on top of TCP are persistant (large file transfer type applications). Also note that the buffer requirement is for the switches only. In other words, the queues are pushed by ABR to the edged of the network, and the edge routers need to use proprietary mechanisms to manage the edge queues. We shall address these assumptions, and their implications at the end of the section.

Note also that, under certain extreme conditions (like large RTT of satellite networks) some of the factors (RTT, feedback delay, Averaging interval) may dominate over the others (eg: the feedback delay over the round trip time in satellite networks). Another scenario is a LAN where the averaging interval dominates over both RTT and feedback delay. The round trip time for a ABR segment (delimited by VS/VD switches) is twice the maximum one-way delay within the segment, and not the end-to-end delay of any ABR connection passing through the segment. These factors further reduce the buffer requirements in LAN switches interfacing to large networks, or LAN switches which have connections passing through segmented WANs.

8.14.2 Derivation of the buffer requirement

1. **Initial TCP behavior:** TCP load doubles every RTT initially when the bottleneck is not loaded (see also section 8.3). During this phase, TCP sources are window-limited, i.e., their data transmission is bottlenecked by their congestion window sizes and not by the network directed rate.

   Initially all the TCP sources are in their exponential rise phase. In its exponential rise phase, TCP doubles its window every RTT. For every cell output

on a link, two cells are expected as input to the link in the next RTT. This is true irrespective of the number of sources.

This can be viewed in three ways:

- the number of cells in the network (in an RTT) doubles
- the overload measured over an RTT doubles
- the "active period" of the burst doubles every RTT

2. **Time to reach rate-limited operation**: The minimum number of RTTs required to reach rate-limited operation decreases as the logarithm of the number of sources. In other words, the more the number of sources, the faster they all reach rate-limited operation. Rate-limited operation occurs when the TCP sources are constrained by the network directed ABR rate rather than their congestion window sizes. The minimum number of RTTs required is derived as follows:

   Suppose we find that find that TCP packets are available, but the source is not transmitting. There are two reasons for this: either the source has exhausted its window, or it is waiting for the next transmission opportunity at the current ACR. In the first case, we call the source *window-limited*. In the second case, it is *rate-limited*.

   Initially, the window is small (starting from one). The sources are window-limited (and not rate-limited). That is, each source exhausts its window and may remain idle for a while before it sends its next burst. As observed, the number of cells in the network doubles every RTT. Stable closed loop rate-control can be established only after there are enough cells to fill the pipe. The

number of cells in the first RTT depends upon the number of sources starting up together (one MSS for each source). Henceforth, the number of cells only double irrespective of the number of sources. Hence, the number of RTTs required to fill the pipe depends upon the number of sources as follows:

After K RTTs,

window $W = 2^{K-1} \times MSS$

Note that MSS = 512 bytes + Overhead = 12 cells

Or,

Effective rate = $MSS \times 2^{K-1} \times N/RTT$

Here, N = number of sources

The pipe is filled when the effective input rate first exceeds capacity (or overload becomes greater than 1, for ERICA)

$MSS \times 2^{K-1} \times N/RTT \geq L$

Here, L is the link rate in cells per second.

or

$K - 1 = \log_2 (\frac{Link\_Bandwidth \times RTT}{(MSS * N)})$

This shows that the number of RTTs decreases as the log of N. Note that once the link (or path/pipe) becomes fully loaded, the TCP windows increase linearly and not exponentially (see section 8.3 earlier in this chapter). However, the sources may still not be rate-limited (the ACRs are still large, though the bottleneck load is greater than unity). The sources reach rate-limited steady

state only after the switch algorithm brings the bottleneck load down to unity again by reallocating the rates.

3. **Switch algorithm issues:** We have claimed that the switch algorithm cannot, in general, give correct feedback when the network load is bursty (i.e., has active and idle periods). Some of problems observed by common switch algorithms are discussed below:

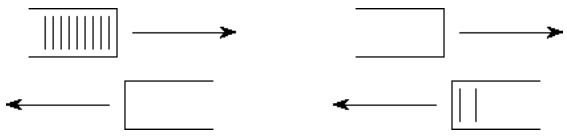


Figure 8.11: Out-of-phase Effect (TCP over ABR)

a) Out-of-phase effect: No load or sources are seen in the forward direction while sources and RM cells are seen in the reverse direction (figure 8.11).

Figure 8.12: Clustering Effect (TCP over ABR)

b) Clustering effect: The cells from TCP connections typically come in clusters [82] (figure 8.12). Hence, the activity of multiple connections is difficult to sense over small averaging intervals, though the corresponding load may be high.

c) Variation in load: As described in section 8.3, even an infinite traffic source running on top of TCP looks like a bursty source at the ATM layer. When a number of such sources aggregate, the load experienced at the switch can be highly variant.

d) Variation in capacity: The ABR capacity depends upon the link bandwidth, and the bandwidth usage of the higher priority classes like CBR and VBR, and can exhibit variation accordingly.

Due to these effects, switches may make errors in measuring quantities which they use to calculate feedback. Due to the out-of-phase effect, the input rate and overload measured in ERICA over the last interval is zero, because no cells are seen in the forward direction. Due to the clustering effect, the number of active sources may be underestimated in any interval (for example, N different sources may be seen in each interval when the total number of sources is 2N), leading to overallocation of rates in ERICA.

The third problem is variation in load. Due to the variation in load, it is possible to have a long period of underload, followed by a sudden burst which builds queues. As a result the maximum queue may be large even though the utilization/throughput is low. Schemes like ERICA can track the variation in load and filter it, because we use the average load as a metric. However, several schemes use the queue length metric exclusively. Queue length has a higher variation than the average load, and it also varies depending upon the available capacity. Further, a queue length of zero yields little information about the utilization of the link. It has been argued that schemes which look at only the

queue length is less susceptible to errors than schemes which use several metrics (like input rate, MACR , number of active sources etc). But, the use of several independent metrics gives more complete information about the system [49], and variation reduction can be done by using simple averaging techniques.

The fourth problem is the effect of ABR capacity variation. This effect, when combined with the latency in giving feedback to sources, results in an alternating series of high and low rate allocations by the switch. If the average total allocation exceeds the average capacity, this could result in unbounded queueing delays.

These effects reduce as the network path gets completely filled by TCP traffic, and the ABR closed loop control becomes effective. The switch scheme then controls the rate of the sources. Note that we have designed averaging techniques in ERICA (see chapter 6) specifically to counter such conditions, i.e., reduce the error in measurement and handle boundary cases. The residual error even after these modifications manifests as queues at the bottleneck. We handle this using the queue control algorithm in ERICA which scales down the ABR capacity as a function of queueing delay.

4. **Switch algorithm convergence time:** After the pipe just becomes full (TCP has sent data continuously for one RTT), the maximum queue which can build up before the switch algorithm convergences to steady state $2 \times \text{RTT} \times$ link bandwidth.

   Note that the first feedback after the link is fully loaded can take as much as a round trip time to be effective. Also note (from the discussing in section 8.3)

that if not controlled, the TCP load increases at most linearly after the bottleneck is loaded, even though the sources are in their exponential rise phase. In other words, the load may change in the following pattern: cycle 0, load = 0.25; cycle 1, load = 0.5; cycle 2, load = 1, cycle 3, load = 2, cycle 4, load = 3, and so on. Note that the load change from 0.25 to 1 was exponential, and then became linear because of the bottleneck becoming fully loaded. However, even the linear load increase can result in unbounded switch queues unless the sources are controlled. The switch algorithm is therefore the key which decides how much queues build up before the sources rates stabilize. The above statements assume that the TCP windows increase smoothly and not in bursts (i.e., TCP acknowledgements are not bunched together on receipt at the sources).

With ERICA, once the link is fully loaded, the measurements (input rate, number of active source etc) can be made more reliably. There is also a continuous flow of BRM cells in the reverse direction, which can carry the rate feedback to the sources. This results in accurate feedback to sources. The sources are asked to reduce their rates, and the effect is seen at the switch within one feedback delay. Recall that the feedback delay is defined as the sum of the delay from the switch to the source and from the source to the same switch. Feedback delay is always less than a round-trip time. In the worst case, the feedback delay is equal to the round trip time.

From our observation that the overload increases at most by a factor of two every round-trip time, the maximum overload with ERICA in the first RTT after the link is fully loaded is two. The typical convergence time of ERICA (time to reach the steady state) is about two round trip times (one round trip

to equalize rates for fairness, and one round trip to bring the load to unity). See the performance evaluation of ERICA in chapter 6 for further details. Assuming that the rate allocations do not result in the increase of the total load, the link is overloaded by a factor of two during the convergence period.

The queue growth during this period is $2 \times RTT \times Link\_Bandwidth$ (the rate of queue growth is at most $Link\_Bandwidth$, since the maximum overload factor is two). However, in general, the switch algorithm takes a few round trips and a few feedback delays before it converges. The feedback delay becomes a factor because many switch algorithms give feedback as the RM cell travels in the reverse direction (rather than in the forward direction), which results in a sources responding faster to feedback. Also note that we have used an explicit rate scheme (ERICA) which results in faster convergence and hence smaller buffer requirements. A binary (EFCI) feedback scheme or an ER-scheme which is sensitive to variation may require larger buffers.

After this convergence phase, the source rates are controlled. In other words, the sources transition from a "window-limited" state to a "rate-limited" state. This is a stable state. The switch queues built up during the convergence phase are now drained out and the system reaches its "congestion avoidance" operating point of "high utilization and low queueing delay."

Note that even if a new VC carrying TCP traffic starts transmitting data, the additional load is small. This is because the TCP source sends only one segment (due to the slow start algorithm which starts with a window of one) irrespective of the Initial Cell Rate (ICR) of the VC. Note that switches have to provide additional buffering for sources which may carry non-TCP sources and may

start transmitting at ICR. ICR negotiation is therefore important for buffer allocation. Further, we also assume that the RTTs are sufficiently larger than the MSS, so that the addition of one segment of size MSS from every new source starting (at a window of one) does not increase the queue size significantly. Another assumption, is that the applications which run on TCP are either persistant, or have sufficient idle time between transmissions to allow resetting of TCP windows . The latter condition is because TCP implementations reset their congestion windows to one if there is no source activity for more than a TCP timeout period (typically a few hundreds of milliseconds) [81].

5. **Queue Backlogs:** In the above analysis, we have ignored the queue backlog due to bursts smaller than RTT. This backlog is built up as follows:

   The TCP sources have active periods and idle periods. The idle periods are used by the switch to clear out the backlogs created during the active periods. In general, TCP creates an temporary overload of 2 during the active period (which is less than RTT initially). This active period is followed by an idle (zero load) period for $RTT - activeperiod$ which is used to clear the backlogs created during the active period. We will assume that the switch algorithm is totally ineffective as long as the active period is smaller than RTT. The active period doubles every RTT until it is greater than RTT. Once the active period is greater than RTT, the switch rate control takes over.

   Suppose idle period is T. The corresponding active period is $RTT - T$. The maximum queue buildup during this active period is:

$$QB = Link\_bandwidth \times (RTT - T)$$

and the maximum queue drain during the idle period is:

$$QD = Link\_bandwidth \times T$$

Observe that $QD > QB$ when $T > RTT - T$, i.e., when the idle period $T > RTT/2$

In other words, the backlogs build up only when the idle periods become smaller than $RTT/2$. From the nature of TCP traffic, we know that initially the idle times are large, and then they reduce in size exponentially. When the idle time becomes smaller than $RTT/2$, the maximum queue backlog in each cycle (till the link becomes fully loaded) is $Link\_Bandwidth \times (RTT - 2T)$. But, observe that such a backlog can be created only once. This is because a burst needs to have an idle time T which satisfies $0 < T < RTT/2$ in order to create a backlog. Since, the active period doubles every RTT, there cannot be two RTTs where the idle time T satisfies $0 < T < RTT/2$.

6. **Effect of two-way traffic:** The above analysis has assumed undirectional TCP traffic (typical of file-transfer applications). We will briefly study the effect of two-way traffic on the buffer requirements. It has been noted [82] that bidirectional traffic complicated TCP dynamics considerably leading to more bursty behavior by TCP. This is called the "Ack Compression" phenomenon.

TCP smoothes out its window increase by using small steps on the receipt of every new ack from the destination. If the flow of acks is smooth, the window increases are smooth. As a result, TCP data cannot load the network in sudden bursts. However, when the traffic flows in both directions on TCP connections, the behavior is more complex. This is because the acknowledgements may be received in bursts, rather than being spaced out over time. As a result, the TCP window increases by amounts larger than one MSS, and the source sends data in larger bursts.

In the worst case, all the acknowledgements for the previous cycle are received at once, i.e., acknowledgements for $D = 1 \times \text{RTT} \times \text{Link\_bandwidth}$ bytes of data is received at once. This results in the $2D$ bytes of data (D due to data using the portion of the windows acknowledged, and D due to data using the expansion of the window by the slow-start exponential increase policy) sent in the current cycle. Assuming that there is bandwidth available to transmit this data upto the bottleneck, the worst case queue is $2D$. Observe that the total average load measured over an RTT is still twice the Link_bandwidth, however there is an extra instantaneous queue of $D = 1 \times \text{RTT} \times \text{Link\_bandwidth}$. This is because the entire queue builds up within a cell transmission time. In the case when TCP load is smooth, two cells are being input for every cell drained from the queue.

Observe that once the sources are rate-limited, the TCP burstiness due to ack compression is hidden from the ATM network because the excess cells are

buffered at the end-system, and not inside the ATM network. This is a significant performance benifit considering the fact that without rate-control the buffer requirement may be very large [82].

7. **Effect of VBR backgrounds:** The presence of higher priority background traffic implies that the ABR capacity is variable. There are two implications of the variation in capacity: **a)** the effect on the rate of TCP acks and the window growth, and, **b)** the effect on the switch rate allocation algorithm (ERICA).

   Part a): The effect of VBR background on the TCP "ack clock" (i.e., the rate of TCP acknowledgements (ACKs) is described below.

   - If VBR comes on during zero ABR load, it does not affect the TCP ACK clock because there are no ABR cells in the pipe. This is the case when VBR comes on during the idle periods of TCP, especially in the initial startup phase of TCP.
   - If VBR comes on during low load (initial load less than unity), *such that the resulting total load is less than unity,* it increases the TCP packet transmission time. Increased packet transmission time means that the inter-ACK time is increased (called *"ACK expansion"*). Increase in inter-ACK time slows down the window increase since the "ACK clock" runs slower now. Slower window increases imply reduced ABR load in the next cycle, irrespective of whether VBR is off or on.

     Specifically, if VBR goes off, the inter-ACK spacing starts decreasing by half every round trip (due to TCP window doubling every round trip). Since we have assumed that the system was in low load previously, this

change in VBR load does not affect the ABR load immediately. The TCP continues its exponential increase as if it started at the current window levels. The analysis is then similar to the case where VBR is absent. We are assuming that the source is not rate-limited by ABR during this phase. There is no excess ABR queues due to

- If VBR comes such that the resulting total load is greater than unity, then queues build up, and the ACK expansion occurs. The TCP load grows at a slower rate, and the ABR feedback mechanism throttles the source rates, leading to the excess TCP load being buffered at the sources. The excess network queue due to this case is $1 \times RTT \times VBRBandwidth$.

  TCP does experience variation in the rate of acknowledgements. The rate of acknowledgements determine how rapidly the TCP window grows. Note that if the sum of the windows is such that it fully loads the bottleneck (no idle periods), then the variable rate of acknowledgements only determines how quickly excess data is dumped onto the ATM sources by TCP. If the ABR sources are rate-limited, and have TCP queues built up, the excess data affects the queue at the source and not the network. The network queue is affected by the rate changes caused by the switch algorithm, as described in part b. If the source queue is close to zero, then the excess data dumped by TCP due to the variable "ack clock" affects the network queue as well. The effect of VBR on TCP ack clock is as follows:

  However, once the VBR load goes away, the ABR feedback (see part b) determines how many cells are admitted into the network and how many remain at the sources.

Part b): When ABR load goes away, then the switches see a sudden underload and allocate high rates to sources. Note that a switch which sees no cells in an interval or a small number of cells due to the fact that VBR load disappears is dealing with transient, incomplete metric information. As a result, if it overallocates rates, then sudden queue spikes are seen. In the worst case, the queues may grow unboundedly as a result of several such high priority VBR on/off phases. The buffering required under this condition is heavily dependent upon the switch algorithm. We present the problem, simulation results, modifications we made to the ERICA algorithm, and results with the improved algorithm in section 8.16 later in this chapter. We also model MPEG-2 traffic over VBR and study its effect on TCP traffic over ABR in section 8.17.

We have seen that the round trip time, feedback delay, the bandwidth variability caused by VBR, and the switch algorithm are key factors which determine the buffer requirements for TCP over ABR.

From items 4 (switch convergence time) and 5 (queue backlogs) above, we find that for file transfer over ABR, we require at least $3 \times RTT$ worth of buffer. Items 6 (two-way traffic) and 7 (VBR traffic) require a buffer of at least $5 \times RTT$. Note that the effect of the averaging interval parameter dominates in LANs (because it is much larger than RTT or feedback delay). Similarly, the effect of the feedback delay dominates in satellite networks because it can be much smaller than RTT. We substantiate our claims with simulation results, where we will observe that the buffer requirement is at most $3 \times RTT$.

Though the maximum ABR network queues are small, the queues at the sources are high. Specifically, the maximum sum of the queues in the source and the switches

is equal to the sum of the TCP window sizes of all TCP connections. In other words the buffering requirement for ABR becomes the same as that for UBR if we consider the source queues into consideration. This observation is true only in certain ABR networks. If the ATM ABR network is an end-to-end network, the source end systems can directly flow control the TCP sources. In such a case, the TCP will do a blocking send, i.e., and the data will go out of the TCP machine's local disk to the ABR source's buffers only when there is sufficient space in the buffers. The ABR service may also be offered at the backbone networks, i.e., between two routers. In these cases, the ABR source cannot directly flow control the TCP sources. The ABR flow control moves the queues from the network to the sources. If the queues overflow at the source, TCP throughput will degrade. We substantiate this in section 8.15.5

Note that we have studied the case of infinite traffic (like a large file transfer application) on top of TCP. In section 8.19, we show that bursty (idle/active) applications on TCP can potentially result in unbounded queues. However, in practice, a well-designed ABR system can scale well to support a large number of applications like bursty WWW sources running over TCP [79].

8.15 Factors Affecting Buffering Requirements of TCP over ATM-ABR Service

In this section we present sample simulation results to substantiate the preceding claims and analyses. We also analyze the effect of some important factors affecting the ABR buffering requirements. The key metric we observe is the maximum queue length. We look at the effect of VBR in two separate sections, the second of which deals with multiple MPEG-2 sources using a VBR connection.

All our simulations presented use the n Source configuration presented earlier. Recall that it has a single bottleneck link shared by n ABR sources. All links run at 155.52 Mbps and are of the same length. We experiment with the number of sources and the link lengths.

All traffic is unidirectional. A large (infinite) file transfer application runs on top of TCP for the TCP sources. N may assume values 1, 2, 5, 10, 15 and the link lengths 1000, 500, 200, 50 km. The maximum queue bounds also apply to configurations with heterogenous link lengths.

The TCP and ABR parameters are set in the same way as in the earlier simulations. For satellite round trip (550 ms) simulations, the window is set using the TCP window scaling option to 34000×2^8 bytes.

8.15.1 Effect of Number of Sources

In Table 8.2, we notice that although the buffering required increases as the number of sources is increased, the amount of increase slowly decreases. As later results will show, three RTTs worth of buffers are sufficient even for large number of sources. In fact, one RTT worth of buffering is sufficient for many cases: for example, the cases where the number of sources is small. The rate allocations among contending sources were found to be fair in all cases.

8.15.2 Effect of Round Trip Time (RTT)

From Table 8.3, we find that the maximum queue approaches

$3 \times \text{RTT} \times \text{link bandwidth}$,

particularly for metropolitan area networks (MANs) with RTTs in the range of 6 ms to 1.5 ms. This is because the RTT values are lower and in such cases, the effect

Number of Sources	RTT(ms)	Feedback delay(ms)	Max Q (cells)	Thoughput
5	30	10	10597 = 0.95*RTT	104.89
10	30	10	14460 = 1.31*RTT	105.84
15	30	10	15073 = 1.36*RTT	107.13

Table 8.2: Effect of number of sources

of switch parameters on the maximum queue increases. In particular, the ERICA averaging interval parameter is comparable to the feedback delay.

Number of Sources	RTT(ms)	Feedback Delay (ms)	Max Q size(cells)	Thoughput
15	30	10	15073 = 1.36*RTT	107.13
15	15	5	12008 = 2.18*RTT	108.00
15	6	2	6223 = 2.82*RTT	109.99
15	1.5	0.5	1596 = 2.89*RTT	110.56

Table 8.3: Effect of Round Trip Time (RTT)

8.15.3 LANs: Effect of Switch Parameters

In Table 8.4, the number of sources is kept fixed at 15. The averaging interval is specified as a pair (T, n), where the interval ends when either T ms have expired or N cells have been processed, whichever happens first. For the parameter values shown in the table, the number of cells determined the length of the averaging interval since under continuous traffic 1000 ATM cells take only 2.7 ms.

Averaging Interval (ms,cells)	RTT(ms)	Feedback Delay (ms)	Max Q size(cells)	Thoughput
(10,500)	1.5	0.5	2511	109.46
(10,1000)	1.5	0.5	2891	109.23
(10,500)	0.030	0.010	2253	109.34
(10,1000)	0.030	0.010	3597	109.81

Table 8.4: Effect of Switch Parameter (Averaging Interval)

From Table 8.4, we observe that the effect of the switch parameter, averaging interval, dominates in LAN configurations. The ERICA averaging interval is much greater than the RTT and feedback delay and it determines the congestion response time and hence the queue lengths. configurations. The ERICA averaging interval becomes much greater than

8.15.4 Effect of Feedback Delay

We conducted a 3×3 full factorial experimental design to understand the effect of RTT and feedback delays [49]. The results are summarized in Table 8.5. The thoughput figures for the last three rows (550 ms RTT) are not available since the throughput did not reach a steady state although the queues had stabilized.

Observe that the queues are small when the feedback delay is small and do not increase substantially with round-trip time. This is because the switch scheme limits the rate of the sources before they can overload for a substantial duration of time.

RTT(ms)	Feedback Delay (ms)	Max Q size(cells)	Thoughput
15	0.01	709	113.37
15	1	3193	112.87
15	10	17833	109.86
30	0.01	719	105.94
30	1	2928	106.9
30	10	15073	107.13
550	0.01	2059	NA
550	1	15307	NA
550	10	17309	NA

Table 8.5: Effect of Feedback Delay

8.15.5 TCP Performance over ATM Backbone Networks

The ATM source buffer requirement is derived by examining the maximum queues at the source when TCP runs over ABR. We also study the performance when sufficient buffers are not provided and discuss the implications for ATM backbone networks.

Source End System Queues in ABR

Table 8.6 shows the results with a 15-source configuration with link lengths of 1000 km (there is no VBR background). The link lengths yield a round trip time (propagation) of 30 ms and a feedback delay of 10 ms. We vary the size of the source end system buffers from 100 cells to 100000 cells per VC (second column). These values are compared to the maximum receiver window size (indicated as "Win" in the table) which is 1024 kB = 24576 cells. The switch has infinite buffers and uses

a modified version of the ERICA algorithm including the averaging feature for the number of sources and an averaging interval of (5 ms, 500 cells) as described in Section 8.16.1.

The maximum source queue values (third column) are tabulated for every VC, while the maximum switch queue values (fourth column) are for all the VCs together. When there is no overflow the maximum source queue (third column) measured in units of cells is also presented as a fraction of the maximum receiver window. The switch queues are presented as a fraction of the round trip time (indicated as "RTT" in the table). The round trip time for this configuration is 30 ms which corresponds to a "cell length" of 30 ms $\times$ 368 cells/ms = 11040 cells.

The last column tabulates the aggregate TCP throughput. The maximum possible TCP throughput in our configuration is approximately: 155.52 $\times$ (0.9 for ERICA Target Utilization) $\times$ (48/53 for ATM payload) $\times$ (512/568 for protocol headers) $\times$ (31/32 for ABR RM cell overhead) = 110.9 Mbps.

#	Source Buffer (cells)	Max Source Q (cells)	Max Switch Q (cells)	Total Throughput
1.	100 (< Win)	> 100 (overflow)	8624 (0.78×RTT)	73.27 Mbps
2.	1000 (< Win)	> 1000 (overflow)	17171 (1.56×RTT)	83.79 Mbps
3.	10000 (< Win)	> 10000 (overflow)	17171 (1.56×RTT)	95.48 Mbps
4.	100000 (> Win)	23901 (0.97×Win)	17171 (1.56×RTT)	110.90 Mbps

Table 8.6: Source Queues in ABR

In rows 1, 2 and 3 of Table 8.6, the source has insufficient buffers. The maximum per-source queue is equal to the source buffer size. The buffers overflow at the source and cells are dropped. TCP then times out and retransmits the lost data.

Observe that the switch queue reaches its maximum possible value for this configuration (1.56×RTT) given a minimum amount of per-source buffering (1000 cells = 0.04×Win). The switch buffering requirement is typically 3×RTT as discussed earlier in this chapter.

The sources however require one receiver window's worth of buffering per VC to avoid cell loss. This hypothesis is substantiated by row 4 of Table 8.6 which shows that the maximum per-source queue is 23901 cells = 0.97×Win. The remaining cells (0.03×Win) are traversing the links inside the ATM network. The switch queues are zero because the sources are rate-limited by the ABR mechanism. The TCP throughput (110.9 Mbps) is the maximum possible given this configuration, scheme and parameters.

The total buffering required for N sources is the sum of the N receiver windows. Note that this is the same as the switch buffer requirement for UBR [35]. In other words, the ABR and UBR services differ in whether the sum of the receiver windows' worth of queues is seen at the source or at the switch.

Implications for ATM Backbone Networks

If the ABR service is used end-to-end, then the TCP source and destination are directly connected to the ATM network. The source can directly flow control the TCP source. As a result, the TCP data stays in the disk and is not queued in the end-system buffers. In such cases, the end-system need not allocate large buffers.

ABR is better than UBR in these (end-to-end) configurations since it allows TCP to scale well.

