Traffic Management for Point-to-Point and Multipoint 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

Sonia Fahmy, B.Sc., M.S.

* * * * *

The Ohio State University

1999

Dissertation Committee:

Professor Raj Jain, Adviser

Professor Wu-chi Feng

Professor Ten-Hwang Lai

Approved by

Adviser

Department of Computer and Information Science

ABSTRACT

Traffic management aims at efficiently allocating network resources and meeting the negotiated quality of service guarantees. Asynchronous transfer mode (ATM) networks allow seamless transport of voice, video and data on the same network. ATM networks provide several service categories for real-time and bulk data transfer. The available bit rate (ABR) service category attempts to provide (possibly non-zero) minimum rate guarantees, achieve fairness, and minimize cell loss by periodically indicating to sources the rate at which to transmit. Thus ABR is ideal for data distribution applications, and can perform well for real-time applications with the appropriate implementation and parameter choices.

Many ATM applications require multipoint communication, where one or more senders concurrently transmit to multiple receivers. Examples of such applications include audio and video conferencing, distance learning, server and database synchronization, and data distribution applications. A flexible and efficient ATM multipoint capability is thus required. This research focuses on the development of a traffic management framework for unicast and multicast connections in ATM networks, with a focus on the ABR service.

We examine point-to-multipoint and multipoint-to-point connection support, forming a foundation for multipoint-to-multipoint connections. Simplicity and scalability of the schemes are our main concerns. We develop definitions of the optimal and

fair bandwidth allocations, based on connections, sources or flows. The operation of branch points that consolidate feedback, and that of merge points that regulate feedback, is designed. The performance of the schemes is analyzed under a large variety of realistic configurations and traffic patterns. Results indicate that the branch point consolidation algorithm exhibits stability and a fast transient response, while the merge point algorithm allocates fair rates and regulates feedback.

In addition, we examine ABR parameters and rate allocation schemes. Formulae are developed for selecting ABR parameter values, and the effect of round trip time and link bandwidths is determined. A novel mechanism for allocating rates, and a method for multiplexing virtual connections on virtual paths, are designed and analyzed. Most of the problems we resolve are not specific to ATM networks, and provide insight into traffic management design in computer networks in general.

Traffic Management for Point-to-Point and Multipoint Available Bit Rate (ABR) Service in Asynchronous Transfer Mode (ATM) Networks

By

Sonia Fahmy, Ph.D.

The Ohio State University, 1999

Professor Raj Jain, Adviser

Traffic management aims at efficiently allocating network resources and meeting the negotiated quality of service guarantees. Asynchronous transfer mode (ATM) networks allow seamless transport of voice, video and data on the same network. ATM networks provide several service categories for real-time and bulk data transfer. The available bit rate (ABR) service category attempts to provide (possibly non-zero) minimum rate guarantees, achieve fairness, and minimize cell loss by periodically indicating to sources the rate at which to transmit. Thus ABR is ideal for data distribution applications, and can perform well for real-time applications with the appropriate implementation and parameter choices.

Many ATM applications require multipoint communication, where one or more senders concurrently transmit to multiple receivers. Examples of such applications

include audio and video conferencing, distance learning, server and database synchronization, and data distribution applications. A flexible and efficient ATM multipoint capability is thus required. This research focuses on the development of a traffic management framework for unicast and multicast connections in ATM networks, with a focus on the ABR service.

We examine point-to-multipoint and multipoint-to-point connection support, forming a foundation for multipoint-to-multipoint connections. Simplicity and scalability of the schemes are our main concerns. We develop definitions of the optimal and fair bandwidth allocations, based on connections, sources or flows. The operation of branch points that consolidate feedback, and that of merge points that regulate feedback, is designed. The performance of the schemes is analyzed under a large variety of realistic configurations and traffic patterns. Results indicate that the branch point consolidation algorithm exhibits stability and a fast transient response, while the merge point algorithm allocates fair rates and regulates feedback.

In addition, we examine ABR parameters and rate allocation schemes. Formulae are developed for selecting ABR parameter values, and the effect of round trip time and link bandwidths is determined. A novel mechanism for allocating rates, and a method for multiplexing virtual connections on virtual paths, are designed and analyzed. Most of the problems we resolve are not specific to ATM networks, and provide insight into traffic management design in computer networks in general.

ACKNOWLEDGMENTS

As I approach the end of my graduate career, I am left with profound feelings of accomplishment, excitement, and gratitude to a number of people whose support has been integral in completing this dissertation.

My advisor, Professor Raj Jain, has been a constant source of inspiration, motivation and guidance throughout the past five years. I have learned so much from his keen insight, his problem solving abilities, and his amazing energy.

I am extremely grateful to my committee members, Professor Steve Lai and Professor Wu-chi Feng, and to Professor Anish Arora, who have carefully examined my dissertation proposal and provided me with valuable feedback. Professors Neelam Soundararajan, Mike Liu, Mary Jean Harrold, Tim Long, Bruce Weide, and Stuart Zweben have also been extremely helpful throughout my graduate school journey. I would also like to thank everyone in the CIS office, especially Elley, Tom, Elizabeth, Deanna, Marty, Sandy and James.

My parents and sister have been extremely supportive of my studies throughout my life, and have always been overwhelmingly encouraging and understanding. I am also grateful to my wonderful friends in Columbus, Ohio, and throughout the world, and my family in Egypt and the United States.

My colleagues and good friends at the networking laboratory, especially Bobby Vandalore, Rohit Goyal, Shivkumar Kalyanaraman, Mukul Goyal, Chunlei Liu, and

Sohail Munir, have helped me through every step of my research. Their aid has been valuable in writing utility programs, running simulations, and proof-reading papers just a few hours before their deadlines.

My work has been sponsored by OSU-CIS teaching assistantships, the OSU presidential fellowship, the National Science Foundation, and Rome Air Force Laboratories. I appreciate their generous support.

I am grateful to many colleagues in the networking community. Professor Melody Moh and Mr. Sastri Kota have been wonderful mentors. Professor Sunny Siu has also provided a lot of help. Many members of the ATM Forum have provided valuable comments and suggestions on various parts of this dissertation.

VITA

1972	Born, Cairo, Egypt
1992	B.Sc. Computer Science, The American University in Cairo, Cairo, Egypt
1992-1994	Software Engineer, Pyramid Systems Development, Cairo, Egypt
1996	M.S. Computer and Information Science, The Ohio State University, Columbus, Ohio, USA
1994-present	Graduate Teaching Associate, Graduate Research Associate, and Presidential Fellow, The Ohio State University, Columbus, Ohio, USA

PUBLICATIONS

Research Publications

S. Fahmy, R. Jain, R. Goyal, B. Vandalore, and S. Kalyanaraman. Design and Evaluation of Feedback Consolidation for ABR Point-to-Multipoint Connections in ATM Networks. *Journal of Computer Communications*, 25 July 1999, Vol. 22, Issue 12, pp. 1085-1103.

S. Fahmy, R. Jain, S. Rabie, R. Goyal and B. Vandalore. Quality of Service for Internet Traffic over ATM Service Categories. *Journal of Computer Communications*, 1999, Vol. 22, Issue 14.

S. Fahmy and R. Jain. ABR Flow Control for Multipoint Connections. *IEEE Network Magazine, ATM Forum Perspectives column,* September-October 1998, Vol. 12, Issue 5, pp. 6-7.

S. Fahmy and R. Jain. Resource Reservation Protocol (RSVP). Chapter in the *Handbook of Communications Technologies: The Next Decade,* CRC Press, R. Osso, editor, July 1999.

R. Jain, S. Kalyanaraman, R. Goyal, R. Vishwanathan and S. Fahmy. ERICA: Explicit Rate Indication for Congestion Avoidance in ATM Networks. *US Patent 5,805,577.*

B. Vandalore, S. Fahmy, R. Jain, R. Goyal, M. Goyal. QoS and Multipoint support for Multimedia Applications over the ATM ABR Service. *IEEE Communications Magazine,* January 1999, pp. 53-57.

R. Jain, S. Kalyanaraman, S. Fahmy, R. Goyal. Source Behavior for ATM ABR Traffic Management: An Explanation. *IEEE Communications Magazine,* vol 34, no 11, pp. 50-57, November 1, 1996.

S. Kalyanaraman, R. Jain, S. Fahmy, R. Goyal and S-C Kim. Performance and Buffering Requirements of Internet Protocols over ATM ABR and UBR Services. *IEEE Communications Magazine,* vol 36, no 6, pp 152-157, June 1, 1998.

R. Goyal, R. Jain, S. Kalyanaraman, S. Fahmy, B. Vandalore. Improving the Performance of TCP over the ATM-UBR service. *Journal of Computer Communications,* Vol. 21, Issue 10, pp. 898-911, July 1998.

S. Kalyanaraman, R. Jain, R. Goyal, S. Fahmy, and S-C Kim. Use-it or Lose-it Policies for the Available Bit Rate (ABR) Service in ATM Networks. *Journal of Computer Networks and ISDN Systems,* vol 30, no 24, pp. 2293-2308, December 1998.

S. Kalyanaraman, R. Jain, J. Jiang, R. Goyal, S. Fahmy and P. Samudra. Design Considerations for the Virtual Source/Virtual Destination (VS/VD) Feature in the ABR Service of ATM Networks. *Journal of Computer Networks and ISDN Systems,* vol 30, no 19, pp. 1811-1824, October 1998.

R. Goyal, R. Jain, S. Kota, M. Goyal, S. Fahmy, B. Vandalore. Traffic Management for TCP/IP over Satellite-ATM Networks. *IEEE Communications Magazine,* March 1999.

FIELDS OF STUDY

Major Field: Computer and Information Science

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

The Internet is experiencing explosive growth. End-to-end quality of service (QoS) is critical to the success of the Internet and its applications. QoS in computer networks is important for several reasons. Critical applications, such as real-time auctions and transactions, should be given priority over less critical ones such as some World Wide Web (WWW) stored video transmissions. Moreover, most multimedia applications require delay or delay variation guarantees for acceptable performance. The ability to give preferential treatment to some flows is important both among customers or aggregates, depending on the tariff or subscription, and also within an aggregate (for example, to prevent starvation among sessions). Traffic management thus becomes essential.

Asynchronous transfer mode (ATM) networks are being deployed throughout the information infrastructure, especially in carrier backbones. ATM is proposed to transport a wide variety of traffic, such as voice, video and data, in a seamless manner. ATM transports data in fixed size 53 byte-long packets, called cells. End systems must set up virtual channel connections (VCCs) of appropriate service categories prior to transmitting information. Service categories distinguish a small number of general ways to provide QoS, which are appropriate for different classes of applications. A

representative list of current and future applications includes video, voice, image and data in conversational, messaging, distribution and retrieval modes. The required service categories can be derived from the properties of the application. ATM service categories distinguish real-time from non-real-time services, and provide simple and complex solutions for each case. The added mechanisms in the more complex categories are justified by providing a benefit or economy to a significant subset of the applications [44]. ATM provides six service categories: Constant Bit Rate (CBR), real-time Variable Bit Rate (rt-VBR), non real-time Variable Bit Rate (nrt-VBR), Available Bit Rate (ABR), Guaranteed Frame Rate (GFR) and Unspecified Bit Rate (UBR) [43].

Traffic management and QoS issues become more complicated in case of multipoint communication. Multipoint communication is the exchange of information among multiple senders and multiple receivers, forming a multicast group. Examples of multipoint (or multicast) applications include audio and video conferencing, video on demand, distance learning, tele-metering, distributed games, server and replicated database synchronization, advertising, and data distribution applications. Multipoint support in ATM networks is essential for efficient duplication, synchronization and coherency of data. LAN emulation, Internet protocol (IP) multicasting, and overlay IP networks are extremely important multipoint protocols that must run efficiently on ATM networks. Many agencies and companies are already investing a lot on ATM-based high speed networks, especially in carrier backbones. It is important that such networks efficiently support the main protocols that will be used with them. Defining a flexible ATM multipoint capability is definitely a major step in that direction. In

particular, traffic management for ATM multipoint connections is an important and challenging problem.

In this research, we focus on the design of a traffic management framework for the ATM ABR service category to support both point-to-point (unicast) and multipoint connections. In particular, we address the problems of rate allocation and feedback consolidation and regulation. The goal is to design a network architecture and mechanisms to efficiently support multipoint communication. Many of the problems we tackle are not specific to ATM and apply to multipoint communication in computer networks in general.

In the next section, we further motivate the ATM multipoint traffic management problem. Then we discuss the scope, key contributions and organization of the dissertation.

1.1 Motivation

We first discuss the importance of traffic management and multipoint communication in general, and then we focus on the importance of ATM multipoint traffic management in particular.

1.1.1 Why Traffic Management?

Traffic management is key to high speed network design. The aim of traffic management is to control network congestion, efficiently utilize network resources, and deliver the negotiated quality of service to users [65]. This means that critical or real-time application traffic is given better service at network nodes than less critical traffic. Congestion should be controlled to avoid the performance degradation and

congestion collapse that occur when buffers overflow and packets are lost. The network load should not increase beyond a certain optimal operating point, known as the "knee" of the delay-throughput curves. This is the point beyond which increasing the load level on the network results in a dramatic increase in end-to-end delay, caused by network congestion and retransmissions. Congestion is a dynamic problem and static solutions are insufficient: congestion still occurs even with increase in buffers, bandwidth and processing power [75, 62, 61, 63, 64].

ATM networks provide a powerful framework for supporting various types of services. The end-to-end QoS guarantees for these services are clearly specified. Several traffic management components are provided to optimize resources and provide guarantees to the user. These include connection admission control (CAC), policing, shaping, scheduling, buffer management and feedback control. Some of these mechanisms operate at end systems (e.g., shaping at the source end system), some operate at network switches (e.g., switch buffer management), and some require both end systems and network switches to interoperate (e.g., feedback control). This dissertation focuses on the feedback control mechanism defined for the ABR service.

The key attractive features of ABR are that it (1) gives sources low cell loss guarantees, (2) minimizes queuing delay, (3) provides possibly non-zero minimum rate guarantees, (4) utilizes bandwidth and buffers efficiently, and (5) gives the contending sources fair shares of the available resources. The traffic management schemes we develop attempt to achieve these requirements for the ABR service. The objectives of the schemes include:

1. **Resource utilization:** The available network switch buffers and network link bandwidths should be efficiently utilized.

2. **Queuing delay:** Queuing delay should be small to guarantee low end-to-end delay, and to ensure buffers do not overflow and avoid cell loss.

3. **Fairness:** Connections should be treated fairly according to the fairness criteria we define in chapters 2 and 8.

4. **Stability and fast transient response:** The transmission rates of the sources should not unnecessarily fluctuate, and the scheme should be stable in steady state. It should also react rapidly to changing network conditions.

In addition to these criteria, the scheme should perform at an acceptable level even when there is no steady state (robustness) and be simple to implement.

1.1.2 Why Multipoint?

As previously mentioned, multipoint communication is the exchange of information among multiple parties. Table 1.1 shows a sample of the characteristics and requirements of some multipoint applications. As seen in the table, there is wide variation in group sizes, bandwidth requirements, and group dynamics among applications. Multipoint support can be performed at various layers, e.g., the application layer, the transport layer, the network layer or the data link layer. We believe that multicast support at the data link and network layers is essential for efficient multicast with minimal redundancy.

1.1.3 Why ATM?

ATM defines a powerful QoS and traffic management framework. Even though ATM has not penetrated local networks to the initially forecasted levels, ATM is

Application Type	Group Size	Bandwidth (bit/sec)	Real-time	Static/Dynamic Groups	Membership Changes
File system	Small	10 K-1 M	No	Static	Source
Distributed databases	Small	1-10 M	No	Static	Source
Tele-metering	Medium	<10 K	No	Static	Destination
Service location	Large	<100 K	No	Static	Destination
Audio conferencing	Small	4-64 K	Yes	Dynamic	Either
Video conferencing	Small	2-34 M	Yes	Dynamic	Either
Audio distribution	Large	1-2 M	No	Dynamic	Destination
Video distribution	Large	2-32 M	No	Dynamic	Destination
LAN emulation	Large	1-10 M	No	Static	Either

Table 1.1: Multicast application characteristics

already widely deployed in carrier backbones. A large percentage (80% by some estimates) of Internet traffic already passes through ATM networks. We believe that addressing traffic management and multipoint problems in ATM networks is important. ATM multipoint connections are used for efficient duplication and synchronization of data. In addition to multipoint applications such as conferencing and distance learning, ATM multipoint connections will be used to support IP multicast and LAN applications. ATM multipoint connections can be point-to-point, point-to-multipoint, multipoint-to-point or multipoint-to-multipoint.

Point-to-point ATM connections are being extensively used in Internet backbones today. ATM *point-to-multipoint* connections are very important for supporting LAN emulation and IP multicast over ATM networks. *Multipoint-to-point* connections are especially important for overlaying IP networks and simplifying end systems and edge devices [115]. In multipoint-to-point connections, only one connection needs to be set

up even if there are multiple data sources. *Multipoint-to-multipoint* connections provide a scalable solution for applications with multiple senders and multiple receivers to communicate. Since a single connection is shared by everyone, the connection management overhead does not increase proportional to the number of participating end systems.

There are two scenarios for ATM multipoint connections:

1. The ATM multipoint connection is end-to-end. This situation is depicted in figure 1.1. In this case, ATM hosts communicate though an ATM network.

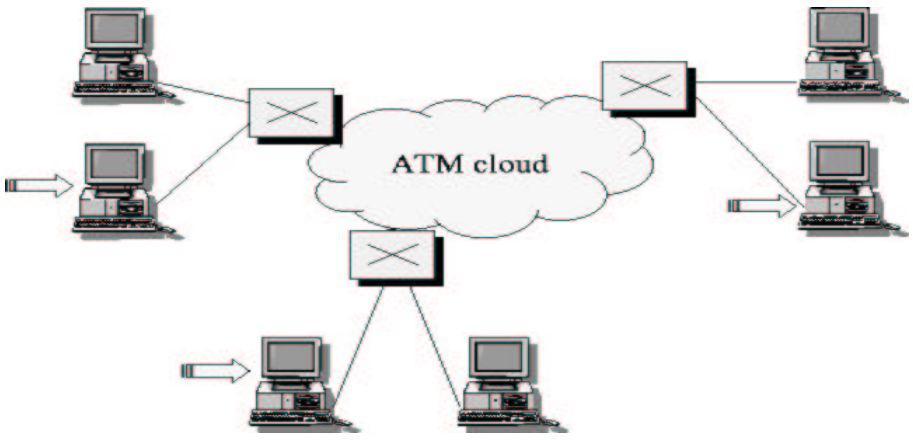


Figure 1.1: ATM multipoint communication

2. ATM forms a part of the multipoint connection. The Internet protocol suite is running on top of ATM, and interfaces to ATM in the ATM portions of the connection.

   Figure 1.2 shows multicasting in the Internet. The Internet Group Management Protocol (IGMP) is used to handle group membership requests. IGMP is the protocol through which hosts indicate to their local routers their interest in

joining a group. The Internet multicast routing protocols are employed in Internet routers to set up the multicast trees used for forwarding the data packets to the appropriate group members.

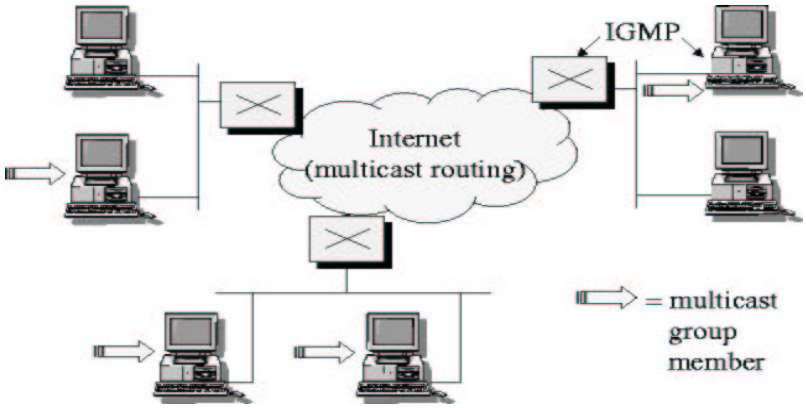


Figure 1.2: Multicasting in the Internet

Table 1.2 lists the key differences between the ATM and IP multipoint architectures, and why techniques developed for IP cannot directly be applied to ATM. Furthermore, the table shows why using ATM multipoint connections to support IP multicast is not straightforward. The main differences, as shown in the table, are the virtual connections set up by ATM, which enable ATM to provide quality of service (QoS) guarantees, but pose difficulties in cases of changing requirements, and merging cells. Care is exercised in order not to duplicate functionalities at the different layers: in cases when the lower layer provides a certain function, mapping techniques are employed to support this function at the higher layers.

Traffic management is an extremely difficult problem, and current solutions proposed for IP are themselves immature (the integrated services model suffers from

Category	IP/RSVP	ATM UNI 3.1
Connection type	Connectionless	Virtual connections
Cell ordering	Not guaranteed	Guaranteed
QoS	New services are being added to best effort	CBR, rt-VBR, nrt-VBR, ABR, UBR and more (GFR)
QoS setup time	Separate from route establishment	Concurrent with route establishment
Renegotiation	Allowed	Not allowed
Heterogeneity	Receiver heterogeneity	Uniform QoS to all receivers
Tree Orientation	Receiver-based	Sender-based (UNI 4.0 adds leaf-initiated join)
State	Soft (periodic renewal)	Hard
Directionality	Unidirectional	Unidirectional point-to-multipoint VCs
Tree construction	Different algorithms	Multicast servers or meshes

Table 1.2: IP/RSVP multicast versus ATM multipoint

scalability problems, while it is not clear what service and guarantees the differentiated services provide). As multimedia applications become more popular, the importance of traffic management and flow control will become paramount, as users send and receive video requiring real-time guarantees. Multipoint communication will also increase in popularity as video and data distribution to a large number of receivers becomes more common. We believe that research in the areas of traffic management and multipoint communication are of utmost importance.

As with label switching, resource reservation and signaling, and flow aggregation, we believe that traffic management is an area where the experiences gained with ATM can benefit the Internet protocol suite. The same problems exist, and the same general techniques can be employed. ATM multipoint can benefit from the

experiences gained in designing IP multicast, while IP traffic management can benefit from the experiences gained in defining the ATM service categories, and ATM traffic management techniques in general.

1.2 Scope of the Dissertation

Defining the set of services and protocol suite for ATM point-to-point and multipoint connections is a challenging task. Several issues need to be addressed in addition to traffic management, including multicast address allocation, signaling and group management, resource reservation, routing, providing reliable transport, and fault tolerance. We do not examine these issues in this dissertation.

This dissertation focuses on the feedback control issues in the point-to-point and multipoint ABR service. Little work has been done on multipoint extensions to ABR. The ATM Forum traffic management specification currently provides a few guidelines on traffic management of point-to-multipoint connections, but does not enforce nor suggest a specific strategy. For multipoint-to-point and multipoint-to-multipoint connections, traffic management design is still in its infancy. Chapter 3 gives a precise definition of the set of problems we are tackling, and our methodologies and approaches to solving these problems.

1.3 Main Contributions of this Work

The primary goal of this research is to design rate allocation and feedback consolidation and regulation schemes for unicast and multicast ATM ABR connections. ABR parameters and aggregation issues are also studied to provide a complete ABR framework.

The research should provide insight into network architectures, and into the quality of service achieved by different mechanisms. The schemes proposed here can be used with other traffic management components to deliver quality of service. The schemes are applicable to various networks supporting feedback control. The key contributions of this research include:

1. A sensitivity analysis for ABR parameters and the effect of link bandwidth and round trip times on their values

2. New ABR rate allocation schemes for ATM switches based on the ERICA+ algorithm

3. An architecture for aggregating Internet traffic over ATM using virtual paths

4. A framework for rate allocation of multiplexed virtual connections on a virtual path, and its application to ERICA+

5. A novel consolidation algorithm for point-to-multipoint ABR, and a comprehensive performance analysis and comparison with other algorithms

6. Fairness definitions and an algorithm and performance analysis for multipoint-to-point ABR rate allocation

1.4 Dissertation Outline

The rest of this dissertation is organized as follows. Chapter 2 discusses some background material necessary for understanding the problem of ATM point-to-point and multipoint traffic management. The chapter provides an overview of ATM traffic management and of multicast support in the Internet and in ATM. The chapter

contains a comprehensive survey of the state of the art in multicasting and traffic management in ATM networks, including a comparison of the use of ATM service categories for Internet traffic transport. The chapter also explains the ABR service and the ERICA+ algorithm, and gives precise definitions of the fairness goal.

Chapter 3 defines the problem tackled in this dissertation, and discusses our approach and methodology for its solution. Our approach breaks the problem into five parts discussed in the following five chapters.

Chapter 4 describes the role of ABR parameter values on network performance, and examines the recommended values for each. In addition, the sensitivity of parameter values to link bandwidths and round trip delay is examined.

Chapter 5 examines ABR rate allocation schemes and proposes a new method to determine activity levels of connections. The activity levels are used to compute the maximum allocation that can be given. Performance results show that the algorithm gives fair and efficient allocations.

Chapter 6 proposes the use of virtual paths to connect enterprise sites in virtual private networks. The key result is that ABR virtual paths can be used for both data and real-time traffic. Fairness issues in the multiplexing of ABR virtual connections on virtual paths are explored. An algorithm for feedback determination is proposed and analyzed.

Chapter 7 discusses ABR point-to-multipoint connection support, and proposes a novel algorithm for consolidating feedback information from the multicast tree branches. The algorithm is comprehensively analyzed and compared to previously proposed algorithms.

Chapter 8 discusses ABR multipoint-to-point connection support, and explores the notion of fairness in the case of multiple concurrent senders. An algorithm is proposed to allocate rates and regulate feedback to all the sources in the multipoint connection. Simulation results show that the algorithm performs well in a variety of situations.

Finally, chapter 9 concludes the dissertation with a summary of the results. The chapter also presents future directions for research in traffic management for broadband networks and multicast support.

CHAPTER 2

BACKGROUND AND SURVEY OF RELATED WORK

Internet traffic can be classified according to the application generating it, and its traffic characteristics. Applications may be real-time (voice or video) or non-real-time (data). Both the application type and the transport protocol affect the traffic characteristics. Unlike the user datagram protocol (UDP), the transmission control protocol (TCP) has built-in congestion avoidance mechanisms, which affect the traffic characteristics as seen at lower layers of the protocol stack.

In this chapter, we give a brief background and survey of related work in traffic management, with special emphasis on multipoint communication. We discuss ATM service categories, fairness definitions, ABR rate allocation schemes, multipoint communication, and finally multiple receiver and multiple sender support. We compare the ATM service categories in terms of cost, buffer requirements, and performance with Internet traffic. We conclude that a well-designed ABR service provides good performance for Internet traffic.

2.1 Overview of ATM Service Categories and their Applications

ATM networks currently provide five service categories [43]: constant bit rate (CBR), real-time variable bit rate (rt-VBR), non-real-time variable bit rate (nrt-VBR), unspecified bit rate (UBR), and available bit rate (ABR). The CBR and rt-VBR services are intended to transport real-time traffic, while the nrt-VBR, UBR and ABR services are designed for non-real-time traffic. In addition to these categories, the guaranteed frame rate (GFR) service is currently being standardized at the ATM forum traffic management working group [85]. The ITU-T I.371 also defines similar (but not the same) categories called ATM transfer capabilities.

Service categories relate traffic characteristics and QoS requirements to network behavior. Table 2.1 shows the attributes supported for each service category (this table is adapted from the ATM forum traffic management specifications [43]). The traffic parameters of service categories are the peak cell rate (PCR), cell delay variation tolerance (CDVT), sustainable cell rate (SCR), maximum burst size (MBS), maximum frame size (MFS) and minimum cell rate (MCR). These parameters define the characteristics of the traffic being transported. Three quality of service parameters define the service level that can be expected for the connection. The quality of service parameters are the peak-to-peak cell delay variation (peak-to-peak CDV), maximum cell transfer delay (maxCTD), and cell loss ratio (CLR).

The CBR service is used by connections requesting that a constant amount of bandwidth (characterized by a peak cell rate) be available throughout the connection lifetime. The source can transmit at or below the PCR for any length of time, and

	ATM Service Category					
Attribute	CBR	rt-VBR	nrt-VBR	UBR	ABR	GFR
Traffic Parameters						
PCR and CDVT	specified					
SCR and CDVT	n/a	specified		n/a		only CDVT for MCR
MBS	n/a	specified		n/a		specified for MCR
MFS	unspecified					specified
MCR	n/a				specified	
QoS Parameters						
peak-to-peak CDV	specified		unspecified			
maxCTD	specified		unspecified			
CLR	specified			unspecified	low for conforming	
Other Attributes						
Feedback	unspecified				specified	unspecified

n/a = not applicable

Table 2.1: ATM service categories: Parameters and attributes

the network assures the negotiated quality of service. Examples of applications that may use the CBR service are voice, video and circuit emulation applications requiring tight delay variation constraints.

The rt-VBR service is also intended for real-time applications requiring tight delay and delay variation constraints. Examples of such applications include voice with silence suppression, as well as emerging compressed video traffic. The difference between CBR and rt-VBR is that rt-VBR connections are characterized in terms of a sustainable cell rate and maximum burst size, in addition to the PCR. Thus, the source is expected to transmit at a variable rate. nrt-VBR connections are also characterized in terms of PCR, SCR and MBS. nrt-VBR, however, is intended for

non-real-time bursty traffic with no delay or delay variation bounds, but with a low cell loss ratio requirement (for conforming cells).

The UBR service is the simplest service. It is intended for traditional data traffic, such as file transfer and electronic mail. No delay or loss guarantees are provided; the service is a best effort service. No fairness or isolation of connections can be assumed. Like UBR, the ABR service is intended for data applications with no delay guarantees. ABR, however, attempts to minimize the cell loss ratio, and give minimum cell rate guarantees through a flow control mechanism. The network provides feedback to the sources when network load changes, and the sources adjust their transmission rates accordingly. ABR sources share the available bandwidth *fairly*, and the source is never required to send below its specified MCR.

As with UBR and ABR, the GFR service is intended to support non-real-time applications. The service is particularly targeted at users who are not able to specify all the traffic parameters needed to request services such as VBR, and are not equipped to comply with the end system behavior required by ABR. Although such users can currently request UBR connections, UBR provides no service guarantees. GFR guarantees a minimum rate and low cell loss ratio for conforming frames, while requiring little interaction between users and the network. The key attractive feature of GFR is its frame level visibility and guarantees, resulting in useful data being delivered.

2.2 Mechanisms for Providing Guarantees

This section gives more details on the mechanisms the end systems and network elements use to provide the guarantees for each service category.