However, if the ABR service is used on a backbone ATM network, the end-systems are edge routers which are not directly connected to TCP sources. These edge routers may not be able to flow control the TCP sources except by dropping cells. To avoid cell loss, these routers need to provide one receiver window's worth of buffering per TCP connection. The buffering is independent of whether the TCP connections are multiplexed over a smaller number of VCs or they have a VC per connection. For UBR, these buffers need to be provided inside the ATM network, while for ABR they need to be provided at the edge router. If there are insufficient buffers, cell loss occurs and TCP performance degrades.

The fact that the ABR service pushes the congestion to the edges of the ATM network while UBR service pushes it inside is an important benifit of ABR for the service providers. In general, UBR service requires more buffering in the switches than the ABR service.

8.15.6 Summary of buffering requirements for TCP over ABR

The main results of this section are:

1. A derivation for the buffer requirements of TCP over ABR is given. The factors which affect the buffer requirements are RTT, switch algorithm parameters, feedback delay, presence of VBR traffic, or two-way TCP traffic. For a switch algorithm like ERICA, the buffer requirements are about $3 \times RTT \times Link\_bandwidth$. The derivation is valid for infinite applications (like file transfer) running over TCP.

2. Once the ABR sources are rate-limited, the queues build up at the sources, and not inside the network. This has implications for ATM backbone networks where the edge routers either need large buffers, or need to implement some form of flow control with the TCP end system to avoid loss.

8.16 Effect of ON-OFF VBR Background Traffic

In this section we examine the effect of ON-OFF VBR background on the buffer requirements for TCP over ABR. We use the n source + VBR configuration as before. The parameter changes in the configuration are described below:

We use the ERICA+ scheme in the VBR simulations, and compare the performance with the ERICA scheme in certain cases. Recall that the ERICA+ scheme is an extension of ERICA which uses the queueing delay as a additional metric to calculate the feedback. ERICA+ eliminates the target utilization parameter (set to 1.0) and uses four new parameters: a target queueing delay (T0 = 500 microseconds), two curve parameters (a = 1.15 and b = 1.05), and a factor which limits the amount of ABR capacity allocated to drain the queues (QDLF = 0.5). In certain cases, we use averaging schemes for the metrics used by ERICA, and a longer averaging interval: min(5 ms, 500 cells).

The VBR source when present is an ON-OFF source. The ON time and OFF time are defined in terms of a "duty cycle" and a "period". A pulse with a duty cycle of d and period of p has an ON time of d×p and and OFF time of (1-d)×p. Our previous results of TCP over VBR used a duty cycle of 0.5 resulting in the ON time being equal to the OFF time. Unequal ON-OFF times used in this study cause new effects that were not seen before. The VBR starts at t = 2 ms to avoid certain initialization

problems. During the ON time, the VBR source operates at its maximum amplitude. The maximum amplitude of the VBR source is 124.41 Mbps (80% of link rate). VBR is given priority at the link, i.e, if there is a VBR cell, it is scheduled for output on the link before any waiting ABR cells are scheduled.

8.16.1 Simulation Results

Table 8.7 shows the results of a 3x3 full-factorial experimental design [49] used to identify the problem space with VBR background traffic. We vary the two VBR model parameters: the duty cycle (d) and the period (p). Recall that, with parameters d and p, the VBR ON time is d×p and the VBR OFF time is d×(1-p). Each parameter assumes three values. The duty cycle assumes values 0.95, 0.8 and 0.7 while the period may be 100 ms (large), 10 ms (medium) and 1 ms (small).

The maximum switch queue is also expressed as a fraction of the round trip time (30 ms = 30 ms × 368 cells/ms = 11040 cells).

Effect of VBR ON-OFF Times

Rows 1,2 and 3 of Table 8.7 characterize large ON-OFF times (low frequency VBR). Observe that the (maximum) queues are small fractions of the round trip time. The queues which build up during the ON times are drained out during the OFF times. Given these conditions, VBR may add at most one RTT worth of queues. ERICA+ further controls the queues to small values.

Rows 4,5 and 6 of Table 8.7 characterize medium ON-OFF times. We observe that rows 5 and 6 have divergent (unbounded) queues. The effect of the ON-OFF time on the divergence is explained as follows. During the OFF time the switch experiences underload and may allocate high rates to sources. The duration of the OFF time

#	Duty Cycle(d) (ms)	Period (p) (cells)	Max Switch Q
1.	0.95	100	2588 (0.23×RTT)
2.	0.8	100	5217 (0.47×RTT)
3.	0.7	100	5688 (0.52×RTT)
4.	0.95	10	2709 (0.25×RTT)
5.	0.8	10	DIVERGENT
6.	0.7	10	DIVERGENT
7.	0.95	1	2589 (0.23×RTT)
8.	0.8	1	4077 (0.37×RTT)
9.	0.7	1	2928 (0.26×RTT)

Table 8.7: Effect of VBR ON-OFF Times

determines how long such high rate feedback is given to sources. In the worst case, the ABR load is maximum whenever the VBR source is ON to create the largest backlogs.

On the other hand, the VBR OFF times also allow the ABR queues to be drained out, since the switch is underloaded during these times. Larger OFF times may allow the queues to be completely drained before the next ON time. The queues will grow unboundedly (i.e., diverge) if the queue backlogs accumulated after ON and OFF times never get cleared.

Rows 7,8 and 9 of Table 8.7 characterize small ON-OFF times. Observe again that the queues are small fractions of the round trip time. Since the OFF times are small, the switch does not have enough time to allocate high rates. Since the ON times are small, the queues do not build up significantly in one ON-OFF cycle. On the other

hand, the frequency of the VBR is high. This means that the VBR changes much faster than the time required for sources to respond to feedback. ERICA+ however controls the queues to small values in these cases.

Effect of Feedback Delays with VBR

Another factor which interacts with the VBR ON-OFF periods is the feedback delay. We saw that one of the reasons for the divergent queues was that switches could allocate high rates during the VBR OFF times. The feedback delay is important in two ways. First, the time for which the switch may allocate high rates is the minimum of the feedback delay and the VBR OFF-time. This is because, the load due to the high rate feedback is seen at the switch within one feedback delay. Second, when the load due to the high rate feedback is seen at the switch, it takes at least one feedback delay to reduce the rates of the sources.

The experiments shown in Table 8.7 have a long feedback delay (10 ms). The long feedback delay allows the switch to allocate high rates for the entire duration of the VBR OFF time. Further, when the switch is overloaded, the sources takes 10 ms to respond to new feedback. Therefore, given appropriate value of the ON-OFF times (like in rows 4,5 of Table 8.7), the queues may diverge.

Table 8.8 shows the effect of varying the feedback delay and round trip time. We select the divergent case (row 4) from Table 8.7 and vary the feedback delay and round trip time of the configuration.

Row 1 in Table 8.8 shows that the queues are small when the feedback delay is 1 ms (metropolitan area network configuration). In fact, the queues will be small when the feedback delay is smaller than 1 ms (LAN configurations). In such configurations, the minimum of the OFF time (2 ms) and the feedback delay (< 1 ms) is the feedback

#	Feedback Delay(ms)	RTT (ms)	Duty Cycle (d)	Period (p) (ms)	Max Switch Q (cells)
1.	1 ms	3 ms	0.8	10 ms	4176 (0.4×RTT)
2.	5 ms	15 ms	0.8	10 ms	DIVERGES
3.	10 ms	30 ms	0.8	10 ms	DIVERGES

Table 8.8: Effect of Feedback Delay with VBR

delay. Hence, in any VBR OFF time, the switch cannot allocate high rates to sources long enough to cause queue backlogs. The new load is quickly felt at the switch and feedback is given to the sources.

Rows 2 and 3 in Table 8.8 have a feedback delay longer than the OFF time. This is one of the factors causing the divergence in the queues of these rows.

Effect of Switch Scheme with VBR

The TCP traffic makes the ABR demand variable. The VBR background makes the ABR capacity variable. In the presence of TCP and VBR, the measurements used by switch schemes are affected by the variation. The errors in the metrics are reflected in the feedback. The errors in the feedback result in queues. Switch schemes need to be robust to perform under such error-prone conditions. Another effect of errors is that the boundary conditions of the scheme are encountered often. The scheme must be designed to handle such conditions gracefully. We study the robustness issues in ERICA and make adjustments needed to reduce the effect of the variation.

As an example, consider the case when the VBR ON-OFF periods are very small (1 ms ON, 1 ms OFF). The resulting variation can be controlled by a switch scheme

like ERICA+ which uses the queueing delay to calculate feedback (in addition to input rate etc). The basic ERICA algorithm without queue control cannot handle this level of variation.

The ERICA+ algorithm uses the queue length as a secondary metric to reduce the high allocation of rates. However, ERICA+ has a limit on how much it can reduce the allocation. Given sufficient variation, the limit can be reached. This means that even the minimum rate allocation by ERICA+ causes the queues to diverge. This reason, along with the discussion on ON-OFF times and feedback delays explains the divergent cases in Tables 8.7 and 8.8.

Reducing the Effects of Variation In ERICA+

We tackle these problems by reducing the effect of variation on the scheme measurements in three ways (described in detail in chapter 6):

1. First, we observe that one way to reduce variation in measurements is to measure quantities over longer intervals. Longer intervals yield averages which have less variance. However, making the intervals too long increases the response time, and queues may build up in the interim.

2. Second, we average the measurements over several successive intervals. The ERICA scheme uses two important measurements: the overload factor (z) which is the ratio of the input rate and the target ABR rate, and the number of active sources (N_a). We re-examine how the scheme depends on these metrics and design an appropriate averaging technique for each of them.

    - The overload factor (z) is used to divide the current cell rate of the source to give what we call the "VC share". The VC share is one of the rates

which may be given as feedback to the source. If the overload factor (z) is underestimated, the VC share increases. The overload factor is usually not overestimated. However, if the interval length is small, the estimated values may have high variation.

The overload factor (z) can suddenly change in an interval if the load or capacity in that interval changes due to the variation. The out-of-phase effect of TCP may lead to no cells being seen in the forward direction ($z = 0$, a huge underestimate !), while BRM cells are seen in the reverse direction. The switch will then allocate a high rate in the feedback it writes to the BRM cell.

We have designed two averaging schemes for the overload as described in chapter 6. Both schemes use an averaging parameter "α_z". The first scheme is similar to the technique of exponential averaging technique for a random variable. However, it differs in that it resets the averaging mechanism whenever the instantaneous value of overload is measured to be zero or infinity. The second scheme does not ignore the outlier values (zero or infinity) of the overload factor. Further, it averages the overload by separately averaging the input rate and capacity, and then taking the ratio of the averages. It can be shown [49] that this is theoretically the right way to average a ratio quantity like overload.

- The number of active sources (N_a) is used to calculate a minimum fairshare that is given to any source. If N_a is underestimated, then the minimum fairshare is high leading to overallocation. If N_a is overestimated, then the

minimum fairshare is low. This may result in slower transient response, but does not result in overallocation.

The number of active sources can fluctuate if some sources are not seen in an interval. Further, due to the clustering effect of TCP, cells from just a few VCs may be seen in an interval leading to an underestimate of N_a.

In averaging N_a, the scheme maintains an activity level for each source. The activity level of the source is set to one when any cell of the source is seen in the interval. However, when no cell from a source is seen in an interval, the scheme "decays" the activity level of the source by a factor, "α_n" (also called $DecayFactor$). Hence, the source becomes inactive only after many intervals. A recommended value of α_n is 0.9. Roughly, the N_a measured with this value of α_n is approximately equal to the N_a measured without averaging over an averaging interval 8 or 9 times larger than the current averaging interval.

3. Third, we modify the response to boundary conditions of the scheme. This allows the scheme to handle the boundary conditions gracefully. Specifically, the number of active sources is set to one if it is measured to be below one. The second method of overload factor averaging does not allow the overload factor be zero or infinity. However, outlier measurements are not ignored in the averaging method.

The ERICA+ scheme with these modifications controls the ABR queues without overly compromising on TCP throughput. Table 8.9 shows the results of representative experiments using these features.

#	Averaging Interval (T ms, n cells)	Averaging of Na on ? ($\alpha_n = 0.9$)	Averaging of z on ? ($\alpha_z = 0.2$)	d	p(ms)	Max Switch Queue (cells)
1.	(1,100)	YES	YES	0.7	20	5223
2.	(5,500)	YES	NO	0.7	20	5637

Table 8.9: Effect of Switch Scheme

Row 1 shows the performance with the averaging of Na and z turned on on a formerly divergent case. Observe that the queue converges and is small. The parameter α_z is 0.2, which is roughly equivalent to increasing the averaging interval length by a factor of 5. Hence, we try the value (5 ms, 500 cells) as the averaging interval length, without the averaging of overload factor. Row 2 shows that the queue for this case also converges and is small.

8.16.2 Summary of ON-OFF VBR background effects

In this section, we have studied the impact of ON-OFF VBR background traffic on switch buffering for ABR service carrying TCP traffic. We find that the ON-OFF times, the feedback delays, and a switch scheme sensitive to variation in ABR load and capacity may combine to create worst case conditions where the ABR queues diverge. We have motivated three enhancements to the ERICA+ scheme. The modifications reduce the effect of the variation and allow the convergence of the ABR queues, without compromising on the efficiency. In the next section, we shall examine the effect of VBR carrying long-range dependent traffic (similar to multiplexed MPEG-2 traffic) on ABR, and show that the buffer requirements are unchanged.

8.17 Effect of Long-Range Dependent (LRD) VBR background traffic

We have studied the ABR model extensively with different source traffic patterns like persistent sources, ON-OFF bursty sources, ping pong sources, TCP sources and source-bottlenecked VCs. Many of these studies have also considered the performance in the presence of ON-OFF VBR background traffic.

In reality, VBR consists of multiplexed compressed audio and video application traffic, each shaped by leaky buckets at their respective Sustained Cell Rate (SCR) and Peak Cell Rate (PCR) parameters. Compressed video has been shown to be long-range dependent by nature [13]. Compressed audio and video streams belonging to a single program are expected to be carried over an ATM network using the MPEG-2 Transport Stream facility as outlined in reference [34].

In this section, we first present a model of multiplexed MPEG-2 transport streams carried over ATM using the VBR service. Each stream exhibits long-range dependence, i.e., correlation over large time scales. We then study the effect of this VBR background on ABR connections carrying TCP file transfer applications on WAN and satellite configurations. The effect of such VBR traffic is that the ABR capacity is highly variant. We find that a proper switch algorithm like ERICA+ can tolerate this variation in ABR capacity while maintaining high throughput and low delay. We will present simulation results for terrestrial and satellite configurations.

8.17.1 Overview of MPEG-2 over ATM

In this section, we give a short introduction to the MPEG-2 over ATM model and introduce some MPEG-2 terminology. For a detailed discussion, see reference [77].

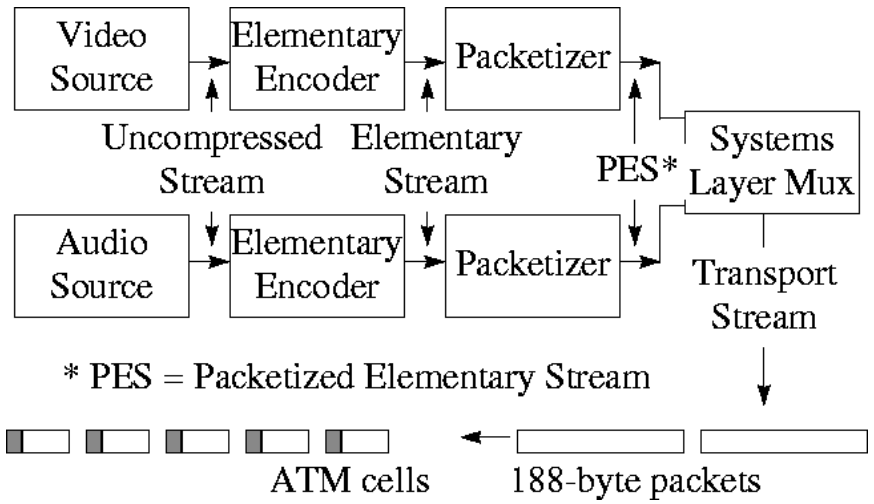


Figure 8.13: Overview of MPEG-2 Transport Streams

The MPEG-2 standard specifies two kinds of streams to carry coded video: the "Transport Stream" and the "Program Stream". The latter is used for compatibility with MPEG-1 (used for stored compressed video/audio), while the former is used to carry compressed video over networks which may not provide an end-to-end constant delay and jitter-free abstraction.

A Transport Stream can carry several programs multiplexed into one stream. Each program may consist of several "elementary streams," each containing MPEG-2 compressed video, audio, and other streams like close-captioned text, etc.

Figure 8.13 shows one such program stream formed by multiplexing a compressed video and a compressed audio elementary stream. Specifically, the figure shows the uncompressed video/audio stream going through the MPEG-2 elementary encoder to form the elementary stream. Typically, the uncompressed stream consists of frames generated at constant intervals (called "frame display times") of 33 ms (NTSC format) or 40 ms (PAL format). These frames (or "Group of Pictures" in MPEG-2

terminology) are called "Presentation Units." MPEG-2 compression produces three different types of frames: I, P and B frames, called "Access Units," as illustrated in Figure 8.14.

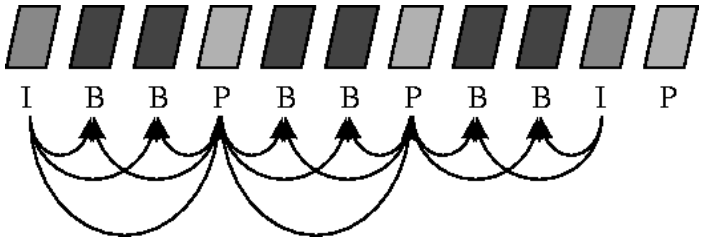


Figure 8.14: The I, P and B frames of MPEG-2

I (Intra-) frames are large. They contain the base picture, autonomously coded without need of a reference to another picture. They might take about 4-5 frame display times (approximately 160 ms) to be transmitted on the network depending upon the available rate.

P (Predictive-) frames are medium-sized. They are coded with respect to previous I or P frame. Transmission times for P frames is typically about 0.5-1 frame display times.

B (Birectionally predicted-) frames are very small. They are coded with respect to previous and later I or P frames and achieve maximum compression ratios (200:1). Transmission times for B frames is typically about 0.2 frame display times or even less.

As shown in Figure 8.13, the access units are packetized to form the "Packetized Elementary Stream (PES)". PES packets may be variable in length. The packetization process is implementation specific. PES packets may carry timestamps (called

Presentation Timestamps (PTS) and Decoding Timestamps (DTS)) for long-term synchronization. The MPEG-2 standard specifies that PTS timestamps must appear at least once every 700 ms.

The next stage is the MPEG-2 Systems Layer which does the following four functions. First, it creates fixed size (188 byte) transport packets from PES packets. Second, the transport packets of different PESs belonging to one program are identified as such in the transport packet format. Third, it multiplexes several such programs to create a single Transport Stream. Fourth, it samples a system clock (running at 27 MHz) and encodes timestamps called "MPEG2 Program Clock References" (MPCRs, see [34]) in every multiplexed program. The time base for different programs may be different.

The MPCRs are used by the destination decoder to construct a Phase Locked Loop (PLL) and synchronize with the clock in the incoming stream. The MPEG-2 standard specifies that MPCRs must be generated at least once every 100 ms. Due to AAL5 packetization considerations, vendors usually also fix a maximum rate of generation of MPCRs to 50 per second (i.e. no less than one MPCR per 20 ms).

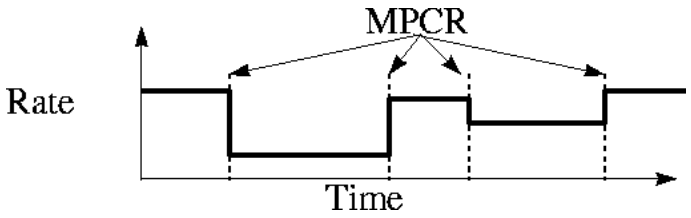


Figure 8.15: Piecewise constant nature of MPEG-2 Single Program Transport Streams

The key point is that the MPEG-2 rate is piecewise-CBR. As shown in Figure 8.15, the program's rate (not the transport stream's rate) is constant between successive MPCRs. The maximum rate is bounded by a peak value (typically 15 Mbps for HDTV quality compressed video [77]). The choice of the rates between MPCRs is implementation specific, but in general depends upon the buffer occupancy, and the rate of generation of the elementary streams.

The transport stream packets are encapsulated in AAL5 PDU with two transport stream packets in a single AAL5 PDU (for efficiency). The encapsulation method does not look for MPCRs in a transport packet and might introduce some jitter in the process. Alternate methods and enhancements to the above method have been proposed [77, 39].

An ATM VBR connection can multiplex several transport streams, each containing several programs, which in turn can contain several elementary streams. We model the multiplexing of several transport streams over VBR. But in our model, we will have only one program per Transport Stream (called the "Single Program Transport Stream" or "SPTS").

MPEG-2 uses a constant end-to-end delay model. The decoder at the destination can use techniques like having a de-jittering buffer, or restamping the MPCRs to compensate for network jitter, [77]. There is a Phase Locked Loop (PLL) at the destination which locks onto the MPCR clock in the incoming stream. The piecewise-CBR requirement allows the recovered clock to be reliable. Engineering of ATM VBR VCs to provide best service for MPEG-2 transport streams and negotiation of rates (PCR, SCR) is currently an important open question.

8.17.2 VBR Video modeling

There have been several attempts to model compressed video, see references [13, 33, 21] and references therein. Beran et al [13] show that long-range dependence is an inherent characteristic of compressed VBR video. But, they do not consider MPEG-2 data. Garrett and Willinger [33] show that a combination of distributions is needed to model VBR video. Heyman and Lakshman [21] argue that simple markov chain models are sufficient for traffic engineering purposes even though the frame size distribution may exhibit long-range dependence.

The video traffic on the network may be affected further by the multiplexing, renegotiation schemes, feedback schemes and the service category used. Examples of renegotiation, feedback schemes and best-effort video delivery are found in the literature, [38, 65, 23].

We believe that a general model of video traffic on the ATM network is yet to be discovered. In this paper, we are interested in the performance of ABR carrying TCP connections when affected by a long-range dependent, highly variable VBR background. We hence need a model for the video background. We have attempted to design the model to resemble the MPEG-2 Transport Stream.

There are three parameters in the model: the compressed video frame size, the inter-MPCR interval lengths, and the rates in these inter-MPCR intervals. In our model, the inter-MPCR intervals are uniformly distributed and the rates in the inter-MPCR intervals are long-range dependent. In real products, the rates are chosen depending upon the buffer occupancy at the encoder, which in turn depends upon the frame sizes of the latest set of frames generated. Further, the range of inter-MPCR intervals we generate follows implementation standards. We believe that this models

the MPEG-2 Transport Stream, and still incorporates the long-range dependence property in the video streams. The effect of this VBR model on ABR is to introduce high variation in ABR capacity. As we shall see, the ERICA+ algorithm deals with the variation in ABR capacity and successfully bounds the maximum ABR queues, while maintaining high link utilization.

8.17.3 Modeling MPEG-2 Transport Streams over VBR

We model a "video source" as consisting of a transport stream generator, also called encoder (E) and a network element (NE). The encoder produces a Transport Stream as shown in Figure 8.13 and discussed in section 8.17.1 . In our model, the Transport Stream consists of a single program stream. The network element encapsulates the transport packets into AAL5 PDUs and then fragments them into cells. The output of the network element (NE) goes to a leaky bucket which restricts the peak rate to 15 Mbps. This leaky bucket function can alternatively be done in the encoder, E (which does not send transport packets beyond a peak rate).

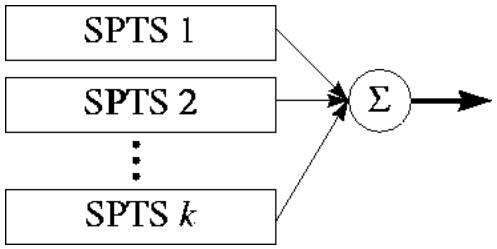


Figure 8.16: Multiplexing MPEG-2 Single Program Transport Streams (SPTSs) over VBR

Several (N) such video sources are multiplexed to form the VBR traffic going into the network as shown in Figure 8.16. Each encoder generates MPCRs uniformly

distributed between 20 ms and 100 ms. The reason for this choice (of maximum and minimum MPCRs) is explained in section 2. The rate of an encoder is piecewise-constant between successive pairs of MPCRs.

We generate the rates as follows. We choose the rate such that the sequence of rate values is long-range dependent. Specifically, we use a fast-fourier transform method [71] to generate the fractional gaussian noise (FGN) sequence (an independent sequence for each source). We ignore values above the maximum rate to 15 Mbps and below the minimum rate (0 Mbps). This reason for this choice is discussed in the following section. We choose different values of mean and standard deviation for the generation procedure. When we generate an inter-MPCR interval Ti and a corresponding rate Ri, the video source sends cells at a rate Ri uniformly spaced in the interval Ti. Due to the ignoring of some rate values, the actual mean of the generated stream may be slightly greater or lesser than the input means. We later measure the actual mean rate and use it to calculate the efficiency metric.

Though each video source sends piecewise-CBR cell streams, the aggregate VBR rate need not be piecewise-CBR. It has a mean (SCR) which is the sum of all the individual means. Similarly, it has a maximum rate (PCR) which is close to the sum of the peak rates (15 Mbps) of the individual video streams. These quantities depend upon the number of video sources. In our model, we use N equal to 9 to ensure that the PCR is about 80% of total capacity. VBR is given priority at any link, i.e, if there is a VBR cell, it is scheduled for output on the link before any waiting ABR cells are scheduled. Further, since each video stream is long-range dependent, the composite VBR stream is also long-range dependent. Therefore, the composite VBR stream and the ABR capacity has high variation.

8.17.4 Observations on the Long-Range Dependent Traffic Generation Technique

The long-range dependent generation technique described in [71] can result in negative values and values greater than the maximum possible rate value. This occurs especially when the variation of the distribution is high (of the order of the mean itself). Fortunately, there are a few approaches in avoiding negative values and bounding values within a maximum in such sequences. We considered these approaches carefully before making a choice.