2.2.1 CBR, rt-VBR and nrt-VBR

The three categories: CBR, rt-VBR and nrt-VBR provide open loop traffic control and preventive congestion avoidance. During connection admission control (CAC), reservations are made in the network nodes to meet the traffic contract and QoS commitments. The traffic contract can be met by the source end system if appropriate traffic shaping is performed. Alternatively, the network or the destination end system may enforce the contract with usage parameter control (UPC) functions. Traffic shaping and UPC can be performed using the generic cell rate algorithm (GCRA) which essentially uses leaky bucket mechanisms. For each cell arrival, GCRA determines whether the cell conforms to the traffic contract of the connection or not. Non-conforming cells may be tagged/marked (i.e., their cell loss priority (CLP) bit is set to one) or dropped. All tagged cells are dropped before any untagged cell is dropped (i.e., untagged cells have a higher priority). Two GCRA leaky buckets are needed to shape or control traffic according to the VBR parameters. CBR, rt-VBR and nrt-VBR provide complete isolation between connections: connections exceeding their traffic contract should not affect the QoS experienced by the other connections [43].

2.2.2 UBR

The basic UBR service has no explicit congestion control mechanisms. UBR signaling and parameters are minimal: only PCR is specified, and even that may not be subject to CAC or UPC procedures. Switches respond to congestion by dropping UBR cells when their buffers become full. Intelligent switch drop policies and end system policies can improve the performance of UBR with limited buffers. Intelligent

packet discard mechanisms are also applicable to all services which use ATM adaptation layer 5 (AAL5), such as VBR-nrt and ABR. Partial packet discard (PPD) and early packet discard (EPD) [108] have been shown to improve throughput. Per-VC accounting drop methods, and per-VC queuing and scheduling have been shown to improve both throughput and *fairness* [49]. A service using these mechanisms is usually referred to as UBR+. Throughout the remainder of this chapter, we mean simple vanilla UBR implemented in first generation switches when referring to UBR.

2.2.3 ABR

ABR allows the network to divide the available bandwidth fairly and efficiently among active sources. The ABR traffic management model is: (1) "rate-based" because the sources transmit at a specified "rate," rater than using a window; (2) "closed-loop" because, unlike CBR and VBR, there is continuous feedback of control information to the source throughout the connection lifetime; and (3) "end-to-end" because control cells travel from the source to the destination and back to the source [68].

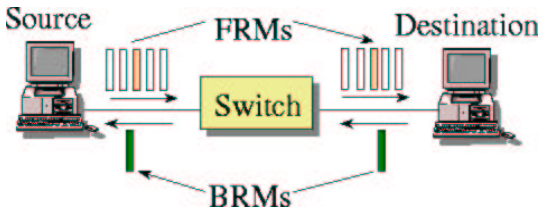


Figure 2.1: Forward and backward RM cells carry feedback information to the sources.

The components of the ABR traffic management framework are shown in figure 2.1. To obtain network feedback, the sources send resource management (RM)

cells after every $Nrm - 1$ (Nrm is a parameter with default value 32) data cells. Destinations simply return these RM cells back to the sources. The RM cells contain the source current cell rate (CCR), in addition to several fields that can be used by the network to provide feedback to the sources. These fields are: the explicit rate (ER), the congestion indication (CI) flag and the no increase (NI) flag. The ER field indicates the rate that the network can support for this connection at that particular instant. The ER field is initialized at the source to a rate no greater than the PCR, and the CI and NI flags are usually reset. Each switch on the path reduces the ER field to the maximum rate it can support, and sets CI or NI if necessary. The RM cells flowing from the source to the destination are called forward RM cells (FRMs) while those returning from the destination to the source are called backward RM cells (BRMs) (refer to figure 2.1). When a source receives a BRM cell, it computes its allowed cell rate (ACR) using its current ACR value, the CI and NI flags, and the ER field of the RM cell [7, 17, 38, 43, 65, 68]. Several other operations are performed by the end systems to ensure correct operation even under exceptional circumstances. Chapter 4 examines those operations and performs a sensitivity analysis for the ABR parameter values.

2.2.4 GFR

The GFR service requires user data to be divided into frames that can be delineated at the ATM layer. If the user sends frames not exceeding the maximum frame size (MFS) in a burst that does not exceed the maximum burst size (MBS), the user can expect its frames to be delivered with minimum losses. GFR also allows the user

to send in excess of the MCR (and the associated MBS), and delivers excess traffic if resources are available. Such resources should be shared "fairly" among users [85].

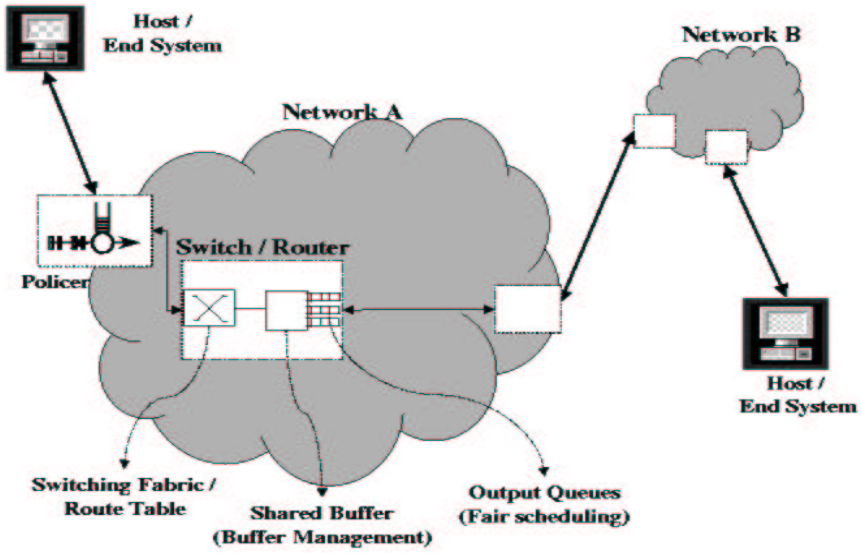


Figure 2.2: A network can use tagging, buffer management and scheduling to meet guarantees.

There are three design options that can be used by the *network* to provide the per-VC guarantees for GFR [48] (refer to figure 2.2): **(1) Tagging (Marking):** Network based tagging (or policing) can be used as a means of marking non-conforming frames. This requires some per-VC state information to be maintained by the network. Tagging can isolate the non-conforming traffic of each VC so that other rate enforcing mechanisms can schedule the conforming traffic in preference to non-conforming traffic. Policing can be used to discard non-conforming packets. **(2) Buffer management:** If multiple VCs share a common buffer space, per-VC buffer management can control the buffer occupancies of individual VCs. Per-VC buffer management

uses per-VC accounting to keep track of buffer occupancies [49]. **(3) Scheduling:** Scheduling and queuing strategies determine how packets are scheduled onto the next hop. First-in first-out (FIFO) queuing cannot isolate packets from various VCs. Per-VC queuing, on the other hand, maintains a separate queue for each VC in the buffer and can isolate VCs.

2.3 Cost/Performance Tradeoffs among ATM Service Categories

Tables 2.2 and 2.3 compare the six ATM service categories in terms of the complexity of connection admission and connection aggregation, the cost of end systems (network interface cards or NICs) and network elements (switches), and the buffer requirements and guarantees provided. As seen in the tables, UBR is the simplest service in terms of signaling, as it has a single parameter, PCR. Furthermore, end systems and network elements are simple as they are not required to perform any functions, though they may police on PCR, and UBR+ may provide intelligent drop policies (thus needing to recognize frame boundaries). One simple first-in first-out (FIFO) queue is adequate for vanilla UBR. UBR, however, gives no guarantees and requires significant switch buffering.

CBR and rt-VBR give the strictest guarantees, but they require the user to specify a number of parameters to exactly define the traffic contract and requirements. CBR requires the specification of the PCR and CDVT, in addition to the required QoS parameters, and it gives strict delay, delay variations and loss guarantees. The end systems and network elements are quite simple, since they only need to perform the functions required at connection admission control (provisioning), and perform policing functions. Very little buffering is needed at the switches. The problem with

CBR, however, is that it is wastes bandwidth if traffic is bursty, and most data traffic is bursty. CBR is bandwidth-inefficient because during connection admission, the network elements assume that the connection will always be sending at the peak rate, and resources must be reserved to satisfy the QoS requirements under such conditions. There is minimal statistical multiplexing gain.

rt-VBR gives the same strict guarantees as CBR, and does not suffer from the bandwidth inefficiency problem. However, applications are required to specify their traffic precisely in terms of a peak cell rate, a sustained cell rate, maximum burst size and tolerance values. Applications, however, rarely know their traffic characteristics precisely. Connection admission decisions, billing, and aggregation of VBR connections are also difficult. This is because the accumulation of values, such as the cell delay variation, is not straightforward (not additive, for example). End systems and network elements are slightly more complex than with CBR because policing is based on the sustained cell rate as well. nrt-VBR also gives loss and rate guarantees (though no delay guarantees) and provides isolation, but it also requires the user to define a specific traffic contract. Though QoS parameters and connection admission control decisions are simpler than with rt-VBR, they are non-trivial. Large buffers are required in network elements for nrt-VBR to reduce cell loss.

ABR minimizes cell loss for well-behaved connections and can give minimum rate guarantees, in addition to isolation and fairness, but the end system and switches need to perform complex functions. A large number of parameters are signaled, though the setting of these parameters is well understood. The connection admission is simply based on MCR, but it is slightly more complex than CBR because an overbooking factor must be determined. ABR does provide superior traffic management

	Complexity of				Overall
Category	Connection Admission	Connection Aggregation	End System	Network Element	**Complexity**
CBR	Low+	Low	Low+	Low	Low
rt-VBR	High	High	Medium	Medium	Medium
nrt-VBR	High	High	Medium	Medium	Medium
UBR	Low	Low+	Low	Low	Low
ABR	Medium	Medium	High	High	High
GFR	Medium	Medium (frame size)	Medium+	Medium+	Medium+

Table 2.2: Cost/Complexity of ATM service categories

though, since it maximizes buffer and bandwidth utilization (because the end systems precisely know the network state). Switch buffer requirements are small, and extra buffering at the end systems or routers can be utilized. No other service category provides end systems with network state information. Little buffering is required at routers or end systems if Internet transmission control protocol (TCP) acknowledgment regulation schemes are used to convey the ABR rate information to the TCP flow control mechanism.

GFR gives similar guarantees to ABR, but reduces the signaling and end system complexity. A minimum cell rate value is negotiated and the user may request tagging. End systems only need to provide tagging and policing functions. Network elements, however, provide the required minimum rate guarantees and fairness among connections. In addition to tagging, intelligent buffer allocation and scheduling may need to be performed. Buffer requirements at the switches may be high [36].

	Buffer Requirements		Guarantees			
Category	Network Element	End System (or Router)	Fairness and Isolation	Delay	Loss	Rate
CBR	Very Low	Depends on traffic	Yes, but no full multiplexing	Yes	Yes	Yes
rt-VBR	Medium	Low	Yes	Yes	Yes	Yes
nrt-VBR	High	Low	Yes	No	Yes	Yes
UBR	High	Low	No	No	No	No
ABR	Low	High (except when acknowledgment regulation for TCP is used)	Yes	No	Yes	Yes
GFR	High	Low	Yes	No	Yes (frame-based)	Yes

Table 2.3: Performance of ATM service categories

2.4 Performance of TCP over ATM

In this section, we compare the performance of TCP over different ATM service categories. The bursty nature of TCP traffic makes transporting it over CBR a poor design choice, as TCP traffic rarely requires delay guarantees, and the statistical multiplexing benefits of ATM are not fully utilized. Before we explore the transport of TCP over ATM, we will first discuss the TCP congestion avoidance mechanism and how it affects TCP traffic patterns as seen at the ATM layer.

2.4.1 TCP Congestion Avoidance

TCP is the most widely used transport protocol. It provides reliable transfer of data using a window-based flow and error control algorithm. The key TCP congestion

control mechanism is the TCP slow start [59]. TCP connections use a window to limit the number of packets that the source can send. The sender window is computed as 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 (ack) 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 size, and then CWND is set to one. The source retransmits the lost packet and increases its congestion window by one every time a packet is acknowledged. This continues until the window reaches SSTHRESH. After that, the window w is increased by $1/w$ for every packet that is acked. The source window is always limited by the receiver window size. The typical changes in the source window plotted against time are illustrated by figure 2.3.

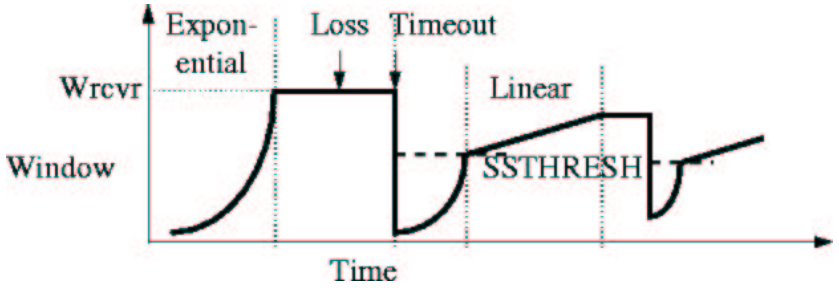


Figure 2.3: The TCP slow start and congestion avoidance mechanism manages traffic by controlling the growth of the TCP window.

2.4.2 TCP over UBR

The UBR service depends upon the transport layer to provide congestion and flow control functions. A single cell drop at the ATM layer results in an entire packet drop at the destination. The source TCP times out and retransmits the lost packet. Low throughput and unfairness result from the time lost in the timeouts and retransmissions of packets. A TCP source stops increasing its transmission rate only when its congestion window reaches a maximum value. *TCP using basic vanilla UBR requires network buffers approaching the sum of the maximum window sizes of all TCP connections to completely avoid cell loss.* Thus, vanilla UBR does not scale well in this sense. With limited buffering, however, TCP throughput and fairness can be improved using UBR+ [81] by: **(1) Drop policies** decide when to drop cells. PPD and EPD [108] drop full packets instead of random cells from multiple packets. **(2) Buffer allocation policies** decide how to divide the available buffer space among the cells from contending connections. Fair buffer allocation (FBA) schemes [49] improve fairness by selectively discarding frames from flows that are sending more than their fair share. **(3) Scheduling policies** divide the available bandwidth among contending queues. Scheduling may be implemented at a coarse granularity to divide bandwidth among service categories, or at a fine granularity to divide bandwidth among connections within a service category.

2.4.3 TCP over ABR

For TCP over ABR, the TCP window-based control is running on top of the ABR rate-based control. A steady flow of RM cells results in a steady flow of feedback from the network. In this state, the ABR control loop has been established, and source

rates are primarily controlled by the network feedback (closed-loop control). 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). When the traffic is bursty, open-loop control may be exercised at the beginning of every active period.

ABR switch algorithms allocate high rates to ABR sources if insufficient load is experienced at the switches. This is likely to be the case when a new TCP connection starts data transmission. The connection is bottlenecked by the TCP congestion window size and not by the ABR source rate. The TCP active periods double every round trip time and eventually load the switches and appear as persistent traffic at the ATM layer. The switches ask sources to reduce their rates, and data is bottlenecked by the ABR source rate, not by the TCP congestion window size. Once the ABR rates converge to optimal values, the lengths of the ABR queues at the switches decrease.

Therefore, ABR flow control pushes the queues from the network to the end systems. In [81], we show that ABR is scalable for persistent applications running over TCP/IP (such as long file transfers) in the sense that, given the right implementation and parameters, its network buffer requirements for *zero* packet loss do *not* grow linearly with the number of TCP connections. If buffers overflow, smaller TCP timer granularity (which controls timeout durations) can help improve throughput.

Sample Simulation Results

As a sample result, we show the throughput and maximum queue length obtained for a simple 15 source configuration, as shown in figure 2.4 (with $n = 15$). The

configuration has a single bottleneck link shared by 15 ABR sources. All links run at 155.52 Mbps and are of the same length. We experiment with various link lengths.

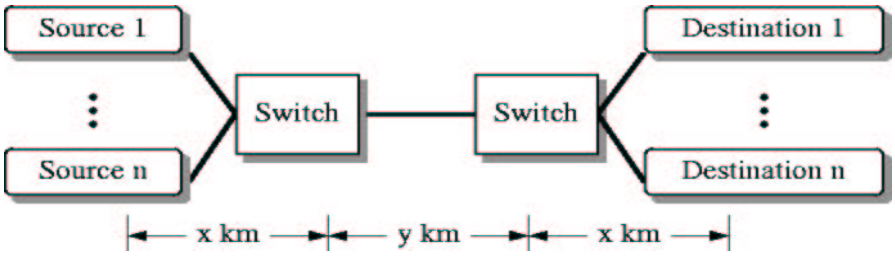


Figure 2.4: The n source configuration is a simple configuration with n sources sending to n destinations using the same bottleneck link.

All traffic is unidirectional. A large (infinite) file transfer application runs on top of TCP. The link lengths are 1000 km×3 links, 500×3, 200×3, 50×3 (for round trip times (RTTs) of 30, 15, 6 and 1.5 ms respectively). We have verified that maximum queue bounds also apply to configurations with heterogeneous link lengths, multiple bottlenecks and with VBR traffic in the background causing variance in ABR capacity and errors in measurement. We use a TCP maximum segment size (MSS) of 512 bytes. The window scaling option is used so that the throughput is not limited by path length. The TCP window is set to 16 × 64 kB = 1024 kB. 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.

Table 2.4 shows that the worst case maximum queue is less than 3 × RTT × link bandwidth, even with transient bursts. Therefore, ABR is scalable because the maximum queue size is a small multiple of the RTT, and does not grow linearly with

the number of connections. ABR buffer requirements are also much smaller (than the values shown) after the system reaches steady state.

Number of Sources	RTT (ms)	Feedback Delay (ms)	Max Queue Size (cells)	Throughput
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 2.4: Maximum queue size and throughput for TCP over ABR (transient bursts)

2.4.4 TCP over GFR

Edge devices can use GFR to transport multiple TCP connections over a single GFR connection. The bursty nature of TCP traffic makes it difficult to provide per-connection GFR rate guarantees using FIFO queuing. *Per-VC queuing* and scheduling are recommended to provide rate guarantees to TCP connections when GFR VCCs are fully using the buffers. Good TCP performance has been observed in such cases. Under conditions of low buffer allocation, however, it is possible to control TCP rates, even with FIFO queuing, by manipulating the TCP congestion window through setting buffer thresholds to drop packets. This assumes that in cases where the offered load is low, a queue is not built up and TCP is allowed to use as much capacity as it can. The average throughput achieved by a connection is proportional to the fraction of the buffer occupancy used by the cells of that connection. As long as the fraction of buffer occupancy of TCP can be controlled, its relative throughput depends primarily on the fraction of packets of that TCP in the buffer. At a very high

buffer utilization, packets may be dropped due to buffer unavailability. This results in larger variations in TCP throughputs. At very high thresholds, the queuing delay also increases significantly, and may cause the TCP sources to time out [48].

2.4.5 TCP over VBR

TCP can be transported over the VBR service, given that the user selects an appropriate PCR, SCR, MBS and CDVT values. In [55], experiments are performed to evaluate the performance of TCP over VBR. TCP over VBR performed well except with 100% utilization and when MBS was set to extremely small values. The reason behind the poor performance of VBR for unreasonably small MBS is that a large number of the frames are partially tagged. This causes the corruption of frames, although they consumed tokens from the GCRA (leaky bucket). If the sum of reservations is only 50% of the link rate, VBR showed good fairness than in the full utilization case, even with small MBS. The reason for this is that if the bucket size is large, frame boundaries make no difference: in a large burst, only the one frame will be partially tagged [55]. As a rule of thumb, MBS must be greater than the maximum frame size (preferably greater than the maximum TCP window of 64 kbytes plus overhead). Most carriers set MBS in the 9 kbytes to 64 kbytes range.

2.4.6 Comparison of TCP Performance over UBR, ABR, GFR and VBR

Table 2.5 summarizes the previous discussion on the performance of TCP over the four service categories: VBR, UBR, ABR and GFR. Vanilla UBR performs poorly unless buffers are large or intelligent drop policies (UBR+) are employed. ABR pushes the queues to the ATM network edges and, in that sense, it is scalable because the

network queue sizes are not a function of the number of connections, but of the round trip times, feedback delays and switch congestion avoidance scheme employed. GFR exhibits good efficiency and fairness with intelligent drop or tagging, and/or per connection queuing. VBR performs quite well except in cases where utilization is high and MBS is too small, because incomplete frames are tagged.

Since TCP losses result in long idle times waiting for a timeout and performing slow start, high utilization is directly linked to low packet loss. ABR provides control over queue length, and hence the low loss and high bandwidth utilization. In end-to-end ATM, ABR and CBR minimize losses. GFR and UBR can use fair buffer allocation or per-VC queuing to fairly distribute losses among the VCCs, since each TCP flow will most likely be carried on a separate VC. This is not the case for an ATM backbone situation, where each VC will carry multiple TCP flows and VCCs are only used between edge routers. In this case, most ABR and CBR losses are in the routers and not the ATM switches, so schemes such as Random Early Detection (RED) gateways [41] (or flow RED gateways) that perform selective drop at the routers can provide fairness for TCP over ABR. With VBR, UBR and GFR, most losses occur in the ATM switches and not at the edge routers. Fair buffer allocation in the ATM switches can ensure fairness among the VCCs, but not among the flows multiplexed on the same VCC. Hence, our results show that ABR is most suitable for bursty TCP traffic, followed by GFR, then VBR, then UBR and finally CBR.

2.5 Performance of UDP over ATM

The User Datagram Protocol (UDP) has no built-in flow control mechanism like the TCP slow start and congestion avoidance mechanisms. Therefore, losses may

Category	Performance of TCP
UBR	Low efficiency and fairness, especially without intelligent drop and with FIFO queuing and scheduling, unless buffer approaches sum of receiver windows.
GFR	Very good performance with intelligent drop and tagging (better than VBR because of frame visibility).
VBR	Good (except with high utilization and very small MBS values).
CBR	Unsuitable for bursty TCP traffic.
ABR	Pushes queues to ATM network edges and provides high utilization and fairness. Network queues only depend on round trip time, feedback delay and switch scheme.

Table 2.5: Performance of TCP over ATM service categories

continue and have more effect than in the case of TCP. Several client-server transaction applications use UDP. An example of such servers is authentication servers used for security. Such applications handle retransmission if necessary. In addition to loss-sensitive data traffic, UDP is also used to transport loss-tolerant traffic, such as voice over IP. Loss-tolerant applications are usually delay-sensitive applications, for example, voice applications have no use for the packets after a certain time delay, and thus can tolerate its a moderate amount of loss.

As with TCP, CBR is not ideally suited to UDP traffic which is generally bursty. VBR, UBR and GFR can make use of the drop priority to drop lower priority packets before dropping higher priority ones. However, since these categories have no information on the network state, cells may be dropped inside the ATM network. ABR, on the other hand, provides low cell and packet loss rates inside the ATM network, and most drops occur at the edge routers where the ABR queues may grow. If data is hierarchically coded and drop preference is indicated, these routers may be able to drop lower priority information before dropping the higher priority information.

2.6 Fairness Definitions

The optimal operation of a distributed shared resource is usually given by a criterion called the *max-min allocation* [60]. This fairness definition is the most commonly accepted one, though other definitions are also possible.

Definition: A component c_j is said to be *downstream* of another component c_i in a certain connection if c_j is on the path from c_i to the destination. In this case, c_i is said to be *upstream* of c_j. □

Definition: Given a configuration (sources, destinations, switches, links, connections) with n contending sources, suppose the i^{th} source is allocated a bandwidth x_i. The allocation vector:

$$\{x_1, x_2, \ldots, x_n\}$$

is *feasible* if all link load levels are less than or equal to 100%. □

Definition: Max-min allocation: The max-min allocation vector is a feasible vector where the allocation of the sources with the minimum allocation is maximized. Given an allocation vector $\{x_1, x_2, \ldots, x_n\}$, the source that is getting the least allocation is, in some sense, the "unhappiest source." To achieve max-min allocations, find the feasible vectors that give the maximum allocation to this unhappiest source. Now remove this "unhappiest source" and reduce the problem to that of the remaining $n - 1$ sources operating on a network with reduced link capacities. Again, find the unhappiest source among these $n - 1$ sources, give that source the maximum allocation, and reduce the problem by one source. Repeat this process until all sources have been allocated the maximum that they can get. □

Intuitively, this means that all sources bottlenecked on the same link get equal rates, and if a source cannot utilize its fair share, the left over capacity is shared fairly among those who can use it. An extension of this definition guarantees a minimum cell rate (MCR) for each source, and shares the left-over capacity in a weighted manner. This is called the general weighted fair allocation [128].

Definition: General weighted fair allocation: Given a weight vector

$$\{w_1, w_2, \ldots, w_n\}$$

that denotes the weight to be given to each source switched to a certain output port, and an MCR vector $\{MCR_1, MCR_2, \ldots, MCR_n\}$ denoting the minimum cell rate for each source switched to that port, the allocation for each source is denoted by:

$$x_i = MCR_i + \frac{w_i \times (\text{Capacity} - \sum_{i=1}^{n} MCR_i)}{\sum_{j=1}^{n} w_j}$$

□

We extend these definitions for aggregate connections in chapter 6, point-to-multipoint connections in chapter 7 and multipoint-to-point connections in chapter 8.

2.7 Rate Allocation Schemes for ATM-ABR

Several switch algorithms have been developed to compute the feedback to be indicated to ABR sources in RM cells [1, 3, 15, 46, 74, 73, 79, 112, 124]. Two excellent surveys of these schemes are provided in references [3, 80]. The standard "explicit rate indication for congestion avoidance+" (ERICA+) algorithm [66, 73, 69] is one of these algorithms. We refer to it as ERICA+ in this dissertation. The main advantages of ERICA+ are its low complexity, fast transient response, high efficiency, and small

queuing delay. This section gives a brief overview of the ERICA+ algorithm. For a more complete description of the algorithm and its performance, refer to [73].

The ERICA+ algorithm works at every ABR queuing point. Time is divided into consecutive equal-sized slots called *"switch averaging intervals."* The measured load in the forward direction in each slot is used to provide feedback in the reverse direction in the next slot. The feedback may be computed at the end of each slot or when a BRM is received. ERICA+ switches give *at most one* feedback value per flow during any averaging interval. This precludes the switch from giving multiple conflicting feedback indications in a single averaging interval using the same control values.

ERICA+ periodically monitors the load on each link and determines the ABR capacity, the load factor (z), and the number of active virtual connections (N) during each "averaging interval." Below we present the key steps in ERICA+ as pseudo-code. The variable *MaxAllocPrevious* represents the maximum allocation given during the previous averaging interval to any source transmitting to this output link. Similarly, *MaxAllocCurrent* is used to determine the maximum allocation given to any source so far in the current averaging interval.

Initialization:

MaxAllocPrevious $\leftarrow$ MaxAllocCurrent $\leftarrow$ FairShare

End of Averaging Interval:

$$\text{Total ABR Capacity} \leftarrow \text{Link Capacity} - \text{CBR/VBR Capacity} \quad (2.1)$$

$$\text{Target ABR Capacity} \leftarrow \textit{Fraction} \times \text{Total ABR Capacity} \quad (2.2)$$

$$z \leftarrow \frac{\text{ABR Input Rate}}{\text{Target ABR Capacity}} \quad (2.3)$$

$$\text{FairShare} \leftarrow \frac{\text{Target ABR Capacity}}{\text{Number of Active VCs}} \tag{2.4}$$

$$\text{MaxAllocPrevious} \leftarrow \text{MaxAllocCurrent} \tag{2.5}$$

$$\text{MaxAllocCurrent} \leftarrow \text{FairShare} \tag{2.6}$$

When an FRM is received:

CCR[VC] $\leftarrow$ CCR_in_RM_Cell

When a BRM is received:

$$\text{VCShare} \leftarrow \frac{CCR[VC]}{z} \tag{2.7}$$

IF $(z > 1 + \delta)$

$$\text{THEN ER} \leftarrow \text{Max (FairShare, VCShare)} \tag{2.8}$$

$$\text{ELSE ER} \leftarrow \text{Max (MaxAllocPrevious, VCShare)} \tag{2.9}$$

$$\text{MaxAllocCurrent} \leftarrow \text{Max (MaxAllocCurrent,ER)} \tag{2.10}$$

IF (ER > FairShare AND CCR[VC] < FairShare)

$$\text{THEN ER} \leftarrow \text{FairShare} \tag{2.11}$$

$$\text{ER\_in\_RM} \leftarrow \text{Min (ER\_in\_RM, ER, Target ABR Capacity)} \tag{2.12}$$

The target ABR capacity is a fraction of the total ABR capacity (equation (2.2)), where the fraction may be determined based upon queuing delays. The function $f(Q)$ or the "queue control function" allows only a specified fraction of the available capacity to be allocated to the sources. The remaining capacity is used to drain the current queues.

The value of $f(Q)$ depends on the current switch queuing delay. Figure 2.5 shows an example of a queue control function. A target queuing delay, Q_0, is specified, and the function is an inverse hyperbolic function for queuing delays larger than

the specified value ($Q > Q_0$). The function, however, does not decrease beyond a minimum value, called the queuing delay limit factor (F_{min} in the figure). For more details on the performance of different queue control functions, refer to [130].

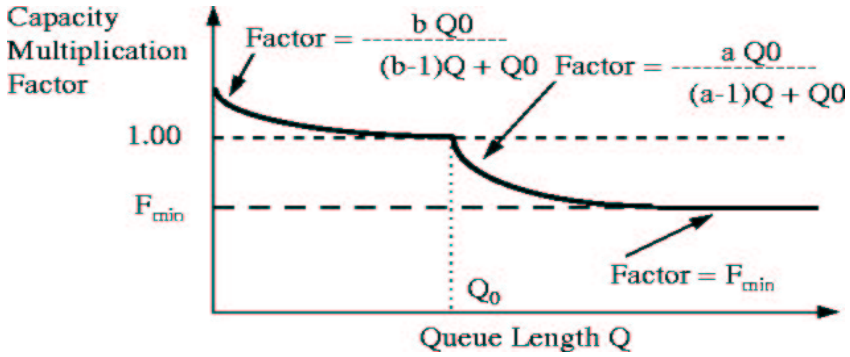


Figure 2.5: The inverse hyperbolic function can be used for queue control.

The key metric used in ERICA+ is the load factor (z) which is the ratio of the measured input rate at the port to the target ABR capacity, as given by equation (2.3). The aim of z is to drive the system towards an efficient operating point, defined as the neighborhood of $z = 1$. The simplest way to achieve efficiency is to reduce each VC's activity by a factor of z. In other words, each VC's allocation ("VCShare" in the pseudo-code above) is set to the VC current cell rate (CCR) divided by the load factor z, or $\frac{CCR[VC]}{z}$. Here, CCR is the estimate of the source current rate. CCR may be read from the forward RM cells of the VC, or measured by the switch. Either way, the CCR value is stored in a table and used for this calculation.

Though VCShare can be used to achieve efficiency, it may not be a fair allocation. A mechanism is required to equalize the rate allocations, while ensuring that the bottleneck load factor remains in the neighborhood of unity. ERICA+ uses a two step

process. A "FairShare" is computed based upon the target ABR capacity and the number of active connections. All sources sending below the FairShare are allowed to rise to FairShare, and those sending above FairShare are allowed to rise to MaxAllocPrevious, which is the maximum allocation given to any VC in the previous switch averaging interval. These features and mechanisms (VCShare, rate equalization, at least FairShare, at most FairShare if rate is low) are incorporated into the ERICA+ algorithm as presented in equations (2.7) through (2.11). The parameter δ is used for the equalization of allocations (equation (2.9)) and defines the "neighborhood of unity."

Chapter 5 shows alternative methods for computing the optimal max-min rate allocations.

2.8 ATM Virtual Paths

One of the key distinguishing design aspects of ATM networks is the use of labels for switching. The use of labels speeds up the switching functions, and improves scalability since the labels need not be globally unique. This technique has now been adopted into the Internet in the form of multiprotocol label switching (MPLS).

An interesting feature of label usage in ATM is the aggregation mechanism defined by the two level hierarchy of virtual path connections (VPCs) and virtual channel connections (VCCs). VPCs provide an elegant method for combining several VCCs between two end points. This technique is essential for scalability in backbone networks where there is a large number of flows. Using VPCs in the backbone reduces complexity, improves utilization, and lowers cost. The mechanisms to perform traffic

management for aggregate flows are currently being debated at the differentiated services working group at the Internet Engineering Task Force (IETF). The scalability of the future Internet that combines real time and non real time traffic is affected by the outcome of this work. Chapter 6 explores the use of virtual paths for traffic aggregation and proposes a framework for multiplexing VCCs on VPCs.

2.9 Multipoint Communication

Multipoint support can be performed at different layers of the protocol stack, including the data link layer, network layer, transport layer, or application layer. However, without proper network support, multipoint communication can be extremely inefficient. For example, suppose multipoint communication is only supported at the application layer. In this case, data is unnecessarily duplicated on the path from the sender to each of the receivers. Clearly, duplication at the branch points would be more efficient, i.e., multicasting is better handled by lower layers of the protocol stack.

For a protocol to support multipoint communication, it needs to establish methods for carrying out each of the following steps:

1. Define group address format.
2. Allocate group addresses.
3. Dynamically define group memberships.
4. Determine the routes.
5. Forward the packets.

6. Recover from faults.

There are subtle differences between the terms *"multicast," "multipoint,"* and *"multiway."* In IP terminology, the communication between a group of participants is referred to as "multicasting." The ATM Forum has adopted the term "multiway" since "multicast" can be confused with *point-to-multipoint* communication. Some researchers argue that since the term "multicast" was originally a restriction of "broadcast" on a broadcast medium, it is not appropriate for use when referring to non-broadcast situations, such as the ATM networks. Either of the terms "multipoint" or "multiway" should be used to avoid this confusion. "Multipoint" is simply a generalization of "point-to-point" to indicate multiple parties communicating with each other, and "multiway" is a similar concept that further emphasizes that multiple parties can be communicating at the same time.

In this dissertation, we mostly use the term "multipoint," but we use the two terms "multipoint" and "multicast" interchangeably at times. We use the term "point-to-multipoint" to specifically indicate *one* sender sending to a group of receivers.

2.9.1 Internet Multicasting

Multicasting in the Internet was defined in the late 1980s by specifying multicast extensions to IP hosts, as well as the behavior of the multicast routers. IP multicast was designed to enable a sender to send to multiple receivers by specifying a *single* group address in the IP destination field. Members of a multicast group can be located anywhere, and can join and leave the group at will. To improve scalability, the sender need not be aware of the group membership, and need not itself be a member of the group. IP multicast is also fault tolerant because all memberships are

periodically renewed (what is known as "soft state"). IP needs no special solution for multipoint-to-multipoint communication.

IP multicast is implemented as: (1) the Internet group management protocol (IGMP) at the hosts, and (2) one of the Internet multicast routing protocols (such as the distance vector multicast routing protocol) in multicast routers. Multicast routers ("mrouters") handle the forwarding of datagrams and the propagation of routing information. This subsection briefly overviews IP multicasting.

Internet Group Management Protocol (IGMP)

IGMP allows IP hosts to join and leave multicast groups. As previously mentioned, the membership of multicast groups is dynamic, and there is no restriction on the location or number of members in a group. A host can be a member of any number of groups. Host groups can be permanent or transient.

Multicast IP addresses start with the reserved 4-bit sequence 1110 (class D IP addresses), and the rest of the address (the remaining 28 bits) indicates the multicast group number. (IP version 6 defines more sophisticated group addressing.)

Several extensions to IP, the IP interface, and the network interface are implemented to support IP multicast. The underlying Ethernet (or local net) multicast is used, and IP multicast addresses map to the Ethernet multicast address space. The routines "JoinHostGroup" and "LeaveHostGroup" are specified at both the IP and the network interfaces. IGMP provides messages used to query hosts about their group memberships. Only one host per subnet needs to reply. The queries are periodically broadcast to the net. A random timer is used to prevent collisions. Hosts only need to inform routers of join requests, and not leave requests [22].

IGMP Version 2 [39] extends the basic IGMP protocol by adding a procedure for the election of the query-broadcasting router for each local area network, explicit leave messages for faster pruning, and group-specific query messages. IGMP Version 3 enables a host to specify a set of hosts in a multicast group from which it wants to receive multicast messages.

IP Multicast Routing Protocols

A number of algorithms have been proposed for building trees to be used for multicast data forwarding. All multicast routing protocols employ one or more of the following techniques: flooding, spanning trees, reverse path forwarding, pruning, Steiner trees and center-based trees.

Multipoint-to-multipoint IP multicast can use two approaches for data distribution, namely: the shared tree approach (sometimes called sparse mode), and the (per) source-based tree (or dense mode) approach. The shared tree approach uses a common multicast tree that is shared by all sources (senders), whereas the source-based (or sender-specific) tree approach requires each source to maintain its own multicast tree. Core based trees (CBTs), and protocol-independent multicast (PIM)-sparse mode (SM) are examples of the shared tree approach, while the distance vector based multicast routing protocol (DVMRP), multicast extensions to open shortest path first (MOSPF), and protocol-independent multicast (PIM)-dense mode (DM) PIM are examples of the source-based tree approach. Refer to [57] for a detailed tutorial on IP multicast routing, and all the IP multicast routing algorithms.

The core-based tree (which is a shared tree idea) is one of the most popular approaches. This is because it is relatively simple to implement. The approach is also suitable for ATM networks. RFC 1458 [9] gives more information on the

requirements for multicast protocols, including routing protocols, and group address and membership authority.

2.9.2 ATM Multicasting

ATM multipoint communication is being studied at the ATM Forum multiway BOF (birds of a feather) and at the International Telecommunications Union (ITU) study groups 11 and 13 [76]. The Internet Engineering Task Force (IETF) has also studied the mapping of IP multicasting to ATM networks, especially in the Internetworking Over Non-broadcast multiple access (ION), and integrated services over specific link layers (ISSLL) working groups.

Unlike IP multicasting, ATM multicasting is still in its earlier phases of definition. The ATM user to network interface (UNI) signaling currently supports multipoint communication via *point-to-multipoint VCs* only. The ATM UNI 3.1 signaling standard supports the source-based tree approach for multicast data distribution, using root-initiated joins for multicast tree construction. This means that only the root can setup the point-to-multipoint connection, add leaves, and send ATM cells. Since receiver or leaf initiated join (LIJ) is a more scalable approach, UNI 4.0 signaling supports such joins [111]. The ATM private network to network interface (PNNI) 1.0 does not define routing for multipoint connections. The second phase of PNNI will define routing for UNI 4.0 multipoint connections. However, pure multipoint-to-point and multipoint-to-multipoint services are not yet supported by signaling or routing.

The Multicast Address Resolution Server (MARS) architecture [2] uses the point-to-point and point-to-multipoint VCs supported by UNI 3.1 signaling to forward packets within a cluster, and uses multicast routers to go outside a cluster. Thus,

sources or servers need to know which receivers are listening to which multicast group. This incurs state management overhead, leading to scalability problems for large multicast groups.

A number of proposals have attempted to tackle some of the problems that the MARS and multicast server (MCS) proposals have attacked without requiring the use of dedicated servers. Such approaches use multipoint-to-multipoint VCs to achieve better scalability and reduce the signaling load. In multipoint-to-multipoint VCs, multipoint communication is not implemented as a set of sender-specific trees, rather it is implemented as a shared tree. The existence of multiple senders in the same VC, however, introduces a number of problems, which we discuss next.

Cell Interleaving Solutions

In ATM networks, the virtual path identifier (VPI)/virtual connection identifier (VCI) fields in the cell header are used to switch ATM cells. The ATM adaptation layer (AAL) at the sender segments packets into ATM cells, marking the last cell of each packet. The AAL at the receiver uses the VPI/VCI fields and the end of packet marker to reassemble the data from the cells received.

ATM adaptation layer 5 (AAL5), used for most data traffic, does not introduce any multiplexing identifier or sequence number in ATM cells. If cells from different senders are merged and interleaved on the links in a multipoint connection (implemented as a shared tree), the AAL5 at the receiver cannot assemble the data. This is because all traffic within the group uses the same VPI/VCI. The AAL5 uses the end-of-message bit to determine the end of each packet, but since the cells of different packets are interleaved, these packets may get corrupted, as illustrated in figure 2.6.

The identity of the sender is not indicated in each cell. Hence, alternate solutions must be implemented.

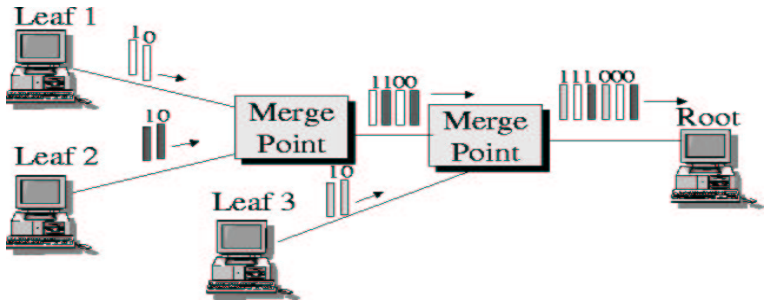


Figure 2.6: The cell interleaving problem prevents correct packet reassembly

Solutions to the interleaving problem either: (a) avoid merging entirely, e.g., by using different connections, or (b) prevent interleaving of cells of packets originating from different sources in the same multipoint connection at merge points, or (c) provide enough information in the cell headers to enable the receivers to reassemble the packets even if their cells are interleaved.

The solutions proposed to the cell interleaving problem include:

1. **AAL3/4:** AAL3/4 can be used instead of AAL5. AAL3/4 contains a 10-bit multiplexing identifier (MID) field, part of which can be used to distinguish the senders in the multipoint VC. This can make switching fast and connection management simple. However, AAL3/4 suffers from excessive overhead and is not well supported. (An alternative AAL, AAL5+, was proposed in [88].)

2. **VC mesh:** Point-to-multipoint VCs can be overlaid, forming a VC mesh [2]. In this case, cells from different senders can be differentiated based on their

VPI/VCI fields. This solution does not scale well as it requires N point-to-multipoint VCs for N senders.

3. **Multicast servers (MCSs):** In this case, all senders send to the MCS, which forwards data on a point-to-multipoint VC [116]. This approach is simple. The problem with it is that it is inefficient, and the MCS needs large amounts of buffering. In addition, the MCS can become a single point of congestion, making it difficult to guarantee quality of service requirements.

4. **Election:** Election can be used to coordinate senders. In this approach, a sender must acquire a control message (token) before it can transmit data, and there is only one token for each VC. Hence, only one sender can transmit at a time, and no cell interleaving can occur. This approach is used in the SMART scheme [45]. Although this mechanism is feasible, the overhead and delay of the scheme are high.

5. **VP merge:** This approach uses multipoint virtual paths (VPs). Only the VPI field is used for switching cells of a multipoint connection, and the VCI field is used to uniquely identify the sender. Connection management is simple in this case, but the approach requires receivers to have a global assignment of VCs within VPs. The main problem, however, is that VPs should not be used by end-systems, since network providers use VPs for aggregation in the backbone. Finally, there are only $2^{12} = 4096$ unique VPI values possible at each hop, and hence it is possible to run out of VPI values.

6. **Variable VP merge:** Different VPI field sizes are used in this approach [132]. The switches support both 12-bit VPI fields, as well as 18-bit VPI fields. Distributed schemes to assign globally unique VCIs within each VP are proposed using collision avoidance. This approach overcomes the VP scarcity problem of VP merge, but still has the problem of using VPs. Furthermore, it complicates the switch design because two distinct VP tables need to be maintained.

7. **VC merge:** The VC merge approach buffers cells of other packets at the switch until all cells of the current packet go through (as shown in figure 2.7). The technique is also called "cut-through forwarding," and it is used in the SEAM [52] and ARIS schemes. It entails implementing a packet-based scheduling algorithm at the merge point, and maintaining separate queues for each sender. The AAL5 end-of-message bit is used to signal to the switch that a packet from a different port can now be forwarded. The approach is fast and simple but requires more memory at the switches, and adds to the burstiness and latency of traffic. However, studies [137] have shown that the effect of VC merge on traffic is minimal.

8. **Sub-channel multiplexing:** A sub-channel is a "channel within a VC." Each sub-channel can be assigned an identifier called the sub-channel number to distinguish between multiple sub-channels in a VC [126]. Four bits from the Generic Flow Control (GFC) bits in the ATM cell header can carry this number. Each burst of cells is preceded by a "start" resource management (RM) cell, and followed by an "end" RM cell. The sub-channel is allocated on the "start" cell and released on the "end" cell. Sub-channel identifiers can change at every

switch. This approach allows dynamic sharing by using on-the-fly mapping of packets to sub-channels. However, four bits only allow up to fifteen concurrent senders (sub-channel number hexadecimal FF indicates an idle sub-channel). If no sub-channel is available, the burst of cells is lost, so this solution may not be very scalable.

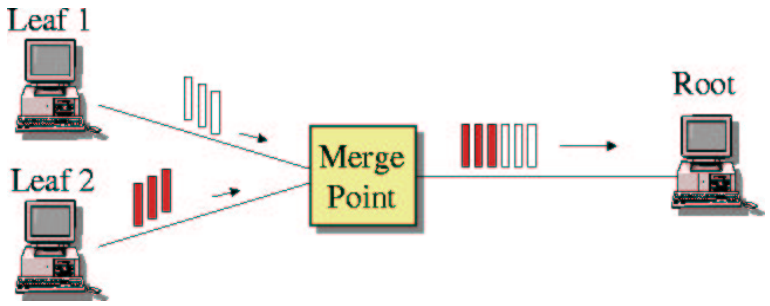


Figure 2.7: The VC merge approach buffers cells of other flows until all cells of the current packet go through

Although each of the approaches has its own merits and drawbacks, VC merge and VP merge are the most popular approaches today. This dissertation emphasizes the issues involved if VC merge and VP merge are implemented.

ATM Multipoint Proposals

Proposals for supporting ATM multipoint connections as a single shared tree include the SMART [45] and SEAM [52] protocols.

The scalable and efficient ATM multipoint-to-multipoint (SEAM) scheme [52] attempts to achieve scalability by using a single shared tree for all senders and receivers, and a CBT-like core (as in IP) for routing. Each conversation in SEAM has a unique identifier called a group handle, which identifies all packets associated with a given

multicast group. The core is used as a focal point for routing signaling messages for the group. Both member-initiated and core-initiated joins of the multicast group are allowed. Little changes in signaling are required. A technique called "short-cutting" is also used whereby cells are forwarded by switches on a spanning tree. This mechanism allows packets to follow the shortest path along the shared tree by emulating reverse path forwarding. The VC merge technique is also employed at the switches to avoid the cell interleaving problem.

A SEAM based environment can co-exist with islands of non-SEAMable switches, where only the boundary SEAM switches need to be concerned with interoperating with non-SEAMable islands. Such islands can exploit the point-to-multipoint capabilities of current ATM signaling. Refer to [131] for an alternative ATM PNNI routing mechanism for group multicast based upon dynamic Steiner trees.

The shared many-to-many ATM reservations (SMART) protocol [45], like SEAM, uses a shared tree for each many-to-many VC connection. SMART serializes the communication of various senders to a set of receivers, thus avoiding the cell interleaving problem. This is done by allowing a sender to send to the group only after it has acquired a token, where each VC only contains one token.

The main idea of the SMART scheme is that the resources for the requested service are reserved, and resource management (RM) cells are used as special control messages (called "GRANT" and "REQUEST" messages) to control the link access. These messages are associated with each VC connection. When a system receives a "GRANT" message, this means that the sender of the "GRANT" is willing to receive data on this VC connection. If two "GRANT" messages coming from different directions cross each other on a link, a "bias" (initially negotiated among every two

neighbors) is used to resolve the conflict. In addition, RM cell races are avoided by using a two-bit sequence number in the RM cells. The protocol grants requests (sends tokens) in such a way that the reservations are respected.

Since SMART uses control messages to guarantee the serialization of communication, it not only solves the cell interleaving problem, but also guarantees that the traffic contract associated with the VC connections is respected, thus performing a simple traffic management function.

2.10 Fairness and Flow Control for Multipoint Connections

We first discuss the case of multiple receivers and branch point algorithms, then we discuss the case of multiple senders and merge point algorithms.

2.10.1 Fairness for Multiple Receivers

In the case of multiple receivers, the rate at which the source can send depends on the rates that can be supported by the all links on the path from the source to *all* the receivers. The choice of which fairness definition to adopt is application specific [42]. The most commonly adopted fairness definition for the ABR service is a simple extension of the max-min fairness definition such that the source sends at the bottleneck rate indicated by switches on all the paths. In other words, the source attempts to minimize loss on all the branches by sending at no more than the rate that can be supported by the slowest branch. This is the best approach if none of the receivers can tolerate loss.

Sending at the minimum rate can result in low link utilization on all the multicast tree branches other than the most congested branch. Therefore, several alternative

approaches have been proposed. These approaches attempt to maximize the *intra-session* fairness (or intra-group fairness), i.e., they provide a receiver with a rate according to its capabilities and the capacity of the path leading to it, regardless of the capabilities of the other receivers [77].

A popular approach to maximize intra-group fairness is using multiple multicast groups. The source transmits data to multiple groups at different rates. Each receiver subscribes to the group where the data is being sent at or lower than the bottleneck rate on the path to that receiver. Receivers can dynamically unsubscribe and subscribe to groups when conditions change. Several algorithms have been based on this idea [4, 18]. Layering the transmitted data [95, 133], using forward error correction [133], and using hierarchical approaches [23, 42, 93] have also been proposed in the context of reliable multicast transport and congestion control.

A number of studies have also proposed that the sender transmit at a rate higher than that requested by the slowest branch, in order to maximize a certain fairness metric. This "quasi-reliability" can be used for information dissemination applications where the receiver can "tune" the level of reliability requested [138]. One of the metrics to be maximized is the *inter-receiver* fairness, which is defined as the ratio between the minimum of the sender rate and the bottleneck rate supported for a branch, to the maximum of the two. This can be maximized in a weighted manner for all branches, as long as the application-specific loss tolerance is met [77].

Inter-session fairness (or inter-group fairness) is defined as providing a receiver with a rate according to its capabilities and the capacity of the path leading to it, regardless of the capabilities of the other receivers in the group *and other groups* [92]. A similar concept called bounded fairness or *essential fairness* is proposed among

TCP and point-to-multipoint connections [136]. If the throughput of TCP is denoted by λ_{TCP} and that of multicast connections is denoted by λ_m, then a multicast session is essentially fair if:

$$a \times \lambda_{TCP} < \lambda_m < b \times \lambda_{TCP}$$

where a and b can be selected as desired.

2.10.2 ABR Point-to-Multipoint Algorithms

A branch point replicates cells from the root to each branch in the responding state and consolidates their feedback. In point-to-multipoint connections, consolidation of feedback information from different leaves of the tree is necessary. This is due to the "feedback implosion" problem: feedback information provided to the sender should not increase proportional to the number of leaves in the connection. This problem resembles the feedback implosion problem for transport layer flow control (see, e.g., [24, 123]), and we have examined that area and considered many of the proposed ideas. Our work can provide insight into the solution of this problem as well.

The source in an ABR point-to-multipoint VC is usually required to send at the minimum of the rates allowed by all the destination nodes. Unless the destinations can tolerate cell loss, allowing the source to send at a rate larger than the minimum rate tolerated by receivers will result in lost cells at at least one receiver. The minimum rate is the technique most compatible with the typical data requirements where no data should be lost, and the network can take whatever time it requires to deliver the data intact.

A number of consolidation algorithms have been proposed in [14, 19, 77, 78, 96, 98, 103, 105, 106, 113, 127, 134, 135, 141]. The algorithms differ in which component (destination or branch point) generates the BRM, and the condition that triggers the generation of the BRM, in addition to other details.

The first point-to-multipoint ABR algorithm was proposed in [106]. In this algorithm, a register MER (minimum explicit rate), maintains the minimum feedback indicated by the BRM cells received from the branches. Whenever an FRM cell is received, it is multicast to all branches, and a BRM is returned with the MER value as the explicit rate. MER is then reset.

This algorithm suffers from the "consolidation noise" problem when a BRM generated by a branch point does not consolidate feedback from all tree branches [58]. In fact, if a BRM generated by the branch point does not accumulate feedback from *any* branch, the feedback can be given as the peak cell rate (if that branch point itself is not overloaded). In [103, 113] some solutions to this problem are proposed. A simple enhancement to alleviate this problem is to maintain a flag, and only generate the BRM cell if a BRM has been received from at least one leaf after the last BRM was sent by the branch point [113].

To reduce the complexity of the scheme, Ren and Siu [103] propose to forward one of the BRM cells returned by the leaves, instead of turning around the FRM cells of the source. Another alternative would be to pass back the BRM cell only when BRM cells from *all* branches have been received after the last feedback. This idea is also used in [19], but the BRM cell that is allowed to pass back to the source is the last BRM cell to be received with a certain sequence number. These approaches suffer from a slow transient response.

Other schemes, such as those in [14, 78, 96, 134, 135] have been recently developed. Some of these schemes use timers at the branch points, and the BRM cells are sent by the branch points when the timer expires. Timers, however, are expensive to implement in network elements. Others [78] store all values returned by the BRM cells from the different branches and update the values as BRM cells are received. The minimum of all the rates has to be computed every time a BRM cell is to be sent. This can be a complex operation. Thus, the main problem with new schemes is the complexity of implementation. Chapter 7 gives a simple consolidation algorithm with little consolidation noise and a fast transient response.

2.10.3 Fairness for Multiple Senders

Multipoint-to-point connections are especially important for overlaying IP networks and simplifying end systems and edge devices [115]. In this case, only one multipoint-to-point connection needs to be set up even if there are multiple senders. This reduces the connection management and control overhead over using multiple point-to-point connections.

Little work has been done to define fairness and traffic management rules for multipoint-to-point connections. Bandwidth management is an important issue. In the presence of multiple senders, fairness definition within a multicast group and among multicast groups and point-to-point connections becomes complex. Billing and pricing issues play an important role in such cases.

Multipoint-to-point connections require feedback to be returned to the appropriate sources at the appropriate times. The bandwidth requirements for a VC after a merge point is the sum of the bandwidths used by all senders whose traffic is merged. This

is because the aggregate data rate after a merge point is the sum of all incoming data rates to the merge point [76]. Similarly, the number of RM cells after merging is the sum of those from different branches. Hence, if all sources in the VC use the same RM cell to data cell ratio (as denoted by the parameter Nrm), the ratio remains the same after merging.

2.10.4 ABR Multipoint-to-point Algorithms

Ren and Siu [104] describe an algorithm for multipoint-to-point congestion control that allocates bandwidth fairly among contending sources. The algorithm assumes that a multipoint-to-point VC is defined as a shared tree, and that VC merging is employed to prevent the cell interleaving problem. The authors prove that, given a max-min fair point-to-point rate allocation algorithm, their proposed multipoint-to-point extension framework is also max-min fair among sources. The authors do not discuss why fairness among sources is the best approach.

The idea of Ren and Siu's algorithm is similar to point-to-multipoint algorithms [26, 96, 103, 106]. The algorithm operates as follows. When a forward resource management (FRM) cell originating at a leaf is received at the merge point, it is forwarded to the root, and the merge point returns a backward resource management (BRM) cell to the flow which had sent the FRM cell. The explicit rate in the BRM cell is set to the value of a register called MER (explicit rate), maintained at the merge point for each VC. The MER register is then reset to the peak cell rate. When a BRM cell is received at the merge point, the ER value in the BRM is used to set the MER register, and the BRM cell is discarded. Although the algorithm is fair, it can cause

rate oscillations and is complex to implement due to the RM cell turn-around at the merge points.

The same authors remedy these problems in [105]. The algorithm maintains a bit at the merge point for each of the flows being merged. The bit indicates that an FRM has been received from this flow after a BRM had been sent to it. Therefore, when an FRM is received at the merge point, it is forwarded to the root and the bit is set, but the RM cell is not turned around as in the previous algorithm. When a BRM is received at the merge point, it is duplicated and sent to all the branches that have their bit set, and then the bits are reset. Only destinations turn around RM cells in this case. The problem with Ren and Siu's work is that it does not clearly state which types of rate allocation algorithms the proposed multipoint extension works for. In fact, the extension does not work for many popular ABR schemes that perform per VC accounting, since this is no longer equivalent to per-source accounting, as discussed in section 8.3.

Recently more complex algorithms have been developed [13, 97] for multipoint-to-point and multipoint-to-multipoint connections respectively. The algorithm in [13] aims at fairness among the sources as in [104]. The algorithm in [97] adds a *weight* in RM cells to allow scaling of the rates to give the appropriate allocations to sources. The throughput of a unicast source is given a pre-determined weight with respect to that of a sender in a multicast session. This technique adds more flexibility at the expense of complexity in RM cells and processing. Weight assignment is also very difficult. Chapter 8 proposes fairness definitions for multipoint-to-point connections and gives a simple distributed algorithm for rate allocation.

2.11 Concluding Remarks

Table 2.6 gives a summary of the comparison of ATM service categories as discussed in this chapter. As seen in the table, CBR, rt-VBR and nrt-VBR provide high quality of service, provided that the user can specify the traffic characteristics and quality of service requirements of the connection. Vanilla UBR is simple, but gives no guarantees. ABR provides fair and efficient utilization of bandwidth and exhibits good performance, though it requires the user to comply with the end system operations, and the network elements to indicate congestion state in the cells. The GFR service gives minimum rate and low loss without requiring end system cooperation, but network elements need to perform frame-level tagging, fair buffer allocation or scheduling operations to provide the guarantees.

ABR is unique because of its feedback control. It allows easier handling of drop preferences and priorities, and can best utilize added buffering. The edge devices can intelligently mark, drop and schedule flows based on the enterprise policy. Thus, our analysis indicates that a well-designed and engineered ABR implementation is capable of providing the most flexible QoS-based transport of enterprise traffic over ATM backbones. To support multimedia applications, the ABR service should evolve to provide end-to-end delay guarantees through the carrier network. It should also efficiently support connections with multiple senders and multiple receivers. This dissertation focuses on the traffic management and multicast support issues for the ABR service.

Category	Nature
CBR	Gives strict guarantees, but requires the user to define its traffic characteristics and QoS requirements, and is unsuitable for bursty traffic (little statistical multiplexing gains).
rt-VBR	Gives strict guarantees, but requires the user to define traffic characteristics and QoS requirements.
nrt-VBR	Gives loss guarantees, but requires the user to define traffic characteristics. Requires large network buffers.
UBR	Extremely simple, but gives no guarantees.
GFR	Gives loss and rate guarantees, but network elements must perform frame-level tagging/policing, scheduling or buffer allocation functions.
ABR	Gives loss and rate guarantees, but sources and network elements must perform a number of complex functions. Provides adaptive closed-loop feedback, and hence gives excellent control and utilization. Pushes queues to edge routers, with small queues inside the ATM network.

Table 2.6: Comparison of ATM service categories

CHAPTER 3

PROBLEM STATEMENT AND METHODOLOGY

Connection management, routing, and traffic management become complex when multiple senders and multiple receivers are parties in the same connection. We focus on the ABR point-to-point and multipoint traffic management service.

3.1 Problem Statement

This dissertation resolves traffic management issues for point-to-point, point-to-multipoint, multipoint-to-point and multipoint-to-point connections, and studies flow control parameters. This includes:

1. Investigation of the connection of enterprise sites using ABR virtual paths, and study of fairness and rate allocation schemes for virtual paths and virtual connection multiplexing.

2. Study of the effect of ABR end system parameters and ABR end system rules, and development of formulae and guidelines for setting for these parameters to achieve the best performance. This includes the study of the effect of round trip time and link bandwidths on parameter values.

3. Study of fairness issues in ABR rate allocation schemes, and development of distributed fair ABR rate allocation algorithms.

4. Development of an ABR point-to-multipoint traffic management framework that resolves the noise and slow transient response problems and balances the tradeoff involved. The framework must also have a low overhead and complexity, and give efficient and fair bandwidth allocations.

5. Development of a precise definition of the optimal allocations for ABR multipoint-to-point connections, and development of a traffic management framework for managing bandwidth for those connections. The framework must achieve a set of objectives, including optimality of allocations and low overhead.

6. Comprehensive analysis of the performance of ATM multipoint traffic management schemes under a variety of configurations and traffic patterns, ensuring scalability. Buffer requirements are studied, in addition to transient performance, noise, and delays. VBR traffic is used as cross traffic.

The use of multipoint-to-multipoint shared tree connections for ATM multipoint communication ensures the scalability of the multipoint service. In this case, nodes points can combine the algorithms we develop for branch and merge points. The point-to-point algorithm is simply be a special case of the multipoint one.

3.2 Methodology

We will develop novel schemes for traffic management for point-to-point and multipoint connections, and develop formulae and guidelines for setting parameter values. We explain the methodologies used in the next few paragraphs.

3.2.1 ABR End System Parameters

The main aim of this study is to develop formulae and guidelines for setting ABR end system parameters. Sensitivity analysis for the parameters is also performed. Our approach is to determine and explore the range of each parameter value, and give recommendations on how to set the parameters at the end system and at the network switches. The effect of the round trip time of the connection, as well as the bandwidths of the links on the connection path, on these parameter value should be determined. We use various traffic models and background traffic models, and various network configurations for our simulations.

3.2.2 ABR Rate Allocation for Unicast Traffic

Our goal is to understand how a distributed rate allocation algorithm computes max-min fair allocations, and develop such algorithms. We develop the notion of a maximum share that a connection can get, and compute this maximum share as the target capacity divided by the sum of activity levels for all connections. We prove that allocating the maximum share results in max-min fair allocations. We analyze the performance of the proposed algorithm in a configuration specifically designed to test the fairness of the scheme. We also examine the design and performance of an algorithm that does not estimate the effective number of active connections.