The first approach is to generate a long-range dependent sequence $x_1, x2, ..., xn$ and then use the sequence $e^{x_1}, e^{x_2}, ..., e^{x_n}$ in our simulation. The values e^{x_i} is rounded off to the nearest integer. This method always gives zero or positive numbers. The new distribution still exhibits long-range dependence, though it is no longer a fractional gaussian noise (FGN) (like the originally generated sequence) [71]. Another problem is that all significant negative values are truncated to zero leading to an impulse at zero in the new probability density function (pdf). Further, the mean of the new sequence is not the exponentiated value of the old mean. This makes it difficult to obtain a sequence having a desired mean.

A second technique is to avoid exponentiation, but simply truncate negative numbers to zero. This approach again has the problem of the pdf impulse at zero. Also the mean of the entire distribution has increased.

The third technique is a variation of the second, which truncates the negative numbers to zero, but subtracts a negative value from the subsequent positive value.

This approach is aimed to keep the mean constant. But, it not only has the side-effect of inducing a pdf impulse at zero, but also changes the shape of the pdf, thus increasing the probability of small positive values.

The fourth and final technique is to simply ignore negative values and values greater than the maximum. This approach keeps the shape of the positive part of the pdf intact while not introducing a pdf impulse at zero. If the number of negative values is small, the mean and variance of the distribution would not have changed appreciably. Further, it can be shown that the new distribution is still long-range dependent.

We choose the fourth approach (of ignoring negative values and values greater than the maximum) in our simulations.

8.18 Simulation Configuration and Parameters

We use the n Source + VBR configuration described in section 8.7 earlier in this chapter. Recall that the configuration has a single bottleneck link shared by the N ABR sources and a VBR VC carrying the multiplexed stream. Each ABR source is a large (infinite) file transfer application using TCP. All traffic is unidirectional. All links run at 149.76 Mbps. The links traversed by the connections are symmetric i.e., each link on the path has the same length for all the VCs. In our simulations, N is 15 and the link lengths are 1000 km in WAN simulations. In satellite simulations, the feedback delay may be 550 ms (corresponds to a bottleneck after the satellite link) or 10 ms (corresponds to a bottleneck before the satellite link). This is illustrated in Figures 8.17 and 8.18.

For the video sources, we choose means and standard deviations of video sources to have three sets of values (7.5 Mbps, 7 Mbps), (10 Mbps, 5 Mbps) and (5 Mbps, 5 Mbps). This choice ensures that the variance in all cases is high, but the mean varies and hence the total VBR load varies. The number of video sources (N) is 9 which means that the maximum VBR load is 80% of 149.76 Mbps link capacity. As discussed later the effective mean and variance (after bounding the generated value to within 0 and 15 Mbps) may be slightly different and it affects the efficiency measure. We also compare certain results with those obtained using an ON-OFF VBR model described in section 8.16.

The Hurst parameter which determines the degree of long-range dependence for each video stream is chosen as 0.8 [13].

Recall that when TCP data is encapsulated over ATM, a set of headers and trailers are added to every TCP segment. We have 20 bytes of TCP header, 20 bytes of IP header, 8 bytes for the RFC1577 LLC/SNAP encapsulation, and 8 bytes of AAL5 information, a total of 56 bytes. Hence, every MSS of 512 bytes becomes 568 bytes of payload for transmission over ATM. This payload with padding requires 12 ATM cells of 48 data bytes each. The maximum throughput of TCP over raw ATM is (512 bytes/(12 cells $\times$ 53 bytes/cell)) = 80.5%. Further in ABR, we send FRM cells once every Nrm (32) cells. Hence, the maximum throughput is $31/32 \times 0.805 = 78\%$ of ABR capacity. For example, when the ABR capacity is 149.76 Mbps, the maximum TCP payload rate is 116.3 Mbps. Similarly, for a MSS of 9140 bytes, the maximum throughput is 87% of ABR capacity.

We use a metric called "efficiency" which is defined as the ratio of the TCP throughput achieved to the maximum throughput possible. As defined above the

maximum throughput possible is 0.78×(mean ABR capacity). The efficiency is calculated as follows. We first measure the the aggregate mean VBR rate (since it is not the sum of the individual mean rates due to bounding the values to 0 and 15 Mbps). Subtract it from 149.76 Mbps to get the mean ABR capacity. Then multiply the ABR capacity by 0.78 (or 0.87) to get the maximum possible throughput. We then take the ratio of the measured TCP throughput and this calculated value to give the efficiency.

8.18.1 Effect of High Variance and Total VBR Load

In this section, we present simulation results where we vary the mean and the standard deviation of the individual video sources such that the total variance is always high, and the total maximum VBR load varies.

In Table 8.10, and Table 8.11, we show the maximum queue length, the total TCP throughput, VBR throughput, ABR throughput, and efficiency for three combinations of the mean and standard deviation. Table 8.11 is for TCP MSS = 512 bytes, while Table 8.11 is for TCP MSS = 9140 bytes.

	Video Sources		ABR Metrics		
#	Mean per-source rate (Mbps)	Standard Deviation (Mbps)	Max Switch Q (cells)	Total TCP T'put	Efficiency (% of Max throughput)
1.	5	5	6775 (1.8×F/b Delay)	68.72 Mbps	94.4%
2.	7.5	7	7078 (1.9×F/b Delay)	59.62 Mbps	94.1%
3.	10	5	5526 (1.5×F/b Delay)	82.88 Mbps	88.4%

Table 8.10: Effect of Variance and VBR Load: MSS = 512 bytes

	Video Sources		ABR Metrics		
#	Mean per-source rate (Mbps)	Standard Deviation (Mbps)	Max Switch Q (cells)	Total TCP Throughput	Efficiency (% of Max throughput)
1.	5	5	5572 (1.5×F/b Delay)	77.62 Mbps	95.6%
2.	7.5	7	5512 (1.5×F/b Delay)	67.14 Mbps	95%
3.	10	5	5545 (1.5×F/b Delay)	56.15 Mbps	95.6%

Table 8.11: Effect of Variance and VBR Load: MSS = 512 bytes

Observe that the measured mean VBR thoughput (column 6) is the same in corresponding rows of both the tables. This is because irrespective of ABR load, VBR load is given priority and cleared out first. Further, by bounding the MPEG-2 SPTS source rate values between 0 and 15 Mbps, we ensure that the total VBR load is about 80of the link capacity.

For row 1, measured VBR throughput (column 6) was 56.44 Mbps (against $9 \times 5 = 45$ Mbps expected without bounding). For row 2, it was 68.51 Mbps (against $9 \times 7.5 = 67.5$ Mbps expected without bounding). For row 3, it was 82.28 Mbps(against $9 \times 10 = 90$ Mbps expected without bounding). Observe that when the input mean is higher, the expected aggregate value is lower and vice-versa.

The efficiency values are calculated using these values of total VBR capacity. For example, in row 1 of Table 8.10, the ABR throughput is is 149.76 - 56.44 = 93.32 Mbps. For a MSS of 512, the maximum TCP thoughput is 78% of ABR throughput = 72.78 Mbps (not shown in the table). Given that TCP thoughput achieved is 68.72 Mbps (Column 5), the efficiency is 68.72/72.78 = 94.4%. For Table 8.11, since the

MSS is 9140 bytes, the maximum TCP thoughput is 87% of ABR throughput, and this is the value used to compare the total TCP throughput against.

Observe that the efficiency achieved in all cases is high (above 90%) in spite of the high variation in ABR capacity. Also observe that the total TCP throughput is higher (as well as the efficiency) for TCP MSS = 9140 bytes in all cases.

The maximum queue length is controlled to about three times the feedback delay (or one round trip time) worth of queue. The feedback delay for this configuration is 10 ms, which corresponds to (10 ms) $\times$ (367 cells/ms) = 3670 cells worth of queue when the network is on the average overloaded by a factor of 2 (as is the case with TCP). The round-trip time for this configuration is 30 ms.

The queue length is higher when the mean per-source rate is lower (i.e., when the average ABR rate is higher). This is explained as follows. Whenever there is variation in capacity, the switch algorithm may make errors in estimating the average capacity and may overallocate rates temporarily. When the average ABR capacity is higher, each error in allocating rates will result in a larger backlog of cells to be cleared than for the corresponding case when the average ABR capacity is low. The combination of these backlogs may result in a larger maximum queue before the long-term queue reduction mechanism of the switch algorithm reduces the queues.

8.18.2 Comparison with ON-OFF VBR Results

Recall that in section 8.16 we have studied the behavior of TCP over ABR in the presence of ON-OFF VBR sources. We studied ranges of ON-OFF periods from 1 ms through 100 ms. Further, we had looked at results where the ON period was not equal to the OFF period. The worst cases were seen in the latter simulations.

However, with modifications to ERICA+ and a larger averaging interval we found that the maximum switch queue length was 5637 cells. This experiment has a duty cycle of 0.7 and a period of 20ms i.e., the ON time was 14 ms and the off time was 6 ms. Since we use the same switch algorithm parameters in this study, we can perform a comparison of the two studies.

We observe that, even after the introduction of the long-range dependent VBR model, the queues do not increase substantially (beyond one round trip worth of queues) and the efficiency remains high (around 90%). This is because the ERICA+ switch algorithm has been refined and tuned to handle variation in the ABR capacity and ABR demand. These refinements allow the convergence of the ABR queues, without compromising on the efficiency.

8.18.3 Satellite simulations with Short Feedback Delay

In this section and the next, we repeat the experiments with some links being satellite links. In the first set of simulations, we replace the bottleneck link shared by 15 sources with a satellite link as shown in Figure 8.17. The links from the second switch to the destination nodes are 1 km each. The total round trip time is 550 ms, but the feedback delay remains 10 ms.

Table 8.12 and Table 8.13 (similar to Tables 8.10 and 8.11) show the maximum switch queue length, the total TCP throughput, VBR throughput, ABR throughput, and efficiency for three combinations of the mean and standard deviation. Table 8.12 is for TCP MSS = 512 bytes, while Table 8.13 is for TCP MSS = 9140 bytes.

Note that the TCP startup time in this configuration is large because the round trip time (550 ms) is large and TCP requires multiple round trips to be able to use its

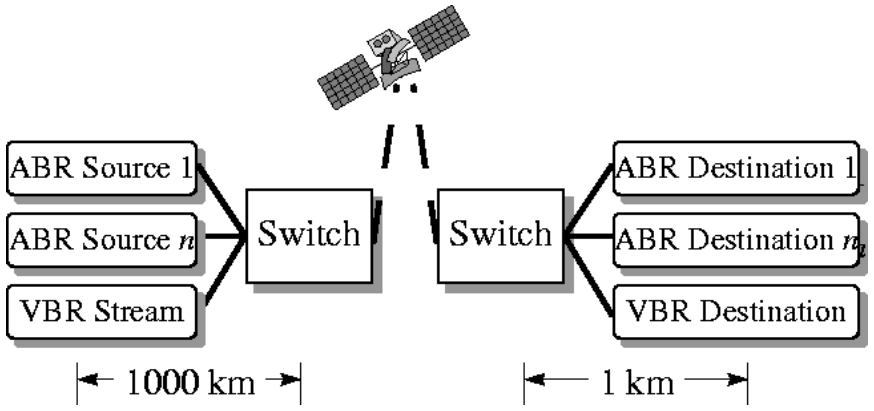


Figure 8.17: The "N Source + VBR" Configuration with a satellite link

full capacity. However, the effect on total TCP throughput is minimal since there is no loss and the feedback delays are small (10 ms) compared to round trip time, allowing ABR to control sources more effectively. Throughputs are high, and efficiency values are high.

	Video Sources		ABR Metrics				
#	Avg Src rate (Mbps)	STD (Mbps)	Max Switch Q (cells)	Total TCP T'put	VBR T'put	ABR T'put	Effcy (% Max T'put)
1.	5	5	5537 (1.5×f/b)	74.62	47.33	102.43	93.4%
2.	7.5	7	4157 (1.1×f/b)	67.34	57.23	92.53	93.3%
3.	10	5	3951 (1.1×f/b)	60.08	67.55	82.21	93.7%

Table 8.12: Maximum Queues for Satellite Networks with Short Feedback Delay: MSS=512 bytes

	Video Sources		ABR Metrics				
#	Avg Src rate (Mbps)	STD (Mbps)	Max Switch Q (cells)	Total TCP T'put	VBR T'put	ABR T'put	Effcy (% Max T'put)
1.	5	5	11294 (3×f/b delay)	84.72	47.33	102.43	95.1%
2.	7.5	7	9074 (2.5×f/b delay)	76.49	57.23	92.53	95.0%
3.	10	5	6864 (1.9×f/b delay)	68.18	67.55	82.21	95.3%

Table 8.13: Maximum Queues for Satellite Networks with Short Feedback Delay : MSS=9140 bytes

The tables shows that maximum queues are small (in the order of three times the feedback delay), irrespective of the mean and variance. In such satellite configurations, we observe that the feedback delay is the dominant factor (over round trip time) in determining the maximum queue length. As discussed earlier, one feedback delay of 10 ms corresponds to 3670 cells of queue for TCP.

8.18.4 Satellite simulations with Long Feedback Delay

In our second set of satellite simulations, we examine the effect of longer feedback delays. Consider a switch A at the end of a satellite link or a switch downstream of A. It will have a feedback delay of about 550 ms. This is the scenario we model. We form a new configuration as shown in Figure 8.18 by replacing the links in the feedback path to sources with satellite link. All other links are of length 1 km each. As a result, the round trip time and the feedback delay are both approximately equal to 550 ms.

Tables 8.14 and 8.15 (similar to Tables 8.10 and 8.11) show the maximum switch queue length, the total TCP throughput, VBR throughput, ABR throughput, and

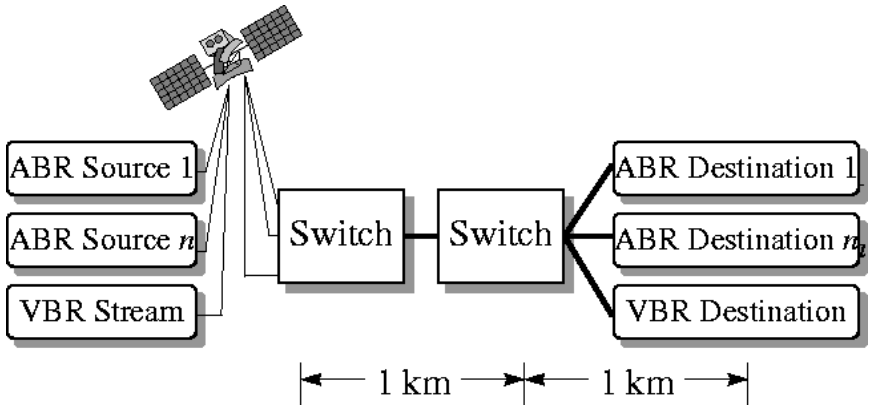


Figure 8.18: The "N Source + VBR" Configuration with satellite links and long feedback delays

efficiency for three combinations of the mean and standard deviation. Table 8.14 is for TCP MSS = 512 bytes, while Table 8.15 is for TCP MSS = 9140 bytes.

Observe that the queue lengths are quite large, while the total TCP throughput and efficiency are smaller (by 10-15%) compared to the values in Tables 8.12 and 8.13 (1000 km feedback delay cases) respectively. The total queue is still a small multiple of the feedback delay or RTT (a feedback delay of 550 ms corresponds to 201850 cells). This indicates that satellite switches need to provide at least so much buffering to avoid loss on these high delay paths. A point to consider is that these large queues should not be seen in downstream workgroup or WAN switches, because they will not provide so much buffering. Satellite switches can isolate downstream switches from such large queues by implementing the VSVD option as described in chapter 10.

	Video Sources		ABR Metrics				
#	Avg Src rate (Mbps)	STD (Mbps)	Max Switch Q (cells)	Total TCP T'put	VBR T'put	ABR T'put	Effcy (% Max T'put)
1.	5	5	255034 (1.3×f/b delay)	63.4	47.33	102.43	79.35%
2.	7.5	7	189276 (0.9×f/b delay)	59.8	57.23	92.53	82.86%
3.	10	5	148107 (0.7×f/b delay)	53.4	67.55	82.21	83.33%

Table 8.14: Maximum Queues for Satellite Networks with Long Feedback Delay: MSS=512 bytes

	Video Sources		ABR Metrics				
#	Avg Src rate (Mbps)	STD (Mbps)	Max Switch Q (cells)	Total TCP T'put	VBR T'put	ABR T'put	Effcy (% Max T'put)
1.	5	5	176007 (0.9×f/b delay)	71.92	47.33	102.43	80.70%
2.	7.5	7	252043 (1.3×f/b delay)	67.86	57.23	92.53	84.29%
3.	10	5	148646 (0.7×f/b delay)	59.33	67.55	82.21	82.95%

Table 8.15: Maximum Queues for Satellite Networks with Long Feedback Delay: MSS=9140 bytes

8.18.5 Summary of the effect of long-range dependent VBR

In this section, we have described how to model several multiplexed MPEG-2 video sources over VBR. Compressed video sources exhibit long-range dependence in the traffic patterns they generate. The effect of this long-range dependence is to introduce high variation in the ABR capacity. However a good switch scheme like ERICA+ is sufficient to handle this variation in ABR capacity. This results in

controlled ABR queues and high utilization. The maximum ABR queue length is a function of the feedback delay and round trip time. This implies that switches terminating satellite links should provide buffers proportional to the length of the satellite link in order to deliver high performance. Further, if they implement the VSVD option, they can isolate downstream workgroup switches from the effects of the long delay satellite path. We also briefly survey VBR video modeling techniques, the MPEG-2 over ATM approach, and propose a model for MPEG-2 video over VBR which incorporates the long-range dependence property in compressed video.

8.19 Effect of bursty TCP applications

In a related work [79], we have studied the effect of bursty applications running on top of TCP. An example of such an application is the World Wide Web application. The WWW application sets up TCP connections for its data transfers [30]. The WWW application differs from a large file transfer application in that while the latter looks like an "infinite or persistant" application to TCP, the former looks like a "bursty" application (with active and idle transmission periods). The effect of this on traffic management is described below.

TCP increases its "congestion window" as it receives acknowledgements for segments correctly received by the destination. If the application (eg. file transfer or WWW server/client) has data to send, it transmits the data. Otherwise, the window remains open until either the application has data to send or TCP times out (using a timer set by its RTT estimation algorithm). If the timer goes off, TCP reduces the congestion window to one segment (the minimum possible), and rises exponentially ("slow start") once the source becomes active again.

On the other hand, if the application remains idle for a period smaller than the timeout, the window is still open when the source becomes active again. If acknowledgements (corresponding to the data sent) are received within this idle interval, the window size increases further. Since no new data is sent during the idle interval, the usable window size is larger. The effect is felt when the application sends data in the new burst. Such behavior is possible by WWW applications using the HTTP/1.1 standard [30].

When TCP carrying such a WWW application runs over ATM, the burst of data is simply transferred to the network interface card (NIC). Assuming that each TCP connection is carried over a separate ABR VC, the data burst is sent into the ATM network at the VC's ACR. Since this VC has been idle for a period shorter than the TCP timeout (typically 500 ms for ATM LANs and WANs), it is an "ACR retaining" VC. Source End System (SES) Rule 5 specifies that the ACR of such a VC be reduced to ICR if the idle period is greater than parameter ADTF (which defaults to 500 ms). With this default value of ADTF, and the behavior of the TCP application, we are in a situation where the ACR is not reduced to ICR. This situation can be potentially harmful to the switches if ACRs are high and sources simultaneously send data after their idle periods.

Observe that an infinite application using TCP over ABR does not send data in such sudden bursts. As discussed in previous sections, the aggregate TCP load at most doubles every round trip time (since two packets are inserted into the network for every packet transmitted, in the worst case). Bursty TCP applications may cause the aggregate ABR load to more than double in a round trip time.

Note that the service capabilities in such a situation is also affected by a Use-it-or-Lose-it (UILI) implementation at the source or switch (as described in chapter 7). The UILI mechanism would reduce the source rate of the VC's carrying bursty TCP connections, and hence control the queues at the network switches. The effect of UILI in such conditions is for future study.

However, it has been shown that such worst case scenarios may not appear in practice due to the nature of WWW applications and the ABR closed-loop feedback mechanism [79]. Note that since the WWW traffic exhibits higher variation, techniques like averaging of load, and compensation for residual error (queues) as described in section 8.16.1 need to be used to minimize the effects of load variation. In summary, though bursty applications on TCP can potentially result in unbounded queues, a well-designed ABR system can scale well to support a large number of applications like bursty WWW sources running over TCP.

8.20 Summary of TCP over ABR results

This section unifies the conclusions drawn in each of the sections in this chapter (see sections 8.13, 8.15.6, 8.16.2, 8.18.5, and 8.19). In brief, the ABR service is an attractive option to support TCP traffic scalably. It offers high throughput for bulk file transfer applications and low latency to WWW applications. Further, it is fair to connections, which means that access will not be denied, and performance will not be unnecessarily degraded for any of the competing connections. This chapter shows that it is possible to achieve zero cell loss for a large number of TCP connections with a small amount of buffers. Hence, the ABR implementators can tradeoff the complexity

of managing buffers, queueing, scheduling and drop policies with the complexity of implementing a good ABR feedback scheme.

In section 8.13 we first noted that TCP can achieve maximum throughput over ATM if switches provide enough buffering to avoid loss. The study of TCP dynamics over the ABR service showed that initially when the network is underloaded or the TCP sources are starting to send new data, they are limited by their congestion window sizes (window-limited), rather than by the network-assigned rate (rate-limited). When cell losses occur, TCP loses throughput due to two reasons - **a)** a single cell loss results in a whole TCP packet to be lost, and, **b)** TCP loses time due to its large timer granularity. Intelligent drop policies (like avoiding drop of "End of Message (EOM)" cells, and "Early Packet Discard (EPD)" can help improve throughput). A large number of TCP sources can increase the total thoughput because each window size is small and the effect of timeout and the slow start procedure is reduced. We also saw that the ATM layer "Cell Loss Ratio (CLR)" metric is not a good indicator of TCP throughput loss. Further, we saw that the switch buffers should not be dimensioned based on the ABR Source parameter "Transient Buffer Exposure (TBE)". Buffer dimensioning should be based upon the performance of the switch algorithm (for ABR), and the round trip time.

In section 8.15.6, we summarized the derivation and simulation of switch buffer requirements for maximum throughput of TCP over ABR. The factors affecting the buffer requirements are round trip time, switch algorithm parameters, feedback delay, presence of VBR traffic, or two-way TCP traffic. For a switch algorithm like ERICA, the buffer requirements are about $3 \times RTT \times Link\_bandwidth$. The derivation is valid for infinite applications (like file transfer) running over TCP. Though the queueing

inside the ATM network can be controlled with the ABR service, in a heterogeneous network environment, the cells belonging to TCP streams may queue at the edge routers, i.e., at the entrance to the ATM network. Some form of end-to-end flow control involving the TCP end system is still necessary to avoid cell loss under such conditions.

In section 8.16.2 and 8.18.5, we studied the effect of the VBR background traffic patterns on the buffer requirements for TCP over ABR. The effect of the background traffic is to create variation in ABR capacity. The switch algorithm needs to be robust to handle the variation in ABR capacity (due to VBR) and in ABR demand (due to TCP dynamics). We motivate three enhancements to the ERICA+ scheme which reduce the effect of the variation and allow the convergence of the ABR queues, without compromising on efficiency. We then use a model of several multiplexed MPEG-2 video sources over VBR. In this effort, we also briefly survey VBR video modeling techniques, the MPEG-2 over ATM approach, and propose a model for MPEG-2 video over VBR which incorporates the long-range dependence property in compressed video. Compressed video sources exhibit long-range dependence in the traffic patterns they generate. We verify that the ERICA+ algorithm is robust to the variation introduced by such background traffic and can control the ABR queues.

Finally, in section 8.19, we refer to a related study of the effect of bursty applications (such as the World Wide Web application) running on top of TCP. Bursty applications can potentially cause unbounded ABR queues since they can use open TCP windows to send bursts of data. However, Vandalore et al [79] show that since ABR switches respond to load increases, if the aggregate load increases as a function of the number of applications, then the switch will assign lower rates to sources and

hence control the total load on the network. In other words, a well-designed ABR system can scale well to support a large number of applications like persistant file transfer or bursty WWW sources running over TCP.

CHAPTER 9

THE VIRTUAL SOURCE/VIRTUAL DESTINATION (VS/VD) FEATURE: DESIGN CONSIDERATIONS

One of the architectural features in the ABR specification is the Virtual Source/Virtual Destination (VS/VD) option. This option allows a switch to divide an end-to-end ABR connection into separately controlled ABR segments by acting like a destination on one segment, and like a source on the other. The coupling in the VS/VD switch between the two ABR control segments is implementation specific. In this section, we model a VS/VD ATM switch and study the issues in designing coupling between ABR segments. We identify a number of implementation options for the coupling. We show that a good choice significantly improves the stability and transient performance of the system and reduces the buffer requirements at the switches.

As mentioned, the VS/VD option allows a switch to divide an ABR connection into separately controlled ABR segments. On one segment, the switch behaves as a destination end system, i.e., it receives data and turns around resource management (RM) cells (which carry rate feedback) to the source end system. On the other segment the switch behaves as a source end system, i.e., it controls the transmission rate of every virtual circuit (VC) and schedules the sending of data and RM cells. We

call such a switch a "VS/VD switch". In effect, the end-to-end control is replaced by segment-by-segment control as shown in Figure 9.1.

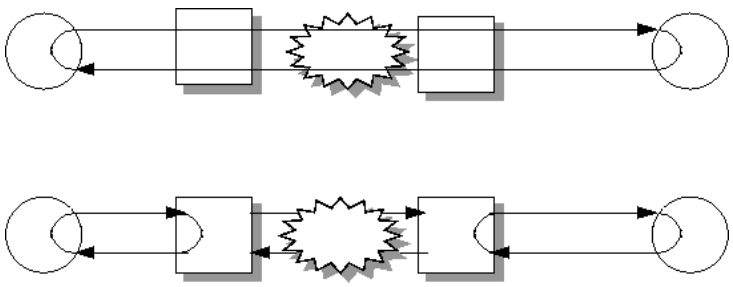


Figure 9.1: End-to-End Control vs VS/VD Control

One advantage of the segment-by-segment control is that it isolates different networks from each other. One example is a proprietary network like frame-relay or circuit-switched network between two ABR segments, which allows end-to-end ABR connection setup across the proprietary network and forwards ATM packets between the ABR segments (signaling support for this possibility is yet to be considered by the ATM Forum). Another example is the interface point between a satellite network and a LAN. The gateway switches at the edge of a satellite network can implement VS/VD to isolate downstream workgroup switches from the effects of the long delay satellite paths (like long queues). A second advantage of segment-by-segment control is that the segments have shorter feedback loops which can potentially improve performance because feedback is given faster to the sources whenever new traffic bursts are seen. The VS/VD option requires the implementation of per-VC queueing and scheduling at the switch.