3.2.3 Multiplexing on ABR Virtual Path Connections

We propose that ABR virtual paths be used to connect enterprise sites in a virtual private network. ABR minimizes cell loss and delay inside the network. Hence if the appropriate scheduling and management mechanism is implemented at the edge

devices, ABR can well support different QoS and differentiated services in the backbone. We design the architecture of the edge device, and devise the ABR flow control mechanism. We extend max-min fairness definitions for fair bandwidth allocations of ABR VPCs and the ABR VCCs multiplexed on them. In addition, we develop a distributed algorithm for computing fair bandwidth allocations for the VCCs. The main idea of our algorithm is to use the VPC rate and perform accounting separately for each VPC, in order to estimate the capacity and load of the multiplexed VCCs. The VPC queue length is also an important consideration for our algorithm.

3.2.4 ABR Point-to-Multipoint Traffic Management

A novel approach that reduces consolidation noise, without unnecessarily slowing down the feedback during excessive overload, is designed. To eliminate consolidation noise, feedback is only passed back towards the source when feedback from all branches has been received after the last feedback. This can be done by maintaining a separate flag for each branch to indicate if feedback has been received from the branch after the last feedback was sent. The slow transient response in this algorithm, caused by waiting for feedback from possibly distant leaves, can be avoided when a severe overload situation (severe overload means the rate to be returned is less than a multiplicative factor of the last rate returned) has been detected. We call this fast overload indication. In the cases when the branch point itself is a switch and queuing point, the branch point can invoke the switch scheme to compute the new rate value whenever feedback is received. Hence, overload at the branch point can be detected and indicated according to the fast overload indication idea.

The fast overload indication may increase the BRM cell overhead, since the ratio of source-generated FRM cells to BRM cells received by the source can exceed one. We alleviate this problem using a simple counter mechanism.

3.2.5 ABR Multipoint-to-Point Traffic Management

We define fairness for a network with multiple point-to-point and multipoint connections. The max-min fairness definition discussed in chapter 2 can be applied to a network with multipoint connections at the source level, the VC level, or the flow level, where we define a flow as a VC coming on an input link. This results in four different fairness definitions, which we explore in chapter 8.

Since sources, VCs, and flows are equivalent for point-to-point connections, but different for multipoint-to-point connections, it is important to note the differences between per-source accounting, per-VC accounting and per-flow accounting. We give general guidelines for adapting rate allocation algorithms for multiple sender scenarios. We give precise information on the values that can be used by the algorithm for rate estimation, and explain how to measure each of these values. We also develop a rate allocation algorithm to achieve source-based fairness. The algorithm uses some of the techniques developed in the ERICA+ switch algorithm, as discussed in chapters 2 and 5. The approach relies on computing the explicit rate by taking the maximum of the current cell rate of the connection, divided by an overload factor (computed as in the ERICA+ scheme), and the maximum explicit rate given in the previous interval. Unlike many ABR schemes, the approach does not need to know the number or rates of the sources, and at the same time, the approach exhibits a fast transient response and good steady state properties.

3.2.6 Performance Analysis

We use simulation techniques to demonstrate that the proposed frameworks are suitable for the expected range of traffic patterns, number of VCs, bandwidth bottlenecks, and round trip times. We use VBR background traffic in many of our simulations to show the effect of variable capacity. We also use bursty traffic to demonstrate the effect of variable demand. We use various numbers of connections, and various numbers of receivers and senders in each connection. We also use point-to-point and multipoint connections in the same simulations. We experiment with various link lengths and round trip times, as well as various link bandwidths. We use different locations of bottleneck links with respect to the branch and merge point locations. Local area networks, wide area networks and satellite networks are studied. The optimality of allocations, transient performance, noise, and delays are studied. Buffer requirements are also evaluated.

3.3 Chapter Summary

In this chapter, we have defined the problem of traffic management of point-to-point and multipoint communication in ATM networks. The problem consists of five main components: (1) ABR parameter study, (2) fair rate allocation algorithm design, (3) aggregation design, (4) feedback consolidation algorithm design, and (5) fairness and rate allocation algorithm design for multiple senders. The next five chapters of this dissertation are devoted to each of these five problems.

CHAPTER 4

ROLES AND GUIDELINES FOR SETTING ABR PARAMETERS

This chapter discusses the role of parameters negotiated and used for flow control in the ATM ABR service, and gives guidelines for setting each parameter. The effect of the speed of the links on the path from the source to the destination, and the round trip time of the connection, on the ABR parameter values is discussed. We also give simulation results to illustrate the effect of various parameter values on the performance of the ABR connection, in terms of both throughput and queue lengths.

4.1 ABR Parameters

At the time of connection setup, ABR sources negotiate several parameters with the network. A complete list of parameters used in the ABR mechanism is presented in table 4.1. The relevant parameters are further explained as they occur in the ensuing discussion.

4.2 Rate Upper and Lower Bounds: PCR and MCR

We first discuss the role of the bounds, then discuss how to set their values.

Label	Expansion	Units and Range	Default Value	Signaled?
PCR	Peak Cell Rate	cells/second from 0 to 16M	–	down
MCR	Minimum Cell Rate	cells/second from 0 to 16M	0	down to MCRmin
ACR	Allowed Cell Rate	cells/second from 0 to 16M	–	no
ICR	Initial Cell Rate	cells/second from 0 to 16M	PCR	down
TCR	Tagged Cell Rate	constant	10 cells/s	no
Nrm	Number of cells between FRM cells	power of 2 from 2 to 256	32	optional
Mrm	Controls bandwidth allocation between FRM, BRM and data cells	constant	2	no
Trm	Upper Bound on Inter-FRM Time	milliseconds, 100 × power of 2 from −7 to 0	100 ms	optional
RIF	Rate Increase Factor	power of 2 from 1/32768 to 1	1	down
RDF	Rate Decrease Factor	power of 2 from 1/32768 to 1	1/32768	up, or down by ≤ RIF decrease factor
ADTF	ACR Decrease Time Factor	seconds, from 0.01 to 10.23 seconds in steps of 10 ms	0.5 s	optionally down
TBE	Transient Buffer Exposure	cells from 0 to 16,777,215	16,777,215	down
CRM	Missing RM-cell Count	integer of unspecified size	$\lceil \frac{TBE}{Nrm} \rceil$	computed
CDF	Cutoff Decrease Factor	zero or a power of 2 from 1/64 to 1	1/16	optionally up
FRTT	Fixed Round-Trip Time	microseconds from 0 to 16.7 seconds	–	accumulated

Table 4.1: ABR parameters and their ranges

4.2.1 Role

The peak cell rate (PCR) and the minimum cell rate (MCR) are used in **source rule 1**. The rule states that source should always transmit at a rate equal to or below its computed ACR, which cannot exceed PCR and need not go below MCR, i.e.,

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

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

PCR is the maximum value at which a source can transmit. It must be negotiated down and has no default value. MCR is the minimum value that a source need not reduce its rate beyond. It is negotiated down to the minimum acceptable MCR, MCRmin, only if MCRmin is signaled.

4.2.2 Values

The sources can initially set PCR to the maximum possible value if they are willing to pay for it (for example, they can set it according to the capacity of the application or host, or the link bandwidth of the link from the host to the next node). Of course, billing and pricing considerations play an important role here, and sources may select a lower PCR value if they are not willing to pay for a large one.

MCR can be set according to the user requirements (e.g., video applications require some minimum rate guarantee), and the billing and pricing policy as well. Unless the traffic is high priority, MCR can be set to zero, to make the service a best effort one. Most applications, especially TCP/IP applications, however, work better with an MCR greater than zero, to prevent timeouts.

Observe that charging considerations may limit PCR to be a multiple of MCR, i.e.,

$$PCR = k \times MCR$$

where:

$$2 \leq k \leq 10$$

This makes it easier for traffic to be shaped.

The switches can reduce the PCR and MCR according the connection admission control (CAC) algorithm. One possible simple policy is to ensure that the following is satisfied:

$$\Sigma PCR_{CBR} + \Sigma SCR_{VBR} + \Sigma MCR_{ABR}(\text{including the new connection}) \leq \text{link bandwidth}$$

Hence, the MCR of the new connection can be computed as:

$$MCR_i \leq min(\text{User-requested:}MCR_i, \text{link bandwidth} - \Sigma PCR_{CBR} - \Sigma SCR_{VBR} - \Sigma_{j \neq i} MCR_{ABRj})$$

If the signaled MCR is less than the minimum acceptable MCR, i.e., $MCR_i < MCRmin_i$, the connection is rejected.

As for the PCR of the ABR connection, it is only limited by the bandwidth of the links on the path from the source to the destination.

$$PCR_i = min(PCR_i, \forall j, j \in \text{links on path from source to destination, minimum (link bandwidth)}_j)$$

Thus, PCR and MCR are dependent on the bottleneck link bandwidth, but not on the round trip time (RTT) of the connection.

Refer to chapter 6 for algorithms and results with MCR guarantees.

4.3 Control of Frequency of RM Cells: Nrm, Mrm and Trm

We first discuss the role of the control parameters, then discuss how to set their values.

4.3.1 Role

The three parameters Nrm, Mrm and Trm control the frequency of generation of resource management cells at the source. They are used in **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.

The sources are required to send an FRM after every Nrm cells. But 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 should send an FRM cell if more than Trm milliseconds have elapsed since the last FRM was sent. This introduces another problem for low rate sources. In some cases, at every transmission opportunity, the source may find that it has exceeded Trm 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 (Mrm) other cells between FRMs.

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. The second and third part of source rule 3 ensure that BRMs are not

unnecessarily delayed and that all available bandwidth is not used up by the RM cells. Figure 4.1 illustrates the scheduling of FRMs, BRMs and data cells.

Figure 4.1: Scheduling of forward RM, backward RM, and data cells

4.3.2 Values

Mrm is constant at 2, and is not negotiated at connection setup. We next discuss the setting of Nrm and Trm.

Nrm

The specifications [43] select a default value of 32 for Nrm to ensure that the control overhead does not exceed approximately 6% (the value with window-based flow control). During normal operation, $\frac{1}{32}^{nd}$ or 3% of all cells are FRM cells. Another 3% of cells are BRM cells resulting in a total overhead of 6% [65]. Nrm is independent of link speed and round trip time, since it is simply a ratio.

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 Mbps, the inter-RM cell time is 86.4 μs while it is 8.60 ms for the same connection running at 1.55 Mbps. The inter-RM interval determines the responsiveness of the system. Sources, destinations, and switches may wish to increase Nrm if their processing power is limited, or if they wish to minimize the rate variations of the ABR

connection, or increase the data cell frequency. They may wish to decrease Nrm if fast rate changes are desirable, and responsiveness to network feedback is advantageous.

At high data rates, a small RM cell interval can result in high frequency rate variations caused by the ABR feedback. If traffic such as real-time video is being transported over ABR, the rate variations must be minimized to reduce variations in the quality of service perceived by the user. One way of reducing the ABR rate changes is to send RM cells less frequently, i.e., set Nrm to a large value, instead of 32.

In the experiments shown next, we vary Nrm and examine the allowed cell rates at the sources, as well as the queue lengths at the switches, the link utilizations and the throughput at the destinations. Since the Nrm value must be a power of two that is allowed to range between 2 and 256 (according to the current specifications [43]), we have conducted experiments with all the allowed Nrm values (2, 4, 8, 16, 32, 64, 128 and 256). However, we only show the simulation results for Nrm = 8, 32 and 256 here. The reason why we have selected these values is that values smaller than 8 incur a very high control cell overhead and are not very realistic. 32 is the default value, and 256 is the maximum allowed value. In our simulations, all links are 155.52 Mbps links. The initial cell rate (ICR) of all sources is set to 150 Mbps, while the remaining ABR parameters are set to their default values as specified in the specifications. In particular, note that the value of the rate increase factor (RIF) parameter is set to 1/16. The ERICA [73] scheme is used in this study, with the switch averaging interval set to a fixed time of 5 ms, and target utilization set to 90% of the link capacity. The configuration used in the simulations is an N-source configuration (figure 2.4), consisting of two ABR sources: source 1 sends data at its

ACR throughout the simulation, while source 2 is a transient source that comes on at 100 ms and sends data for about 100 ms. All link lengths are 1000 km. The main aim of this configuration is to test how the responsiveness of the system is affected by the Nrm value, both during rate increases and rate decreases.

ABR performance for the two source transient configuration is shown in figures 4.2 through 4.4 for Nrm = 8, 32, and 256 respectively. The figures show the ACRs of the two sources, the queue length for switch 1, and the link utilization at the bottleneck link. In all cases, source 1 ACR quickly comes down to its target value of about 140 Mbps. When source 2 starts to send data, the ACRs of both sources are brought down to 70 Mbps. When source 2 stops sending data, the ACR for source 1 comes back up to 140 Mbps. There is a difference in the rate of increase of ACR for the three Nrm values. Since RIF is set to 1/16, the ACR comes up in steps on the receipt of every BRM cell. With Nrm = 8, the source receives BRMs more frequently than with Nrm = 256. As a result, the ACR for source 1 first reaches 140 Mbps faster for Nrm = 8. The overhead with small Nrm values is quite high, however. This can be clearly observed by measuring the throughput at the application layer at the destinations (these plots are not shown here). Another interesting observation is that for smaller Nrm values, source 1 does not start rising as fast as with larger Nrm values because the high RM cell overhead causes the data of the second source to take a longer time to be transmitted, and hence the two sources must share the bottleneck link for a longer time.

Table 4.2 shows the variation of inter-RM cell time with link speed, and with Nrm value. The source is assumed to be sending at link rate for the values shown in the

Total ABR Capacity	DS0 64 kbps	T1 1.5 Mbps	OC-3 155 Mbps	OC-24 1.2 Gbps
Nrm = 8	0.5 s	24 ms	24 μs	3 μs
Nrm = 32	2.3 s	96 ms	96 μs	12 μs
Nrm = 256	18.4 s	768 ms	768 μs	96 μs

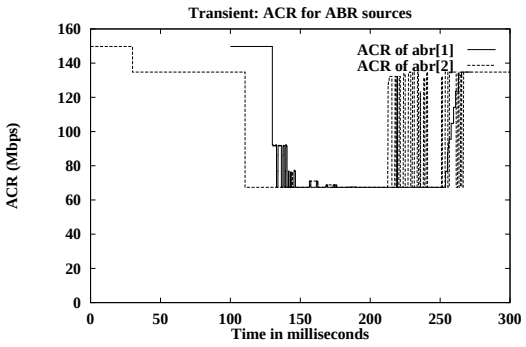
Table 4.2: Inter-RM cell time for different speeds and Nrm values

table. A general heuristic is to use Nrm of 32 at speeds below OC-3 and to use Nrm of 256 for OC-3 and higher speeds.

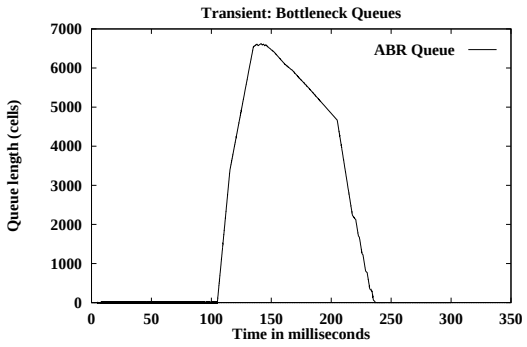
Trm

The Trm parameter is used with low rate sources, when Trm milliseconds are compared to the time elapsed since the last in-rate FRM cell was sent. Sources may get a low ACR due to high amplitude VBR traffic sharing the same resources as the ABR connection, a large number of ABR sources, or low bottleneck link speeds (T1 links). Trm allows low rate sources to sense the network state more frequently than normal. When more bandwidth suddenly becomes available, the network may not be able to allocate the source more bandwidth until it sees an RM cell from the source.

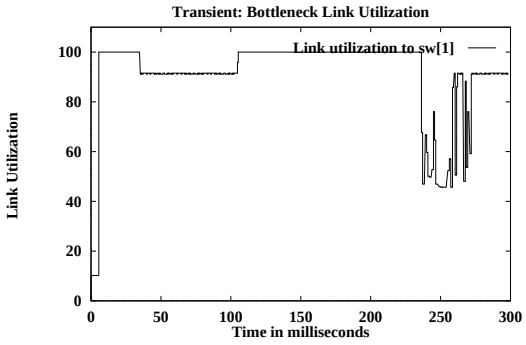
Smaller Trm values result in shorter time between RM cells, leading to faster transient response (rise from low rate to high rate). Small Trm values, however, cause high overhead with low rate sources. The choice of Trm depends on the link speed. For example, at a rate of 155 Mbps, the inter-cell time is 2.7 μs, while at a rate of 1.5 Mbps, the inter-cell time is 270 μs, and at a rate of 2.4 Gbps, the inter-cell time is 0.42 ns. Thus, a Trm value of 100 ms seems more appropriate for 1.5$-$155 Mbps than with higher (2.4 Gbps+) speeds, where a Trm of 100 ms is too long to



(a) Allowed Cell Rate in Mbps

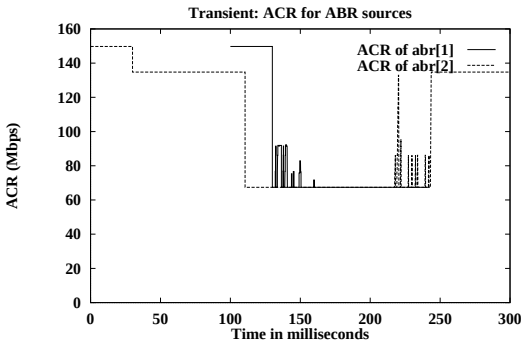


(b) ABR Queue Lengths in Cells

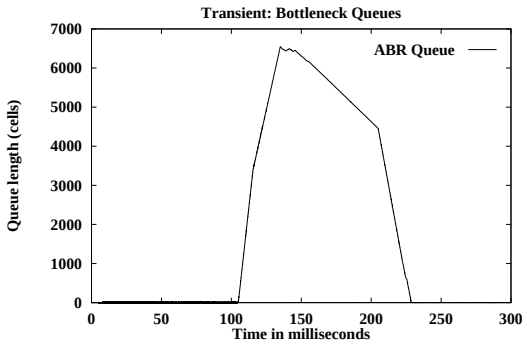


(c) Link Utilization

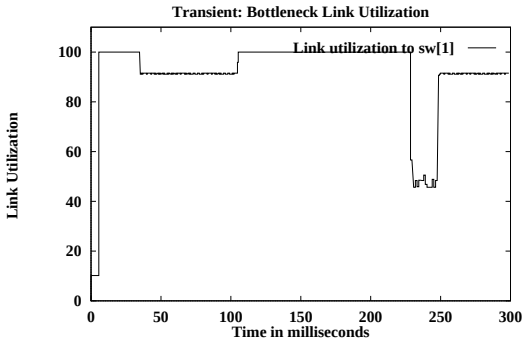
Figure 4.2: Simulation results for a WAN transient configuration, Nrm = 8



(a) Allowed Cell Rate in Mbps

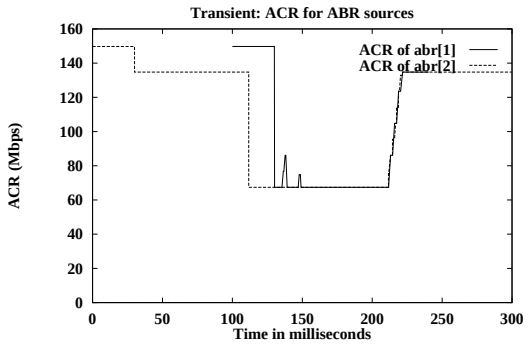


(b) ABR Queue Lengths in Cells

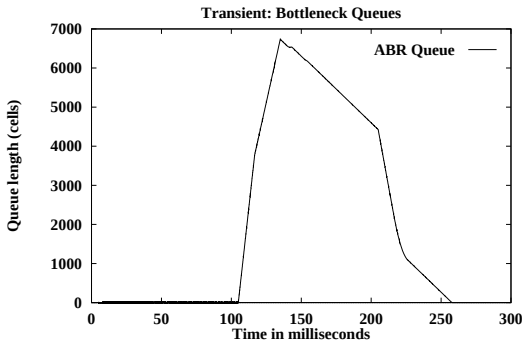


(c) Link Utilization

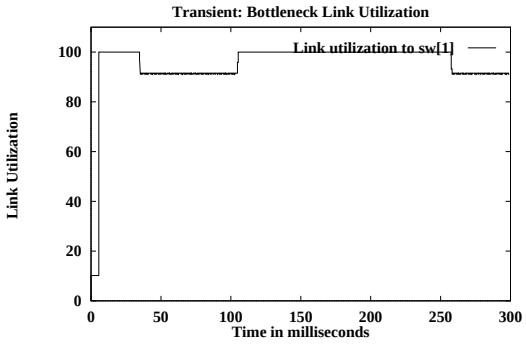
Figure 4.3: Simulation results for a WAN transient configuration, Nrm = 32



(a) Allowed Cell Rate in Mbps



(b) ABR Queue Lengths in Cells



(c) Link Utilization

Figure 4.4: Simulation results for a WAN transient configuration, Nrm = 256

wait before sending an FRM cell to sense the state of the network. Trm should be reduced in such cases. The switches or destination can compare Trm to the inter cell time calculated as the reciprocal of the negotiated PCR (which may indicate the bottleneck link bandwidth). A good value for Trm (based on heuristics) would be:

$$Trm = \frac{1}{PCR} \times c$$

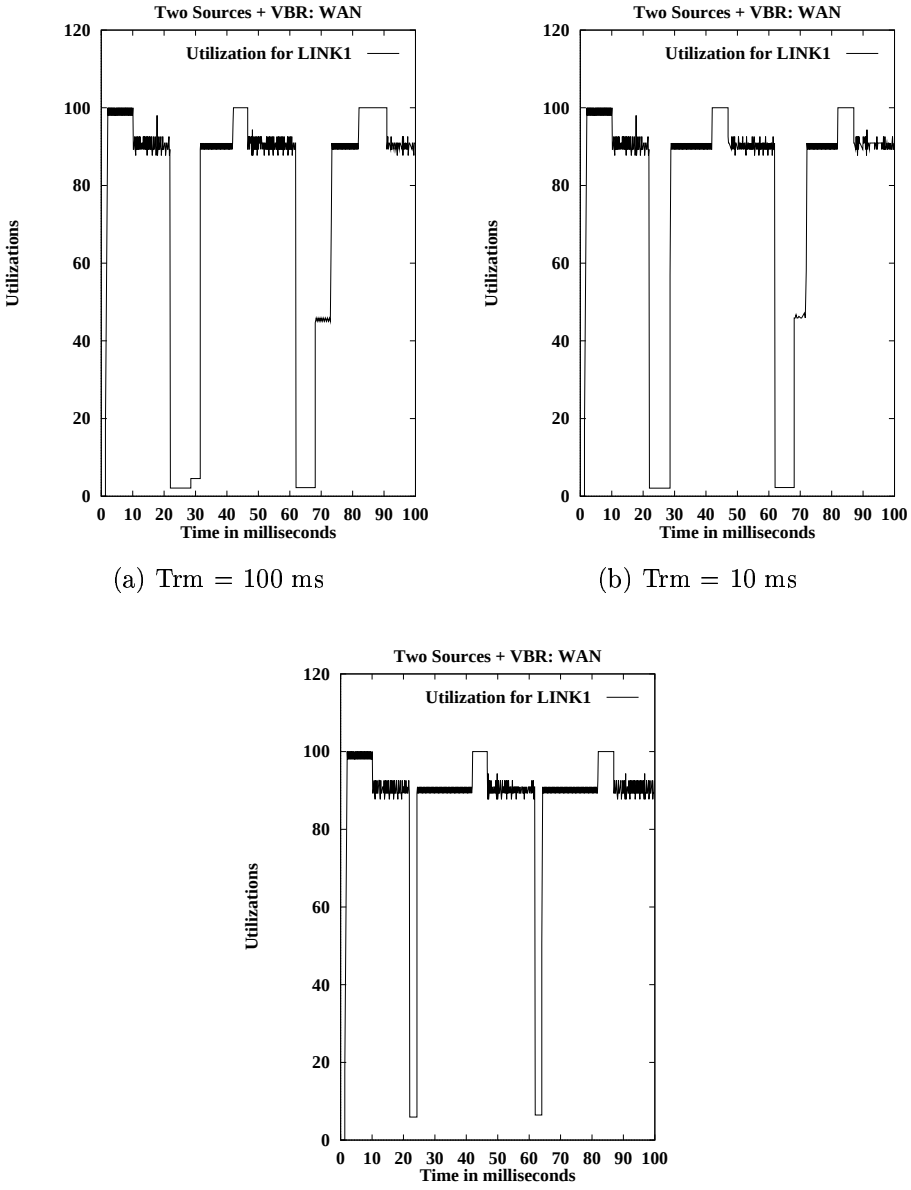
One choice of c can be $\frac{1,000,000}{27}$. This is based upon the intuition that 100 ms was observed to be suitable for OC-3 links (2.7 microsecond = 0.0027 millisecond inter-cell time).

Trm is independent of the round trip time, i.e., whether the connection is local to a LAN, crosses a WAN, or traverses a satellite link of hundreds of milliseconds delay. This is because Trm is compared to the time since the last in-rate FRM cell was sent, so it is independent of the time the RM cell reached the destination, or the time the RM cells returns back to the source.

We have performed an experiment with various Trm values (1, 10 and 100 ms). We assume VBR background traffic always has higher priority. We also assume a simple on/off VBR model where VBR is on for 20 ms and off for 20 ms. When VBR is on, it sends at a rate of 138 Mbps. Simulation results (see figure 4.5) show that in this case, capacity may be unused for a long time for large Trm values (100 ms), when VBR goes away and capacity for ABR becomes available. Lower Trm (1 ms and 10 ms) results in more frequent RM cells, and hence a faster response [70]. *This is especially important for small or zero minimum cell rate.*

4.4 Rate Increase and Decrease Factors: RIF and RDF

We first discuss the role of the factors, then discuss how to set their values.



(a) Trm = 100 ms

(b) Trm = 10 ms

(c) Trm = 1 ms

Figure 4.5: Link utilization simulation results for a WAN two source and VBR configuration with different Trm values

4.4.1 Role

The rate increase factor (RIF) and rate decrease factor (RDF) are used in **source rules 8 and 9**. Source rules 8 and 9 describe how the source reacts to network feedback. The feedback consists of the explicit rate (ER), congestion indication bit (CI), and no increase bit (NI). A source does not simply change its ACR to the new ER due to the following reasons:

1. If the new ER is very high compared to current ACR, switching to the new ER may cause sudden overload 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.

2. If there are any EFCI or relative rate marking (RRM) switches in the path, they do not change the ER field. Instead, they set EFCI bits in the cell headers, or CI and NI bits in RM cells. The destination monitors EFCI bits in data cells, 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 the rate decrease factor (RDF) parameter. Unlike the increase, which is additive, the decrease is multiplicative in the sense that:

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

3. The no-increase (NI) bit handles mild congestion by allowing a switch to specify an ER, but instruct the source not to increase its rate if ACR is already below the specified ER.

The actions corresponding to the various values of CI and NI bits are as follows:

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)

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 [70].

4.4.2 Values

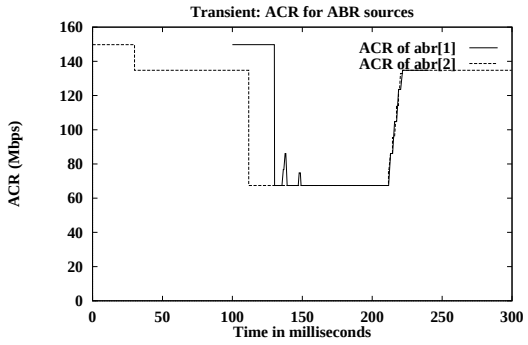
RIF and RDF play an important role when the connection passes through EFCI or RRM switches. In addition, some ER schemes work better with conservative RIF values, while others, such as ERICA [73] are insensitive to the RIF value, and work well with an RIF of 1.

RIF

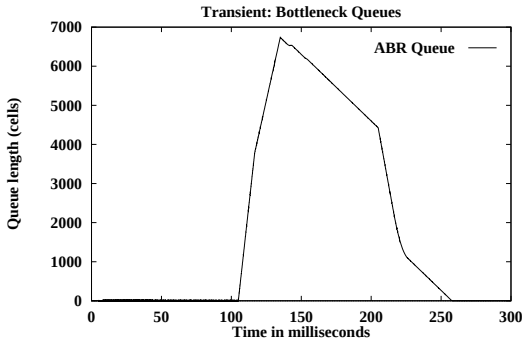
The rate increase factor determines the maximum increase when a BRM cell indicating underload is received. If the RIF is set to a fraction less than one, the maximum increase at each step is limited to RIF $\times$ the peak cell rate for the VC. Setting RIF to small values is a more conservative strategy that controls queue growth and oscillations, especially during transient periods. It, however, may slow down the response of the system when capacity suddenly becomes available, leading to underutilization.

If there are no EFCI switches in a network, setting RIF to 1 allows ACRs to increase as fast as the network directs it (through the ER field). 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 [87]. Thus, sources can initially set RIF to large values, even 1, according to the application requirements. During connection setup, any switch which does not implement an explicit rate scheme or implements a scheme which requires a conservative RIF (such as EPRCA) must reduce RIF to a conservative value such as 1/16 or less. RIF can be set to more conservative (smaller) values for high speeds (as indicated by PCR) and long round trip times (long delay links) to avoid congestion loss. The fixed part of the round-trip time (FRTT) is accumulated during connection setup. This is the minimum delay along the path and does not include any queueing delay [71].

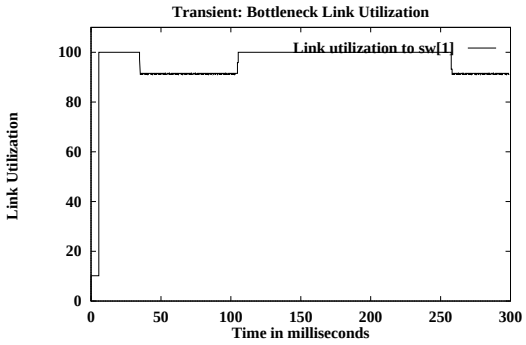
Figures 4.6 and 4.7 compare the performance of a transient configuration (same as the configuration used in the Nrm experiments) with RIF set to 1/16 (the default value) and RIF set to 1. The basic ERICA scheme [73] is used in these simulations. Nrm is set to 256 to slow down the feedback rate, in order to emphasize the effect of RIF. All other parameters are the same as with the Nrm experiments. It is clear from figure 4.6 that an RIF value of 1/16 results in a step increase of the rate of the non-transient source when the transient source stops transmission. With RIF set to 1 (figure 4.7), the rate of the non-transient source increases to the full rate as soon as the inactivity of the transient source is detected.