The goal of this study is find answers to the following questions:

- Do VS/VD switches really improve ABR performance?
- What changes to switch algorithms are required to operate in VS/VD environments?
- Are there any side-effects of having multiple control loops in series?

Specifically, we study the requirements to implement the ERICA algorithm in a VS/VD switch. We describe our switch model and the use of the ERICA algorithm in sections 9.1 and 9.2. The VS/VD design options are listed and evaluated in sections 9.3 and 9.4, and summarized in section 9.6.

9.1 Switch Queue Structure

In this section, we first present a simple switch queue model for the non-VS/VD switch and later extend it to a VS/VD switch by introducing per-VC queues. The flow of data, forward RM (FRM) and backward RM (BRM) cells is also closely examined.

9.1.1 A Non-VS/VD Switch

A minimal non-VS/VD switch has a separate FIFO queue for each of the different service classes (ABR, UBR etc.). We refer to these queues as "per-class" queues. The ABR switch rate allocation algorithm is implemented at every ABR class queue. This model of a non-VS/VD switch based network with per-class queues is illustrated in Figure 9.2.

Besides the switch, the figure shows a source end system, S, and a destination end system, D, each having per-VC queues to control rates of individual VCs. For example, ABR VCs control their Allowed Cell Rates (ACRs) based upon network feedback.

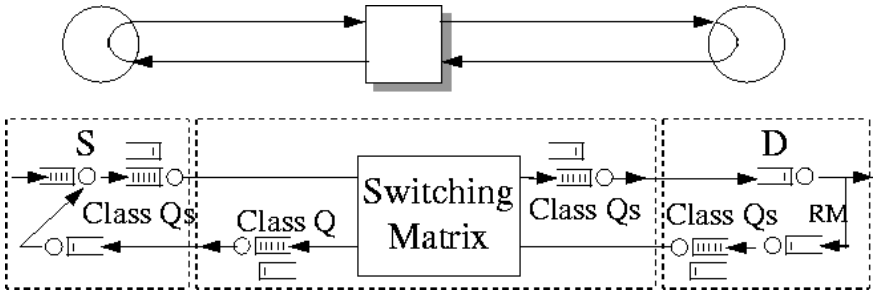


Figure 9.2: Per-class queues in a non-VS/VD switch

We assume that the source/destination per-VC queues feed into corresponding per-class queues (as shown in the figure) which in turn feed to the link. This assumption is not necessary in practice, but simplifies the presentation of the model. The contention for link access between cells from different per-class queues (at the switch, the source and the destination) is resolved through appropriate scheduling.

9.1.2 A VS/VD Switch

The VS/VD switch implements the source and the destination end system functionality in addition to the normal switch functionality. Therefore, like any source and destination end-system, it requires per-VC queues to control the rates of individual VCs. The switch queue structure is now more similar to the source/destination structure where we have per-VC queues feeding into the per-class queues before each link. This switch queue structure and a unidirectional VC operating on it is shown in Figure 9.3.

The VS/VD switch has two parts. The part known as the Virtual Destination (VD) forwards the data cells from the first segment ("previous loop") to the per-VC queue at the Virtual Source (VS) of the second segment ("next loop"). The other part

or the Virtual Source (of the second segment) sends out the data cells and generates FRM cells as specifed in the source end system rules.

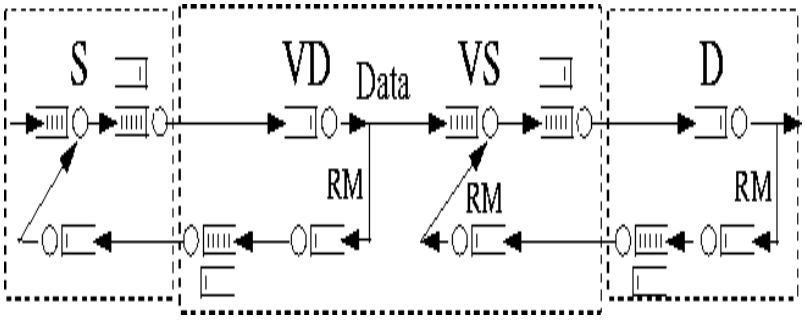


Figure 9.3: Per-VC and per-class queues in a VS/VD switch (a)

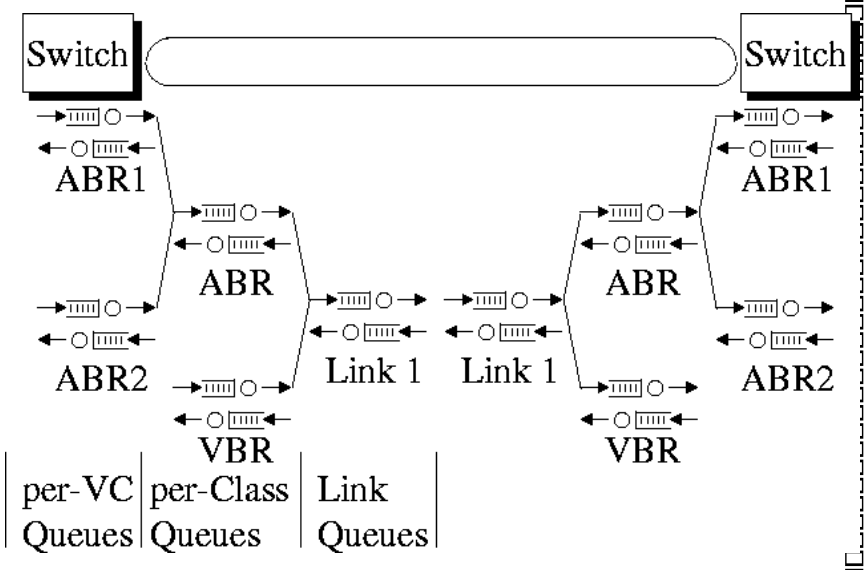


Figure 9.4: Per-VC and per-class queues in a VS/VD switch (b)

The switch also needs to implement the switch congestion control algorithm and calculate the allocations for VCs depending upon its bottleneck rate. A question

which arises is where the rate calculations are done and how the feedback is given to the sources. We postpone the discussion of this question to later sections.

9.1.3 A VS/VD Switch with Unidirectional Data Flow

The actions of the VS/VD switch upon receiving RM cells are as follows. The VD of the previous loop turns around FRM cells as BRM cells to the VS on the same segment (as specified in the destination end system rules (see chapter 2)). Additionally, when the FRM cells are turned around, the switch may decrease the value of the explicit rate (ER) field to account for the bottleneck rate of the next link and the ER from the subsequent ABR segments.

When the VS at the next loop receives a BRM cell, the ACR of the per-VC queue at the VS is updated using the ER field in the BRM (ER of the subsequent ABR segments) as specified in the source end system rules). Additionally, the ER value of the subsequent ABR segments needs to be made known to the VD of the first segment. One way of doing this is for the VD of the first segment to use the ACR of the VC in the VS of the next segment while turning around FRM cells.

The model can be extended to multiple unidirectional VCs in a straightforward way. Figure 9.5 shows two unidirectional VCs, VC1 and VC2, between the same source S and destination D which go from Link1 to Link2 on a VS/VD switch. Observe that there is a separate VS and VD control for each VC. We omit non-ABR queues in this and subsequent figures.

9.1.4 Bi-directional Data Flow

Bi-directional flow in a VS/VD switch (Figure 9.6) is again a simple extension to the above model. The data on the previous loop VD is forwarded to the next loop VS.

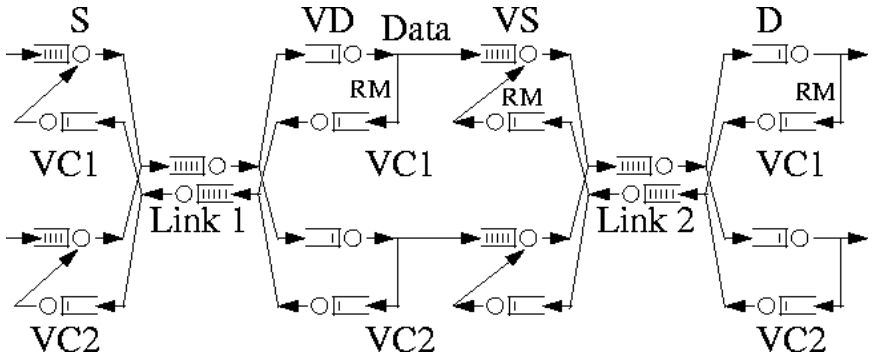


Figure 9.5: Multiple unidirectional VCs in a VS/VD switch

FRMs are turned around by the previous loop VD to the previous loop VS. BRMs are processed by the next loop VS to update the corresponding ACRs.

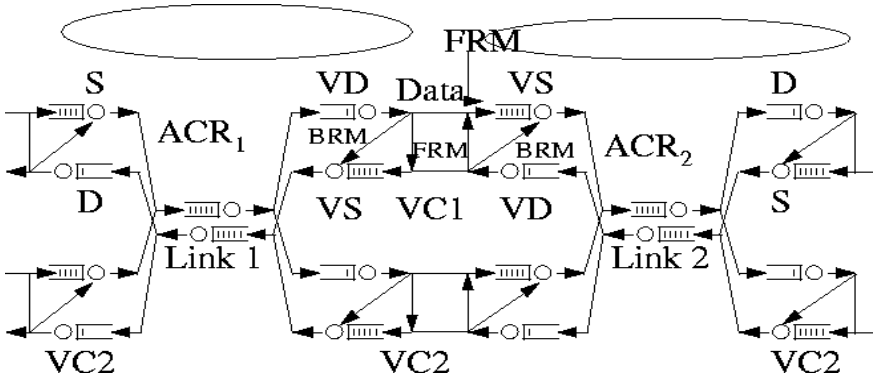


Figure 9.6: Multiple bi-directional VCs in a VS/VD switch

We will discuss the rates and allocations of *VC1 only*. VC1 has two ACRs: ACR_1 in the reverse direction on Link1 and ACR_2 in the forward direction on Link2. Henceforth, the subscript 1 refers to the "previous loop" variables and subscript 2 to the "next loop" variables of VC1.

9.2 The ERICA Switch Scheme: Renotated

In this section, we introduce some new notation for the ERICA algorithm which we use later in this section to explain its implementation in a VS/VD switch.

The ERICA target rate is set as follows:

Target Rate = Target Utilization × Link Rate - VBR (high priority) Rate.

ERICA measures the input rate to the ABR queue and the number of active ABR sources.

To achieve fairness, the VC's Allocation (VA) has a component:

$VA_{fairness}$ = Target Rate / Number of Active VCs

To achieve efficiency, the VC's Allocation (VA) has a component:

$VA_{efficiency}$ = VC's Current Cell Rate / Overload, where Overload = Input Rate / Target Rate;

Finally, the VC's allocation on this link (VAL) is calculated as:

VAL = Max{ $VA_{efficiency}$, $VA_{fairness}$ } = Function{ Input Rate, VC's current rate }

We use this basic algorithm to illustrate the VS/VD implementation. The implementation of the full scheme can be derived as a simple extension to the description given in this section.

9.2.1 Rate Calculations in a non-VS/VD Switch

The non-VS/VD switch calculates the rate (VAL) for sources when the BRMs are processed in the reverse direction and enters it in the BRM field as follows:

ER in BRM = Min{ ER in BRM, VAL }

At the source end system, the ACR is updated as:

ACR = Function{ ER, VC's current ACR }

9.2.2 Rate Calculations in a VS/VD Switch

Figure 9.7 shows the rate calculations in a VS/VD switch. Specifically, the segment starting at Link2 ("next loop") returns an ER value, ER_2 in the BRM, and the FRM of the first segment ("previous loop") is turned around with an ER value of ER_1. The ERICA algorithm for the port to Link2 calculates a rate (VAL_2) as: VAL_2 = Function { Input Rate, VC's Current Rate }. The rate calculations at the VS and VD are as follows:

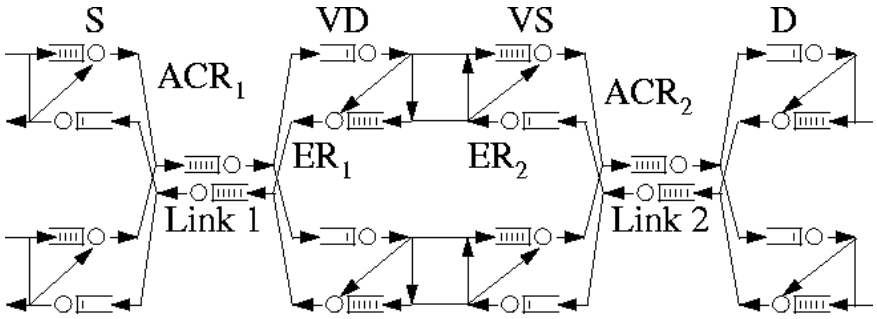


Figure 9.7: Rate calculations in VS/VD switches

- Destination algorithm for the *previous loop*:

  ER_1 = Min { ER_1, VAL_2, ACR_2 }

- Source Algorithm for the *next loop*:

  Optionally, ER_2 = Min { ER_2, VAL_2 }

  ACR_2 = Fn { ER_2, ACR_2 }

The unknowns in the above equations are the input rate and the VC's current rate. We shall see in the next section that there are several ways of measuring VC rates and input rates, combining the feedback from the next loop, and updating the ACR of the next loop. Note that though different switches may implement different algorithms, many measure quantities such as the VC's current rate and the ABR input rate.

9.3 VS/VD Switch Design Options

In this section, we aim at answering the following questions:

- What is a VC's current rate? (4 options)
- What is the input rate? (2 options)
- Does the congestion control actions at a link affect the next loop or the previous loop? (3 options)
- When is the VC's allocation at the link (VAL) calculated? (3 options)

We will enumerate the 72 ($= 4 \times 2 \times 3 \times 3$) option combinations and then study this state space for the best combination.

9.3.1 Measuring the VC's Current Rate

There are four methods to measure the VC's current rate:

1. The rate of the VC is declared by the source end system of the previous loop in the Current Cell Rate (CCR) field of the FRM cell (FRM1) received by the VD. This declared value can be used as the VC's rate.

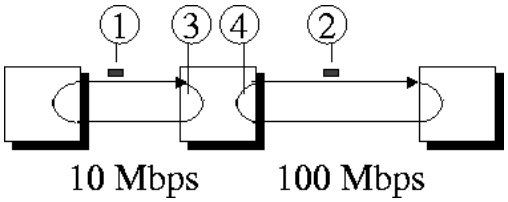


Figure 9.8: Four methods to measure the rate of a VC at the VS/VD switch

2. The VS to the next loop declares the CCR value of the FRM sent (FRM2) to be its ACR (ACR_2). This declared value can be used as the VC's rate.

3. The actual source rate in the *previous loop* can be measured. This rate is equal to the VC's input rate to the per-VC queue. This measured source rate can be used as the VC's rate.

4. The actual source rate in the *next loop* can be measured as the VC's input rate to the per-class queue (from the per-VC queue). This measured value can be used as the VC's rate.

Figure 9.8 illustrates where each method is applied (note the position of the numbers in circles).

9.3.2 Measuring the Input Rate at the Switch

Figure 9.9 (note the position of the numbers in circles) shows two methods of estimating the input rate for use in the switch algorithm calculations. These two methods are:

1. The input rate is the sum of input rates to the per-VC ABR queues.

2. The input rate is the aggregate input rate to the per-class ABR queue.

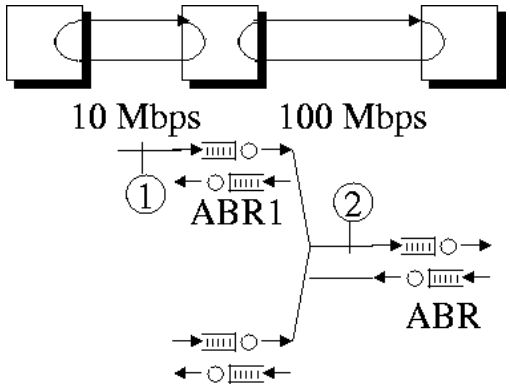


Figure 9.9: Two methods to measure the input rate at the VS/VD switch

9.3.3 Effect of Link Congestion Actions on Neighboring Links

The link congestion control actions can affect neighboring links. The following actions are possible in response to the link congestion of Link2:

1. Change ER_1. This affects the rate of the *previous loop only*. The change in rate is experienced only after a feedback delay equal to twice the propogation delay of the loop.

2. Change ACR_2. This affects the rate of the *next loop only*. The change in rate is experienced instantaneously.

3. Change ER_1 and ACR_2. This affects *both the previous and the next loop*. The next loop is affected instantaneously while the previous loop is affected after a feedback delay as in the first case.

9.3.4 Frequency of Updating the Allocated Rate

Recall that the ERICA algorithm in a non-VS/VD switch calculates the allocated rate when a BRM cell is processed in a switch. However, in a VS/VD switch, there are three options as shown in Figure 9.10:

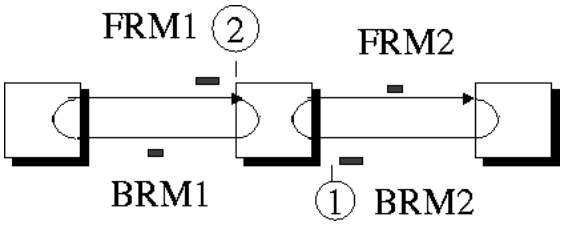


Figure 9.10: Three methods to update the allocated rate

1. Calculate allocated rate *on receiving BRM2 only.* Store the value in a table and use this table value when an FRM is turned around.

2. Calculate allocated rate *only when FRM1 is turned around.*

3. Calculate allocated rate *both when FRM1 is turned around as well as when BRM2 is received.*

In the next section, we discuss the various options and present analytical arguments to eliminate certain design combinations.

9.4 VS/VD Switch Design Options

9.4.1 VC Rate Measurement Techniques

We have presented four ways of finding the the VC's current rate in section 9.3.1, two of them used declared rates and two of them measured the actual source rate. We show that measuring source rates is better than using declared rates for two reasons.

First, the declared VC rate of a loop naively is the minimum of bottleneck rates of *downstream loops only*. It does not consider the bottleneck rates of upstream loops, and may or may not consider the bottleneck rate of the first link of the next loop. Measurement allows better estimation of load when the traffic is not regular.

Second, the actual rate of the VC may be lower than the declared ACR of the VC due to dynamic changes in bottleneck rates upstream of the current switch. The difference in ACR and VC rate will remain *at least* as long as the time required for new feedback from the bottleneck in the path to reach the source plus the time for the new VC rate to be experienced at the switch. The sum of these two delay components is called the "feedback delay." Due to feedback delay, it is possible that the declared rate is a stale value at any point of time. This is especially true in VS/VD switches where per-VC queues may control source rates to values quite different from their declared rates.

Further, the measured source rate is already available in a VS/VD switch because it is measured as part of one of the source end system rules (SES Rule 5) (see chapter 7).

9.4.2 Input Rate measurement techniques

As discussed earlier, the input rate can be measured as the sum of the input rates of VCs to the per-VC queues or the aggregate input rate to the per-class queue. These two rates can be different because the input rate to the per-VC queues is at the previous loop's rate while the input to the per-class queue is related to the next loop's rate. Figure 9.11 shows a simple case where two adjacent loops can run at very different rates (10 Mbps and 100Mbps) for one feedback delay.

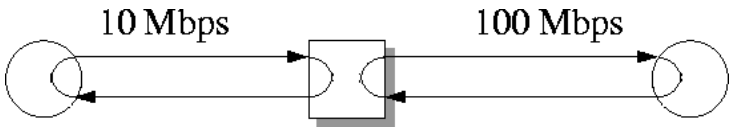


Figure 9.11: Two adjacent loops may operate at very different rates for one feedback delay

9.4.3 Combinations of VC rate and input rate measurement options

Table 9.1 summarizes the option combinations considering the fact that two adjacent loops may run at different rates. The table shows that four of these combinations may work satisfactorily. The other combinations use inconsistent information and hence may either overallocate rates leading to unconstrained queues or result in unnecessary oscillations. We can eliminate some more cases as discussed below.

The above table does not make any assumptions about the queue lengths at any of the queues (per-VC or per-class). For example, when the queue lengths are close to

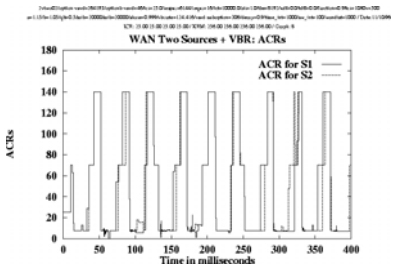
#	VC Rate Method rates (Mbps)	Σ VC	Input Rate Method	Input Rate Value	Choice (YES/NO)
1.	From FRM1	10	Σ per-VC	10	**YES**
2.	From FRM1	10	per-class	10-100	NO
3.	From FRM2	100	Σ per-VC	10	NO
4.	From FRM2	100	per-class	100	**YES**
5.	At per-VC queue	10	Σ per-VC	10	**YES**
6.	At per-VC queue	10	per-class	10-100	NO
7.	At per-class queue	100	Σ per-VC	10	NO
8.	At per-class queue	100	per-class	100	**YES**

Table 9.1: Viable combinations of VC rate and input rate measurement

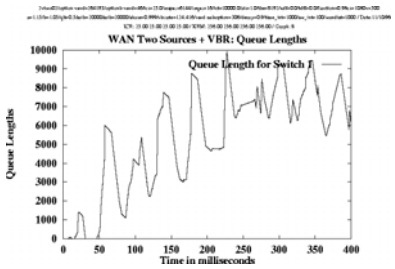
zero, the actual source rate might be much lower than the declared rate in the FRMs leading to overallocation of rates. This criterion can be used to reject more options.

The performance of one such rejected case is shown in Figure 9.12 (corresponding to row 4 in Table 9.1). The configuration used has two ABR infinite sources and one high priority VBR source contending for the bottleneck link's (LINK1) bandwidth. The VBR has an ON/OFF pattern, where it uses 80% of the link capacity when ON. The ON time and the OFF time are equal (20 ms each). The VS/VD switch overallocates rates when the VBR source is OFF. This leads to ABR queue backlogs when the VBR source comes ON in the next cycle. The queue backlogs are never cleared, and hence the queues diverge. *In this case, the fast response of VS/VD is harmful* because the rates are overallocated.

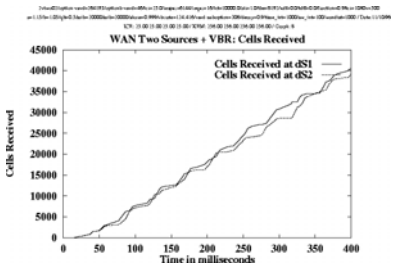
In this study, we have not evaluated row 5 of the table (measurement of VC rate at entry to the per-VC queues). Hence, out of the total of 8 combinations, we consider two viable combinations: row 1 and row 8 of the table. Note that since row



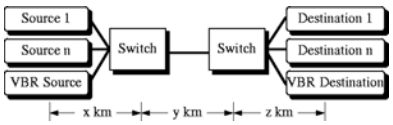
(a) ACR



(b) Queue Lengths



(c) Cells Received



(d) Configuration

Figure 9.12: 2-source+VBR configuration. Unconstrained queues due to overallocation.

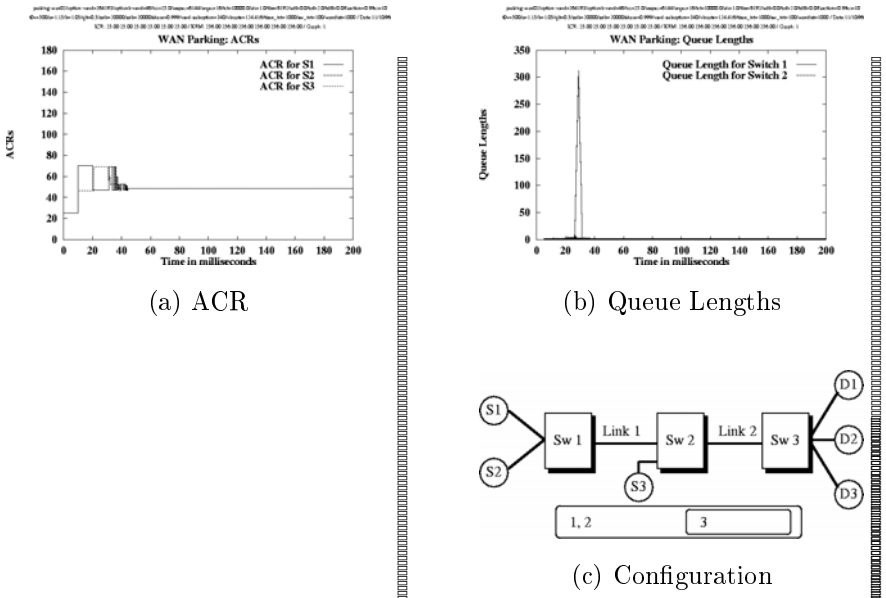
8 uses source rate measurement, we expect it to show better performance. This is substantiated by our simulation results presented later in the paper.

9.4.4 Effect of Link Congestion Control Actions

In a network with non-VS/VD switches only, the bottleneck rate needs to reach the sources before any corresponding load change is seen in the network. However, a VS/VD switch can enforce the new bottleneck rate immediately (by changing the ACR of the per-VC queue at the VS). This rate enforcement affects the utilization of links in the next loop. Hence, the VS/VD link congestion control actions can affect neighboring loops. We have enumerated three options in an earlier section.

We note that the second option ("next loop only") does not work because the congestion information is not propagated to the sources of the congestion (as required by the standard [32]). This leaves us with two alternatives. The third option ("both loops") is attractive because, when ACR_2 is updated, the switches in the next loop experience the load change faster. Switch algorithms may save a few iterations and converge faster in these cases.

Figure 9.13 shows the fast convergence in a parking lot configuration when such a VS/VD switch is used. The parking lot configuration (Figure 9.13(c)) consists of three VCs contending for the Sw2-Sw3 link bandwidth. Link lengths are 1000 km and link bandwidths are 155.52 Mbps. The target rate of the ERICA algorithm was 90% of link bandwidth i.e., 139.97 Mbps. The round trip time for the S3-D3 VC is shorter than the round trip time for the other two VCs. The optimum allocation by ERICA for each source is 1/3 of the target rate on the Sw2-Sw3 (about 46.7 Mbps).



(a) ACR

(b) Queue Lengths

(c) Configuration

Figure 9.13: Parking lot: best VS/VD option converges fast

Figure 9.13(a) shows that the optimum value is reached at 40 ms. Part (b) of the figure shows that the transient queues are small. Further, the allocation is fair.

9.4.5 Link Congestion and Allocated Rate Update Frequency: Viable Options

The allocated rate update has three options:

a) update upon BRM receipt (in VS) and enter the value in a table to be used when an FRM is turned around,

b) update upon FRM turnaround (at VD) and no action at VS,

c) update both at FRM (VD) and at BRM (VS) without use of a table.

The last option recomputes the allocated rate a larger number of times, but can potentially allocate rates better because we always use the latest information.

The allocated rate update and the effects of link congestion actions interact as shown in Figure 9.14. The figure shows a tree where the first level considers the link congestion (2 options), i.e., whether the next loop is also affected or not. The second level lists the three options for the allocated rate update frequency. The viable options are those highlighted in bold at the leaf level.

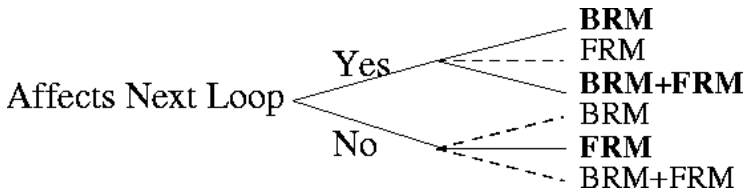


Figure 9.14: Link congestion and allocated rate update: viable options

Other options are not viable because of the following reasons. In particular, if the link congestion does not affect the next loop, the allocated rate update at the FRM turnaround is all that is required. The allocated rate at the BRM is redundant in this case. Further, if the link congestion affects the next loop, then the allocated rate update has to be done on receiving a BRM, so that ACR can be changed at the VS. This gives us two possibilities as shown in the figure (BRM only, and BRM+FRM).

Hence, we have three viable combinations of link congestion and the allocated rate update frequency. A summary of all viable options (a total of 6) is listed in Table 9.2.

The next section evaluates the performance of the viable VS/VD design options through simulation.

Option #	VC Rate Method	Input Rate Measurement point	Link Congestion Effect	Allocated Rate Updated at
A	From FRM1	per-VC	prev loop only	FRM1 only
B	At per-class Q	per-class	both loops	FRM1 only
C	From FRM1	per-VC	both loops	FRM1 only
D	At per-class Q	per-class	both loops	FRM1 and BRM2
E	From FRM1	per-VC	both loops	BRM2 only
F	At per-class Q	per-class	both loops	BRM2 only

Table 9.2: Summary of viable VS/VD design alternatives

9.5 Performance Evaluation of VS/VD Design Options

9.5.1 Metrics

We use four metrics to evaluate the performance of these alternatives:

- **Response Time:** is the time taken to reach near optimal behavior on startup.
- **Convergence Time:** is the time for rate oscillations to decrease (time to reach the steady state).
- **Throughput:** Total data transferred per unit time.
- **Maximum Queue:** The maximum queue before convergence.

The difference between response time and convergence time is illustrated in Figure 9.15. The following sections present simulation results with respect to the above metrics. Note that we have used greedy (infinite) traffic sources in our simulations. We have studied the algorithmic enhancements in non-VS/VD switches for non-greedy

sources in chapter 6. We expect the best implementation option (see below) to work well and produce consistent results when such (bursty) traffic is used.

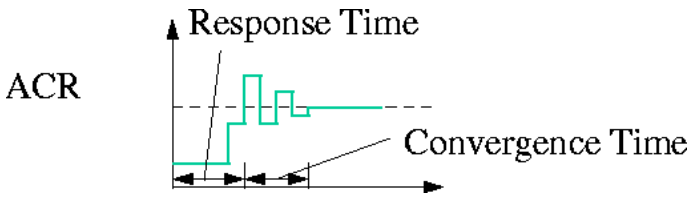


Figure 9.15: Response time vs Convergence time

Response Time

Without VS/VD all response times are close to the round-trip delay. With VS/VD, the response times are close to the feedback delay from the bottleneck. Since VS/VD reduces the response time during the first round trip, it is good for long delay paths. The quick response time (10 ms in the parking lot configuration which has a 30 ms round trip time) is shown in Figure 9.13.

Response time is also important for bursty traffic like TCP file transfer over ATM which "starts up" at the beginning of every active period (when the TCP window increases) after the corresponding idle period (see chapter 7).

Throughput

The number of cells received at the destination is a measure of the throughput achieved. These values are listed in Table 9.3. The top row is a list of the configuration codes (these codes are explained in Table 9.2. The final column lists the throughput values for the case when a non-VS/VD switch is used. The 2 source-VBR and the parking lot configurations have been introduced in earlier section.

The upstream bottleneck configuration shown in Figure 5.19 has a bottleneck at Sw1 where 15 VCs share the Sw1-Sw2 link. As a result the S15-D15 VC is not capable of utilizing its bandwidth share at the Sw2-Sw3 link. This excess bandwidth needs to be shared equally by the other two VCs. The table entry shows the number of cells received at the destination for either the S16-D16 VC or the S17-D17 VC.

In the 2 source+VBR and the upstream bottleneck configurations, the simulation was run for 400 ms (the destination receives data from time = 15 ms through 400 ms). In the parking lot configuration, the simulation was run for 200ms.

VS/VD Opt # → Config ↓	A	B	C	D	E	F	No VS/VD
2 source + VBR	31	31	32.5	34	32	33	30
Parking lot	22	22	23	20.5	23	20.5	19.5
Upstream bottleneck	61	61	61	60	61	61	62

Table 9.3: Cells received at the destination per source in Kcells

As we compare the values in each row of the table, we find that, in general, there is *little difference between the alternatives in terms of throughput.* However, there is a slight increase in throughput when VS/VD is used over the case without VS/VD switch.

Convergence Time

The convergence time is a measure of how fast the scheme finishes the transient phase and reaches steady state. It is also sometimes called "transient response." The convergence times of the various options are shown in Table 9.4. The "transient"

configuration mentioned in the table has two ABR VCs sharing a bottleneck (like the 2 source + VBR configuration, but without the VBR VC). One of the VCs comes on in the middle of the simulation and remains active for a period of 60 ms before going off.

VS/VD Opt # → Config ↓	A	B	C	D	E	F	No VS/VD
Transient	50	50	65	**20**	55	25	60
Parking lot	120	100	170	**45**	125	50	140
Upstream bottleneck	95	75	75	**20**	95	**20**	70

Table 9.4: Convergence time in ms

Observe that the convergence time of VS/VD option D (highlighted) is the best. Recall that this configuration corresponds to measuring the VC rate at the entry to the per-class queue, input rate measured at the per-class queue, link congestion affecting both the next loop and the previous loop, the allocated rate updated at both FRM1 and BRM2.

Maximum Transient Queue Length

The maximum transient queues gives a measure of how askew the allocations were when compared to the optimal allocation and how soon this was corrected. The maximum transient queues are tabulated for various configurations for each VS/VD option and for the case without VS/VD in Table 9.5.

The table shows that VS/VD option D has very small transient queues in all the configurations and the minimum queues in a majority of cases. This result, combined

VS/VD Opt # → Config ↓	A	B	C	D	E	F	No VS/VD
2 Source + VBR	**1.2**	1.4	2.7	1.8	2.7	1.8	2.7
Transient	1.4	1.1	1.4	**0.025**	1.3	1.0	6.0
Parking lot	1.9	1.9	1.4	**0.3**	3.7	0.35	2.0
Upstream bottleneck	0.025	0.08	0.3	**0.005**	1.3	**0.005**	0.19

Table 9.5: Maximum queue length in Kcells

with the fastest response and near-maximum throughput behavior confirms the choice of option D as the best VS/VD implementation.

Observe that the queues for the VS/VD implementations are in general lesser than or equal to the queues for the case without VS/VD. However, the queues reduce much more if the correct implementation (like option D) is chosen.

9.6 Conclusions

In summary:

- VS/VD is an option that can be added to switches which implement per-VC queueing. The addition can potentially yield improved performance in terms of response time, convergence time, and smaller queues. This is especially useful for switches at the edge of satellite networks or switches that are attached to links with large delay-bandwidth product. The fast response and convergence times also help support bursty traffic like data more efficiently.

- The effect of VS/VD depends upon the switch algorithm used and how it is implemented in the VS/VD switch. The convergence time and transient queues

can be very different for different VS/VD implementations of the same basic switch algorithm. In such cases the fast response of VS/VD is harmful.

- With VS/VD, ACR and actual rates are very different. The switch cannot rely on the RM cell CCR field. We recommend that the VS/VD switch and in general, switches implementing per-VC queueing measure the VC's current rate.

- The sum of the input rates to per-VC VS queues is not the same as the input rate to the link. It is best to measure the VC's rate at the output of the VS and the input rate at the entry to the per-class queue.

- On detecting link congestion, the congestion information should be forwarded to the previous loop as well as the next loop. This method reduces the convergence time by reducing the number of iterations required in the switch algorithms on the current and downstream switches.

- It is best for the the rate allocated to a VC to be calculated both when turning around FRMs at the VD as well as after receiving BRMs at the next VS.

We have shown that the VS/VD provision in the ABR traffic management framework can potentially improve performance of bursty traffic and reduce the buffer requirements in switches. The VS/VD mechanism achieves this by breaking up a large ABR loop into smaller ABR loops which are separately controlled. However, further study is required in the following areas:

- Effect of VS/VD on buffer requirements in the switch.

- Scheduling issues with VS/VD.

- Effect of different switch algorithms in different control loops, and different control loop lengths.
- Effect of non-ABR clouds and standardization issues involved.

CHAPTER 10

IMPLEMENTATION ISSUES

At the time of this writing, the Traffic Management 4.0 [32] which includes the ABR specification has been available for a year and a half. However, the first products implementing ABR are just entering the market. The reason for this long delay is in part because of the complexity of ABR implementation. We explore some of the issues in this chapter and study the implementation and performance of one of the ABR options, namely, Virtual Source/Virtual Destination, in depth. We will also mention some of the efforts currently underway to make the ABR service more attractive.

10.1 ATM Service Categories Revisited

ATM provides multiple classes of service to realize the goal of an integrated services network. The CBR and VBR services were designed primarily for voice and isochronous traffic like video. These services required the network to reserve resources. As a result, the method used to reserve resources limited the total number of CBR or VBR connections that could be setup. Data traffic did not require such resource reservations, and it could potentially use the bandwidth "left over" by CBR and VBR. Therefore, the ATM Forum decided to develop a "best-effort" service category for data traffic which uses the "left over" capacity on a physical channel. Initial

ATM data users found that their packets were being dropped indiscriminately by the network. The reason was that due to fragmentation, even a single cell loss resulted in a packet loss.

So, there was a need for a service which provides control over cell loss. ABR was initially designed to meet this need through the use of feedback control. Basically, the network explicitly distributes the "left over" capacity among the active ABR sources. During the development of the ABR service, it was realized that feedback control could also be used to provide high throughput, low delay and fairness among contending sources. The tradeoff was performance versus complexity. There was the debate between credit-based framework and the rate-based framework, and the latter was standardized because it mandated lesser *required* complexity. The standard requires the network interface card (NIC) manufacturers to implement a set of source and destination end system rules. The switches minimally need to give some kind of feedback. They can set EFCI bits on data cells and/or process RM (control) cells sent by the sources once every Nrm cell times to give feedback. Target ABR applications are file transfer, WWW, email, variable quality video and voice.

The UBR service is "unspecified" in the sense that the only standard support required from switches is the capability to accept a UBR connection request from the source. By default, there is no resource to be reserved and the connection admission control (CAC) procedure is very simple. Another implication of the service being unspecified is that nothing is guaranteed. In particular, if network gets congested, UBR cells may be dropped. The network switches may provide an enhanced UBR service by using techniques like intelligent drop policies, buffer allocation and scheduling [36]. Network monitoring traffic, email and news are examples of the UBR applications.

With an enhanced UBR service, applications like file transfer and WWW browsing and downloading become viable over this service.

The service categories of ATM can also be compared to those offered by airlines [51]. CBR is confirmed reservations with no recourse if you do not show up. VBR is like confirmed reservation but you do not pay if you do not show up. ABR is standby. You get to go if seats are available. Having standby service is good for the airline. They can fill their seats that would otherwise would have gone empty. The service is also good for passengers. They can travel cheaply particularly if they don't have to be at their destinations at a certain time. UBR service is not currently offered by the airlines. Passengers travelling on UBR class may be allowed to board a plane but may be strangled at the subsequent airports forever if seats are not available. ABR users would generally be asked to stay home as much as possible if their routes are congested.

10.2 Issues in ABR Implementation Complexity

From an architecture viewpoint, currently, ABR is a complex service to implement. The important architectural tradeoffs we will encounter involve requirements in terms of processing speed, latency, memory, and compactness. We will encounter processing speed mismatches for RM cells versus data cells. Further, the RM cell might need to be processed in both the forward and reverse directions. The latency issue arises when RM cells are processed separately, and/or in software, and/or block on a slow shared DRAM for information access. Memory requirements and access speed requirements vary depending upon the RM cell processing strategy. Compactness and overall cost depends upon the particular implementation (for example: ASIC or FPGA). In this

section, we outline some of the implementation problems for both switches and NICs, and suggest solutions.

10.2.1 Switch issues

1. ABR requires the switch to process RM cells. The processing of RM cells takes a longer time than processing data cells. As a result, the processing of RM cells may disrupt the switch pipeline mechanism. Note that a pipeline mechanism processes a job in several stages of a "pipeline" and assumes that the processing time at each stage is simple and involves the same (small) amount of time. Any task with disproportionate processing requirements disrupts the pipeline. One solution is to extract such tasks from the stream before they enter the pipeline, process them separately, and reinsert them into the stream. In this case, we require a special hardware/software design to extract, process and reinsert RM cells to/from the ABR VC. Note that this solution might extract an RM cell from one point in the stream and reinsert it at a different point. However, the traffic management 4.0 standard allows RM cells to be extracted, processed separately, and reinserted, as long as the RM cell sequence within each VC is maintained. Note that the correlation of the declared parameters with the actual stream is lost under such conditions. For example, the CCR field may not be indicative of the rate of the VC (as measured) when the RM cell is processed. Software processing of RM cells is possible if the Nrm (RM cell frequeny parameter) is negotiated appropriately (eg: use a value like 192, instead of the default value, 32).

2. Some switch schemes have a processing requirement for both the forward and backward going RM cell. This requires extra processing at the switch. Another related problem which arises in this case is that the switch scheme requires exchange of information from one port to another (the ingress chip to the egress chip). The assumption made by the scheme is that the switch has a shared memory which is used for tables facilitating the information exchange. This assumption is based on the fact that early switches had the VC table (which maps a cell of a VC from one port to another) was in such a shared memory. The problem with the shared memory is that in the worst case it needs to support accesses from all ports in a single cell time. Modern switches have evolved to use cheaper (and slower memory) to build a distributed VC table – based on the assumption that VC label allocations are relatively static (written only during connection setup, read by the local port only), and local between pairs of ports (except point-to-multipoint VCs which could involve multiple ports). One disadvantage of the distributed memory implementation is that sharing information between ports via memory is not possible. A solution to this problem is to have a cheap low speed shared memory (DRAM) for storing shared tables which are accessed when RM cells are processed. We take advantage of the fact that RM cells on every VC arrive at a frequency of at most one in $Nrm = 32$ cells. Even in the case when RM cells of multiple VCs arrive together, they need to be processed at a rate much smaller than the link rate. As mentioned before, RM cells can be staggered with respect to the data stream as long as the sequence integrity on a VC is maintained.

The ERICA scheme also works best when calculations are done at the receipt of the FRM and the BRM cell, and information is exchanged. However, it is possible to implement ERICA such that feedback can be given when the RM cell is seen in the forward direction. In this implementation, certain fields of the ERICA table (eg, the CCR field) are not required. Further, the per-VC state can be stored in memory local to the port. Also, the computations usually done when an RM cell is received can be avoided by precalculating the feedback at the end of an averaging interval for the set of active sources. In general, the averaging interval computation can be done in software as a background process. A lazy evaluation technique for the same is also possible.

3. Many switch implementations provide per-VC queueing and scheduling in order to ensure isolation of traffic and provide fairness among VCs. The ERICA algorithm does not require per-VC queueing and scheduling. But, it does assume that misbehaving sources (which do not send data according to their allocations). In a corporate network, the source end-system cards or NICs can be chosen such that they schedule the traffic depending upon the current rate. If the NIC technology cannot handle the scheduling of cells when ACRs vary rapidly, the VC output rates at the NICs may be close to, but not conform to ACRs. Under such conditions, the policing function needs to be done at the edge switch. This switch does require per-VC queues (but a smaller number because it is an edge switch), large buffers, and it needs to monitor and enforce the ACRs of VCs. The non-edge switches can provide simple FIFO queueing, and relatively smaller buffers, simple drop policies, and tradeoff the complexity of the switch feedback scheme.

4. Large legacy switches have a problem in that they are very expensive to replace, but provide few hooks for adding new functionality. Counters, registers and pin-outs are always at a premium on a chip, and are rarely left unused. For example, one might decide to use a switch hook (a pin in the chip) to implement an improved switch scheme. However, if the scheme requires measurement of different quantities, it cannot be done due to the lack of hooks. For example, the ERICA switch scheme requires the measurement of quantities such as the input rate and the number of active ABR sources. But, the only available metric in the switch might be just the queue length, which does not allow the implementation of the algorithm to be retrofitted on the switch. Simple algorithms which use just the queue metric need to be used for such cases.

5. Some ABR features such as the Virtual Source/Virtual Destination feature require the implementation of per-VC queueing. Recall that the requirement of per-VC queueing was one of the key reasons why the credit-based framework was rejected. It might seem contradictory to see per-VC queueing implemented by all major vendors. However, note that the current switches typically support upto 128K VCs per port. When the number of VCs grows further (millions of VCs), the accounting information required and the scheduling overhead is expected to become prohibitively expensive. In such cases, VCs will be aggregated into classes and supported by a few thousand class queues.

   VS/VD, on the other hand, requires the implementation of the source end system rules, scheduling of VC cells at a variable ACR, and the maintainence of a large amount of state per-VC. This has resulted in the VS/VD option to be implemented only in very large switches (like satellite switches) where the

advantages of the mechanism justify the cost of implementation. The implementation issues of VS/VD are further discussed in section 9.

6. Another issue of importance to long-distance ATM service providers is how to price the ABR service for ISPs. This requires switch support in terms of management software for usage-based billing. The definition of "usage" is "the number of cells delivered to the destination end-system." This definition implies that the measurements of usage have to be made in the egress switches. This adds cost and complexity to the implementation of edge switches. Currently, no cost estimates exist for ABR service.

7. There is a cost-performance tradeoff in implementing the various options of the ABR service. LAN switches are typically lower end, and the EFCI feedback mechanism provides sufficient performance, since the round trip times are small. It is anticipated that ABR will be the service of choice for WAN and satellite networks. This is because the ABR service (ER-based implementations) can provide throughput and delay performance, and is more scalable in terms of buffer requirements than the UBR service. WAN switches are expected to use ER-based ABR implementations. Complex alternatives like per-VC queueing and scheduling are required for ABR only at the edge switches. Interior network switches typically would use ER-based feedback, simple FIFO queueing, allocate small amount of buffers, and have simple buffer management and cell drop policies. The VS/VD alternative is required at a few edge switches of a very large delay-bandwidth product network. This option allows network managers to isolate the effects of the large network on downstream small networks (see

section 9. Further, WAN switches would typically allocate some bandwidth for aggregate ABR traffic to avoid zero capacity problems during congestion (when the feedback loops are disrupted), and in general to reduce the variation in available capacity for the service (which allows the switch algorithm to allocate rates more aggressively without worrying about effects of errors due to variation).

10.2.2 End-system issues

1. The end-system (which is the network interface card (NIC) for end-to-end ATM, or an edge router in a backbone network) needs to implement the end-system rules for ABR as specified in the standard. A mechanism needed for this implementation is one which schedules cells of different VCs based upon a dynamically changing set of ACRs. In practice, only a few ACR levels may be possible which can lead to link underutilization. SES Rule 9 allows the source to reschedule a cell on a VC based upon a new rate allocation. The implementation of this mechanism per-VC can be difficult.

2. In the ABR service, the network allocated rate may not match the input rate of a VC at a NIC. Mechanisms to control the actual sources of traffic are necessary to avoid cell loss at the NIC. Further, the NIC needs to have a large number of buffers and typically needs to manage per-VC queues. Current implementations are possible since the number of VCs per NIC is small. When ATM is deployed in the backbone alone, then the cost increases in the edge routers/switches.

3. Typically, the number of end systems is one order of magnitude greater than the number of switches. This has a multiplier effect on the cost of an ATM network supporting ABR. The service can be made attractive only if:

   a) cheap ABR end-system implementations are available;

   b) there are mechanisms which carry the benifits of ABR (cell loss control, thoughput, controlled delay, fairness) to the applications (an application would not choose ABR if its performance degrades due to cell loss at the ABR end-system;

   c) an application programming interface (API) is available for end-to-end ATM implementations which maps applications to ABR;

   d) a larger class of applications (like variable quality voice, audio, video) can be scalably supported using the ABR service (and allow higher/costlier classes of service for applications willing to pay the price for the higher quality of service).

CHAPTER 11

SUMMARY AND FUTURE WORK

11.1 Summary of Contributions

This dissertation has examined the design of several traffic management mechanisms and methodologies for the ABR service in ATM networks. The ideas presented in this dissertation have significantly impacted the shape of the ATM Traffic Management 4.0 standard. This section summarizes the contributions of this work.

In **Chapter 1** we gave a specification of the control problem in ABR traffic management. We presented an open-loop equation in this chapter and presented the requirements for the closed loop solution in **Chapter 3**. The goals we seek to address include: efficiency (high throughput and low delay), fairness, steady state and transient performance, buffer requirements, robustness, implementation complexity and scalability. **Chapter 2** gives a tutorial introduction to the source, destination and switch rules as defined by the ATM Traffic Management 4.0 standard.

A large body of related work (ABR switch schemes) are surveyed in **Chapter 4. Chapters 5 and 6** describe the OSU, ERICA and ERICA+ schemes and related performance analyses.

The OSU scheme was one of the first explicit rate schemes designed for ABR. It exposed some of the pitfalls in using window-based control techniques in rate-based control. The key contributions of the algorithm are:

- Choice of congestion indicator (input rate i.e. aggregate demand, instead of queue length)
- Application of the congestion avoidance concept in rate-based control (use of the target utilization parameter).
- Use of the "overload factor" and "equal fairshare" metrics instead of simply the queue length.
- Small number of parameters
- Measurement of the number of active sources. In general, the scheme uses measurement instead of beleiving the declared values of metrics.
- O(1) time complexity.
- A proof that the fairness algorithm does achieve fairness.

The drawbacks of the scheme are its slow convergence in complex configurations, and the fact that it is incompatible with the final version of the standard (since it was developed at a time when the standards themselves were not finalized).

Three different options that further improve the performance over the basic scheme are also described. These allow the fairness to be achieved quickly, oscillations to be minimized, and feedback delay to be reduced.

The OSU scheme drawbacks were addressed in the ERICA schemes. The ERICA set of schemes use an optimistic approach to provide good steady state as well as

good transient performance. Since real networks are in a transient state most of the time (sources are rarely infinite and ABR capacity varies), the transient performance of a scheme deployed under these conditions is of importance.

The difference in approach between the OSU scheme and ERICA is that while the former attempts to achieve efficiency and fairness one after another, the latter attempts to move towards efficiency and fairness at the same time. It uses an aggressive core algorithm - the maximum of an "efficiency term" (based on the overload factor and the source's current rate) and a "fairness term" (based on the available capacity and the number of active sources). The ERICA schemes still rely on measurement of load, capacity and the number of active sources to calculate the rates. The ERICA+ scheme attempts to achieve an operating point of 100% utilization and a target queueing delay. The use of the queueing delay metric allows the scheme to be robust to errors in measurement and feedback delays (which manifest as queues at the switch). Simulation results with different configurations and traffic patterns have also been presented.

Chapter 7 examines the design of source rules in the ATM Traffic Management framework, i.e., how "open-loop"control complements the "closed-loop" feedback system. This dissertation work has helped develop a number of the rules in the international standard. However, two of these issues are investigated in depth in this chapter.

The first issue is the design of "Use-it-or-Lose-it" policies. These policies take away a source's assigned rate if the source does not use it. The choice of the policy has a significant impact on ABR service capabilities, affecting the performance of bursty (on-off) sources and sources bottlenecked below their network-assigned rate. We

present a survey of the proposed approaches including our approach. The approaches can broadly be classified as source-based approaches (where the source end-system implements the policy) and switch-based policies (where the switch may implement proprietary measures to address the problem). After a long debate, the ATM Forum decided not to standardize an elaborate source-based UILI policy. A simple timeout is mandated for the source, where sources keep their rate allocations until a timeout (parameter ATDF, of the order of 500 ms) expires. We present a detailed study of the various alternatives in this chapter.

The second issue is the efficient support of low-rate sources. We study three mechanisms - tuning the Trm parameter setting, the TCR parameter which controls the rate of out-of-rate RM cells, and a source rescheduling policy which may trigger when the source receives a rate increase indication. The tradeoffs in these mechanisms are examined in this chapter.