(a) Allowed Cell Rate in Mbps

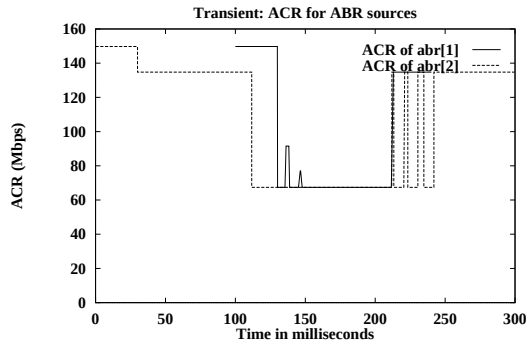


(b) ABR Queue Lengths in Cells

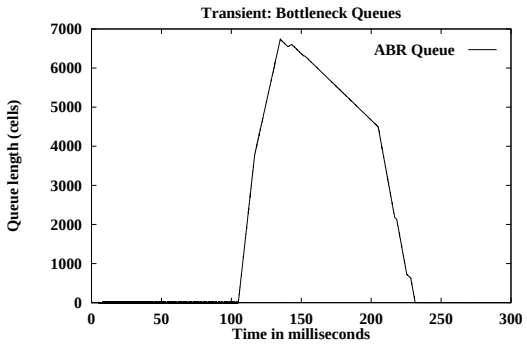


(c) Link Utilization

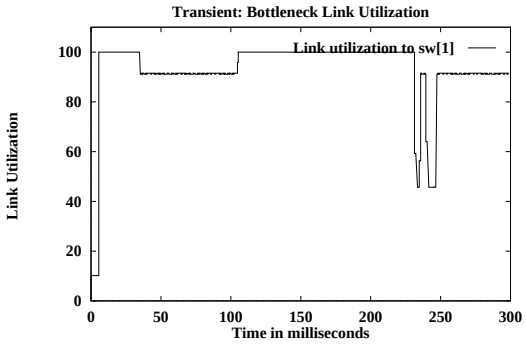
Figure 4.6: Simulation results for a WAN transient configuration (Nrm = 256) RIF = 1/16



(a) Allowed Cell Rate in Mbps



(b) ABR Queue Lengths in Cells



(c) Link Utilization

Figure 4.7: Simulation results for a WAN transient configuration (Nrm = 256) RIF = 1

RDF

When the network is congested (the CI bit is set), the source reduces its rate as follows:

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

Thus the RDF parameter determines how fast the rate is reduced in case of congestion. This multiplicative decrease only occurs if the CI bit is set, either by the switches, or by the destination when the EFCI bits of data cells are set.

The source should initially set RDF to a moderate value. Switches should reduce RDF dependent on the schemes they use for setting EFCI bits or CI bits. *Explicit rate switches need not modify RDF.* The RDF parameter should be set more conservatively (to smaller values) for higher speeds and longer round trip times to avoid a large amount of cell loss during congestion. Switches can examine the round trip time (in the FRTT field) and bottleneck link speed (as indicated by the PCR), and reduce RDF accordingly. If the switch or destination detects a large FRTT or large PCR (indicating a high bottleneck link speed), then RDF should be reduced.

4.5 Rate Reduction under Abnormal Conditions and Startup after Idle Periods: TBE, CRM, CDF, ICR and ADTF

Since CRM and CDF are used with source rule 6, they are both discussed together. Both CRM and ICR are computed using the TBE parameter, so TBE and ICR are also discussed here, as well as ADTF that is used in conjunction with ICR.

4.5.1 Role

We first discuss the rule 6 parameters, and then we discuss the rule 2 and 5 parameters.

TBE, CRM and CDF

The three parameters transient buffer exposure (TBE), missing RM cell count (CRM), and cutoff decrease factor (CDF) are used in **source rule 6**. This rule deals with the following scenario: if a network link fails, or becomes highly congested, RM cells are blocked and the source does not receive 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.

In steady state, a source should receive one BRM for every FRM sent. The sources keep a count of the RM cells sent, and if no backward RM cells are received for a long time, the sources reduce their rate by a factor of "Cutoff Decrease Factor (CDF)." The "long time" is defined as the time to send CRM forward RM cells at the current rate. 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.

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,

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

ICR and ADTF

At the beginning of a connection, sources start at the initial cell rate (ICR) as specified in **source rule 2**. During the first round trip, 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:

$$ICR \leq \frac{TBE}{FRTT}$$

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

$$\text{ICR used by the source} = \min(\text{ICR negotiated with the network}, \frac{TBE}{FRTT})$$

According to **source rule 5**, the rate allowed to a source is valid only for approximately ADTF seconds. 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 the source ACR is already below ICR, it should not increase to ICR. The timeout interval is set to the ACR Decrease Time Factor (ADTF) parameter, whose default value is 500 ms.

Rule 5 is intended to solve the problem of *ACR retention*, when a source retains a rate allocated to it under light loads, and uses it when the network is highly loaded, causing congestion. Several solutions to this problem (called *use it or lose it* (UILI) solutions) were proposed [102, 72]. The ATM Forum standardized a policy that

reduces ACR to ICR when the timeout ADTF expires. Vendors are free to implement additional proprietary restraints at the source or at the switch.

4.5.2 Values

We first discuss the value of TBE and the two parameters which depend on it (CRM and ICR). Then, we discuss CDF, and finally ADTF.

TBE, CRM, and ICR

As previously mentioned, TBE determines the "exposure" of the switch to sudden traffic transients. It determines the the number of cells that may be received at the switch during initial start up (or after any long idle period of time). TBE is specified in cells while CRM is specified in RM cells. Since there is one RM cell per Nrm cells, the relationship between CRM and TBE is as follows:

$$\text{CRM} \leftarrow \lceil \text{TBE/Nrm} \rceil$$

In negotiating TBE, the switches have to consider their buffer resources. 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 or with very high speeds. Thus TBE is highly affected by speed and round trip time (the delay bandwidth product of the connection).

TBE can thus be set to:

$$PCR \times FRTT + \Sigma_i, i \in \{\text{switches on path}\}, \text{buffer sizes}$$

to account for the speed, link delays, and buffer sizes.

Effect of speed and round trip time on TBE and CRM:

For long-delay links, such as satellite links, our simulation results reveal that source rule 6 can unnecessarily trigger and cause oscillations during start up and after idle periods, unless TBE is large enough. This can degrade the throughput considerably. Figure 4.8 shows the configuration used to illustrate the problem. All the links are OC-3 links operating at a rate of 155.52 Mbps. The link connecting the two switches is a satellite link, while the links connecting the switches to the end systems are each 1 km long. The one-way propagation delay of the satellite link is 275 ms, while the propagation delay of each LAN link is 5 microseconds. The traffic is bidirectional, and the sources are persistent. The ERICA [73] algorithm is used with target utilization 90%. The ABR source parameter values are as follows: PCR = 155.52 Mbps, MCR = 0 Mbps, ICR = 0.9×PCR = 140 Mbps, Nrm = 32, RIF = 1, CDF = 1/16, and CRM = 32, 256, 1024, 4096, 6144, 8192.

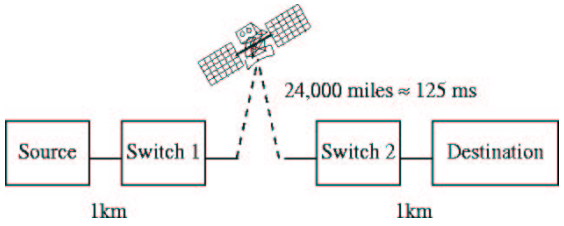
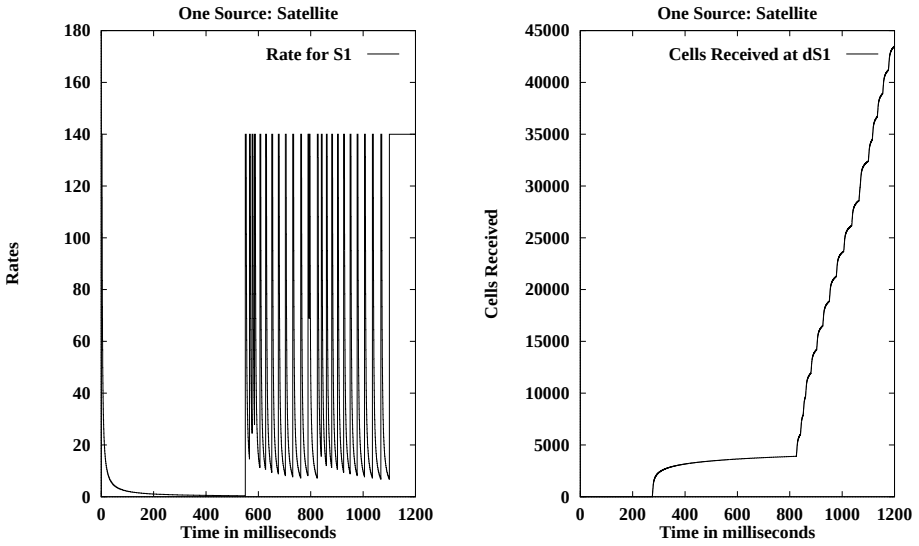


Figure 4.8: One source satellite configuration

Figure 4.9 illustrates the performance of the system with CRM set to 32 (the default value before August 1995). Figure 4.9(a) shows the allowed cell rate of the source over 1200 ms, and figure 4.9(b) shows the number of cells received at the destination during the same period of time [34]. As seen in figure 4.9(a), the initial

rate is 140 Mbps (90% of 155 Mbps). After sending 32 RM cells (or CRM×Nrm = 32×32 = 1024 cells), rule 6 triggers and the rate rapidly drops. The first feedback is received from the network after around 550 ms (275 ms×2), because the one-way delay of the satellite link is 275 ms. The network asks the source to go up to 140 Mbps. The source increases its rate but rule 6 triggers again. The rule triggers again because the time between returning RM cells is large (they were sent at a low rate). This phenomenon of increase and decrease repeats resulting in high-frequency oscillations between very low rates and very high rates. The rapid rate drops occur due to the triggering of source rule 6, while the rate increases occur because the network feedback is consistently at 140 Mbps (90% of 155 Mbps).



(a) Allowed Cell Rate in Mbps (b) Cells Received at Destination

Figure 4.9: Simulation results for a one source satellite configuration. CRM = 32

Figure 4.9(b) shows the number of cells received at the destination. From this figure, it is possible to compute instantaneous throughput by computing the slope of the curve. It is also possible to compute average throughput over any interval by dividing the cells received (increase in the y-value) during that interval by the period of time (x-value) of the interval. The average throughput during the interval from 275 ms to 825 ms is 32 Mbps and that during the interval from 825 ms to 1200 ms is 45 Mbps. During the first 550 ms, the source is mostly sending at a very low rate until the first feedback is received after about 550 ms. The effect of the receipt of feedback can be observed at the destination after 550+275=825 ms. After the first feedback is received, the rate oscillations result in reduced throughput. The results do not significantly vary for different values of CDF. The low throughput values in figure 4.9(b) are a result of the unnecessary triggering of source rule 6 for small CRM values. Rule 6 limits the number of cells that can be in flight during start up periods.

For full throughput, we need to set the value of TBE such that the number of cells in flight can be as large as those required to fill the path both ways. This number is equal to the round trip time (FRTT) multiplied PCR. Hence, the number of RM cells in flight (CRM) should be (1/Nrm)th of this value:

$$\text{CRM} \geq \frac{\text{FRTT} \times \text{PCR}}{\text{Nrm}}$$

For 155 Mbps links, CRM should be greater than or equal to 6144 (550 ms×365 cells per ms/32 cells). For 622 Mbps links, CRM should be greater than or equal to 24576 (6144×4). For two 622 Mbps satellite hops, CRM should be greater than or equal to 49152 (24576×2). For n 622 Mbps satellite hops, CRM should be greater than or equal to 24576×n. Since the size of the TBE parameter is 24 bits and Nrm is normally

32, a 24-bit TBE allows a 19-bit CRM, which is sufficient for most situations (long delay links and high speeds).

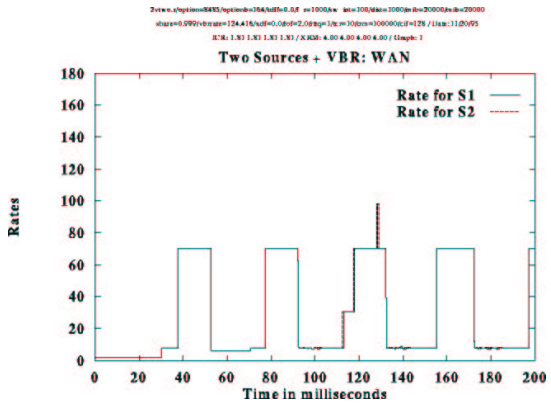
Effect of TBE on queue sizes:

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 show in [67] that it is possible to construct workloads where queue sizes could be unreasonably high even when TBE is very small. TBE limits the queue length only during initial startup and after idle periods when there are no previous cells in the network from the same VC. In this case, the queue length can be given by the following equation:

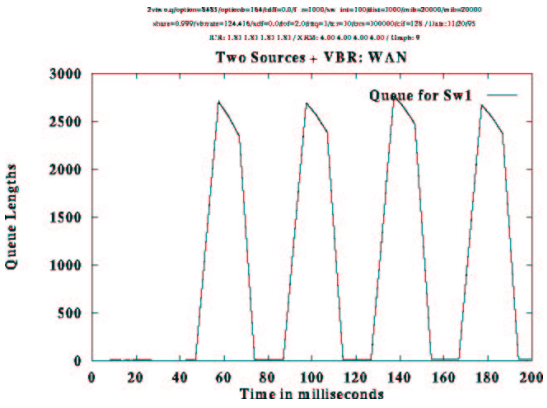
$$\text{Queue length} = (\text{number of sources} - 1) \times min(TBE, \text{burst size})$$

TBE cannot be relied upon during the closed-loop operation phase of a connection. During this latter phase, the contribution of a VC to the queue at a switch can be more than its TBE. The buffer usage at a switch can be more than the sum of TBEs allocated to active VCs. In steady state, rule 6 rarely triggers and is overridden by subsequent explicit feedbacks. Since the reverse flow is not stopped completely, the forward flow continues and keeps filling the queues. TBE does not significantly affect the maximum queue length.

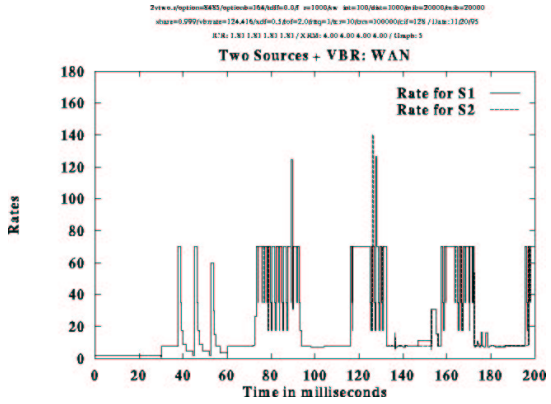
Figures 4.10 and 4.11 show ACR and queue lengths for a network consisting of two ABR and one VBR sources going through two switches to corresponding destination. All simulation results use ERICA switch algorithm [73]. All links are 155 Mbps and 1000 km long. All VCs are bidirectional, that is, D1, D2, VD1 are also sending traffic to S1, S2 and VS1. The following parameter values are used: PCR = 155.52 Mbps, MCR = 0 Mbps, ICR = min155.52, TBE/FRTT, RIF = 1, Nrm = 32, RDF = 1/512, CRM = TBE/Nrm, Trm = 100 ms, FRTT = 30 ms, TBE = {128, 512, 1024} (three



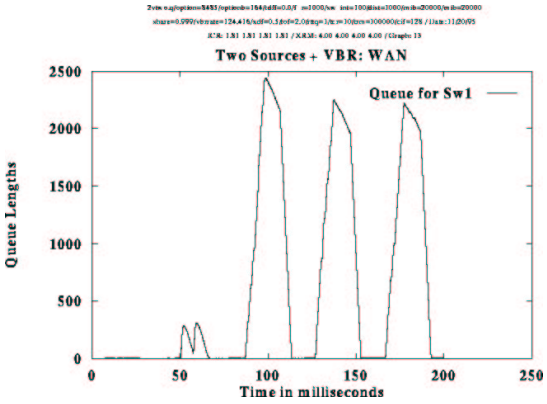
(a) Allowed Cell Rate: Without Rule 6



(b) Queue Length: Without Rule 6



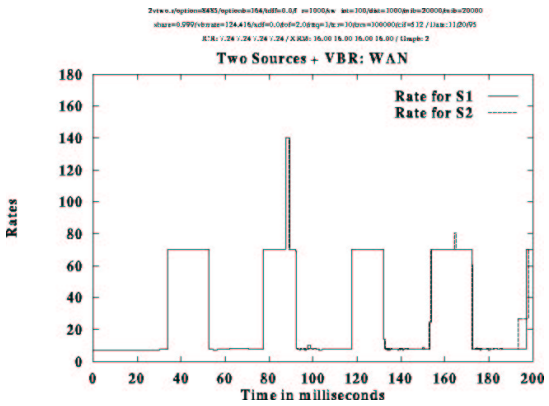
(c) Allowed Cell Rate: With Rule 6



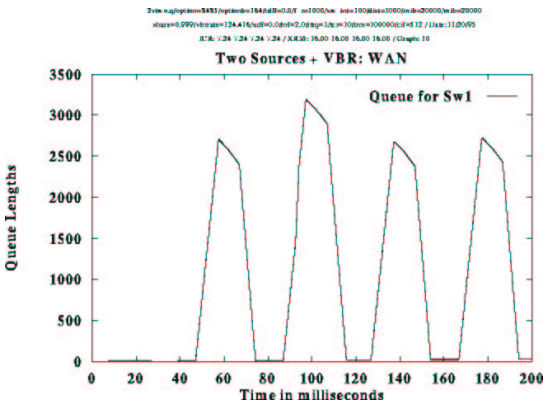
(d) Queue Length: With Rule 6

Figure 4.10: Simulation results for a WAN two sources and VBR configuration. TBE = 128 cells

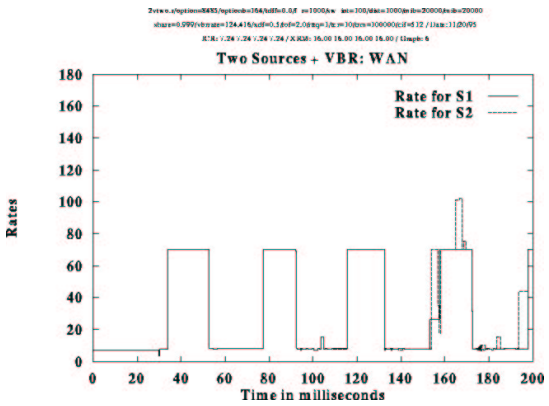
values), CDF = {0, 0.5} = {Without rule 6, With Rule 6}. The VBR source generates a square waveform of 20 ms on and 20 ms off. During the on period, its amplitude is 80% of the link rate. During the off period, the amplitude is zero. The first VBR pulse starts at t=2 ms. Thus, it is on from 2 to 22 ms and off from 22 to 42 ms and so on. The target utilization is 90%. The scheduler gives preference to VBR and so there are no VBR queues.



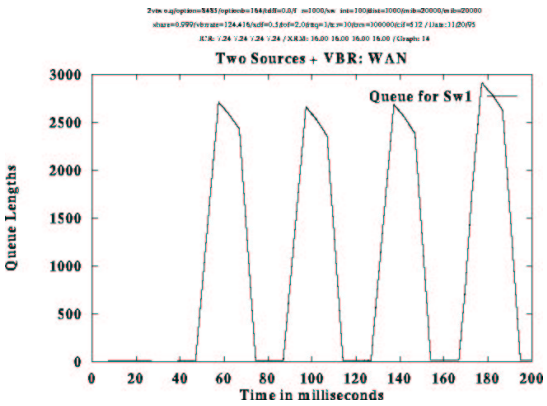
(a) Allowed Cell Rate: Without Rule 6



(b) Queue Length: Without Rule 6



(c) Allowed Cell Rate: With Rule 6



(d) Queue Length: With Rule 6

Figure 4.11: Simulation results for a WAN two sources and VBR configuration. TBE = 512 cells

Figure 4.10 shows the ABR rates and queue graphs for TBE of 128 cells. With just two sources the queue length (without rule 6) is of the order of 2500 cells. The situation does not change significantly with rule 6. Rule 6 does trigger during initial start up, but is not triggered at all once the flow is set up. Figure 4.11 shows ABR rates and queue graphs for TBE of 512 cells. Once again with or without rule 6 the queue length is 2500 to 3000 cells. This queue length is more than that with TBE of 128 but there is no simple relationship between TBE and queue length.

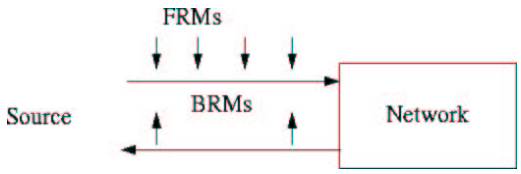


Figure 4.12: Source rule 6 does not trigger if BRM flow is maintained

The reason for the inadequacy of rule 6 in limiting the queue growth can be explained as follows (figure 4.12). Assume that a certain source S is sending forward RM cells at an average rate of R cells per second (cps). The RM cells are turned around by the destination and the backward RM cells are received by S at a different rate r cps. In this case, the inter-forward-RM cell time at the source is $1/R$ while the inter-backward-RM cell time at the source is $1/r$. Source end system Rule 6 will trigger at S if the inter-backward-RM time is much larger (more than CRM times larger) than the inter-forward-RM time [67]. That is, if:

$$1/r \geq \text{CRM} \times (1/R)$$

or:

$$R \geq \text{CRM} \times r$$

In the case of initial startup, r is zero and so after TBE cells, rule 6 triggers and protects the sources. Similarly, in the case of a bursty source, r is zero and rule 6 triggers after TBE cells. However, if the BRM flow is not totally stopped and $R < CRM \times r$, then the cells can accumulate in the network at the rate of $(R-r) \times Nrm$ and not trigger rule 6. In such cases, the queues can grow substantially. The maximum queue length is a function of PCR, the target utilization, and the VBR amplitude, multiplied by the feedback delay [67].

ICR:

ICR should be set by the source as desired according to pricing and the application type. For TCP/IP applications and lower link speeds, ICR should be close to the peak cell rate (PCR).

Switches should reduce their ICR to reflect their availability of buffers, as well as the bandwidth available for the connection. ICR is related to the availability of resources as computed during connection setup, and should correspond to the anticipated ACR for the connection at that time. Finally, the source takes the minimum of that ICR and $\frac{TBE}{FRTT}$ to correspond to the rate at which the source should initially send for the first round trip or after idle periods, before feedback is received. ICR depends on the bottleneck link speed and the round trip time.

CDF

When source rule 6 is triggered, the source reduces its rate by a factor of CDF, but not below the minimum cell rate. That is,

$$\text{ACR} \leftarrow max(\text{MCR}, \text{ACR} - \text{ACR} \times \text{CDF})$$

where the value of CDF can be zero (for no rate decrease), or it can be a power of two that ranges from 1/64 to 1.

This means that after CRM RM cells are sent (or CRM×Nrm total cells are sent), and no backward RM cell is received:

$$\text{ACR} = \text{ACR}_{initial} \times (1 - \text{CDF})$$

Note that if rule 6 is triggered once, it usually triggers on sending successive forward RM cells (as long as no backward RM cells are being received).

Thus, after CRM+1 RM cells (or (CRM+1)×Nrm cells) are sent:

$$\text{ACR} = \text{ACR}_{initial} \times (1 - \text{CDF})^2$$

After CRM+k RM cells (or (CRM+k)×Nrm cells) are sent:

$$\text{ACR} = \text{ACR}_{initial} \times (1 - \text{CDF})^{k+1}$$

Such repeated rate reductions result in an exponential rate drop when source rule 6 triggers, as long as no feedback is being received as shown in figure 4.13.

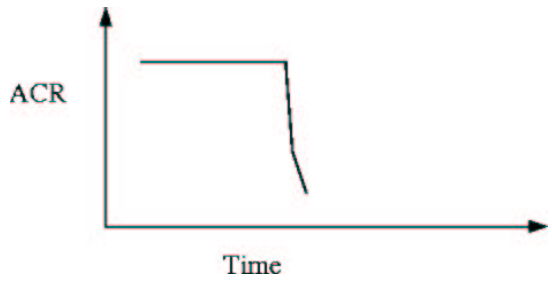


Figure 4.13: Rule 6 results in a sudden drop of rate

The smaller the CDF value, the more rapid the rate decrease when rule 6 is triggered. It is possible to disable source rule 6, by setting CDF to zero. This may

be desirable if TBE cannot be set to a reasonable value, or if TBE must be set to a low value to decrease ICR, but rule 6 should not be triggered unnecessarily. However, disabling rule 6 in this manner risks a large amount of cell loss in case of link failures or congestion collapse. CDF may be set to smaller values for high speeds and long RTTs to avoid big losses. It is set according to the application type, confidence in TBE value, confidence in links, and availability of resources.

ADTF

As previously mentioned, the purpose of the ADTF timeout is to avoid the ACR retention problem that may cause congestion. ACR retention can cause sudden queue growth of:

$$(ACR - \text{source rate}) \times \text{feedback delay} \times (\text{number of sources} - 1)$$

VCs that disable rule 5 (e.g., by setting ICR=PCR) can be vulnerable to sudden arrivals. The default value of 500 ms was selected to correspond to the timer granularity used with most TCP/IP implementations using slow start.

ADTF especially affects bursty traffic. ADTF is independent of the bottleneck link speed of the connection since traffic is smoothed in the ATM network. ADTF should be larger than the round trip time:

$$ADTF > RTT$$

to prevent unnecessary rate reductions for long round trip times. Sources can set ADTF according to the application traffic characteristics (the expected burstiness of the traffic). Switches can reduce ADTF if they have little available resources.

4.6 Upper Bound on Out of Rate RM Cells: TCR

Although the tagged cell rate (TCR) is not signaled, we include a brief discussion here on its role, and the significance of the value chosen for it.

4.6.1 Role

As stated in **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.

4.6.2 Values

Although higher TCR values improve transient response with zero or very low ACRs, since feedback is more frequent, increased TCR does increase the RM cell overhead in such cases. Rescheduling becomes important in cases where ACR is very low and the new ACR will allow cells to be scheduled earlier than their previously scheduled time [70]. There are currently no guidelines on how to space out-of-rate RM cells.

It seems like TCR should depend on the bottleneck link speed, and perhaps a ratio, such as Nrm, should be used. 10 cells per second may be too low for very high speeds, e.g., 2.4 Gbps+. It would be better to state, for example, that no more than a certain percentage, say $2.7 \times 10^{-5}\%$, of the link bandwidth should be used for out-of-rate RM cells. The value $2.7 \times 10^{-5}\%$ is based on the intuition that 10 cells per second seems to be a good value for OC-3 links (10 cells per second out of 365 cells per millisecond).

4.7 Chapter Summary

Tables 4.3 and 4.4 summarize the discussion in this chapter. For each of the parameters, the table indicates what the value the source end system sets for the parameter, how switches and destinations negotiate the parameter, how the parameter is affected by link speeds, and how it is affected by the round trip time of the connection.

Parameter	**Speed?**	**RTT?**	**Source initializes according to**	**Switch/Dest. modifies according to**
PCR	increases	no effect	link bandwidth or host/application capacity and pricing	bottleneck link bandwidth
MCR	increases	no effect	application requirements (e.g, video) and pricing	connection admission control (available resources)
ICR	increases	source takes minimum of signaled ICR and $\frac{TBE}{FRTT}$	pricing, host capacity and application	buffering and resources, PCR and FRTT
Nrm	maybe should increase with speed	no effect	processing speed and application type (real-time should increase it)	switch scheme and switch speed
Trm	decreases	no effect	processing speed and application type	switches can reduce Trm for a high PCR, or increase it for low switch speed

Table 4.3: Parameter value recommendations

Parameter	**Speed?**	**RTT?**	**Source initializes according to**	**Switch/Dest. modifies according to**
RIF	no, but may be decreased	no, but may be decreased	application requirements	EFCI and RRM switches and ER switches sensitive to RIF should reduce it depending on FRTT, PCR and scheme
RDF	no, but may be decreased	no, but may be decreased	application requirements	EFCI and RRM switches should reduce dependent on FRTT, PCR and scheme
ADTF	no effect	no effect (may increase)	application traffic characteristics	if little available resources, reduce ADTF
TBE	increases	increases	application type, pricing and host capacity	buffering and resources, and PCR and FRTT
CDF	may be smaller for high speeds	may be smaller for long RTTs	application type, confidence in TBE value	confidence in TBE value, confidence in links, availability of resources

Table 4.4: Parameter value recommendations (cont'd)

CHAPTER 5

ABR RATE ALLOCATION ALGORITHMS

As discussed in section 2.2, switches must constantly measure the demand and available capacity, and divide the capacity fairly among the contending ABR connections. In order to compute the fair and efficient allocation for each connection, a switch may need to determine the effective number of active connections. In this chapter, we propose a method for determining the number of active connections and the fair bandwidth share for each. We prove the efficiency and fairness of the proposed method analytically, and simulate it for a number of configurations. We also examine an algorithm that eliminates the need of such an estimation.

5.1 Introduction

Determining the fair bandwidth share for the active ABR connections is an extremely complex problem because fairness is commonly measured by the max-min fairness criteria, as defined in section 2.6. Intuitively, fairness means that if a connection is bottlenecked elsewhere, it should be allocated the maximum it can use, and the left over capacity should be fairly divided among the connections that can use it. The switch should indicate this fair bandwidth share to the sources, while also accounting for the load and queuing delays at the switch.

This chapter proposes a method to determine the fair bandwidth share for the active ABR connections, and analyzes the performance of this method using both simple mathematical proofs and simulations. The remainder of the chapter is organized as follows. Sections 5.2 and 5.3 point out some problems with the original ERICA+ algorithm, and describe how ERICA+ has solved these problems. We then describe our proposed method, and give a proof of its correctness, and a number of examples of its operation. We give a sample of the simulation results of the algorithm. Section 5.6 proposes an alternative method that eliminates the need for estimating connection activity, and its performance is also analyzed.

5.2 The Measurement Interval

The standard ERICA+ algorithm (recall section 2.7) measures the required quantities over consecutive intervals and uses the measured quantities in each interval to calculate the feedback in the next interval. The length of the measurement 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 the same feedback value per source during each measurement interval. Longer intervals produce better averages, but slow down the rate of feedback.

The ERICA+ algorithm estimates the number of active VCs to use in the computation of the fair share by considering a connection active if the source sends *at least one cell during the measurement interval.* This can be inaccurate if the source is sending at a low rate and the measurement interval is short. In this chapter, we propose a better method for estimating the number of active connections. The new

method is not as sensitive to the length of the measurement interval. It also eliminates the need to perform some of the steps of the ERICA+ algorithm, as described in the next section.

5.3 ERICA+ Fairness Solution

Assuming that the measurements do not exhibit high variation, the original ERICA+ algorithm converges to efficient operation in all cases. The convergence from transient conditions to the desired operating point is rapid, often taking less than a round trip time. We have, however, discovered cases in which the original 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 connections which are bottlenecked upstream, (3) the source rate for all remaining connections is greater than the $FairShare$. In this case, the system remains in its current state, because the term CCR/z is greater than $FairShare$ for the non-bottlenecked connections.

ERICA+ overcomes this problem by remembering the highest allocation made during each measurement interval, and ensuring that all eligible connections can get this high allocation. To do this, $MaxAllocPrevious$ stores the maximum allocation given in the previous interval. For $z > 1 + \delta$, where δ is a small fraction, we use the basic ERICA+ algorithm and allocate Max (FairShare, VCShare). But, for $z \leq 1+\delta$, we attempt to make all the rate allocations equal, by assigning ER to Max (FairShare, VCShare, MaxAllocPrevious). 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 remainder of this chapter proposes a more accurate method to compute the max-min fair shares for all the contending connections.

5.4 An Accurate Method to Determine the Fair Bandwidth Share

As previously discussed, ERICA+ determines the number of active connections by considering a source as active if at least one cell from this source is sent during the measurement interval. A more accurate method to compute activity and eliminate the need for the proposed solution to the fairness problem is to compute a quantity that we call the "effective number of active VCs" and use this quantity to compute the $FairShare$, as described next.

5.4.1 Basic Idea

We redefine the $FairShare$ quantity to be ***the maximum share a VC could get at this switch under max-min fairness criteria***. Hence, the $FairShare$ is calculated as follows:

$$\text{FairShare} = \frac{\text{ABR capacity}}{\text{Effective number of active VCs}}$$

The main innovation is the computation of the effective number of active VCs. The value of the effective number of active VCs depends on the activity level of each of the VCs. The activity level of a VC is defined as follows:

$$\text{Activity level} = \text{Min}(1, \frac{\text{Source Rate}}{FairShare})$$

Thus, VCs that are operating at or above the $FairShare$ are each counted as one. The VCs that are operating below the $FairShare$ (and are probably not bottlenecked at

this switch) only contribute a fraction. The VCs that are bottlenecked at this switch are considered fully active while other VCs are considered partially active.

The effective number of active VCs is the sum of the activity levels for all VCs:

$$\text{Effective number of active VCs} = \sum_i \text{Activity level of VC}_i$$

Note that the definition of activity level depends upon the $FairShare$, and the definition of the $FairShare$ depends upon the activity levels. Thus, the definitions are recursive.

5.4.2 Examples of Operation

Example 1 (stability):

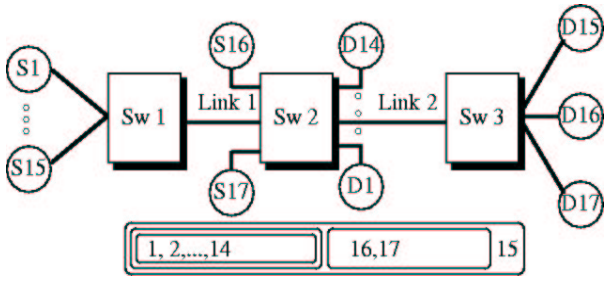


Figure 5.1: The upstream configuration demonstrates fairness

Consider the upstream bottleneck case with 17 VCs shown in figure 5.1. It has been shown in [66] that this configuration demonstrates the necessity of the ERICA+ step described in section 5.3.

Assume that the target capacity is 150 Mbps. For the second switch, when the rates for ($S1$, $S16$, $S17$) are (10, 70, 70):

Iteration 1: Assume FairShare = 70 Mbps

Activity = (10/70, 70/70, 70/70) = (1/7, 1, 1)

Effective number of active VCs = 1 + 1 + 1/7 = 15/7

Iteration 2: FairShare = Target capacity/Effective number of active VCs = 150/2.14 = approximately 70 Mbps

Hence, this example shows that the system is stable at the allocation of (10, 70, 70). At any other allocation, the scheme will calculate the appropriate $FairShare$ that makes the allocation eventually reach this point, as seen in the next two examples.

Example 2 (rising from a low FairShare):

For the same configuration, when the rates are (10, 50, 90):

Iteration 1: Assume FairShare = 50 Mbps

Activity = (10/50, 50/50, 1) = (0.2, 1, 1)

Effective number of active VCs = 0.2 + 1 + 1 = 2.2

Iteration 2: FairShare = 150/2.2 = approximately 70 Mbps

Again, the scheme reaches the optimal allocation within a few round trip times.

Example 3 (dropping from a high FairShare):

For the same configuration, when the rates are (10, 50, 90):

Iteration 1: Assume FairShare = 75 Mbps

Activity = (10/75, 50/75, 1) = (0.13, 0.67, 1)

Effective number of active VCs = 0.13 + 0.67 + 1 = 1.8

Iteration 2: FairShare = 150/1.8 = 83.3 Mbps

Suppose the sources start sending at the new rates, except for the first one which is bottlenecked at 10 Mbps. Also assume that FairShare is still at 83.3 Mbps.

Activity = (10/83.3, 83.3/83.3, 83.3/83.3) = (0.12, 1, 1)

Effective number of active VCs = 0.12 + 1 + 1 = 2.12

FairShare = 150/2.12 = approximately 70 Mbps

Again, the scheme reaches the optimal allocation after the sources start sending at the specified allocations, which is within a few round trip times.

5.4.3 Derivation

The following derivation shows how we have verified the correctness of our method of calculation of the number of active connections. The new algorithm is based upon some of the ideas presented in the MIT scheme [16, 15]. The derivation depends on classifying active VCs as either underloading VCs or overloading VCs. A VC is *overloading* if it is bottlenecked at this switch; otherwise the VC is said to be *underloading*. In the MIT scheme, a VC is determined to be overloading by comparing the computed $FairShare$ value to the desired rate indicated by the VC source. In our scheme, we classify a VC as overloading if its source rate is greater than the $FairShare$ value. Our algorithm only performs one iteration every measurement interval, and is not of the complexity of the order of the number of VCs, as with the MIT scheme.

The MIT scheme has been proved to compute max-min fair allocations for connections within a certain number of round trips (see the proof in [15]). We prove that the MIT scheme reduces to our equation as follows. According to the MIT scheme:

$$FairShare = \frac{\text{ABR Capacity} - \sum_{i=1}^{N_u} Ru_i}{N - N_u}$$

where:

Ru_i = Rate of i^{th} underloading source ($1 \leq i \leq N_u$)

N = Total number of VCs

N_u = Number of underloading VCs

Substituting N_o for the denominator term, this becomes:

$$\text{FairShare} = \frac{\text{ABR Capacity} - \sum_{i=1}^{N_u} Ru_i}{N_o}$$

where:

N_o = Number of overloading VCs ($N_u + N_o = N$)

Multiplying both sides by N_o, we get:

$$FairShare \times N_o = \text{ABR Capacity} - \sum_{i=1}^{N_u} Ru_i$$

Adding $\sum_{i=1}^{N_u} Ru_i$ to both sides produces:

$$FairShare \times N_o + \sum_{i=1}^{N_u} Ru_i = \text{ABR Capacity}$$

Factoring $FairShare$ out in the left hand side:

$$FairShare \times (N_o + \sum_{i=1}^{N_u} \frac{Ru_i}{FairShare}) = \text{ABR Capacity}$$

Or:

$$FairShare = \frac{\text{ABR Capacity}}{N_o + \sum_{i=1}^{N_u} \frac{Ru_i}{FairShare}}$$

Substituting N_{eff}, we get:

$$FairShare = \frac{\text{ABR Capacity}}{N_{eff}}$$

where:

$$N_{eff} = N_o + \sum_{i=1}^{N_u} \frac{Ru_i}{FairShare}$$

This means that the effective number of active VCs is equal to the number of overloading sources, plus the fractional activity of underloading sources.

5.4.4 Algorithm Pseudo-code

This section gives the pseudo-code of the algorithm.

The following variables are introduced:

- N_{last}: Effective number of active VCs in the last measurement interval.
- $N_{current}$: Effective number of active VCs being accumulated for the current measurement interval.
- Activity: This array is maintained for each VC. It is set to one for overloading sources (an overloading source is a source whose CCR exceeds its $FairShare$ value). The activity of a VC is set to the fraction obtained from dividing the CCR of the VC by the $FairShare$ value in the case of underloading sources.
- FirstCellSeen: This is also maintained for each VC, and is only used to avoid the initialization effects of the VC. It is one bit that is set to one if the VC has shown any sign of activity; otherwise, it is set to zero.
- VCsSeen: The sum of the VCs whose FirstCellSeen flag is set. Also used to avoid initialization effects.

INITIALIZATION:

1. N_{last} = number of VCs set up
2. $FairShare$ = ABR Capacity/N_{last}
3. $N_{current} = 0$
4. VCsSeen = 0

5. FOR ALL VCs DO

   Activity [VC] = 0

   FirstCellSeen [VC] = 0

   END (* FOR *)

6. Initialize other ERICA+ variables

END OF MEASUREMENT INTERVAL:

1. IF (VCsSeen >= N_{last})

   N_{last} = max (1, $N_{current}$)

   END (* IF *)

2. $N_{current} = 0$

3. $FairShare$ = ABR Capacity/N_{last}

4. FOR ALL VCs DO

   Activity [VC] = min (1, CCR [VC]/$FairShare$)

   $N_{current} = N_{current}$ + Activity [VC]

   END (* FOR *)

5. Update Overload Factor, and update or reset other ERICA+ variables

CELL IS RECEIVED IN FORWARD DIRECTION:

1. Do NOT update $N_{current}$ as used to be done with ERICA+

2. IF (NOT FirstCellSeen [VC]) THEN

   FirstCellSeen [VC] = 1

VCsSeen = VCsSeen + 1

END (* IF *)

3. Update CCR [VC]

BRM CELL TO BE SENT IN REVERSE DIRECTION:

ER in BRM cell = Max (FairShare, CCR [VC]/Overload Factor)

Observe that the FirstCellSeen array and the VCsSeen counter are only used for the purpose of removing initialization effects from the simulation, and will not exist in a real implementation. Thus, in a real implementation, no steps (other than source rate estimation) will be carried out when a cell is seen, which means that the algorithm will have a low complexity.

5.5 Performance Analysis of ERICA+ with Maximum Share Estimation

The new algorithm has been tested for a variety of networking configurations using several performance metrics. The results were similar to the results obtained with the ERICA+ algorithm [66], except that the new algorithm is max-min fair (without executing the steps described in section 5.3), and is less sensitive to the length of the measurement interval. A sample of the results demonstrating fairness is described in this section [35].

5.5.1 Parameter Settings

Throughout our experiments, the following parameter values are used:

1. All links have a bandwidth of 155.52 Mbps.

2. All 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 to a fixed value of 90%. We use this factor to scale down the ABR capacity term. Alternatively, a queue control function can be used to achieve a target queuing delay and queue lengths [73].

7. The switch measurement interval was set to the minimum of the time to receive 100 cells and 1 ms.

8. All sources are deterministic, i.e., their start/stop times and their transmission rates are known.

5.5.2 Simulation Results

In order to test fairness, we have simulated a three source configuration where one of the sources is bottlenecked at a low rate (10 Mbps). Hence, even though the network gives that source feedback to increase its rate, it never sends at a rate faster than 10 Mbps. The other two sources start transmission at different ICR values. The aim of the configuration is to examine whether the two non-bottlenecked sources will reach the same ACR values, utilizing the bandwidth left over by the

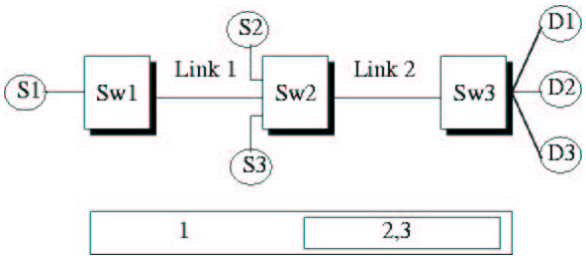


Figure 5.2: A simple three source configuration with a source bottleneck demonstrates fairness

first source. (A number of other configurations was simulated including the standard GFC-2 configuration, and the results indicated good performance of the proposed algorithm.)

Figure 5.2 illustrates the configuration simulated. Note that the round trip time for the $S2$ and $S3$ connections is 30 ms, while that for the $S1$ connection is 40 ms. This configuration is almost identical to the one used in the examples in section 5.4 (figure 5.1), except that connection $S1$ to $D1$ is bottlenecked at the source $S1$ itself, and not at "Link 1." The reason we chose to demonstrate a source bottleneck situation here (and not a link bottleneck situation like figure 5.1) is to demonstrate the effect of using the CCR field in the RM cells versus measuring the source rate.

The results are presented in the form of two graphs for each configuration:

(a) Graph of allowed cell rate (ACR) in Mbps over time for each source.

(b) Graph of the effective number of active VCs N_{eff} at the bottleneck port.

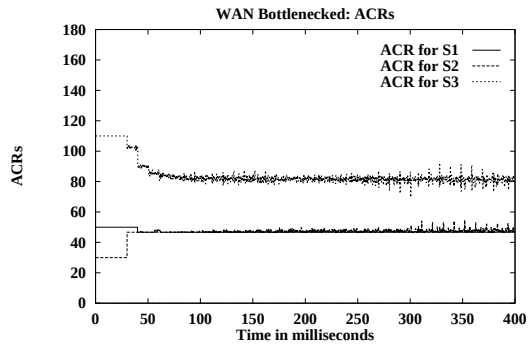
Figure 5.3 illustrates the performance of the original ERICA+ algorithm without the fairness step discussed in section 5.3. Source $S1$ is the bottlenecked source. Sources $S2$ and $S3$ start sending at different ICR (and hence ACR) values. Their ICR values and that of $S1$ add up to little more than the the link rate, so there

is little overload. Observe that the rates of $S2$ and $S3$ remain different, leading to unfairness. The number of active VCs is counted using the original ERICA+ method, so the switch sees 3 sources (see figure 5.3(b)), and the $FairShare$ value remains at around 50 Mbps. Hence, the source $S2$ never increases its rate to make use of the bandwidth left over by $S1$ and only $S3$ utilizes this bandwidth.

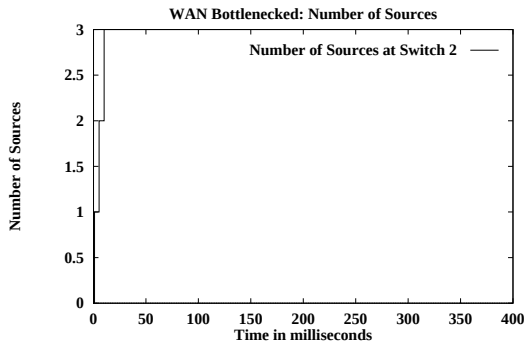
Figure 5.4 illustrates how the fairness problem was overcome in ERICA+ by the change described in section 5.3. In this case, the sources are given the maximum allocation in case of underload or unit load, and hence all sources get an equal allocation. The modified algorithm is max-min fair.

Figure 5.5 illustrates the results with the new method to calculate the fair share of the bandwidth. Observe that the allocations are max-min fair in this case, without needing to apply the maximum allocation algorithm as in the previous case. This is because the method of calculation of the effective number of active connections is different. Figure 5.5 shows that after the initialization period, the effective number of active VCs stabilizes at 1 (for $S2$), plus 1 (for $S3$), plus 10/50 (for $S1$), which gives $1 + 1 + 0.2 = 2.2$ sources. The method also stabilizes to the correct number even *if the length of the measurement interval is short*, unlike the original method where the length of the measurement interval must be long enough to detect cells from all sources, even low-rate sources.

The proposed method works correctly for all cases when there are *link bottlenecks* at various locations (e.g., the configuration in figure 5.1), since it correctly calculates the activity level of each connection based on its CCR value. However, observe that in *source bottleneck* cases, the CCR value cannot be simply obtained from the forward RM cells, but must be measured by the switches. This is because, in source bottleneck



(a) Allowed Cell Rate in Mbps



(b) Number of Active VCs

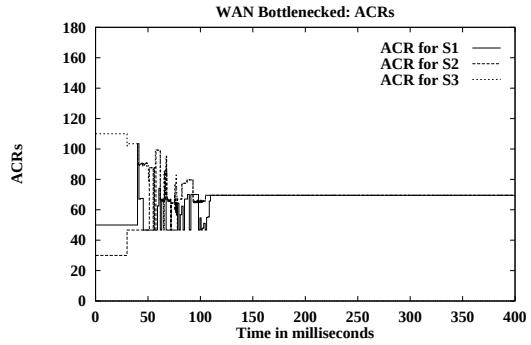
Figure 5.3: Simulation results for a WAN three source bottleneck configuration with the original ERICA+

situations, the source indicates its ACR value in the CCR field of the RM cell, but the source may actually be sending at a much lower rate than its ACR.

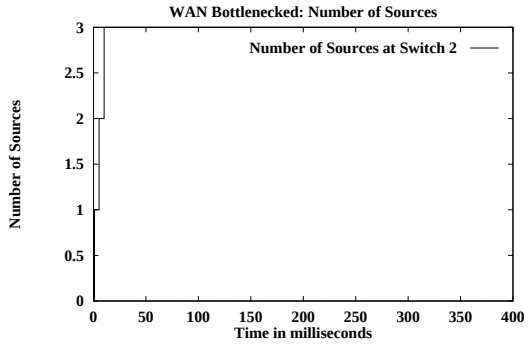
For example, for the configuration discussed above (figure 5.2), assume that we were relying on the CCR values in the RM cells. Figure 5.6 shows that the new method is not fair in this case, since source $S1$ indicates an ACR of 50 Mbps so the effective number of active connections stabilizes at 3 (see figure 5.6(b)), and the $FairShare$ remains at 50 Mbps. But source $S1$ is only sending at 10 Mbps. CCR measurement at the switch detects this, and hence arrives at the correct allocation as shown in figure 5.5.

5.5.3 Observations on the Results

From the simulation results, we can make the following observations about the performance of the proposed algorithm:

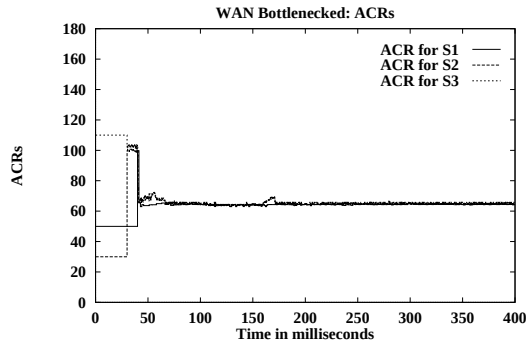


(a) Allowed Cell Rate in Mbps

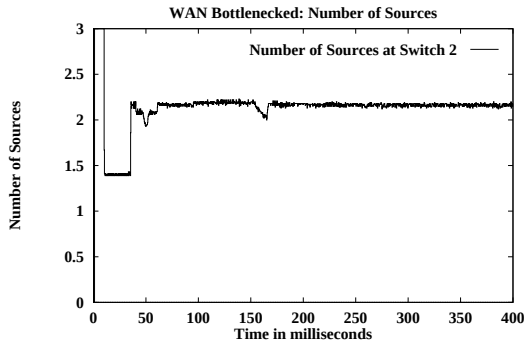


(b) Number of Active VCs

Figure 5.4: Simulation results for a WAN three source bottleneck configuration with ERICA+

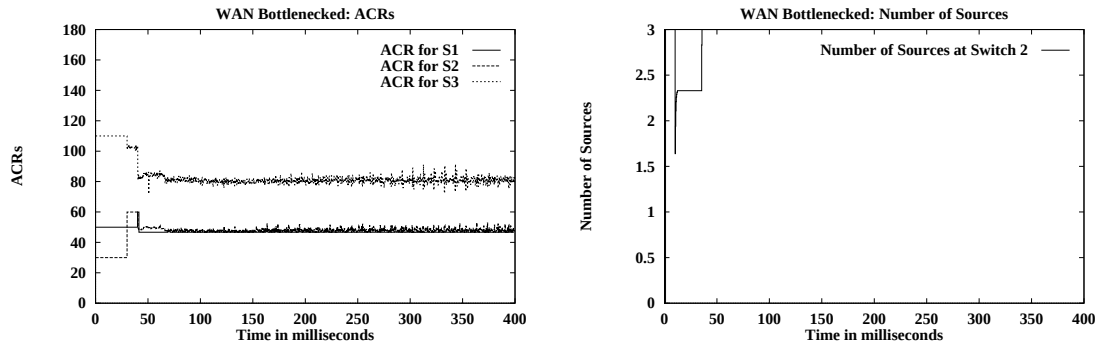


(a) Allowed Cell Rate in Mbps



(b) Number of Active VCs

Figure 5.5: Simulation results for a WAN three source bottleneck configuration with the proposed ERICA+ and source rate measurement at the switch



(a) Allowed Cell Rate in Mbps

(b) Number of Active VCs

Figure 5.6: Simulation results for a WAN three source bottleneck configuration with the proposed ERICA+

- During transient phases, if the $FairShare$ value increases, the N_{eff} value decreases (since it uses the $FairShare$ value in the denominator), and $FairShare$ further increases (since it uses N_{eff} in the denominator), so N_{eff} further decreases, and so on, until the correct values of rates, N_{eff} and $FairShare$ are reached. Then the proposed scheme is provably fair and efficient in steady state (see figure 5.5(a) and (b)). Although the scheme is recursive, its transient response was found to be very fast.

- Even if the measurement interval is so short such that no cells are seen from many low-rate sources, the proposed method can compute the $FairShare$ of the bandwidth correctly (this result is not shown by the simulations in this chapter).

- Without source rate measurement at the switch for each VC, the value of N_{eff} depends on the source ACR, which is not the same as the source rate for source

bottleneck cases. Thus, N_{eff} is too large in those cases, and the $FairShare$ term is less than the CCR by Overload term, leading to unfairness. With per-VC source rate measurement, the value of N_{eff} is correct.

5.6 ERICA+ without Estimation of Number of Connections

One alternative to estimating the number of effective connections, is to avoid using that quantity altogether. Figure 5.7 gives a simplified flow chart of a modified ERICA+ algorithm which avoids using such estimates. This algorithm also supports non-zero MCR values and weights. Again, time is divided into successive intervals, and the algorithm performs a number of computations at the end of every interval. These computations include estimation of the load on the network, estimation of the ABR capacity, and averaging out the values across successive intervals to smooth out measurement variations.

When a BRM cell is received, the algorithm needs to compute the rate to allocate to this connection. It first computes the overload factor as shown in the first step in figure 5.7. The overload factor is the ratio of (A) the average total "excess" load (i.e., after subtracting the MCR values), to (B) the average excess ABR capacity (also after subtracting MCR values), scaled by the queue control function as explained above.

In the next step, the overload is compared to $1+\delta$ (usually δ is set to 0.1). If the overload is greater than 1.1, which means there is high overload, the algorithm scales down the current cell rate of the connection (in excess of MCR) by the overload factor, and then adds the MCR (refer to the rectangle on the right in figure 5.7). This brings down the load. Otherwise, if there is underload (overload is $\leq 1+\delta$), the algorithm also uses an additional quantity. This quantity is the weighted (according

to user specified weights) excess (over MCR) maximum allocation allocated during the previous interval (also averaged). This quantity is the second parameter to the "max" operation in figure 5.7. Bringing up all allocations to this quantity ensures that all connections get fair rates according to the specified weights. Thus the algorithm guarantees MCRs, controlled queuing delays, and weighted allocations. The algorithm is described and analyzed in [128].

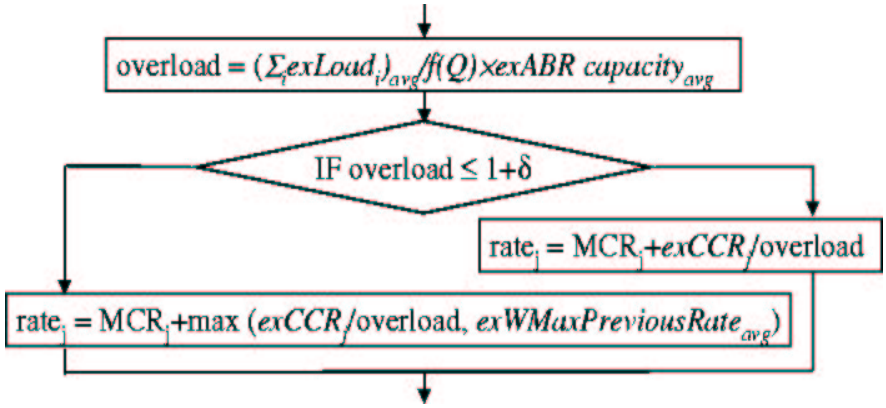


Figure 5.7: The rate allocation algorithm provides weighted fairness with MCR support and controls network queues.

5.7 Performance Analysis of ERICA+ without Estimation of Number of Connections

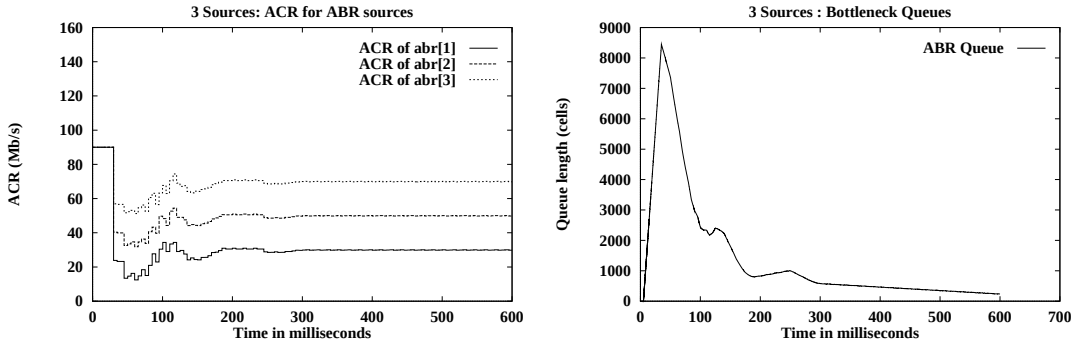
In this section, we give a sample simulation result using the ABR rate allocation algorithm discussed in the previous section. We show a simple configuration here, since the results are easy to interpret. The scheme has also been simulated with more complex configurations, such as the GFC-2 configuration [128]. We use persistent sources (always sending at ACR) in the simulation. The data traffic is only one way,

from source to destination. Using two-way traffic produces similar results, except that the convergence time is longer since the BRM cells travel with traffic from the destination to the source. The network configuration simulated has three sources sending data to three destinations over two switches and a bottleneck link, as shown in figure 2.4 ($n = 3$). All the link bandwidths are 149.76 Mbps (accounting for SONET overhead). The link distances were 1000 km each link.

An ERICA+ interval length of 5 ms was used. As previously mentioned, a dynamic queue control function achieves a constant queuing delay in steady state. The "target delay" parameter (mapped to Q_0 used in figure 2.5) specifies the desired queuing delay. A value of 1.5 ms was used. The hyperbolic function curve parameters used were $a = 1.15$ and $b = 1$. The F_{min} value was set to 0.5. A δ value of 0.1 was used (refer to figure 5.7). Exponential averaging was used to decrease the variation in measured quantities such as the input rate and the available capacity. The exponential averaging parameter used was 0.8.

Weight values of one were used for all connections in this simulation. This corresponds to an allocation of MCR plus an equal share of excess bandwidth for each connection. If weights equal to MCRs are used, the remaining bandwidth will be shared in proportion to the MCR values. The MCR values used in the simulation were 10, 30 and 50 Mbps for sources 1, 2 and 3 respectively. The excess bandwidth $(149.76 - 90 =)$ 59.76 Mbps is divided equally among the three sources. Therefore, the expected allocation vector is $\{10 + 59.76/3, 30 + 59.76/3, 50 + 59.76/3\} = \{29.92, 49.92, 69.92\}$.

Figure 5.8 shows the allowed cell rate values for the three sources. From figure 5.8(a), it can be seen that the expected allocation is achieved for the three sources.



(a) Rates of the 3 sources

(b) Queue length at the bottleneck

Figure 5.8: Simulation results for the three source configuration with ABR show that weighted fairness with MCR is achieved and queues are bounded.

Each source is given its MCR plus an equal share of the remaining bandwidth. The rates converge after a short transient period. Figure 5.8(b) shows that the dynamic queue control function rapidly controls the queuing delay to the specified target. The initially large queues are the result of the large initial cell rate values used for the three sources. We used such large values to show that the algorithm rapidly adapts to the transient overload.

5.8 Chapter Summary

This chapter has proposed and demonstrated two new methods to compute the fair bandwidth share for ABR connections in ATM networks.

The first method relies on distinguishing among underloading connections and overloading connections, and computing the value of the "effective number of active connections." The available bandwidth is divided by the effective number of active connections to obtain the fair bandwidth share of each connection.

The method is provably max-min fair, and can be used to ensure the efficiency and fairness of bandwidth allocations. Integrating this method into the standard ERICA+ tackles the fairness and measurement interval problems of ERICA+, while maintaining the fast transient response, queuing delay control, and simplicity of the ERICA+ scheme.

The second method eliminates the need for estimation of the effective number of active connections altogether, by relying solely on the maximum allocation tracking to achieve fairness.

Analysis and simulation results were used to investigate the performance of both methods. From the results, it is clear that the methods are max-min fair and not excessively sensitive to the measurement interval length.

As a final note, we mention that some switches also implement the use-it-or-lose-it feature. The use-it-or-lose-it concept essentially reduces the rates allocated to any connection that is sending data at a much lower rate. Figure 5.9 demonstrates the problem that can arise when sources are allocated high rates without using them. 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 grow. Reducing the rates of such connections (usually referred to as "ACR retaining" or "ACR promoting" connections) reduces the potential cell loss if such connections all suddenly start transmitting at their full rate. More details on use-it-or-lose-it policies are given in [82].

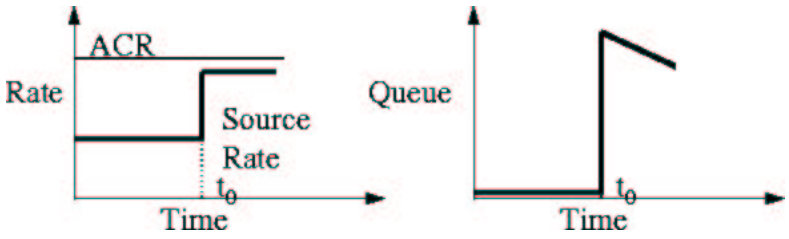


Figure 5.9: ACR retention and promotion can cause sudden network overload when sources use their full ACRs.

CHAPTER 6

ABR VIRTUAL PATHS IN VIRTUAL PRIVATE NETWORKS: ARCHITECTURE AND AGGREGATION ISSUES

Connecting enterprise sites together requires innovative architectures. Virtual private networks (VPNs) linking different organizational sites over the *Internet* are a popular solution. Internet traffic, however, is rapidly growing and becoming increasingly diverse. There is a strong need for quality of service (QoS) support in the Internet.

As ABR service provides a good synergy with the emerging Internet technologies for supporting end-to-end QoS, connecting enterprise networks by ABR virtual path connections (VPCs) can guarantee quality of service and minimize queuing delay and loss in the backbone. In addition, it provides flexibility in supporting various implementations at the edge devices.