Chapter 8 deals with issues in supporting internet applications like file transfer and world wide web (which run over the TCP/IP protocol) over ATM ABR, with different models of higher priority VBR background traffic in the background. We show that a well-designed ABR system can scalably support persistant TCP applications like ftp as well as bursty TCP applications like WWW clients and servers. We study the TCP dynamics and show that when ftp applications using TCP run over ABR, the switch algorithm can control the TCP sources given sufficient amount of buffering. Once the control has been established, given no changes in traffic behavior, TCP can achieve maximum throughput and zero cell loss. The buffer requirements do not depend upon the number of TCP sources - only on parameters like the switch algorithm parameters and round trip time. We verify that this requirement holds despite highly

variant background traffic conditions, and for LAN, WAN and satellite configurations. To introduce highly variant conditions, we use conventional ON-OFF models and also propose new models of MPEG-2 transport streams (resulting in long-range dependent traffic) multiplexed over VBR. We observe that the ABR control system only pushes the queues to the edge of the network - where an edge router has to again handle the issue of large TCP queues.

We note that the system can theoritically be loaded with unbounded queues when bursty traffic like WWW is used. However, under practical conditions, the average load when a large number of WWW exist increases more smoothly than expected. Since the ABR switch scheme reacts to load and can tolerate variation in load and capacity, we see that queues are controlled and high throughput is attained even under such conditions.

In **Chapter 9** we look at the switch design issues for a specific ABR framework option called the "Virtual Source/Virtual Destination" option. In this option, the switch splits the network into two segments and shortens the feedback loop for both segments. We show that this option has the potential to increase the performance of the network, but the implementation can be complex and has to be carefully done. In our study of multiple implementation options of this feature, we found that only a very few performed well, and we identify the properties of the best option. This chapter is followed up by **Chapter 10** where we briefly address certain implementation issues.

11.2 Future Work

At the time of this writing, the ABR service is being actively implemented and is currently facing interesting cost-performance tradeoff questions. Field trials and

interoperability tests are required to ensure that the implementations conform to the specifications and deliver the promised performance. Another step to make the service more attractive in the cost-performance tradeoff is to demonstrate that a large class of applications can be made to run over the service. Currently, ABR is promising for two key internet applications: file transfer and the world wide web. It is interesting to see if ABR can support variable quality voice and video. ABR multicast is also another pre-requisite for supporting a wider class of applications

Another issue with ATM backbone scenarios is that ABR provides control only upto the edge of the ATM network. It is possible that the edge router can use the ABR rate feedback information to pace TCP traffic. This will carry the benifits of ABR to applications (i.e., end-to-end).

The proliferation of high-speed networking will increase the demand for high quality of service (QoS) on access network technologies like wireless and DSL (ADSL, 56 kbps modems etc). Such technologies are characterized by low bit rates and high error rates. The implication of the mapping of ATM on such technologies is that traffic management has to deal with the effect of uniform errors (due to the underlying technology) as well as burst errors (due to congestion). Another scenario is that of satellite networks where the delay is large, the bit rates are smaller, and the satellite technology imposes rigid design constraints. Special schemes are required to handle such scenarios correctly.

APPENDIX A

SOURCE, DESTINATION AND SWITCH RULES

This appendix provides the precise source and destination behavior verbatim from the ATM Forum's Traffic Management 4.0 specification [32]. All table, section, and other references in this appendix refer to those in the TM specification.

5.10.4 Source Behavior

The following items define the source behavior for CLP=0 and CLP=1 cell streams of a connection. By convention, the CLP=0 stream is referred to as in-rate, and the CLP=1 stream is referred to as out-of-rate. Data cells shall not be sent with CLP=1.

1. The value of ACR shall never exceed PCR, nor shall it ever be less than MCR. The source shall never send in-rate cells at a rate exceeding ACR. The source may always send in-rate cells at a rate less than or equal to ACR.

2. Before a source sends the first cell after connection setup, it shall set ACR to at most ICR. The first in-rate cell shall be a forward RM-cell.

3. After the first in-rate forward RM-cell, in-rate cells shall be sent in the following order:

   a) The next in-rate cell shall be a forward RM cell if and only if, since the last in-rate forward RM-cell was sent, either:

i) at least Mrm in-rate cells have been sent and at least Trm time has elapsed, or

ii) Nrm -1 in-rate cells have been sent.

b) The next in-rate cell shall be a backward RM-cell if condition (a) above is not met, if a backward RM cell is waiting for transmission, and if either:

i) no in-rate backward RM-cell has been sent since the last in-rate forward RM-cell, or

ii) no data cell is waiting for transmission.

c) The next in-rate cell sent shall be a data cell if neither condition (a) nor condition (b) is met, and if a data cell is waiting for transmission.

4. Cells sent in accordance with source behaviors #1,#2, and #3 shall have CLP=0.

5. Before sending a forward in-rate RM cell, if ACR > ICR and the time T that has elapsed since the last in-rate forward RM-cell was sent is greater than ADTF, then ACR shall be reduced to ICR.

6. Before sending an in-rate forward RM cell, and following behavior #5 above, if at least CRM in-rate forward RM-cells have been sent since the last backward RM-cell with BN=0 was received, then ACR shall be reduced by at least ACR $\times$ CDF, unless that reduction would result in a rate below MCR, in which case ACR shall be set to MCR.

7. After following behaviors #5 and #6 above, the ACR value shall be placed in the CCR field of the outgoing forward RM-cell, but only in-rate cells sent after the outgoing forward RM-cell need to follow the new rate.

8. When a backward RM-cell (in-rate or out-of-rate) is received with CI=1, then ACR shall be reduced by at least ACR $\times$ RDF, unless that reduction would result in a rate below MCR, in which case ACR shall be set to MCR. If the backward RM-cell has both CI=0 and NI=0, then the ACR may be increased by no more than RIF $\times$ PCR, to a rate not greater than PCR. If the backward RM-cell has NI=1, the ACR shall not be increased.

9. When a backward RM-cell (in-rate or out-of-rate) is received, and after ACR is adjusted according to source behavior #8, ACR is set to at most the minimum of ACR as computed in source behavior #8, and the ER field, but no lower than MCR.

10. When generating a forward RM-cell, the source shall assign values to the various RM-cell fields as specified for source-generated cells in Table 5-4.

11. Forward RM-cells may be sent out-of-rate (i.e., not conforming to the current ACR). Out-of-rate forward RM-cells shall not be sent at a rate greater than TCR.

12. A source shall reset EFCI on every data cell it sends.

13. The source may implement a use-it-or-lose-it policy to reduce its ACR to a value which approximated the actual cell transmission rate. Use-it-or-lose-it policies are discussed in Appendix I.8.

Notes:

1. In-rate forward and backward RM-cells are included in the source rate allocated to a connection.

2. The source is responsible for handling congestion within its scheduler in a fair manner. This congestion occurs when the sum of the rates to be scheduled exceeds the output rate of the scheduler. The method for handling local congestion is implementation specific.

5.10.5 Destination Behavior

The following items define the destination behavior for CLP=0 and CLP=1 cell streams of a connection. By convention, the CLP=0 stream is referred to as in-rate, and the CLP=1 stream is referred to as out-of-rate.

1. When a data cell is received, its EFCI indicator is saved as the EFCI state of the connection.

2. On receiving a forward RM-cell, the destination shall turn around the cell to return to the source. The DIR bit in the RM-cell shall be changed from "forward" to "backward," BN shall be set to zero, and CCR, MCR, ER, CI, and NI fields in the RM-cell shall be unchanged except:

   a) If the saved EFCI state is set, then the destination shall set CI=1 in the RM cell, and the saved EFCI state shall be reset. It is preferred that this step is performed as close to the transmission time as possible;

   b) The destination (having internal congestion) may reduce ER to whatever rate it can support and/or set CI=1 or NI=1. A destination shall either

set the QL and SN fields to zero, preserve these fields, or set them in accordance with ITU-T Recommendation I.371-draft. The octets defined in Table 5-4 as reserved may be set to 6A (hexadecimal) or left unchanged. The bits defined as reserved in Table 5-4 for octet 7 may be set to zero or left unchanged. The remaining fields shall be set in accordance with Section 5.10.3.1 (Note that this does not preclude looping fields back from the received RM cell).

3. If a forward RM-cell is received by the destination while another turned-around RM-cell (on the same connection) is scheduled for in-rate transmission:

   a) It is recommended that the contents of the old cell are overwritten by the contents of the new cell;

   b) It is recommended that the old cell (after possibly having been overwritten) shall be sent out-of-rate; alternatively the old cell may be discarded or remain scheduled for in-rate transmission;

   c) It is required that the new cell be scheduled for in-rate transmission.

4. Regardless of the alternatives chosen in destination behavior #3, the contents of the older cell shall not be transmitted after the contents of a newer cell have been transmitted.

5. A destination may generate a backward RM-cell without having received a forward RM-cell. The rate of the backward RM-cells (including both in-rate and out-of-rate) shall be limited to 10 cells/second, per connection. When a destination generated an RM-cell, it shall set either CI=1 or NI=1, shall set set

BN=1, and shall set the direction to backward. The destination shall assign values to the various RM-cell fields as specified for destination generated cells in Table 5-4.

6. When a forward RM-cell with CLP=1 is turned around it may be sent in-rate (with CLP=0) or out-of-rate (with CLP=1)

Notes-

1. "Turn around" designates a destination process of transmitting a backward RM-cell in response to having received a forward RM-cell.

2. It is recommended to turn around as many RM-cells as possible to minimize turn-around delay, first by using in-rate opportunities and then by using out-of-rate opportunities as available. Issues regarding turning RM-cells around are discussed in Appendix I.7.

5.10.6 Switch Behavior

The following items define the switch behavior for CLP=0 and CLP=1 cell streams of a connection. By convention, the CLP=0 stream is referred to as in-rate, and the CLP=1 stream is referred to as out-of-rate. Data cells shall not be sent with CLP=1.

1. A switch shall implement at least one of the following methods to control congestion at queueing points:

   a) *EFCI marking:* The switch may set the EFCI state in the data cell headers;

   b) *Relative Rate Marking:* The switch may set CI=1 or NI=1 in forward and/or backward RM-cells; item[c)] *Explicit Rate Marking:* The switch may reduce the ER field of forward and/or backward RM-cells (Explicit Rate Marking);

d) *VS/VD Control:* The switch may segment the ABR control loop using a virtual source and destination.

2. A switch may generate a backward RM-cell. The rate of these backward RM-cells (including both in-rate and out-of-rate) shall be limited to 10 cells/second, per connections. When a switch generates an RM-cell it shall set either CI=1 or NI=1, shall set BN=1, and shall set the direction to backward. The switch shall assign values to the various RM-cell fields as specified for switch-generated cells in Table 5-4.

3. RM-cells may be transmitted out of sequence with respect to data cells. Sequence integrity within the RM-cell stream must be maintained.

4. For RM-cells that transit a switch (i.e., are received and then forwarded), the values of the various fields before the CRC-10 shall be unchanged except:

   a) CI,NI and ER may be modified as noted in #1 above

   a) RA, QL and SN shall be set in accordance with ITU-T Recommendation I.371-draft

   MCR may be corrected to the connection's MCR if the incoming MCR value is incorrect.

5. The switch may implement a use-it-or-lose it policy to reduce an ACR to a value which approximates the actual cell transmission rate from the source. Use-it-or-lose-it policies are discussed in Appendix I.8.

Notes-

1. A switch queueing point is a point of resource contention where cells may be potentially delayed or lost. A switch may contain multiple queueing points.

2. Some example switch mechanisms are presented in Appendix I.5.

3. The implications of combinations of the above methods is beyond the scope of this specification.

5.10.7 Virtual Source and Virtual Destination Behavior

VS/VD behavior divides an ABR connection into two or more separately controlled ABR segments. The coupling between adjacent ABR control segments associated with an ABR connection is implementation specific.

The following applies to VS/VD behavior:

1. Each ABR control segment, except the first, is sources by a virtual source. A virtual source assumes the behavior of an ABR source end point. Backward RM-cells received by a virtual source are removed from the connection.

2. Each ABR control segment, except the last, is terminated by a virtual destination. A virtual destination assumes the behavior of an ABR destination end point. Forward RM-cells received by a virtual destination shall be turned around as defined in destination behavior #2, and shall not be forwarded to the next segment of the connection.

3. The coupling between two adjacent ABR control segments associated with an ABR connection is implementation specific.

4. MCR shall be conveyed across VS/VD boundaries.

5. Setting of other parameters at VS/VD is network specific

APPENDIX B

THE OSU SCHEME: PSEUDO CODE

B.1 The Source Algorithm

There are four events that can happen at the source adapter or Network Interface Card (NIC). These events and the action to be taken on these events are described below.

1. Initialization:

   TCR ←Initial Cell Rate;

   Averaging_Interval ←Some initial value;

   IF (BECN_Option) THEN Time_Already_Acted ←0;

2. A data cell or cell burst is received from the host.

   Enqueue the cell(s) in the output queue.

3. The inter-cell transmission timer expires.

   IF Output_Queue NOT Empty THEN dequeue the first cell and transmit;

   Increment Transmitted_Cell_Count;

   Restart Inter_Cell_Transmission_Timer;

4. The averaging interval timer expires.

   Offered_Cell_Rate ←Transmitted_Cell_Count/Averaging_Interval;

   Transmitted_Cell_Count ←0;

   Create a control cell;

   OCR_In_Cell ←Offered_Cell_Rate ;

   TCR_In_Cell ←max{TCR, OCR} ;

   Load_Adjustment_Factor ←0;

   IF (BECN_Option) THEN Time_Stamp_in_Cell ←Current Time;

   Transmit the control cell;

   Restart Averaging_Interval_Timer;

5. A control cell returned from the destination is received.

   IF ((BECN_Option AND Time_Already_Acted < Time_Stamp_In_Cell) OR
   (NOT BECN_Option))
   THEN BEGIN
   New_TCR ←TCR_In_Cell/Load_Adjustment_Factor_In_Cell;
   IF Load_Adjustment_Factor_In_Cell ≥ 1
   THEN IF New_TCR < TCR
   THEN BEGIN
   TCR ←New_TCR ;
   IF(BECN_Option)
   THEN Time_Already_Acted ←Time_Stamp_In_Cell;
   END
   ELSE IF Load_Adjustment_Factor_In_Cell < 1

THEN IF New_TCR > TCR THEN TCR ←New_TCR ;

Inter_Cell_Transmission_Time ←1/TCR;

END; (* of FECN Cell processing *)

Averaging_Interval ←Averaging_Interval_In_Cell;

6. A BECN control cell is received from some switch.

IF BECN_Option

THEN IF Time_Already_Acted <

Time_Stamp_In_Cell

THEN IF Load_Adjustment_Factor_In_Cell ≥ 1

THEN BEGIN

New_TCR ←

TCR_In_Cell/Load_Adjustment_Factor_In_Cell;

IF New_TCR < TCR

THEN BEGIN

TCR ←New_TCR;

Inter_Cell_Transmission_Time ←1/TCR;

Time_Already_Acted ←Time_Stamp_In_Cell;

END;

END;

B.2 The Switch Algorithm

The events at the switch and the actions to be taken on these events are as follows:

1. Initialization:

   Target_Cell_Rate ←Link_Bandwidth × Target_Utilization / Cell_Size ;

   Target_Cell_Count ←Target_Cell_Rate×Averaging_Interval;

   Received_Cell_Count ←0;

   Clear VC_Seen_Bit for all VCs;

   IF (Basic_Fairness_Option OR Aggressive_Fairness_Option)

   THEN BEGIN

      Upper_Load_Bound ←1 + Half_Width_Of_TUB;

      Lower_Load_Bound ←1 - Half_Width_Of_TUB;

   END;

2. A data cell is received.

      Increment Received_Cell_Count;

      Mark VC_Seen_Bit for the VC in the Cell;

3. The averaging interval timer expires.

      Num_Active_VCs ←max{$\sum$ VC_Seen_Bit, 1};

      Fair_Share_Rate ←Target_Cell_Rate/Num_Active_VCs;

      Load_Level ←Received_Cell_Count/Target_Cell_Count;

      Reset all VC_Seen_Bits;

      Received_Cell_Count ←0;

      Restart Averaging_Interval_Timer;

4. A control cell is received.

IF (Basic_Fairness_Option)

THEN IF (Load_Level $\geq$ Lower_Load_Bound)

and (Load_Level $\leq$ Upper_Load_Bound)

THEN BEGIN

IF OCR_In_CELL > Fair_Share_Rate

THEN Load_Adjustment_Decision $\leftarrow$Load_Level/Lower_Load_Bound

ELSE Load_Adjustment_Decision $\leftarrow$Load_Level/Upper_Load_Bound

END (*IF *)

ELSE Load_Adjustment_Decision $\leftarrow$Load_Level;

IF (Aggressive_Fairness_Option)

THEN BEGIN

Load_Adjustment_Decision $\leftarrow$1;

IF (Load_Level < Lower_Load_Bound)

THEN IF ((OCR_In_Cell < Fair_Share_Rate$\times$Load_Level) OR

(Num_VC_Active =1))

THEN Load_Adjustment_Decision $\leftarrow$Load_Level

ELSE IF (OCR_In_Cell < Target_Cell_Rate$\times$Load_Level)

THEN Load_Adjustment_Decision $\leftarrow$Load_Level + (1-

Load_Level)$\times$(OCR_In_Cell/(Load_level$\times$

Fair_Share)-1)/(Num_VC_Active-1)

ELSE Load_Adjustment_Decision $\leftarrow$1

ELSE IF Load_Level $\geq$ Upper_Load_Bound

```
                    THEN IF (OCR_In_Cell ≤ Fair_Share_Rate AND
                        Num_Active_VCs ≠ 1)
                        THEN Load_Adjustment_Decision ←1
                        ELSE IF (OCR_In_Cell <
                          Fair_Share_Rate × Load_Level)
                            THEN Load_Adjustment_Decision ←max{1,
                                OCR_In_Cell/Fair_Share_Rate}
                            ELSE IF (OCR_In_Cell ≤ Target_Cell_Rate)
                                THEN Load_Adjustment_Decision ←
                                  Load_Level
                                ELSE Load_Adjustment_Decision ←
                                    OCR_In_Cell ×
                                      Load_Level/Target_Cell_Rate;
    END (* of Aggressive Fairness Option *)

IF (Precise_Fairshare_Computation_Option)
BEGIN
    OCR_Of_VC_In_Table ←OCR_In_Cell;
    Fair_Share_Rate ←Target_Cell_Rate/Num_VC_Active;
    REPEAT
        Num_VC_Underloading ←0 ;
        Sum_OCR_Underloading ←0 ;
        FOR each VC seen in the last interval DO
        IF (OCR_In_Cell < Fair_Share_Rate)
```

```
            THEN BEGIN
                Increment Num_VC_Underloading ;
                Sum_OCR_Underloading ←
                 Sum_OCR_Underloading + OCR_Of_VC
            END (* IF *)
            Fair_Share_Rate ←(Target_Cell_Rate - SUM_OCR_Underloading)
                /max{1, (Num_VC_Active - Num_VC_Underloading )}
        UNTIL Fair_Share_Rate does not change (* Maximum of 2 iterations *);
Load_Adjustment_Decision ←OCR_In_Cell/Fair_Share_Rate;
END; (* Precise Fairness Computation Option *)

IF (Load_Adjustment_Decision > Load_Adjustment_Factor_In_Cell)
THEN BEGIN
      Load_Adjustment_Factor_In_Cell ←Load_Adjustment_Decision;
     IF BECN_Option and Load_Adjustment_Decision > 1
      THEN SEND_A_COPY_OF_CONTROL_CELL_BACK_TO_SOURCE ;
END (* IF *)
```

APPENDIX C

ERICA SWITCH ALGORITHM: DETAILED DESCRIPTION

C.1 Variables and Flow charts

Notes:

- All rates are in the units of cells/s
- The following pseudo-code assumes a simple fixed-time averaging interval. Extension to a cells and time averaging interval is trivial.

We use a combination of flowcharts and pseudo-code to describe the ERICA algorithm. The following names are used to identify the flow charts:

Flow Chart 1: Flow Chart of the Basic ERICA Algorithm. Figure C.1.

Flow Chart 2: Flow Chart for Achieving Max-Min Fairness. Figure C.2.

Flow Chart 3: Flow Chart for Bi-Directional Counting. Figure C.3.

Flow Chart 4: Flow Chart of averaging number of active sources (part 1 of 2). Figure C.4.

Flow Chart 5: Flow Chart of averaging number of active sources (part 2 of 2). Figure C.5.

Flow Chart 6: Flow Chart of averaging load factor (method 1). Figure C.6.

Flow Chart 7: Flow Chart of averaging load factor (method 2). Figure C.7.

C.2 Pseudocode

Initialization:

(* ABR Capacity and Target Utilization *)

IF (Queue_Control_Option) THEN

Target_Utilization ←1

END (* IF *)

ABR_Capacity_In_cps ←Target_Utilization × Link_Bandwidth −

VBR_and_CBR_Capacity

(* Count of Number of VCs, Cells *)

FOR ALL VCs DO

Contribution[VC] ←0

Seen_VC_In_This_Interval[VC] ←0

Seen_BRM_Cell_In_This_Interval[VC] ←0

END (* FOR *)

ABR_Cell_Count ←ABR_Capacity_In_cps × Averaging_Interval

Number_Active_VCs_In_This_Interval ←Total Number of Setup VCs

```
Number_Active_VCs_In_Last_Interval ←Number_Active_VCs_In_This_Interval

(* Fairshare and Load Factor variables *)
Fair_Share ←ABR_Capacity_In_cps / Number_Active_VCs_In_Last_Interval
Max_Alloc_Previous ←0
Max_Alloc_Current ←Fair_Share
Load_Factor ←ABR_Capacity_In_cps/ FairShare

(* Per VC CCR Option Variables *)
IF (Per_VC_CCR_Option) THEN
   FOR ALL VCs DO
      Number_Of_Cells[VC] ←0
   END (* FOR *)
END (* IF *)
```

A cell of "VC" is received in the forward direction:

```
IF (Averaging_VCs_Option) THEN
   IF (Contribution[VC] < 1) THEN (* VC inactive in current interval *)
      Number_Active_VCs_In_This_Interval ←
            Number_Active_VCs_In_This_Interval − Contribution[VC] + 1
      IF ((Immediate_Fairshare_Update_Option) AND
      (Contribution[VC] < Decay_Factor)) THEN
         Number_Active_VCs_In_Last_Interval ←Number_Active_VCs_In_Last_Interval
```

```
        – (Contribution[VC] / Decay_Factor) + 1
      Fair_Share ←ABR_Capacity_In_cps / Number_Active_VCs_In_Last_Interval
    END (* IF *)
    Contribution[VC] ←1
  END (* IF *)
ELSE
  IF (NOT(Seen_VC_In_This_Interval[VC])) THEN
    Seen_VC_In_This_Interval[VC] ←1
  END (* IF *)
  IF ((Immediate_Fair_Share_Option) AND (NOT(Seen_VC_In_Last_Interval[VC])))
THEN
    Number_Active_VCs_In_Last_Interval ←Number_Active_VCs_In_Last_Interval +
1
    Fair_Share ←ABR_Capacity_In_cps / Number_Active_VCs_In_Last_Interval
    Seen_VC_In_Last_Interval[VC] ←1
  END (* IF *)
END (* IF *)
ABR_Cell_Count ←ABR_Cell_Count + 1
IF (Per_VC_CCR_Option) THEN
  Number_Of_Cells[VC] ←Number_Of_Cells[VC] + 1
END (* IF *)
```

Averaging interval timer expires:

```
IF (NOT(Averaging_VCs_Option)) THEN
   Number_Active_VCs_In_Last_Interval ←
   Max (∑ Seen_VC_In_This_Interval, 1)
   Number_Active_VCs_In_This_Interval ←0
   FOR ALL VCs DO
      Seen_VC_In_Last_Interval[VC] ←Seen_VC_In_This_Interval[VC]
   END (* FOR *)
ELSE
   Number_Active_VCs_In_Last_Interval ←
    Max(Number_Active_VCs_In_This_Interval , 1)
   Number_Active_VCs_In_This_Interval ←0
   FOR ALL VCs DO
      Contribution[VC] ←Contribution[VC] × Decay_Factor
      Number_Active_VCs_In_This_Interval ←Number_Active_VCs_In_This_Interval +
Contribution[VC]
   END (* FOR *)
END (* IF *)

IF (Exponential_Averaging_Of_Load_Method_2_Option) THEN
   ABR_Capacity_In_Cells ←
   Max(Target_Utilization × Link_Bandwidth × Averaging_Interval)
      − VBR_and_CBR_Cell_Count, 0)
   Avg_ABR_Capacity_In_Cells ←
```

$(1-\alpha) \times$ Avg_ABR_Capacity_In_Cells +

$\alpha \times$ ABR_Capacity_In_Cells

Avg_Averaging_Interval $\leftarrow$

$(1-\alpha) \times$ Avg_Averaging_Interval + $\alpha \times$Averaging_Interval

Avg_ABR_Cell_Count $\leftarrow (1-\alpha) \times$Avg_ABR_Cell_Count + $\alpha \times$ABR_Cell_Count

ABR_Input_Rate $\leftarrow$Avg_ABR_Cell_Count / Avg_Averaging_Interval

ABR_Capacity_In_cps $\leftarrow$Avg_ABR_Capacity_In_Cells / Avg_Averaging_Interval

ELSE

VBR_and_CBR_Cell_Rate $\leftarrow$VBR_and_CBR_Cell_Count / Averaging_Interval

ABR_Capacity_In_cps $\leftarrow$

Max(Target_Utilization$\times$Link_Bandwidth $-$ VBR_and_CBR_Cell_Rate, 0)

ABR_Input_Rate $\leftarrow$ABR_Cell_Count / Averaging_Interval

END (* IF *)

IF (Queue_Control_Option) THEN

Target_Queue_Length $\leftarrow$Target_Time_To_Empty_Queue $\times$ ABR_Capacity_In_cps

Queue_Control_Factor $\leftarrow$Fn(Current_Queue_Length)

ABR_Capacity_In_cps $\leftarrow$Queue_Control_Factor $\times$ ABR_Capacity_In_cps

END (* IF *)