The basic concept and architecture developed in this chapter are not only applicable to ATM networks, but also to any network implementing intelligence in the edge device, and flow control in the backbone. This includes frame relay networks using flow control.

6.1 Introduction

Virtual private networks (VPNs) are rapidly gaining popularity. A VPN uses the public Internet to transparently connect private networks or even users, as if they are on the same network. Enterprise sites connected through the Internet are becoming increasingly common, especially within companies with multiple locations separated by long distances (figure 6.1). VPNs provide an attractive solution because of their reduced costs (over leased lines), reduced administration overhead, and support for remote access and collaboration with partners.

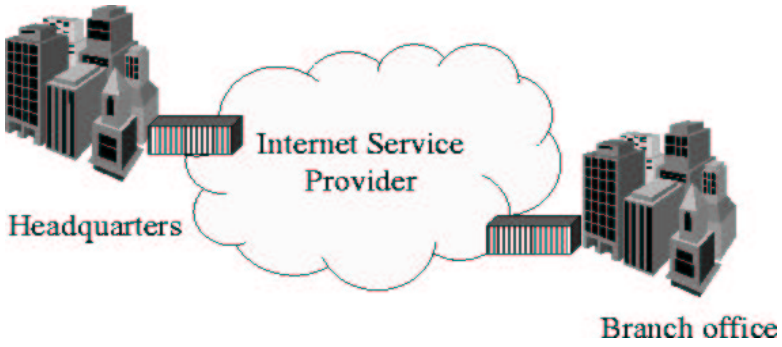


Figure 6.1: Virtual private networks connect enterprise sites over the Internet.

Since ATM is widely deployed in Internet backbones, traffic management for Internet traffic over ATM is becoming an increasingly important problem. Aggregation of Internet (IP) flows is necessary for scalability, overhead reduction, fast re-routing and simplified billing. Examples of aggregation in ATM include the use of virtual path connections (VPCs) that include several VCCs, and sub-multiplexing techniques within a VCC (for example, carrying multiple IP flows within an ATM VCC). In the Internet Engineering Task Force (IETF), Internet differentiated services are the best

example of quality of service for aggregate flows. An example scenario is a customer buying a fixed width pipe, with multiple QoS streams occupying percentages of the pipe: 10% premium or guaranteed, 20% real-time, 30% excellent effort data and 40% best effort data.

This chapter shows that the ATM available bit rate (ABR) service can be used in backbones to connect various enterprise sites with QoS guarantees. The remainder of the chapter is organized as follows. The next section explains our proposed architecture with an emphasis on differentiated services support. Then we discuss the special case of multiplexing ABR connections onto ABR VPCs. We give a framework for developing an algorithm for rate allocation, and apply that framework to the ERICA+ algorithm, giving some sample simulation results.

6.2 Proposed Architecture for Connecting Enterprise Networks

We propose an architecture that employs intelligent edge devices and an ATM backbone to connect enterprise networks, as shown in figure 6.2. Our architecture integrates real-time and data traffic of the enterprise on a single backbone virtual path connection (VPC) between sites. The architecture supports IP differentiated and integrated services traffic and policy control, in addition to ATM and frame relay (FR) traffic. The advantages of separating edge device functionality from backbone functionality include simplification and scalability of the network design and bandwidth management, as well as scalability of the number of connections [109]. Enterprise *voice, video and data* integration within a *single* carrier VPC decreases the costs the enterprise pays (one VPC is used instead of two or more between any two points), and also allows dynamic sharing of voice, video and data bandwidth.

The proposed network is thus a two-tiered network: the outer (access) tier and the inner (backbone) tier. The access tier performs flow identification and QoS management at the flow level. Each switching node manages a relatively small number of flows. It may use ATM, FR, integrated services, or differentiated services for quality of service, or classes of service (COS). Traffic is aggregated at the edge into an ATM backbone (forming the inner tier). The backbone works with aggregate flows, mapped to ATM VPCs or VCCs. The backbone traffic management is simple because of the large number of flows within each connection, and the high speed between the nodes. Backbone traffic management is at the granularity of aggregates, not for traffic within a flow.

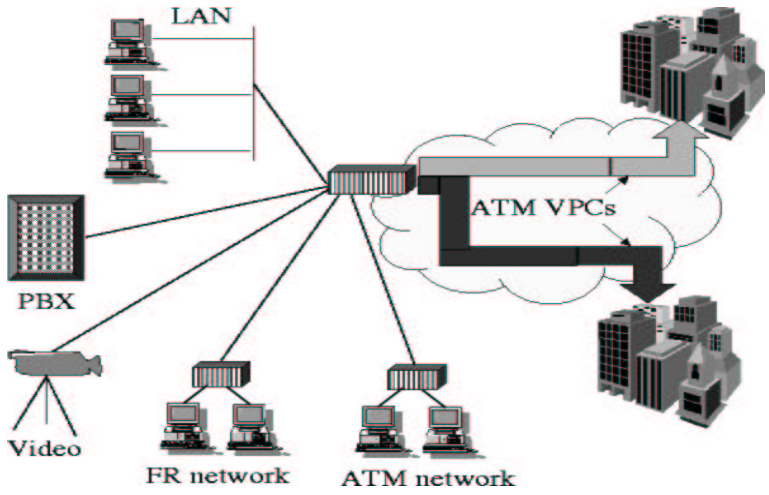


Figure 6.2: The proposed architecture uses a single VPC to connect enterprise sites. Voice, video and data traffic can be multiplexed on this VPC.

This architecture can be used for VPNs, large local area network (LAN) or wide area network (WAN) enterprises, and carrier networks. We will focus on the VPN application. Typically, each enterprise site has a relative abundance of bandwidth

(for example, using Fast or Gigabit Ethernet). The site implements the enterprise policy for managing the traffic. It performs flow identification and classification, QoS assignment, QoS management, and flow mapping within the local area network (the campus or the branch). QoS can be managed through: (1) tagging/marking, (2) dropping, or (3) assigning scheduling priorities. At the edge of the campus enterprise network, traffic is aggregated into the ATM VPCs or VCCs for transport through the carrier network connecting the sites. The edge device uses a weighted fair queuing scheduler for scheduling traffic to the VPC(s), as shown in figure 6.3. The following subsections give more details on the design of the edge device and the choice of ATM service to use in the backbone.

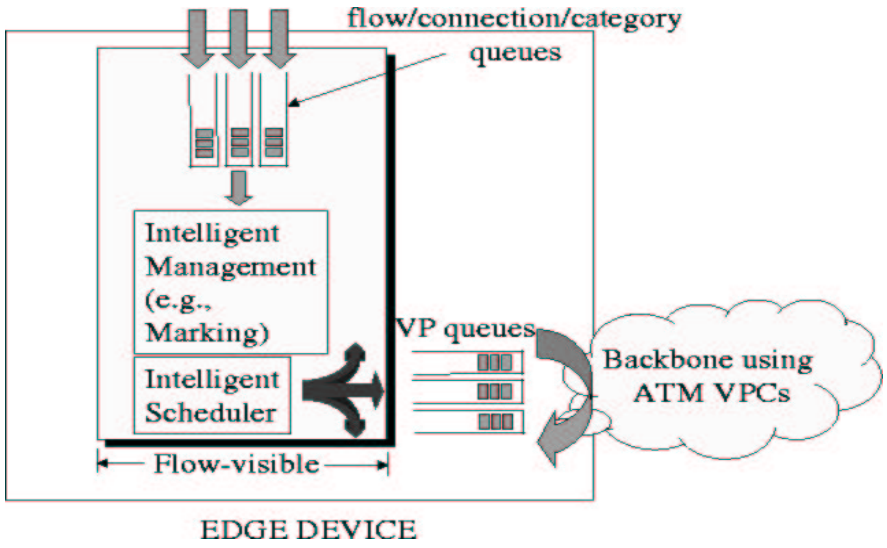


Figure 6.3: The edge device performs traffic management based on the flows, and then intelligently schedules traffic to the backbone VPCs.

6.2.1 The Scheduler

An intelligent scheduling mechanism is required in the edge device to feed traffic from multiple flows into the ATM VPC(s). An example of a scheduler is a weighted fair queuing (WFQ) scheduler for the individual connections or flows into the VPC pipe. Per-connection or per-flow queues may be maintained to control delay and loss, depending on the flow type.

The weights used by the WFQ scheduler for different traffic streams are assigned based upon:

- The enterprise policy rules for users or applications.
- The ATM (or FR) parameters negotiated during connection admission, and the ATM service category, in case of ATM or FR networks at an enterprise site.
- The integrated services requests signaled by the application (if integrated services and the reservation protocol (RSVP) are used at the enterprise site).
- The service requested by the hosts and set in the packet headers using the differentiated services framework (refer to section 6.2.3 for more details).

6.2.2 Choice of Service Category in the Backbone

The choice of service category to use in the ATM backbone is critical to the quality of service experienced by applications sending traffic to another site of the enterprise. As previously mentioned, each site is likely to have abundant bandwidth. Congestion most likely occurs on the relatively low-capacity WAN access link (for example, a Fast or Gigabit Ethernet feeding into a low capacity T1/E1 or T3/E3 link). Depending on the carrier ATM service category, congestion may occur in the carrier network

leading to performance degradation. For example, if VBR is used and the traffic is aggressively shaped to the PCR and not the SCR, losses in the backbone can occur.

ABR performs well in the backbones connecting enterprise networks. The ABR service pushes congestion to the edge devices, where adequate buffering can be provided, and, more importantly, the flows are visible and the enterprise policy can be applied. The ABR VPCs perform flow control for the pipes between enterprise networks. With ABR, there is very little loss in the backbone, and hence higher priority traffic can be transported without loss. On the other hand, the application takes advantage of all the bandwidth given by the network and efficiently utilizes the buffer at the edge device. This is not the case with other services, such as VBR, where either (1) the traffic is shaped according to the SCR to avoid loss in the network, which is clearly inefficient and increases delay, or (2) the traffic is shaped according to the PCR, which risks random losses inside the backbone, unless intelligent cell marking according to SCR is used.

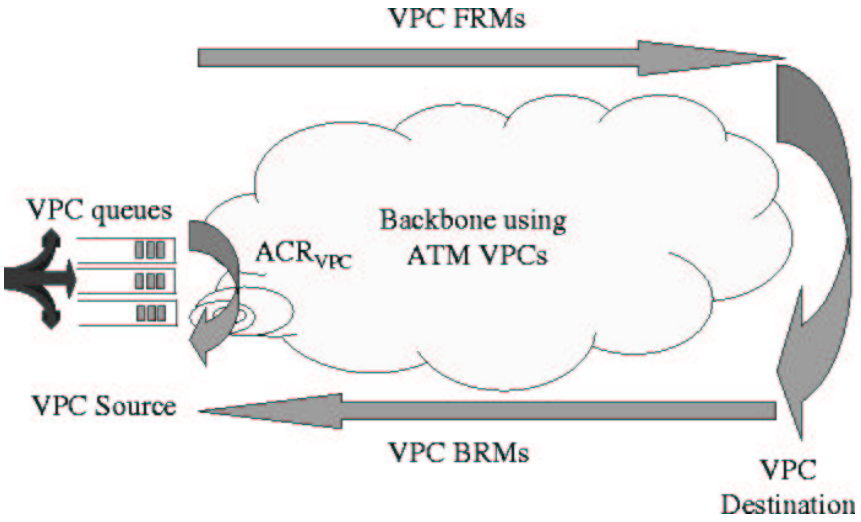


Figure 6.4: ABR VPCs can be used in the network backbones to minimize delay and loss.

Figure 6.4 shows the use of ABR VPCs. Per-VP queues are implemented at the VPC source to control the rate of the ABR VPC to the VP allowed cell rate (ACR), according to the feedback from the VPC BRM cells. In the case of an ABR VCC multiplexed on the ABR VPC, per-VP accounting information and the VPC ACR are used to compute the rate indicated in the VCC BRM cells. Section 6.3 gives an algorithm to perform this rate computation.

As previously mentioned, enterprise real-time and non-real-time traffic can be mixed on an ABR VPC. ABR, however, provides no delay guarantees by the service provider. But the use of minimum cell rate (MCR) guarantees, a good explicit rate (ER) switch algorithm, small switch buffers (thus controlling queuing delays), and an intelligent edge scheduler (as explained in the previous subsection) can give delay and loss guarantees [129]. We emphasize the need for a good ER algorithm to reduce loss.

An example of a good ER algorithm is the ERICA+ algorithm with MCR and weights as explained in chapter 5. ERICA+ gives MCR guarantees, which can provide a minimum acceptable quality of service, even for voice and video applications. The use of weights is allowed to give a generalized form of the fair allocations. Thus, the bandwidth in excess of the MCRs is divided proportional to a predetermined weight, associated with each ABR connection.

6.2.3 Internet Differentiated Services Support

Internet differentiated services enable the deployment of multiple services in large networks, providing an alternative to per-flow processing and per-flow state [6]. The use of the differentiated services model is envisioned in large core networks, and the

use of integrated services with the resource reservation protocol (RSVP) [8] is foreseen to be in peripheral stub networks (for example, campus enterprise networks), as shown in figure 6.5.

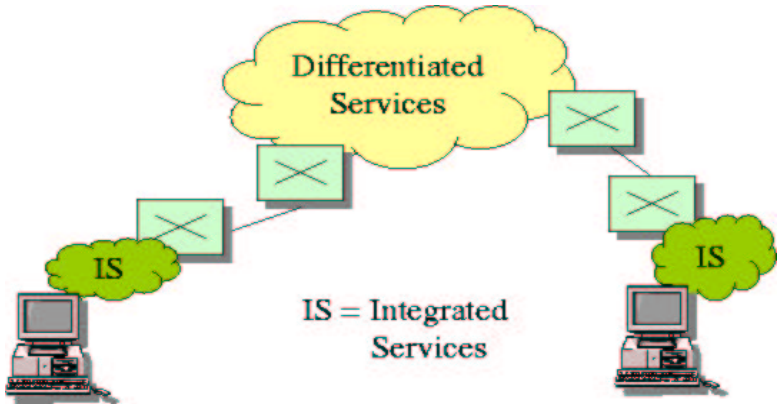


Figure 6.5: Differentiated services are used in Internet backbones, while integrated services are used in peripheral networks.

In large carrier networks, differentiated services IP traffic may be transported over ATM backbones. The mapping of differentiated services to ATM is not straightforward. For example, the Internet assured forwarding behavior provides multiple drop preferences and multiple classes, while ATM has only two drop preferences through the cell loss priority bit, and an unspecified number of queues for each service category.

Consider mapping the assured forwarding behavior onto an ATM backbone using ABR connections. In this case, edge routers can use different drop thresholds for different assured forwarding drop preferences, since most queues are at the edge router itself. Flows are visible at these edge routers, but not inside the ATM network (refer to figure 6.3). CBR may also be used in the backbones, but CBR is unsuitable for bursty traffic. The remaining service categories (rt-VBR, nrt-VBR, UBR and GFR)

cannot easily handle more than two drop preferences since the queues may grow *inside* the network, and it is difficult to control discard priorities at that point (flows are not visible).

Having multiple queues with different priorities and weights (guarantees) may also complicate mapping differentiated services onto ATM. ABR and CBR can implement multiple priority queues by maintaining the queues at the edge routers and multiplexing all the queues onto connections. Again, since flows are visible at the edge routers, intelligent scheduling can be performed there, as discussed in section 6.2.1. This is adequate for giving different guarantees, because ABR (or CBR) queues inside the ATM network are very small, and hence the QoS inside the network is unaffected. The other service categories cannot guarantee bandwidth because priorities must be enforced inside the ATM network, since this is where longer queues exist. Enforcing priorities at the edge router is inadequate, and enforcing priorities inside the network cannot be performed through setting GFR MCR, or VBR SCR and/or PCR.

6.3 Multiplexing ABR VCCs on VPCs

This section proposes an algorithm for aggregating ABR virtual channel connections (VCCs) onto virtual path connections (VPCs). The coupling between the flow control mechanisms for VCCs and VPCs is not standardized. In this section, we propose fairness definitions for rate allocation, and we describe a distributed algorithm for allocating the VPC capacity to the multiplexed VCCs. Preliminary simulation results indicate that the algorithm achieves the required fair allocations, while controlling queue sizes.

First, we give fairness definitions for rate allocation. Then, we propose a framework for the coupling of the VPC and VCC ABR control loops, and use the ERICA+ algorithm as an example mechanism. Preliminary simulation results of the algorithm are then given.

6.3.1 Fair Multiplexing of ABR VCCs on ABR VPCs

The relationship between the service category of the VPC and the VCCs within it is implementation specific. In [109], the authors suggest using a rt-VBR VPC to aggregate CBR and rt-VBR VCCs, and using an ABR VPC to aggregate nrt-VBR, UBR and ABR VCCs. As ABR VPCs provide the more interesting case, we focus on ABR in the remainder of this section.

The ABR service can apply to both VPCs and VCCs. End points of ABR VPCs and those of ABR VCCs comply with the ABR source and destination behavior as given in the specifications [43]. The method used to divide the VPC bandwidth among the VCCs it contains is implementation specific. In the case when link capacity must

be shared between both ABR VPCs and ABR VCCs, the method used to allocate the bandwidth is also implementation specific.

Computation of the ideal allocations for the two level hierarchy (VCCs multiplexed on VPCs) is not straightforward. This is because scenarios are conceivable where a VPC with a larger number of VCCs multiplexed on it should be given more bandwidth than a VPC with a small number of VCCs. The question of how bandwidth is allocated among the VPCs (inter-VPC), and among the VCCs multiplexed on the same VPC (intra-VPC), becomes an important one. This is similar to the intra-group fairness and inter-group fairness for multicast groups discussed in [30, 32].

We use general weighted fairness throughout the remainder of this section.

Example 1: Intra-VPC Fairness:

Consider the simple example in figure 6.6. A VPC has its own flow control loop between the VPC end points (Switch 1 and Switch 3). Assume that the VPC MCR is zero. Suppose that three VCCs are multiplexed on this VPC: a VCC from user A to B, another from user C to D, and a third from user E to F. Assume the 3 VCC MCRs are zero. All available capacities on the links are 150 Mbps, except for the link from user A to Switch 1, which is only 10 Mbps. In this case, the flow control for the VPC will detect that 150 Mbps is available for the VPC, and will allocate it the entire available capacity. The VPC source end system (Switch 1) and the VPC destination end system (Switch 3) will cooperate with the network to regulate the VPC at this rate. The flow control for the VCCs within the VPC will divide the VPC capacity among the active VCCs multiplexed on the VPC. The connection from user A to B will be allocated its bottleneck rate of 10 Mbps. The available capacity of 150 Mbps

– 10 Mbps = 140 Mbps will be equally divided upon the other two connections (C to D and E to F) and each will be allocated $\frac{140}{2} = 70$ Mbps. □

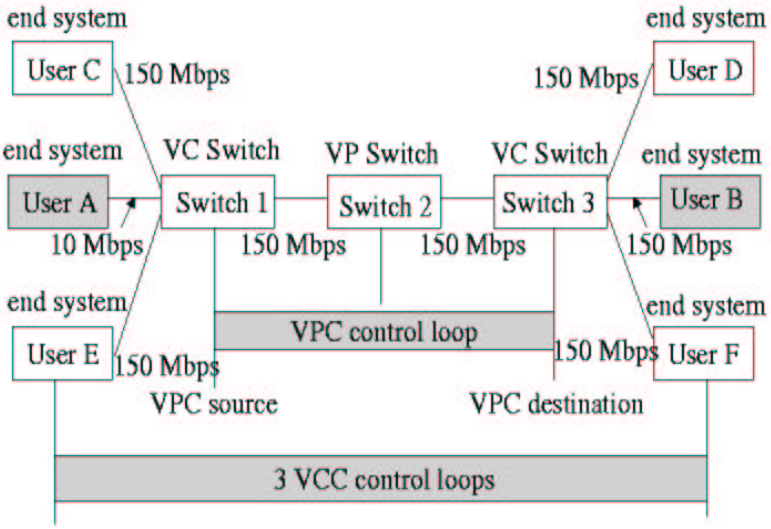


Figure 6.6: Example 1: A single VPC and multiple VCCs

Example 2: Inter-VPC Fairness:

Now consider the example shown in figure 6.7. This is the same as the previous example, except that there is a second ABR VPC between Switch 1 and Switch 3. Suppose that the three VCCs (A to B, C to D, and E to F) are multiplexed on one of the VPCs, while there are 10 VCCs multiplexed on the second VPC (the 10 VCCs are assumed to be bottlenecked on the Switch 1 to Switch 3 path). The weights assigned to the two VPCs at a switch may be equal or different as follows.

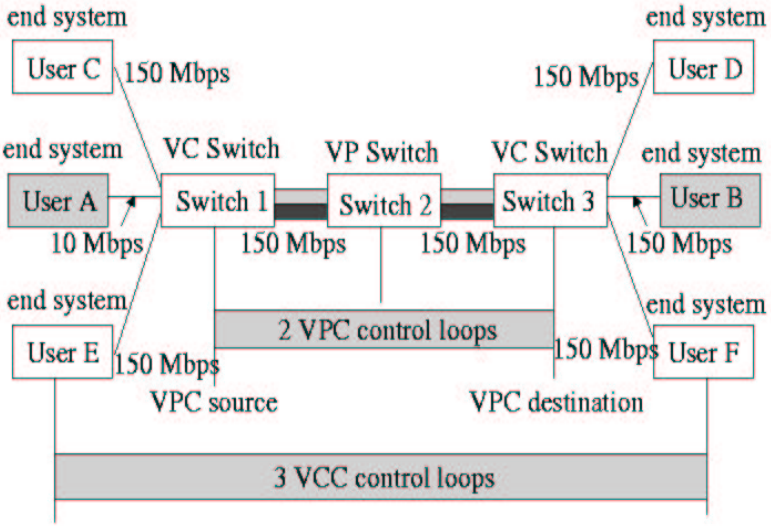


Figure 6.7: Example 2: Two VPCs and multiple VCCs

Case 1: Equal Weights:

$$\forall i, j : i, j \in VPC : i \neq j : w_i = w_j$$

Assuming zero MCRs, each VPC is allocated $\frac{150}{2} = 75$ Mbps. The 75 Mbps is allocated to the 3 VCCs A to B, C to D, and E to F as follows. A to B is allocated 10 Mbps. The remaining bandwidth $75 - 10 = 65$ Mbps is divided equally among the 2 remaining connections so each is allocated $\frac{65}{2} = 32.5$ Mbps.

Case 2: Unequal Weights:

$$\forall i, j : i, j \in VPC : i \neq j : w_i \neq w_j$$

For example, suppose the VPC with 10 VCCs is assigned 5 times the bandwidth of the other VPC. In this case, the VPC with 10 VCCs gets $\frac{5}{6} \times 150 = 125$ Mbps, while the other VPC is allocated 25 Mbps. The 25 Mbps is equally divided upon the three connections, such that each is allocated $\frac{25}{3} = 8.33$ Mbps.

Two interesting special cases arise:

Case 2.1: Weights according to Number of VCCs: Suppose the VPC weights are assigned according to the number of connections multiplexed on them:

$$\forall i, j : i, j \in VPC : i \neq j : \frac{w_i}{NumOfVCC_i} = \frac{w_j}{NumOfVCC_j}$$

In this case, the weight for the VPC with 10 connections is $\frac{10}{3}$ times the weight of the VPC with 3 connections. The weights may need to be updated if new connections join the VPC.

For the above example, the VPC with 10 VCCs is allocated $\frac{10}{13} \times 150 = 115.38$ Mbps and each of the 10 VCCs is allocated $\frac{1}{10} \times 115.38 = 11.54$ Mbps. The VPC with 3 VCCs is allocated $\frac{3}{13} \times 150 = 34.62$ Mbps. User A is bottlenecked at 10 Mbps. Users C and E are allocated $\frac{34.62-10}{2} = 12.31$ Mbps each.

Case 2.2: Weights according to VCC ERs: Suppose the VPC weights are assigned according to the explicit rates of connections multiplexed on them:

$$\forall i, j : i, j \in VPC : i \neq j : \frac{w_i}{\sum_i ER} = \frac{w_j}{\sum_j ER}$$

In this case, the available capacity on each link is divided fairly among the active connections, regardless of which VPC each connection belongs to. The ER (and hence ACR) for the VPC is simply the sum of the ERs for the VCCs it contains. This is a constantly varying quantity.

For the above example, user A is allocated 10 Mbps while all the other users are allocated $\frac{150-10}{12} = 11.67$ Mbps. □

6.3.2 A Framework for Flow Control of ABR VCCs on an ABR VPC

ABR VCCs within a VPC share its capacity in the same way ABR connections share the capacity of a physical link. Virtual source/virtual destination (VS/VD) can be used in the framework as discussed next.

Using VS/VD

One option is to use a virtual destination (VD) for the VCC, and a virtual source (VS) for the VPC at the VPC *source*, and a virtual destination for the VPC, and a virtual source for the VCC at the VPC *destination*. This option is illustrated in figure 6.8.

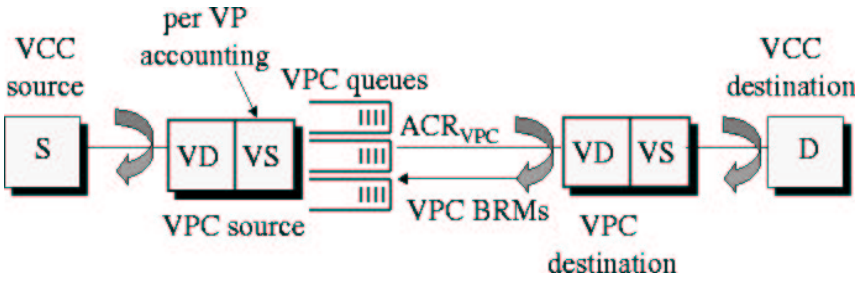


Figure 6.8: Virtual source/virtual destination at the VPC end points

At the VS of the VPC, a separate VPC queue is used to control the VPC rate. The VDs of the corresponding VCCs in the same switch need: (1) per VP accounting information performed at the VPC VS, and (2) the ACR of the VPC, in order to compute the ER values for the VCC.

Terminating/starting the VCC loop at the VPC end points is not required, but it eliminates the per-VC RM cell overhead and VCC RM cell processing inside the VPC loop. Separation of the flow control loops of the VCCs and the VPCs is also useful. VS/VD does incur additional overhead, however, since the end systems and switch functionality must all be provided at the VPC end points.

Without VS/VD

An alternative architecture without VS/VD is shown in figure 6.9. As in the VS/VD case, each VPC has a separate queue at the VPC source. Again, per VP accounting information and the VPC ACR are used to compute the rate indicated in the VCC RM cells at the VPC source. The two architectures and the rate computation operations are quite similar in both cases (with and without VS/VD). In the remainder of this section, we explain the operation of the VCC rate allocation algorithm in more detail.

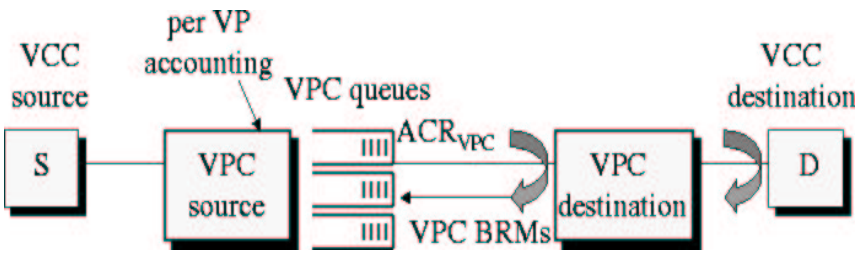


Figure 6.9: VPC/VCC flow control coupling without VS/VD

Flow Control Framework

The framework has two main aspects: capacity estimation, and accounting, as discussed next.

Capacity Estimation. In most ABR rate allocation algorithms, the available capacity for ABR is estimated as follows:

$$\text{Total ABR Capacity} \leftarrow \text{Link Capacity} - \text{CBR/VBR Capacity}$$

This means that higher priority bandwidth is estimated by computing the sum of the number of CBR, rt-VBR and nrt-VBR cells scheduled during a certain interval of time. This sum is then subtracted from the link capacity, and a fraction of that is divided upon the VPCs (other than the CBR/VBR ones) according to the preassigned weights. This ABR capacity estimation operation must be performed by the *VPC* flow control mechanism if a VPC-VCC hierarchy exists. The total ABR capacity for multiplexed VCCs is simply the VPC allowed cell rate (ACR).

Once the total ABR capacity is estimated, the target ABR capacity is computed. For example, ERICA+ [73] computes the target ABR capacity as follows:

$$\text{Target ABR Capacity} \leftarrow \mathit{Fraction} \times \text{Total ABR Capacity}$$

where the *Fraction* can be a constant, or a function of the queuing delay, $f(Q_{VPC})$, of the queue for this VPC at this port of the switch. If the VPC contains VCCs of higher classes (e.g., CBR/VBR) their capacity must first be subtracted from the total ABR capacity.

It is essential to take a *fraction* of the capacity allocated to the VPC. This is because we must allow the VPC queues to drain. These queues are caused by the delay between the instant when the ABR VPC allowed cell rate is controlled to the new value, and the instant the ACRs of all the multiplexed ABR VCCs are controlled. Since there are propagation and queuing delays between the VPC source end system, and the source end systems of the VCCs (refer to figure 6.9), the VPC queue can

grow and must be controlled in the same way any ABR queue (whether a port queue, a VPC queue, or a VCC queue) must be controlled.

Accounting. In addition to the target ABR capacity, other estimates are required to be able to divide the capacity fairly among the active virtual connections. Examples of such metrics used in the ERICA+ scheme are: (1) the ABR input rate, (2) the number of active ABR connections, and (3) the maximum allocation given to any ABR VCC during the previous and current intervals.

In case of a VPC/VCC hierarchy, such computations and estimates must be separately performed for the VCCs on each VPC, and the VCCs on other VPCs should not interfere with this. In other words, estimating the input rate becomes estimating the input rate of the VCCs *on this VPC*, estimating the number of active connections becomes estimating the number of active connections *on this VPC*, and keeping track of the maximum allocation given during a certain interval only considers the allocations given to VCCs *on this VPC*.

Framework Model and Summary. We use the following notation:

$\lambda_{port,VPC}$	input rate of queue for VPC at *port*
$\mu_{port,VPC}$	service rate of queue for VPC at *port*
$Q_{port,VPC}$	queue length of queue for VPC at *port*
ER_{VCC}	explicit rate indicated to the VCC source by the VPC end point
ER_{VPC}	explicit rate indicated to the VPC source
ACR_{VPC}	allowed cell rate computed by the VPC source
$N_{port,VPC}$	number of VCCs multiplexed on VPC at *port*

We need to compute ER_{VCC} such that:

$$\lambda_{port,VPC} \leq f(Q_{port,VPC}) \times \mu_{port,VPC}$$

Or:

$$\sum_{VCC=1}^{VCC=N_{port,VPC}} ER_{VCC} \leq f(Q_{port,VPC}) \times ACR_{VPC}$$

This is performed as follows. Assume that the VPC flow control mechanism assigns an explicit rate value, ER_{VPC} to the VPC (this mechanism must handle the estimation of VBR and CBR bandwidth of other VPCs/VCCs, and the target ABR capacity). The VPC source sets the allowed cell rate of the VPC, ACR_{VPC} to the minimum of ER_{VPC} and $RIF_{VPC} \times PCR_{VPC}$, assuming the CI and NI bits are zero (or decreases the rate by RDF_{VPC} if CI is set) as per the source end system rules specified in [43].