IF (Exponential_Averaging_Of_Load_Method_1_Option) THEN

IF (ABR_Capacity_In_cps ≤ 0) THEN

Load_Factor $\leftarrow$Infinity

ELSE

```
      IF (Load_Factor = Infinity) THEN
        Load_Factor ←ABR_Input_Rate / ABR_Capacity_In_cps
      ELSE
        Load_Factor ←(1−α) × Load_Factor +
          α × ABR_Input_Rate / ABR_Capacity_In_cps
      END (* IF *)
    END (* IF *)
ELSE IF (Exponential_Averaging_Of_Load_Method_2_Option) THEN
    IF (ABR_Capacity_In_cps ≤ 0) THEN
      Load_Factor ←Infinity
    ELSE
      Load_Factor ←ABR_Input_Rate / ABR_Capacity_In_cps
    END (* IF *)
ELSE (* No exponential averaging *)
    IF (ABR_Capacity_In_cps ≤ 0) THEN
      Load_Factor ←Infinity
    ELSE
      Load_Factor ←ABR_Input_Rate / ABR_Capacity_In_cps
    END (* IF *)
END (* IF *)
Fair_Share ←ABR_Capacity_In_cps / Number_Active_VCs_In_Last_Interval
Max_Alloc_Previous ←Max_Alloc_Current
Max_Alloc_Current ←Fair_Share
FOR ALL VCs DO
```

```
    Seen_VC_In_This_Interval[VC] ←0
    Seen_BRM_Cell_In_This_Interval[VC] ←0
END (* FOR *)
ABR_Cell_Count ←0
IF (Per_VC_CCR_Option) THEN
    FOR ALL VCs DO
      CCR[VC] ←Number_Of_Cells[VC]/Averaging_Interval
      Number_Of_Cells[VC] ←0
    END (* FOR *)
END (* IF *)
VBR_and_CBR_Cell_Count ←0
Restart Averaging_Interval Timer
```

A Forward RM (FRM) cell of "VC" is received:

```
    IF (NOT(Per_VC_CCR_Option)) THEN
    CCR[VC] ←CCR_In_FRM_Cell
END (* IF *)
```

A Backward RM (BRM) cell of "VC" is received:

```
    IF (Averaging_VCs_Option) THEN
    IF (Contribution[VC] < 1) THEN (* VC inactive in current interval *)
      Number_Active_VCs_In_This_Interval ←
        Number_Active_VCs_In_This_Interval − Contribution[VC] + 1
      IF ((Immediate_Fairshare_Update_Option) AND
       (Contribution[VC] < Decay_Factor)) THEN
        Number_Active_VCs_In_Last_Interval ←Number_Active_VCs_In_Last_Interval
```

```
            – (Contribution[VC] / Decay_Factor) + 1
          Fair_Share ←ABR_Capacity_In_cps / Number_Active_VCs_In_Last_Interval
      END (* IF (Immediate ...) *)
      Contribution[VC] ←1
   END (* IF (Contribution ... ) *)
ELSE (* NOT (Averaging_VCs_Option) *)
   IF (NOT(Seen_VC_In_This_Interval[VC])) THEN
      Seen_VC_In_This_Interval[VC] ←1
   END (* IF *)
   IF ((Immediate_Fair_Share_Option) AND (NOT(Seen_VC_In_Last_Interval[VC])))
THEN
      Number_Active_VCs_In_Last_Interval ←Number_Active_VCs_In_Last_Interval +
1
      Fair_Share ←ABR_Capacity_In_cps / Number_Active_VCs_In_Last_Interval
      Seen_VC_In_Last_Interval[VC] ←1
   END (* IF ((Immediate ..)) *)
END (* IF-THEN-ELSE (Averaging_VCs_Option) *)

IF (Seen_BRM_Cell_In_This_Interval[VC]) THEN
   ER_Calculated ←Last_Allocated_ER[VC]
ELSE
   VC_Share[VC] ←CCR[VC] / Load_Factor
   (* Max-Min Fairness Algorithm *)
   IF (Load_Factor > 1 + δ) THEN
```

ER_Calculated ←Max (Fair_Share, VC_Share)

ELSE

ER_Calculated ←Max (Fair_Share, VC_Share, Max_Alloc_Previous)

END (* IF *)

Max_Alloc_Current ←Max (Max_Alloc_Current, ER_Calculated)

(* Avoid Unnecessary Transient Overloads *)

IF ((CCR[VC] < Fair_Share) AND (ER_Calculated ≥ Fair_Share)) THEN

ER_Calculated ←Fair_Share

(* Optionally Disable Feedback To This VC For An Averaging Interval *)

END (* IF *)

ER_Calculated ←Min(ER_Calculated, ABR_Capacity_In_cps)

(* Ensure One Feedback Per Switch Averaging Interval *)

Last_Allocated_ER[VC] ←ER_Calculated

Seen_BRM_Cell_In_This_Interval[VC] ←1

END (* IF *)

(* Give Feedback In BRM Cell *)

ER_In_BRM_Cell ←Min (ER_in_BRM_Cell, ER_Calculated)

At each cell slot time schedule cell from a service class using a scheduling policy

Name	Explanation	Flow Chart (FC) or Figure
ABR_Cell_Count	Number of ABR input cells in the current interval	FCs 1 and 7 (step 2)
Contribution[VC]	Contribution of the VC towards the count of the number of active sources	FCs 4 and 5
Seen_VC_In_This_Interval[VC]	A bit which is set when a VC is seen in the current (last) interval	FCs 1,3 and 5
Number_Of_Cells[VC]	Used in Per VC CCR option to count number of cells from each VC in the current interval	
Max_Alloc_Previous	Max rate allocation in previous interval	FC 2
Max_Alloc_Current	Max rate allocation in current interval	FC 2
Seen_BRM_Cell_In_This_Interval[VC]	A BRM from the source has been seen (and feedback given) in this interval. Do not give new feedback	Figure 6.2
Last_Allocated_ER	Unique ER feedback to the source in the current interval	Figure 6.2
Decay_Factor	Factor Used in Averaging the Number of Active Sources $0 < \text{Decay\_Factor} \leq 1$	FCs 4 and 5

Table C.1: Explanation of some of the ERICA Pseudocode variables

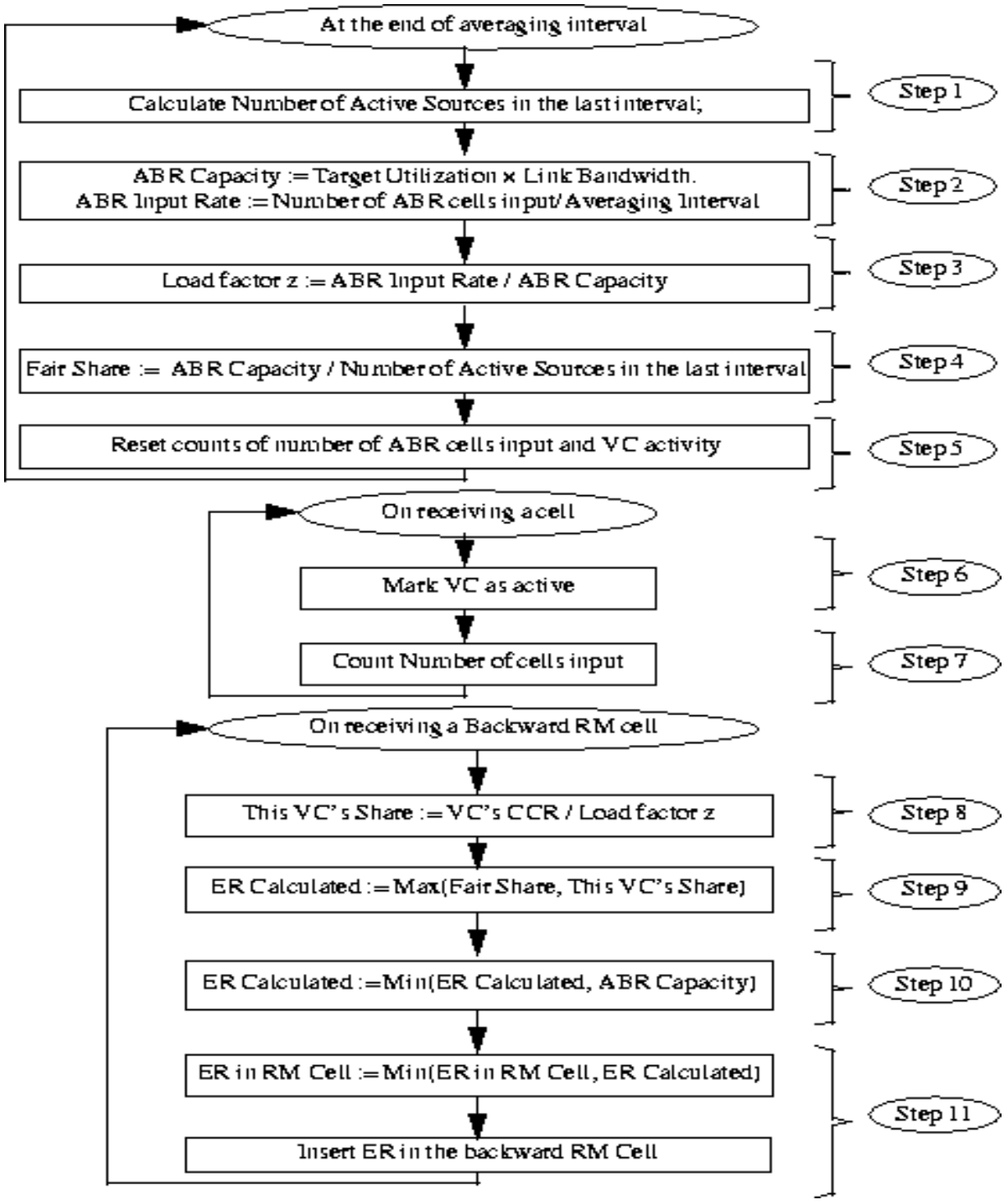


Figure C.1: Flow Chart of the Basic ERICA Algorithm

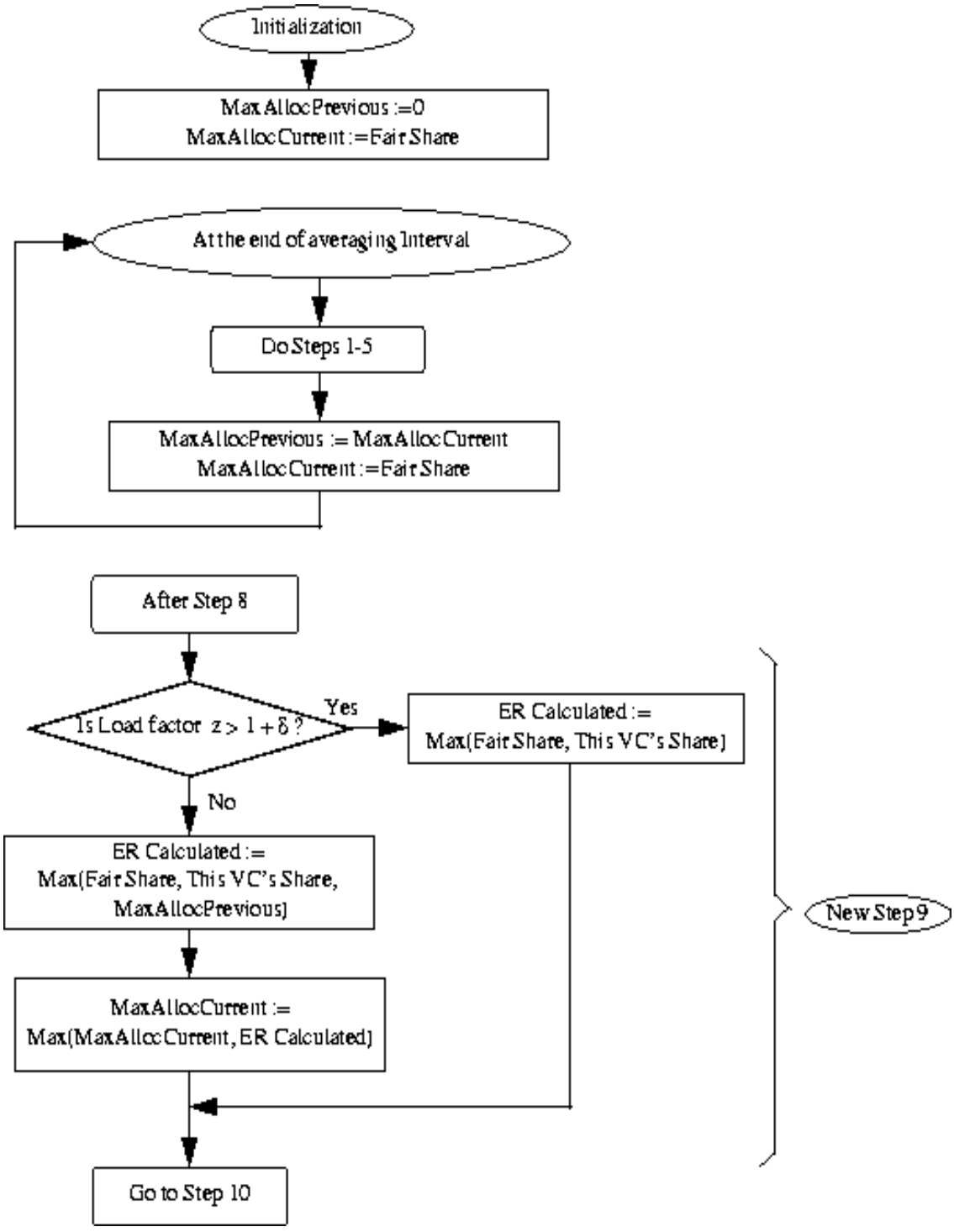


Figure C.2: Flow Chart for Achieving Max-Min Fairness

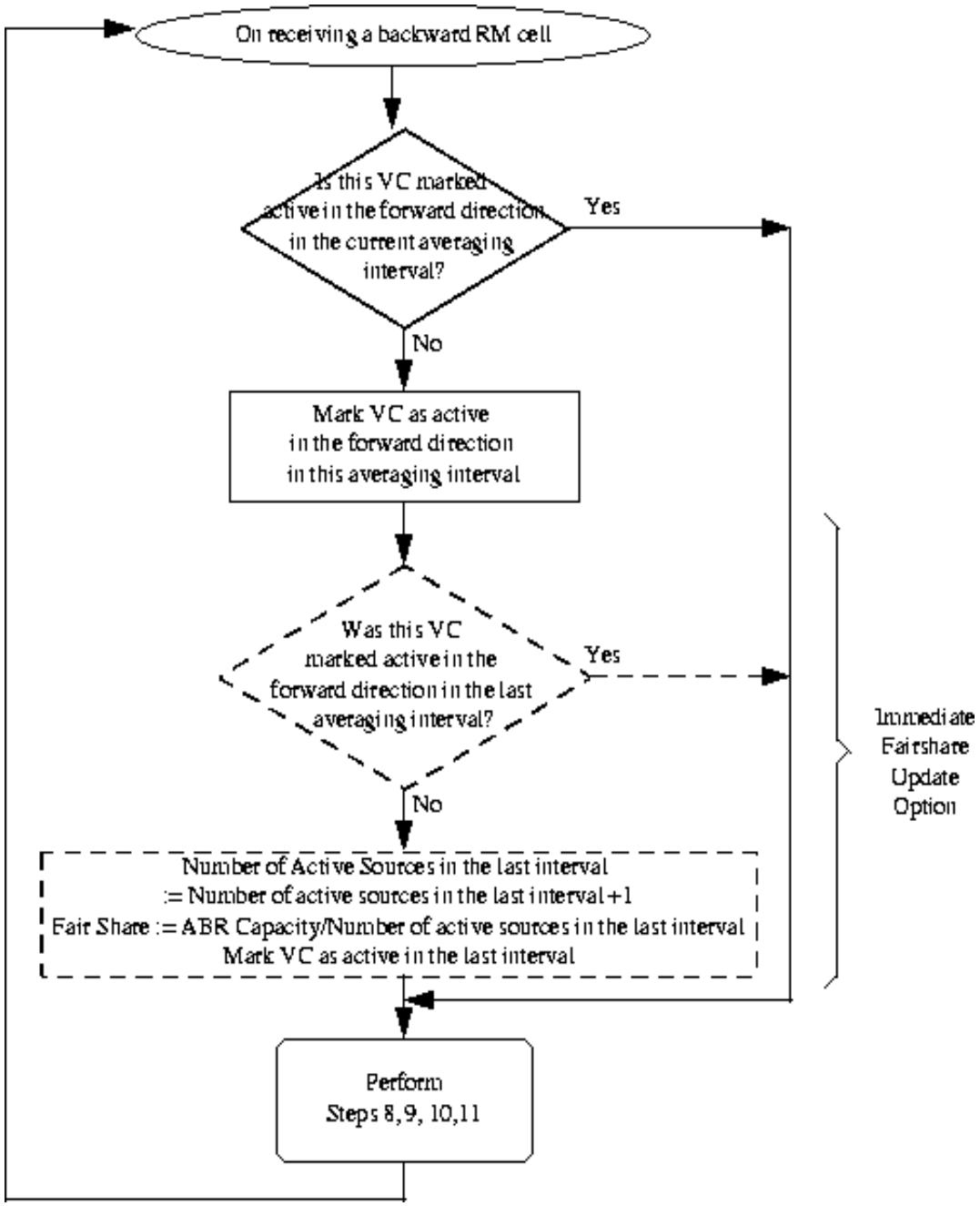


Figure C.3: Flow Chart of Bi-Directional Counting

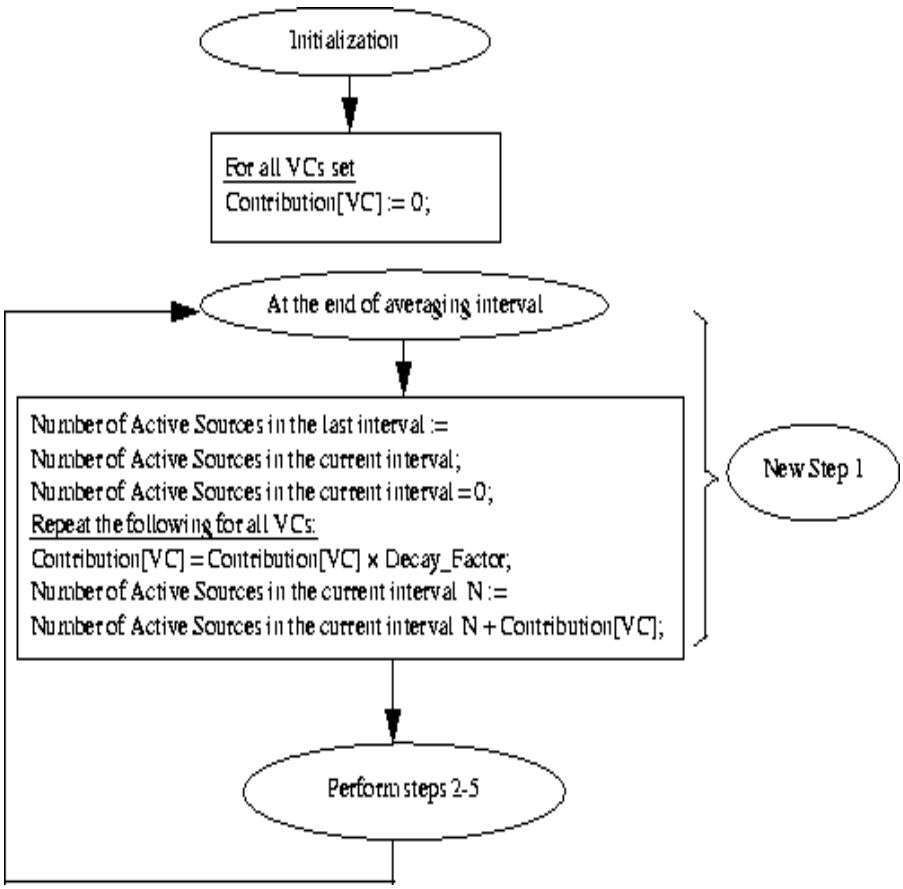


Figure C.4: Flow Chart of averaging number of active sources (part 1 of 2)

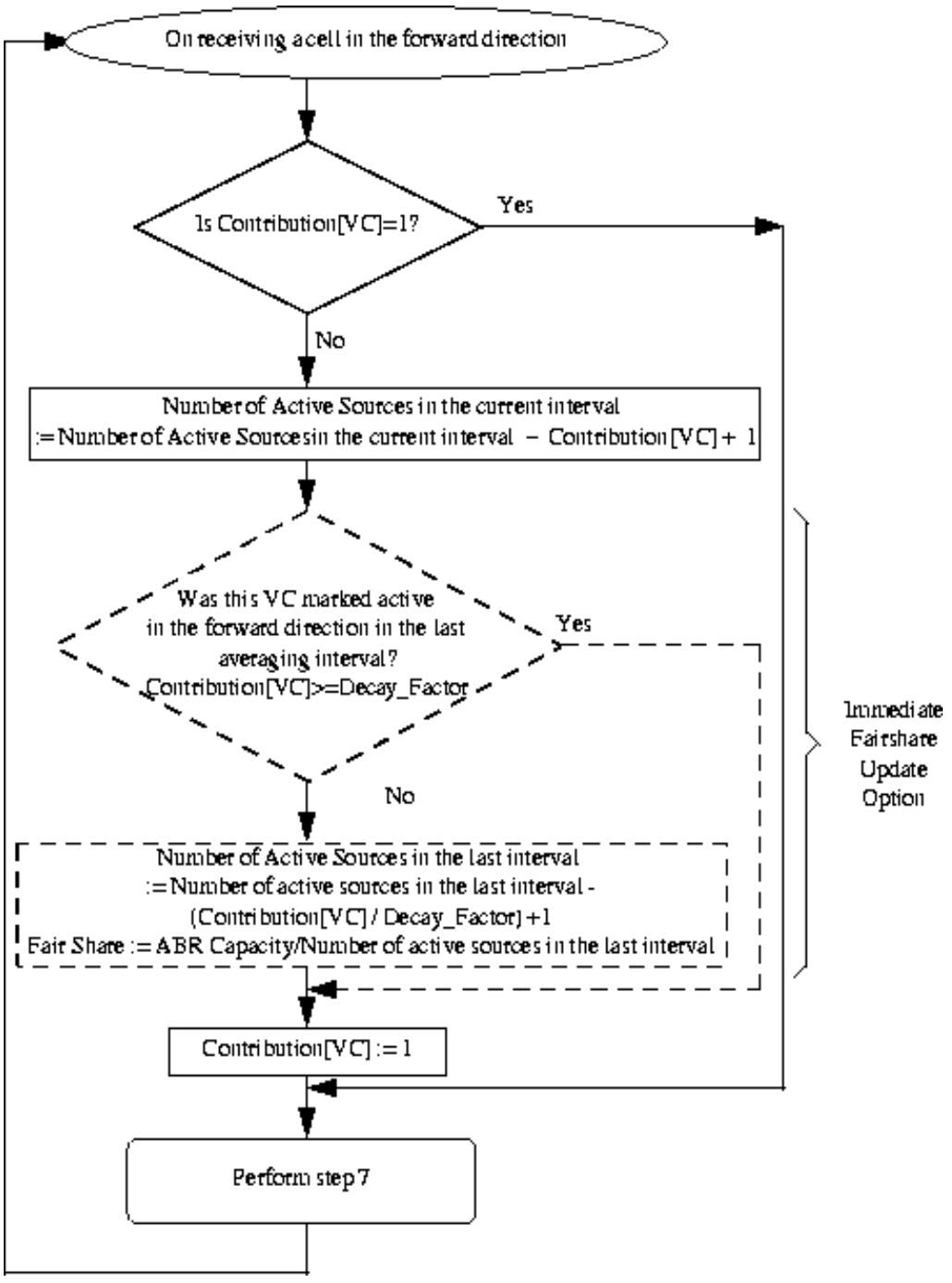


Figure C.5: Flow Chart of averaging number of active sources (part 2 of 2)

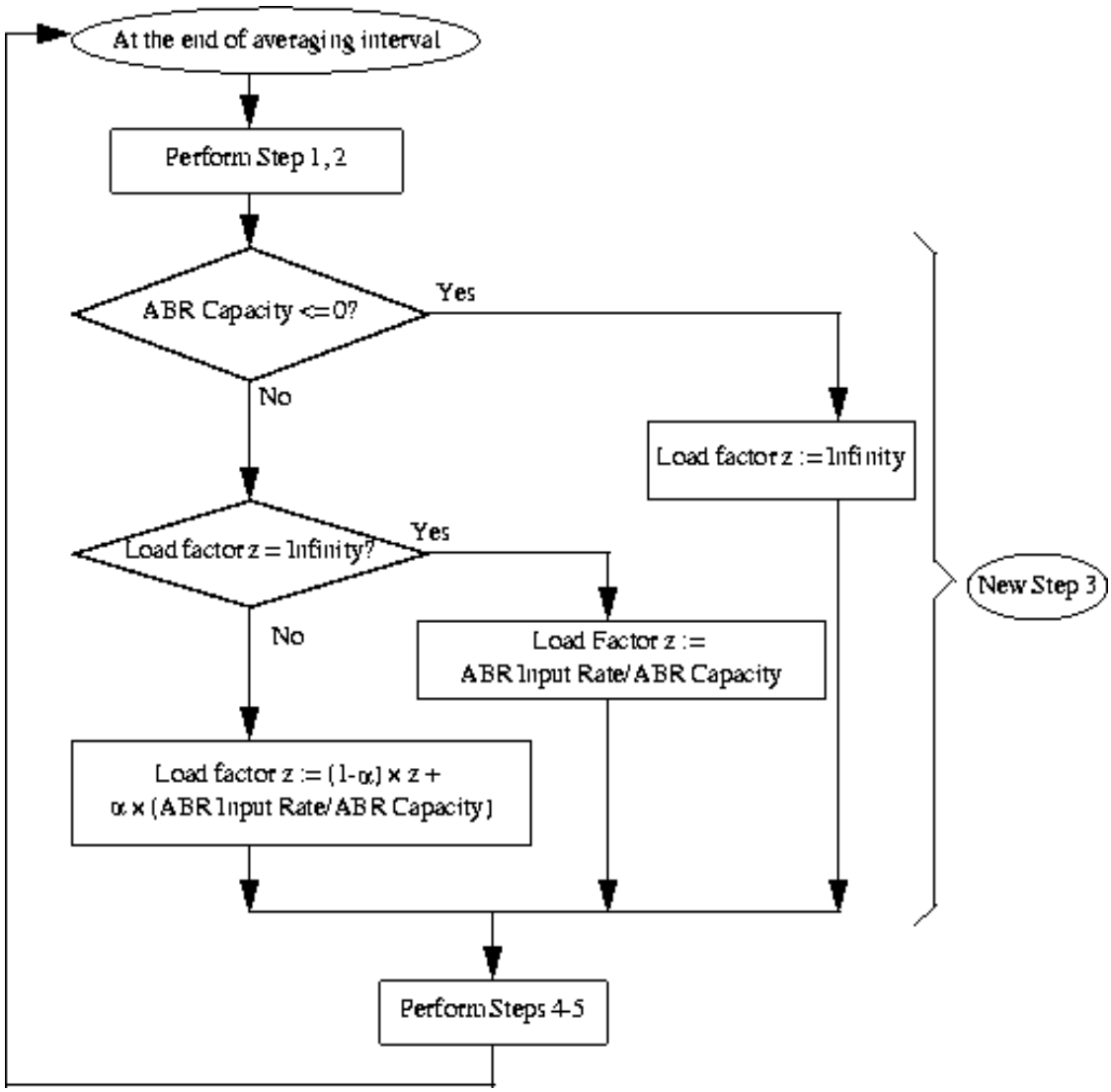


Figure C.6: Flow chart of averaging of load factor (method 1)

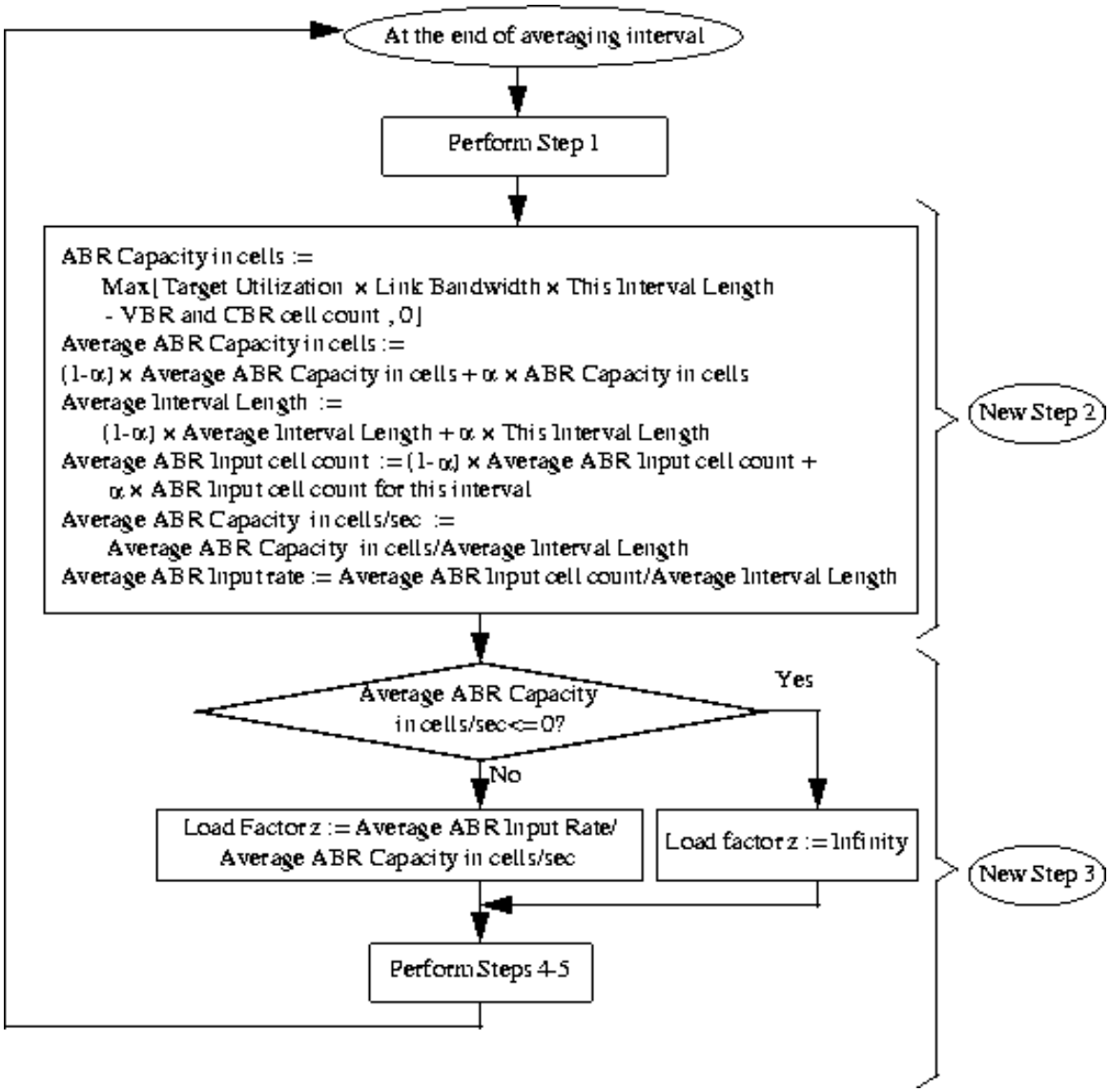


Figure C.7: Flow chart of averaging of load factor (method 2)

C.3 Pseudocode for VS/VD Design Options

The pseudo code describes the combination of the following options:

a) VC's rate from FRM1

b) VC's rate from FRM2

c) VC's rate from measured VC's source rate to per class queues.

A) Measure input rate at entry to per VC queues

B) Measure input rate at entry to per class queues

I) Allocated rate update at FRM turnaround only. Link congestion affects previous loop only.

II) Allocated rate update at BRM receive only. Link congestion affects previous loop and next loop.

III) Allocated rate update at FRM turnaround and BRM receive.Link congestion affects previous loop and next loop.

Observe that the only acceptable combinations are:

a), A), I)

a), A), II)

a), A), III)

c), B), I)

c), B), II)

c), B), III)

C.4 Pseudocode

FRM/BRM/Data receive

(* — switch receives a cell from 'VC' — *)

IF(Opt A) Measure_Input_Rate() ENDIF (* Opt A *)

IF cell is an ABR cell

(* — VD Code: FRM receive — *)

IF cell is an FRM cell

(* — Opt a: CCR update from FRM1 (for ERICA table)— *)

IF (Opt a) CCR ←CCR from FRM ENDIF (* Opt a *)

(* — Link (switch) bottleneck rate — *)

IF (Opt I or III) newER ←Calculate_Allocated_Rate() (* VAL update *)

ELSE IF (Opt II)

newER ←Latest_Allocated_Rate[VC] (* VAL table lookup *)

ENDIF

(* — Bottleneck rate of next loop —- *)

newER ←Minm(newER, ACR of the VC on next loop)

(* — Source bottleneck rate (cell-¿ER) — *)

newER ←Minm(newER,cell-¿ER)

ER_TA for the VC ←newER

(* — Link congestion propogation to the next loop: set ACR — *)

IF (Opt II or III used)

ACR (of the VC on next loop) ←Min(ACR , newER

(* – CCR update from FRM2 (or ACR) — *)

IF (Opt b) used) CCR ←ACR for the VC on next loop ENDIF

ENDIF

turn_around ←1 (* BRM will be generated *)

Turnaround BRM as in DES rules 2-6 (See appendix A)

free the FRM cell

(* —- VS code: BRM receive —- *)

ELSE IF cell is an BRM cell

ACR of the VC ←According to SES rules 8-9

IF (Opt II or III) (* Opt II or III *)

newER ←ERICA algorithm

ACR for the VC ←Minm(ACR, newER)(* Link congestion *)

(* Opt b: CCR update from FRM2 (ACR) *)

IF (Opt b used) CCR ←ACR for the VC

ENDIF

IF (rescheduling option) reschedule the cell sending of this VC ENDIF

free the BRM cell

(* —- VS code: data receive —- *)

ELSE IF cell is a data cell

enqueue the cell to per VC queue in VS of the switch

schedule the sending of this VC queue if necessary

ENDIF (* Cell type: FRM/BRM/data *)

ENDIF (* ABR cell *)

(* — More VS code: FRM/BRM/Data send — *)

(* There is a cell in the VS queue of a source and it

is the scheduling slot for the source ... *)

Follow SES Rules 1-4 (see appendix A).

(* Before Rule 5 *)

(* T = inter FRM time used in Rule 5 test *)

SR ←Nrm/T

IF (Opt c) CCR ←SR ENDIF (* Opt c *)

SES Rules 6 etc (see appendix A).

(* — Before enqueuing the cell (data/FRM/BRM) to per class queue — *)

(* Input rate to per class queue *)

IF (Opt B) Measure_Input_Rate() ENDIF

Enqueue the cell to the ABR per-class queue.

(* END of VS/VD options *)

APPENDIX D

GLOSSARY OF COMMONLY USED ACRONYMNS

ABR - Available Bit Rate

ACR - Allowed Cell Rate

ADTF - ACR Decrease Time Factor

AI - Averaging Interval

APRC Scheme - Adaptive Proportional Rate Control scheme

ATM - Asynchronous Transfer Mode

BECN - Backward Explicit Congestion Notification

BN bit - backward notification bit (RM cell)

BRM - Backward RM cell

CAC - Connection Admission Control

CAPC2 Scheme- Congestion Avoidance using Proportional Control scheme, version 2

CBR - Constant Bit Rate

CCR - Current Cell Rate

CDF - Cutoff Decrease Factor

CI bit - Congestion Indication bit

CLP - Cell Loss Priority

CRM - Missing RM-cell Count

DIR bit - direction bit (RM cell)

DMRCA Scheme - Dynamic Max Rate Control Algorithm

EFCI - Explicit Forward Congestion Indicator

EOM cell - End of Message cell

EPRCA Scheme - Enhanced Proportional Rate Control Algorithm

ER - Explicit Rate

ERICA(+) - Explicit Rate Indication for Congestion Avoidance Schemes

FCVC - Flow Controlled Virtual Circuits (Credit Based Scheme)

FD - Feedback Delay

FIFO - First In First Out

FMMRA Scheme - Fast Max-Min Rate Allocation scheme

FRM - Forward RM cell

FRTT - Fixed Round-Trip Time

HKUST Scheme - Hong Kong University of Science and Technology scheme

ICR - Initial Cell Rate

IRCT - Inter-RM Cell Time

ITU-T - International Telecommunications Union, Telcommunications Sector

LAN - Local area network

LANE - LAN Emulation

LRD traffic - Long range dependendent traffic

MCR - Minimum Cell Rate

MIT Scheme - Massachussetts Institute of Technology Scheme

MPEG-2 standard - Motion Pictures Experts Group Standard for compression, version 2

MPOA - Multiprotocol over ATM

MSS - Maximum Segment Size

Mrm - Controls bandwidth allocation between FRM, BRM and data cells

NI bit - No Increase bit

Nrm - Number of cells between FRM cells

OCR - Offered Average Cell Rate

OSU Scheme - Ohio State University Scheme

PCR - Peak Cell Rate

PRCA - Proportional Rate Control Algorithm

PTI - Payload Type Indicator

QoS - Quality of Service

RDF - Rate Decrease Factor

RIF - Rate Increase Factor

RM cells - Resource Management cells

RTT - Round Trip Time

SFD - Shortest Feedback Delay

SPTS - Single Program Transport Stream (MPEG-2)

TBE - Transient Buffer Exposure

TCP/IP - Transmission Control Protocol/Internet Protocol (layer 4/3 of the Internet)

TCR - a) Transmitted Cell Rate (OSU scheme) b) Tagged Cell Rate (SES parameter)

TM4.0 - ATM Traffic Management Specification, version 4.0

TUB - Target Utilization Band

Trm - Upper Bound on Inter-FRM Time

U - Target Utilization parameter in OSU, ERICA schemes

UCSC Scheme - University of Santa Cruz Scheme

UILI policies - Use-it-or-Lose-it policies

VBR - Variable Bit Rate (comes in the -rt (real-time) and -nrt (non-real time) flavors)

VC - Virtual Circuit

VS/VD - Virtual Source/Virtual Destination Option

WAN - Wide area network

BIBLIOGRAPHY

[1] Santosh P. Abraham and Anurag Kumar. Max-Min Fair Rate Control of ABR Connections with Nonzero MCRs. *IISc Technical Report*, 1997.

[2] Yehuda Afek, Yishay Mansour, and Zvi Ostfeld. Convergence Complexity of Optimistic Rate Based Flow Control Algorithms. In *28th Annual Symposium on Theory of Computing (STOC)*, pages 89–98, 1996.

[3] Yehuda Afek, Yishay Mansour, and Zvi Ostfeld. Phantom: A Simple and Effective Flow Control Scheme. In *Proceedings of the ACM SIGCOMM*, pages 169–182, August 1996.

[4] Anthony Alles. ATM Internetworking. White paper, Cisco Systems, http://www.cisco.com, May 1995.

[5] G.J. Armitage and K.M. Adams. ATM Adaptation Layer Packet Reassembly during Cell Loss. *IEEE Network Magazine*, September 1993.

[6] Ambalavanar Arulambalam, Xiaoqiang Chen, and Nirwan Ansari. Allocating Fair Rates for Available Bit Rate Service in ATM Networks. *IEEE Communications Magazine*, 34(11):92–100, November 1996.

[7] A.W.Barnhart. Changes Required to the Specification of Source Behavior. ATM Forum 95-0193, February 1995.

[8] A.W.Barnhart. Evaluation and Proposed Solutions for Source Behavior # 5. ATM Forum 95-1614, December 1995.

[9] A. W. Barnhart. Use of the Extended PRCA with Various Switch Mechanisms. ATM Forum 94-0898, 1994.

[10] A. W. Barnhart. Example Switch Algorithm for TM Spec. ATM Forum 95-0195, February 1995.

[11] J. Bennett, K. Fendick, K.K. Ramakrishnan, and F. Bonomi. RPC Behavior as it Relates to Source Behavior 5. ATM Forum 95-0568R1, May 1995.

[12] J. Bennett and G. Tom Des Jardins. Comments on the July PRCA Rate Control Baseline. ATM Forum 94-0682, July 1994.

[13] J. Beran, R. Sherman, M. Taqqu, and W. Willinger. Long-Range Dependence in Variable-Bit-Rate Video Traffic. *IEEE Transactions on Communications*, 43(2/3/4), February/March/April 1995.

[14] U. Black. *ATM: Foundation for Broadband Networks*. Prentice Hall, New York, 1995.

[15] D. Cavendish, S. Mascolo, and M. Gerla. SP-EPRCA: an ATM Rate Based Congestion Control Scheme basedon a Smith Predictor. Technical report, UCLA, 1997.

[16] A. Charny, G. Leeb, and M. Clarke. Some Observations on Source Behavior 5 of the Traffic Management Specification. ATM Forum 95-0976R1, August 1995.

[17] Anna Charny. An Algorithm for Rate Allocation in a Cell-Switching Network with Feedback. Master's thesis, Massachusetts Institute of Technology, May 1994.

[18] Anna Charny, David D. Clark, and Raj Jain. Congestion control with explicit rate indication. In *Proceedings of the IEEE International Communications Conference (ICC)*, June 1995.

[19] D. Chiu and R. Jain. Analysis of the Increase/Decrease Algorithms for Congestion Avoidance in Computer Networks. *Journal of Computer Networks and ISDN Systems*, 1989.

[20] Fabio M. Chiussi, Ye Xia, and Vijay P. Kumar. Dynamic max rate control algorithm for available bit rate service in atm networks. In *Proceedings of the IEEE GLOBECOM*, volume 3, pages 2108–2117, November 1996.

[21] D.P.Heyman and T.V. Lakshman. What are the implications of Long-Range Dependence for VBR-Video Traffic Engineering ? *ACM/IEEE Transactions on Networking*, 4(3):101–113, June 1996.

[22] Harry J.R. Dutton and Peter Lenhard. *Asynchronous Transfer Mode (ATM) Technical Overview*. Prentice Hall, New York, 2nd edition, 1995.

[23] H. Eriksson. MBONE: the multicast backbone. *Communications of the ACM*, 37(8):54–60, August 1994.

[24] J. Scott et al. Link by Link, Per VC Credit Based Flow Control. ATM Forum 94-0168, 1994.

[25] M. Hluchyj et al. Closed-Loop Rate-Based Traffic Management. ATM Forum 94-0211R3, April 1994.

[26] M. Hluchyj et al. Closed-loop rate-based traffic management. *ATM Forum 94-0438R2*, 1994.

[27] S. Fahmy, R. Jain, S. Kalyanaraman, R. Goyal, and F. Lu. On source rules for abr service on atm networks with satellite links. In *Proceedings of First International Workshop on Satellite-based Information Services (WOSBIS)*, November 1996.

[28] Chien Fang and Arthur Lin. A Simulation Study of ABR Robustness with Binary-Mode Switches: Part II. ATM Forum 95-1328R1, October 1995.

[29] Chien Fang and Arthur Lin. On TCP Performance of UBR with EPD and UBR-EPD with a Fair Buffer Allocation Scheme. ATM Forum 95-1645, December 1995.

[30] R. Fielding, J. Gettys, J. Mogul, H. Frystyk, and T. Berners-Lee. Hypertext Transfer Protocol – HTTP/1.1. Request For Comments, RFC 2068, January 1997.

[31] ATM Forum. http://www.atmforum.com.

[32] ATM Forum. The ATM Forum Traffic Management Specification Version 4.0. ftp://ftp.atmforum.com/pub/approved-specs/af-tm-0056.000.ps, April 1996.

[33] M. Garrett and W. Willinger. Analysis, modeling, and generation of self-similar vbr video traffic. In *Proceedings of the ACM SIGCOMM*, August 1994.

[34] Matthew S. Goldman. Variable Bit Rate MPEG-2 over ATM: Definitions and Recommendations. ATM Forum 96-1433, October 1996.

[35] Rohit Goyal, Raj Jain, Shiv Kalyanaraman, Sonia Fahmy, and Seong-Cheol Kim. Performance of TCP over UBR+. ATM Forum 96-1269, October 1996.

[36] Rohit Goyal, Raj Jain, Shiv Kalyanaraman, Sonia Fahmy, Bobby Vandalore, Xiangrong Cai, and Seong-Cheol Kim. Selective Acknowledgements and UBR+ Drop Policies to Improve TCP/UBR Performance over Terrestrial and Satellite Networks. ATM Forum 97-0423, April 1997.

[37] Rohit Goyal, Raj Jain, Shiv Kalyanaraman, Sonia Fahmy, Bobby Vandalore, Xiangrong Cai, and Seong-Cheol Kim. Selective Acknowledgements and UBR+ Drop Policies to Improve TCP/UBR Performance over Terrestrial and Satellite Networks. ATM Forum 97-0423, April 1997.

[38] M. Grossglauser, S.Keshav, and D.Tse. RCBR: a simple and efficient service for multiple time-scale traffic. In *Proceedings of the ACM SIGCOMM*, August 1995.

[39] S. Hrastar, H. Uzunalioglu, and W. Yen. Synchronization and de-jitter of mpeg-2 transport streams encapsulated in aal5/atm. In *Proceedings of the IEEE International Communications Conference (ICC)*, volume 3, pages 1411–1415, June 1996.

[40] Van Jacobson. Congestion avoidance and control. In *Proceedings of the ACM SIGCOMM*, pages 314–329, August 1988.

[41] J. Jaffe. Bottleneck Flow Control. *IEEE Transactions on Communications*, COM-29(7):954–962, 1980.

[42] R. Jain. A timeout-based congestion control scheme for window flow-controlled networks. *IEEE Journal on Selected Areas in Communications*, 1986.

[43] R. Jain. The osu scheme for congestion avoidance using explicit rate indication. *ATM Forum 94-0883*, 1994.

[44] R. Jain, S. Kalyanaraman, R. Goyal, S. Fahmy, and F. Lu. A Fix for Source End System Rule 5. ATM Forum 95-1660, December 1995.

[45] R. Jain, S. Kalyanaraman, R. Viswanathan, and R. Goyal. A sample switch algorithm. *ATM Forum 95-0178R1*, 1995.

[46] R. Jain, K. K. Ramakrishnan, and D. M. Chiu. Congestion Avoidance in Computer Networks with a Connectionless Network Layer. Technical Report DEC-TR-506, Digital Equipment Corporation, August 1987.

[47] R. Jain and S. Routhier. Packet Trains - Measurement and a new model for computer network trafic. *IEEE Journal of Selected Areas in Communications,*, 1986.

[48] Raj Jain. Congestion Control in Computer Networks: Issues and Trends. *IEEE Network Magazine*, pages 24–30, May 1990.

[49] Raj Jain. *The Art of Computer Systems Performance Analysis*. John Wiley & Sons, 1991.

[50] Raj Jain. Myths about Congestion Management in High-speed Networks. *Internetworking: Research and Experience*, 3:101–113, 1992.

[51] Raj Jain. ABR Service on ATM Networks: What is it? Network World, 1995.

[52] Raj Jain. Congestion Control and Traffic Management in ATM Networks: Recent advances and a survey. *Computer Networks and ISDN Systems Journal*, October 1996.

[53] Raj Jain, Sonia Fahmy, Shivkumar Kalyanaraman, Rohit Goyal, and Fang Lu. More Straw-Vote Comments: TBE vs Queue sizes. ATM Forum 95-1661, December 1995.

[54] Raj Jain, Shivkumar Kalyanaraman, Sonia Fahmy, and Fang Lu. Out-of-Rate RM Cell Issues and Effect of Trm, TOF, and TCR. ATM Forum 95-973R1, August 1995.

[55] Raj Jain, Shivkumar Kalyanaraman, Sonia Fahmy, and Fang Lu. Straw-Vote comments on TM 4.0 R8. ATM Forum 95-1343, October 1995.

[56] Raj Jain and Shivkumar Kalyanaraman Ram Viswanathan. 'method and apparatus for congestion management in computer networks using explicit rate indication. U. S. Patent application (S/N 307,375), SepJuly 1994.

[57] H. Tzeng K. Siu. Intelligent congestion control for abr service in atm networks. *Computer Communication Review*, 24(5):81–106, October 1995.

[58] Lampros Kalampoukas, Anujan Varma, and K.K. Ramakrishnan. An efficient rate allocation algorithm for atm networks providing max-min fairness. In *6th IFIP International Conference on High Performance Networking (HPN)*, September 1995.

[59] Shivkumar Kalyanaraman, Raj Jain, Sonia Fahmy, Rohit Goyal, and Jianping Jiang. Performance of TCP over ABR on ATM backbone and with various VBR traffic patterns. In *Proceedings of the IEEE International Communications Conference (ICC)*, June 1997.

[60] Shivkumar Kalyanaraman, Raj Jain, Rohit Goyal, and Sonia Fahmy. A Survey of the Use-It-Or-Lose-It Policies for the ABR Service in ATM Networks. Technical Report OSU-CISRC-1/97-TR02, Dept of CIS, The Ohio State University, 1997.

[61] J.B. Kenney. Problems and Suggested Solutions in Core Behavior. ATM Forum 95-0564R1, May 1995.

[62] Bo-Kyoung Kim, Byung G. Kim, and Ilyoung Chong. Dynamic Averaging Interval Algorithm for ERICA ABR Control Scheme. ATM Forum 96-0062, February 1996.

[63] H. T. Kung. Adaptive Credit Allocation for Flow-Controlled VCs. ATM Forum 94-0282, 1994.

[64] H. T. Kung. Flow Controlled Virtual Connections Proposal for ATM Traffic Management. ATM Forum 94-0632R2, September 1994.

[65] T.V. Lakshman, P.P. Mishra, and K.K. Ramakrishnan. Transporting compressed video over atm networks with explicit rate feedback control. In *Proceedings of the IEEE INFOCOM*, April 1997.

[66] L.G.Roberts. Operation of Source Behavior # 5. ATM Forum 95-1641, December 1995.

[67] Hongqing Li, Kai-Yeung Siu, Hong-Ti Tzeng, Chinatsu Ikeda, and Hiroshi Suzuki. Tcp over abr and ubr services in atm. In *Proceedings of IPCCC'96*, March 1996.

[68] S. Liu, M. Procanik, T. Chen, V.K. Samalam, and J. Ormond. An analysis of source rule # 5. ATM Forum 95-1545, December 1995.

[69] P. Newman. Traffic Management for ATM Local Area Networks. *IEEE Communications Magazine*, 1994.

[70] Craig Partridge. *Gigabit Networking*. Addison-Wesley, Reading, MA, 1993.

[71] Vern Paxson. Fast Approximation of Self-Similar Network Traffic. Technical Report LBL-36750, Lawrence Berkeley Labs, April 1995.

[72] K. K. Ramakrishnan, D. M. Chiu, and R. Jain. Congestion Avoidance in Computer Networks with a Connectionless Network Layer. Part IV: A Selective Binary Feedback Scheme for General Topologies. Technical report, Digital Equipment Corporation, 1987.

[73] K.K. Ramakrishnan, P. P. Mishra, and K. W. Fendick. Examination of Alternative Mechanisms for Use-it-or-Lose-it. ATM Forum 95-1599, December 1995.

[74] Larry Roberts. Enhanced PRCA (Proportional Rate-Control Algorithm). ATM Forum 94-0735R1, August 1994.

[75] Allyn Romanov and Sally Floyd. Dynamics of TCP Traffic over ATM Networks. *IEEE Journal on Selected Areas in Communications*, May 1995.

[76] Lucent Technologies. Atlanta chip set, microelectronics group news announcement, http://www.lucent.com/micro/news/032497.html.

[77] Christos Tryfonas. MPEG-2 Transport over ATM Networks. Master's thesis, University of California at Santa Cruz, September 1996.

[78] International Telecommunications Union. http://www.itu.ch.

[79] Bobby Vandalore, Shiv Kalyanaraman, Raj Jain, Rohit Goyal, Sonia Fahmy, Xiangrong Cai, and Seong-Cheol Kim. Performance of Bursty World Wide Web (WWW) Sources over ABR. ATM Forum 97-0425, April 1997.

[80] Bobby Vandalore, Shiv Kalyanaraman, Raj Jain, Rohit Goyal, Sonia Fahmy, and Pradeep Samudra. Worst case TCP behavior over ABR and buffer requirements. ATM Forum 97-0617, July 1997.

[81] Gary R. Wright and W. Richard Stevens. *TCP/IP Illustrated, Volume 2*. Addison-Wesley, Reading, MA, 1995.

[82] Lixia Zhang, Scott Shenker, and D.D.Clark. Observations on the dynamics of a congestion control algorithm: The effects of two-way traffic. In *Proceedings of the ACM SIGCOMM*, August 1991.