As the VPC source rate must be controlled to ACR_{VPC}, per VP queues are required. The value of ACR_{VPC} must be communicated to the rate allocation algorithm for the VCCs at the VPC end point. The rate allocation algorithm will use this value as the estimated capacity and take a fraction of that (minus the CBR/VBR VCCs on the same VPC) as the target capacity. In addition, the algorithm must perform its accounting, e.g., the accounting of the input rate, active connections and maximum allocation, separately for the VCCs of each VPC.

6.3.3 VPC/VCC ERICA+

We apply the general framework proposed above to the ERICA+ algorithm described in chapter 2. The only modifications required for ERICA+ at the VPC source end system are as follows:

1. The allowed cell rate of each VPC is controlled to ACR_{VPC}.

2. The Target ABR Capacity for the VCCs multiplexed on the VPC is computed as a *fraction* of the ACR of the VPC, ACR_{VPC} (minus the capacity of any

CBR/VBR VCCs on this VPC). The fraction may depend on the queuing delay of the VPC queue $f(Q_{VPC})$ (or the VCC queues for the VCCs multiplexed on this VPC).

3. The ABR Input Rate, Number of Active ABR VCCs, MaxAllocPrevious, and MaxAllocCurrent variables only apply for this VPC. Therefore, per-VP accounting must be performed at each output port.

The following is the pseudo-code of the algorithm.

Initialization:

$\text{MaxAllocPrevious}_{VPC} \leftarrow \text{MaxAllocCurrent}_{VPC} \leftarrow \text{FairShare}_{VPC}$

End of averaging interval:

$$\text{Target ABR Capacity}_{VPC} \leftarrow Fraction_{VPC} \times \text{Allowed Cell Rate}_{VPC} - CBR/VBR_{VPC}$$

$$z_{VPC} \leftarrow \frac{\text{ABR Input Rate}_{VPC}}{\text{Target ABR Capacity}_{VPC}}$$

$$\text{FairShare}_{VPC} \leftarrow \frac{\text{Target ABR Capacity}_{VPC}}{\text{Number of Active VCs}_{VPC}}$$

$$\text{MaxAllocPrevious}_{VPC} \leftarrow \text{MaxAllocCurrent}_{VPC}$$

$$\text{MaxAllocCurrent}_{VPC} \leftarrow \text{FairShare}_{VPC}$$

When an FRM is received:

$\text{CCR[VC]} \leftarrow \text{CCR\_in\_RM\_Cell}$

When a BRM is received:

$$\text{VCShare} \leftarrow \frac{CCR[VC]}{z_{VPC}}$$

IF $(z_{VPC} > 1 + \delta)$

$$
\begin{aligned}
\text{THEN ER} &\leftarrow \text{Max (FairShare}_{VPC}, \text{VCShare)} \\
\text{ELSE ER} &\leftarrow \text{Max (MaxAllocPrevious}_{VPC}, \\
&\quad \text{FairShare}_{VPC}, \text{VCShare)} \\
\text{MaxAllocCurrent}_{VPC} &\leftarrow \text{Max (MaxAllocCurrent}_{VPC}, \text{ER)} \\
\text{IF (ER} > \text{FairShare}_{VPC} \text{ AND CCR[VC]} &< \text{FairShare}_{VPC}\text{)} \\
\text{THEN ER} &\leftarrow \text{FairShare}_{VPC} \\
\text{ER\_in\_RM} &\leftarrow \text{Min (ER\_in\_RM, ER,} \\
&\quad \text{Target ABR Capacity}_{VPC}\text{)}
\end{aligned}
$$

6.3.4 Simulation Results

Figure 6.10 shows the configuration used in our preliminary simulations. The configuration consists of three switches separated by 1000 km links. The one way delay between the switches is 5 ms. Five sources send data as shown in the figure. The first hop from the sources to switch 1 is a long delay satellite hop. We simulated a one way delay of 50 ms (LEO satellite delay). The link capacity of link 2 is 45 Mbps, while all other links are 155 Mbps links. Our simulations use persistent ABR sources. ABR initial cell rates are set to 30 Mbps in all experiments. Link 2 is the bottleneck link for all connections.

The simulations demonstrate the basic idea of the algorithm, although they do not show the exact same implementation discussed above. The implementation used is a VS/VD scheme similar to the one explained in section 6.3.2, but using per VCC queues instead of the single queue for each VPC. Thus the control loops for VCCs are terminated/started at the switches. All sources are multiplexed on a single VPC

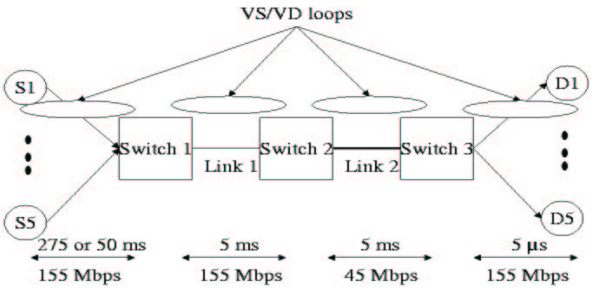
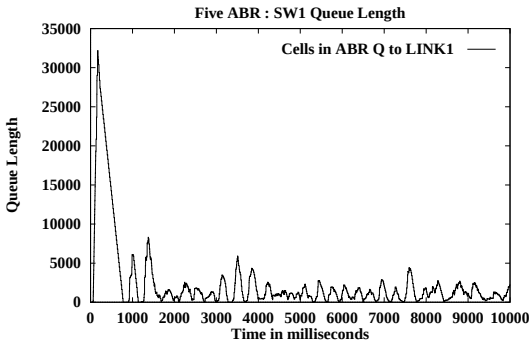


Figure 6.10: Five source satellite configuration

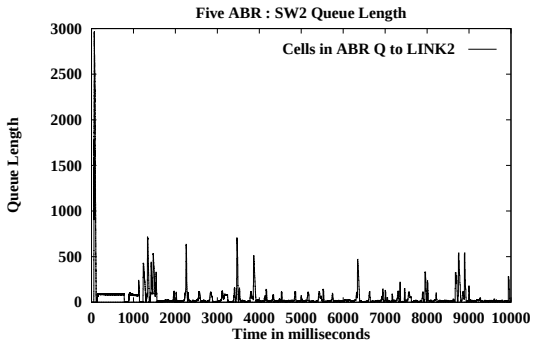
which is allocated a fraction of the link capacity. The resulting VPC ACR becomes the total capacity for all VCCs on this VPC.

Figure 6.11 shows the queue length results. The queue accumulation during the initial open loop period (before the feedback mechanism is in effect) is moved from switch 2 to switch 1 by the VS/VD mechanism. Thus, there are very small queues at switch 2. Pushing the queues to the edge is an important component of the architecture discussed in section 6.2. Moreover, in case of satellite switches as in figure 6.10, it is important to minimize queue length in terrestrial switches (switch 2) which may not have sufficient buffering for an entire satellite round trip. The satellite switch (Switch 1) usually has larger buffers [47].

Figure 6.12 shows the ACRs at the end systems, and at Switch 1 VCC queues. The five sources should each be allocated $\frac{45}{5} = 9$ Mbps. The ACR graphs show that the scheme is fair in the steady state. The transient differences in the ACRs due to the transient differences in the per-VC queue lengths.

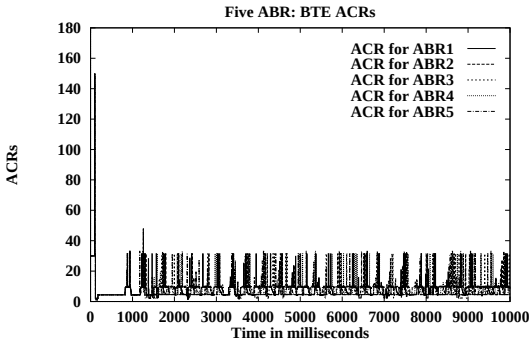


(a) Switch 1 Queue

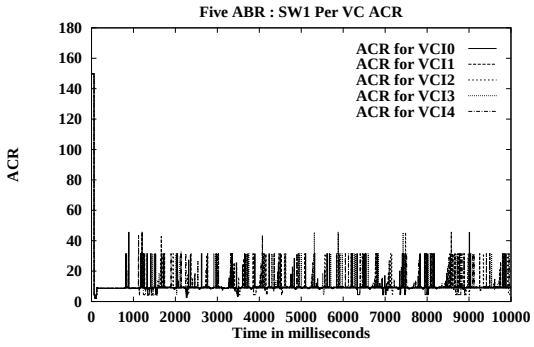


(b) Switch 2 Queue

Figure 6.11: Switch queue lengths for a 5-source LEO configuration



(a) End System ACRs



(b) Switch 1 ACRs

Figure 6.12: Allowed cell rates for a 5-source LEO configuration

6.4 Chapter Summary

This chapter has proposed a framework for connecting enterprise sites on the Internet as a VPN. A single ABR VPC is used to connect two sites, and appropriate scheduling weights and drop policies are employed at the edge devices. This architecture can also be used for supporting differentiated services over ATM through a hierarchical scalable mechanism.

We have also examined the flow control of the ABR virtual path/virtual channel hierarchy. The flow control at the VPC level needs to estimate the bandwidth available for ABR (accounting for CBR/VBR bandwidth), and assign the appropriate weights for different ABR VPCs. We have discussed the issues involved in the VPC/VCC coupling, and have given an example framework. The key aspect of this coupling is the use of the allowed cell rate value for the VPC source as the total capacity available for the VCCs multiplexed on this VPC. This capacity is scaled using the queuing delay of the VPC queue (or the appropriate VCC queues if per-VC queuing is used). In addition, all accounting performed at the output port is performed separately for each VPC. Other VCCs, and VCCs multiplexed on other VPCs, should not interfere with the flow control of VCCs multiplexed on a VPC.

CHAPTER 7

FEEDBACK CONSOLIDATION FOR POINT-TO-MULTIPOINT CONNECTIONS

A number of algorithms have been developed for extending ABR flow control algorithms for point-to-multipoint connections (section 2.10.2. In this case, feedback consolidation is required at the branch points to avoid overwhelming the sender with feedback. This chapter discusses various design options and implementation alternatives for consolidation algorithms, and proposes a number of novel algorithms. The performance of the proposed algorithms and the original algorithms is compared under a variety of conditions. Results indicate that the algorithms we propose eliminate the consolidation noise (caused if the feedback is returned before all branches respond), while exhibiting a fast transient response.

7.1 Introduction

Feedback consolidation at the branch points is necessary for point-to-multipoint connections. The operation of feedback consolidation can be explained by figure 7.1. The consolidation operation avoids the feedback implosion problem, where the number of backward resource management (BRM) cells received by the source is proportional to the number of leaves in the multicast tree. In addition, the allowed rate of

the source should not fluctuate due to the varying feedback received from different leaves.

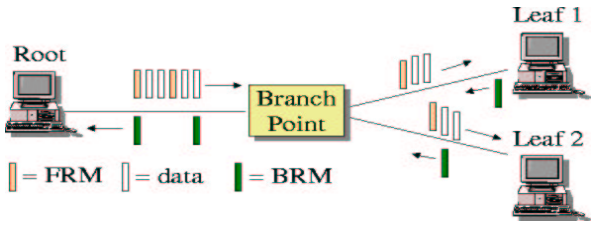


Figure 7.1: Point-to-multipoint ABR connections

In *point-to-point* ABR connections, the source transmits at the minimum rate that can be supported by all the switches on the path from the source to the destination [43]. The natural extension of this strategy for point-to-multipoint connections is controlling the source to the minimum rate that can be supported by the switches on the paths from the source to *all of the leaves* in the multicast tree, as shown in figure 7.2. The minimum rate is the technique most compatible with typical data requirements: no data should be lost, and the network can take whatever time needed for data delivery. If the ABR destinations *can* tolerate a certain amount of cell loss, alternative techniques can be used [77].

A number of consolidation algorithms have been proposed in [14, 19, 78, 77, 96, 98, 103, 105, 106, 113, 127, 134, 135]. Several design and implementation considerations come into play when developing a consolidation algorithm. The oscillations and transient response of the algorithm are important. The algorithm must also be scalable to very large multicast trees. The implementation complexity, feedback delay,

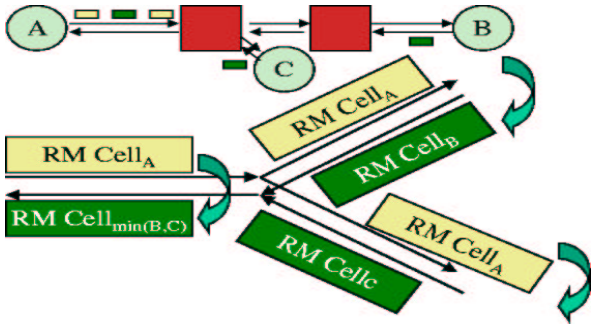


Figure 7.2: Taking the minimum in point-to-multipoint ABR connections

and the overhead of the backward RM cells should not increase with the increase of the number of levels or leaves of the multicast tree.

In this chapter, we propose a set of consolidation algorithms that aim at providing a fast transient response, while eliminating consolidation noise. We examine the performance of the proposed algorithms, and compare it to the previous ones in complexity, transient response, consolidation noise, and scalability. The remainder of the chapter is organized as follows. We begin by discussing the various design and implementation issues involved. An explanation and pseudo-code of the previously proposed consolidation algorithms, as well as the new ones we propose, is presented next. All the algorithms are then simulated and analyzed under a variety of configurations. The chapter concludes with a discussion of the tradeoffs among the algorithms.

7.2 Design Issues

As previously mentioned, there are several ways to implement the feedback consolidation algorithm at branch points. Each method offers a tradeoff in complexity,

scalability, overhead, transient response, and consolidation noise. The tradeoffs can be summarized as follows:

[**A**] Which component generates the BRM cells (i.e., turns around the FRM cells)? Should the branch point, or should the destination, perform this operation?

[**B**] What is the condition to trigger sending a BRM at the branch point? Should the branch point wait for feedback from all the branches before passing the BRM cell upstream? Although this eliminates the consolidation noise, it incurs additional complexity, and increases the transient response of the scheme (especially after idle or low rate periods).

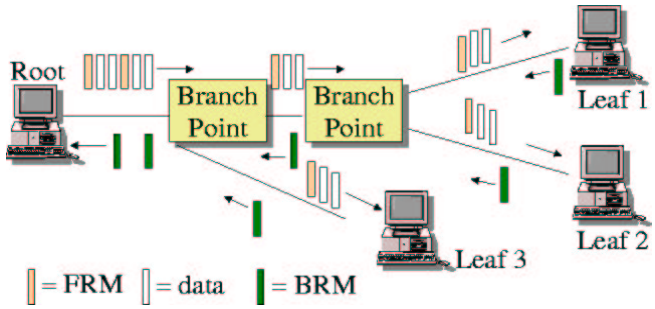


Figure 7.3: A multicast tree with multiple branch points and levels

[**C**] Does the scheme scale well? How can the ratio of FRM cells generated by the source to BRM cells returned to the source be controlled? Will the feedback delay grow with the number of branches? For example, if the algorithm waits for an FRM cell to be received before sending feedback, the delay might increase with the number of levels of the multicast tree (see figure 7.3).

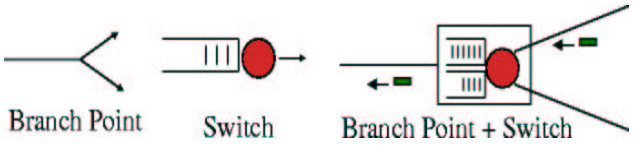


Figure 7.4: Branch points, switches and their coupling

[D] How does the branch point operate when the it is also a switch (queuing point)? (see figure 7.4). The coupling of the switch and branch point functions must be considered. When should the actual rate computation algorithm be invoked?

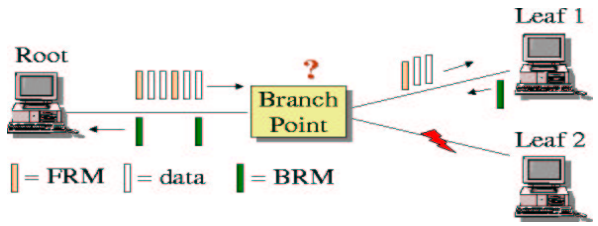


Figure 7.5: A non-responsive branch in a multicast tree should not cause the algorithm to deadlock or cause underload or overload

[E] How are non-responsive branches handled? If the consolidation scheme waits for feedback from *all* the branches before sending a BRM to the source, an algorithm must be developed to determine when a branch becomes non-responsive and handle this case (see figure 7.5). A timeout or outstanding RM cell count mechanism must be implemented. We believe that an outstanding RM cell count (similar to the mechanism used with source end system rule 6 in [43]) is *superior* to the timeout mechanism used in most algorithms, as the timeout value needs to be constantly re-estimated depending on the rate at which the source is sending. Such mechanisms will be the subject of a future study.

[**F**] How is accounting performed at the branch point? Consolidation algorithms use registers to store values such as the minimum rate given by branches in the current iteration, and flags to indicate whether an RM cell has been received since the last one was sent. Some values, if stored per output port, need an efficient data structure. For example, if the rate returned by each branch is stored (instead of only maintaining the minimum value for this connection for this round), a heap is necessary to enable the minimum operation to be rapidly performed.

7.3 Consolidation Algorithms

This section describes some previously proposed consolidation algorithms, while the next section proposes new algorithms. In the algorithms presented, ERICA+ is employed immediately before sending a BRM on the link. This ensures that the most recent feedback information is sent.

We describe the first algorithm (algorithm 1), and then describe the modifications to algorithm 1 resulting in algorithm 2, and so on, until we arrive at our proposed algorithm, algorithm 7. Thus the description of the algorithms is incremental. The modifications are italicized at every step.

7.3.1 Algorithm 1

This algorithm is a modified version of the algorithm in [106]. The main idea of the algorithm is that BRM cells are returned from the branch point when FRM cells are received, and the BRM cells contain the minimum of the values indicated by the BRM cells received from the branches after the last BRM cell was sent. FRM cells are duplicated and multicast to all branches at the branch point.

A register, MER, and two flags, MCI and MNI, are maintained for each multipoint VC. MER stores the minimum of the explicit rate (ER) values, MCI is the logical OR of the congestion indication (CI) values, and MNI is the logical OR of the no increase (NI) values indicated in the BRM cells which were received after the last BRM cell was sent. MER is initialized to the peak cell rate, while CI and NI are initialized to zero. Three temporary variables: MXER, MXCI, and MXNI are also used when an FRM cell is received (their values do not persist across invocations of the algorithm). They store the ER, CI and NI from the FRM cell. The algorithm operates as follows.

Upon the receipt of an FRM cell:

1. Multicast FRM cell to all participating branches
2. Let MXER = ER from FRM cell, MXCI = CI from FRM cell, MXNI = NI from FRM cell
3. Return a BRM with ER = MER, CI = MCI, NI = MNI to the source
4. Let MER = MXER, MCI = MXCI, MNI = MXNI

Upon the receipt of a BRM cell:

1. Let MER = min (MER, ER from BRM cell), MCI = MCI OR CI from BRM cell, MNI = MNI OR NI from BRM cell
2. Discard the BRM cell

When a BRM is to be scheduled:

Let ER = min (ER, ER calculated by rate allocation algorithm for all branches)

7.3.2 Algorithm 2

This algorithm is a modified version of the second algorithm in [103]. The only change from Algorithm 1 (as described above) is ensuring that at least one BRM cell

has been received from a branch before turning around an FRM. For this purpose, a boolean flag, AtLeastOneBRM (initially zero), is set to true when a BRM cell is received from a branch, and reset when a BRM is sent by the branch point (see the italicized statements). As before, MER, MCI, MNI, and here, AtLeastOneBRM, are stored for each multipoint VC, and MXER, MXCI, MXNI are temporary variables.

Upon the receipt of an FRM cell:

1. Multicast FRM cell to all participating branches

2. IF AtLeastOneBRM THEN

A. Let MXER = ER from FRM cell, MXCI = CI from FRM cell, MXNI = NI from FRM cell

B. Return a BRM with ER = MER, CI = MCI, NI = MNI to the source

C. Let MER = MXER, MCI = MXCI, MNI = MXNI

D. Let AtLeastOneBRM = 0

Upon the receipt of a BRM cell:

1. Let AtLeastOneBRM = 1

2. Let MER = min (MER, ER from BRM cell), MCI = MCI OR CI from BRM cell, MNI = MNI OR NI from BRM cell

3. Discard the BRM cell

When a BRM is to be scheduled:

Let ER = min (ER, ER calculated by rate allocation algorithm for all branches)

7.3.3 Algorithm 3

The main idea of this algorithm is that the branch point does not turn around the FRMs, but the BRM that is received from a branch immediately after an FRM

has been received by the branch point is passed back to the source, carrying the minimum values. A boolean flag, AtLeastOneFRM, indicates that an FRM cell has been received by the branch point after the last BRM cell was passed to the source. Again, MER, MCI, MNI, and AtLeastOneFRM are stored per multipoint VC. This is a modified version of the third algorithm in [103].

Upon the receipt of an FRM cell:

1. Multicast FRM cell to all participating branches

2. Let AtLeastOneFRM = 1

Upon the receipt of a BRM cell:

1. Let MER = min (MER, ER from BRM cell), MCI = MCI OR CI from BRM cell, MNI = MNI OR NI from BRM cell

2. IF AtLeastOneFRM THEN

A. Pass the BRM with ER = MER, CI = MCI, NI = MNI to the source

B. Let MER = PCR, MCI = 0, MNI = 0

C. Let AtLeastOneFRM = 0

ELSE Discard the BRM cell

When a BRM is to be scheduled:

Let ER = min (ER, ER calculated by rate allocation algorithm for all branches)

7.3.4 Algorithm 4

A variation of this algorithm was presented in [103] as algorithm 4, and another variation using sequence numbers in RM cells was proposed in [19]. The main idea is that a BRM is passed to the source only when BRM cells have been received from all branches. To count the number of branches from which BRM cells were received at

the branch point (after the last BRM cell was passed by the branch point), a counter, NumberOfBRMsReceived, is incremented the first time a BRM cell is received from each branch (NumberOfBRMsReceived is initialized to zero). As before, the MER, MCI, MNI, and NumberOfBRMsReceived registers are maintained per multipoint VC. The value of the NumberOfBRMsReceived counter is compared to the value of another counter, NumberOfBranches, every time a BRM cell is received by the branch point. If the value of NumberOfBRMsReceived is equal to NumberOfBranches, the BRM cell is passed back to the source, carrying the values of the MER, MCI and MNI registers. (NumberOfBranches stores the number of branches of the point-to-multipoint VC at this branch point. It is also stored for each VC, and initialized during connection setup. In addition, if leaf initiated join is allowed (as in UNI 4.0), NumberOfBranches must be updated every time a new branch is added to a branch point.)

A flag, BRMReceived, is needed for each branch to indicate whether a BRM cell has been received from this particular branch, after the last BRM cell was passed. The flag is stored for each output port and not for each VC, as it is needed for each branch. Note that a mechanism must be implemented to ensure that BRM cell flow is not stopped in the case of non-responsive branches. Timeouts or RM cell counters can be used for that purpose. This will be the subject of a future study.

Upon the receipt of an FRM cell:

Multicast FRM cell to all participating branches

Upon the receipt of a BRM cell from branch i:

1. *IF NOT* $BRMReceived_i$ *THEN*

   A. *Let* $BRMReceived_i = 1$

B. Let NumberOfBRMsReceived = NumberOfBRMsReceived + 1

2. Let MER = min (MER, ER from BRM cell), MCI = MCI OR CI from BRM cell, MNI = MNI OR NI from BRM cell

3. IF NumberOfBRMsReceived is equal to NumberOfBranches THEN

A. Pass the BRM with ER = MER, CI = MCI, NI = MNI to the source

B. Let MER = PCR, MCI = 0, MNI = 0

C. Let NumberOfBRMsReceived = 0

D. Let BRMReceived = 0 FOR all branches

ELSE Discard the BRM cell

When a BRM is to be scheduled:

Let ER = min (ER, ER calculated by rate allocation algorithm for all branches)

7.4 New Algorithms

The main problem with algorithm 4 described in the previous section is its slow transient response. Even when excessive overload is detected, the algorithm has to wait for feedback from (possibly distant) leaves before indicating the overload information to the source. By that time, the source may have transmitted a large number of cells (which would be dropped due to buffer overflows), resulting in performance degradation. This situation is especially problematic when the source has been idle for some time, and then suddenly sends a burst, so there are no RM cells initially in the network.

The main idea behind the algorithms presented next is that the slow transient response problem should be avoided when a *severe overload* situation has been detected. In this case, there is no need to wait for feedback from all the branches,

and the overload should be immediately indicated to the source. In cases of underload indication from a branch, it is better to wait for feedback from all branches, as other branches may be overloaded. This is somewhat similar to the idea behind the backward explicit congestion notification (BECN) cells sent by the switches.

Overload is detected when the feedback to be indicated is *much less* than the last feedback returned by the branch point. The "much less" condition is tested using a multiplicative factor, Threshold. The Threshold value can range from zero to one. A Threshold value of one means that overload is detected when the feedback to be given is less than the current ER (even when it is 99% of the last ER value given); a Threshold value of zero means that overload is not detected except when the new rate to be indicated is zero, which, in effect, disables the fast overload indication feature.

An alternative method would be to compare the feedback to be indicated to the *current cell rate (CCR)* or ACR of the VC. Although this may be better because it accounts for upstream bottlenecks, and prevents the transmission of unnecessary BRM cells in such cases, the CCR information may be stale due to the delay from the source to the branch point (it may also be much larger when the source becomes idle or becomes a low rate source after the last FRM was sent), and a large number of BRMs may be sent in such cases. The last feedback indicated by the branch point is a more current value. The minimum of the CCR and last feedback given can be used in the comparison, but this involves some additional complexity, and may slow down the overload response when the CCR happens to have been small, but is currently large.

Note that when a BRM cell is returned due to overload detection before feedback has been received from all branches, the counters and the register values are not reset.

7.4.1 Fast Overload Indication (Algorithm 5)

In this algorithm, the LastER register maintains the last explicit rate value returned by the branch point (LastER is initialized to the initial cell rate (ICR) of the connection). LastER is stored per multipoint VC and compared to the value of MER in step 5 below.

Two temporary variables: SendBRM and Reset are used. SendBRM is set only if a BRM cell is to be passed to the source by the branch point. Reset is false only if a BRM cell is being used to indicate overload conditions, and hence the register values should not be reset. FRMminusBRM is only used for accounting purposes, and will not exist in a real implementation.

Upon the receipt of an FRM cell:

1. Multicast FRM cell to all participating branches

(* 2. Let FRMminusBRM = FRMminusBRM + 1 *)

Upon the receipt of a BRM cell from branch i:

1. Let SendBRM = 0

2. Let Reset = 1

3. IF NOT BRMReceived$_i$ THEN

   A. Let BRMReceived$_i$ = 1

   B. Let NumberOfBRMsReceived = NumberOfBRMsReceived + 1

4. Let MER = min (MER, ER from BRM cell), MCI = MCI OR CI from BRM cell, MNI = MNI OR NI from BRM cell

5. IF MER < (Threshold × LastER) THEN (overload is detected *)*

   A. IF NumberOfBRMsReceived < NumberOfBranches THEN

      1. Let Reset = 0

B. Let SendBRM = 1

ELSE IF NumberOfBRMsReceived is equal to NumberOfBranches THEN

A. Let SendBRM = 1

6. IF SendBRM THEN

A. Pass the BRM with ER = MER, CI = MCI, NI = MNI to the source

B. IF Reset THEN

1. Let MER = PCR, MCI = 0, MNI = 0

2. Let NumberOfBRMsReceived = 0

3. Let BRMReceived = 0 FOR all branches

(* C. Let FRMminusBRM = FRMminusBRM − 1 *)

ELSE Discard the BRM cell

When a BRM is to be scheduled:

1. Let ER = min (ER, ER calculated by rate allocation algorithm for all branches)

2. Let LastER = ER

7.4.2 RM Ratio Control (Algorithm 6)

The previous algorithm may increase the BRM cell overhead, as the ratio of source-generated FRM cells to BRM cells received by the source can exceed one. To avoid this problem, we introduce the register *SkipIncrease* which is maintained for each multipoint VC (and initialized to zero). SkipIncrease is used to control the RM cell ratio. SkipIncrease is incremented whenever a BRM cell is sent before feedback from all the branches has been received. When feedback from all leaves indicates underload, and the value of the SkipIncrease register is greater than zero, this particular feedback can be *ignored* and SkipIncrease decremented. Note that the value of the SkipIncrease

counter will not increase to large values, as the rate allocation algorithm (such as ERICA) arrives at the optimal allocation within a few iterations, and the explicit rates computed *cannot continue decreasing indefinitely.* Our analysis and simulations have shown that the counter never exceeds small values and quickly stabilizes at zero. A maximum value can also be enforced by the algorithm.

Upon the receipt of an FRM cell:

1. Multicast FRM cell to all participating branches

(* 2. Let FRMminusBRM = FRMminusBRM + 1 *)

Upon the receipt of a BRM cell from branch i:

1. Let SendBRM = 0

2. Let Reset = 1

3. IF NOT BRMReceived_i THEN

   A. Let $\text{BRMReceived}_i = 1$

   B. Let NumberOfBRMsReceived = NumberOfBRMsReceived + 1

4. Let MER = min (MER, ER from BRM cell), MCI = MCI OR CI from BRM cell, MNI = MNI OR NI from BRM cell

5. IF MER $\geq$ LastER AND SkipIncrease $>$ 0 AND NumberOfBRMsReceived is equal to NumberOfBranches THEN

   A. Let SkipIncrease = SkipIncrease − 1

   B. Let NumberOfBRMsReceived = 0

   C. Let BRMReceived = 0 FOR all branches

ELSE IF MER $<$ (Threshold $\times$ LastER) THEN

   A. IF NumberOfBRMsReceived $<$ NumberOfBranches THEN

      1. Let SkipIncrease = SkipIncrease + 1

2. Let Reset = 0

B. Let SendBRM = 1

ELSE IF NumberOfBRMsReceived is equal to NumberOfBranches THEN

A. Let SendBRM = 1

6. IF SendBRM THEN

A. Pass the BRM with ER = MER, CI = MCI, NI = MNI to the source

B. IF Reset THEN

1. Let MER = PCR, MCI = 0, MNI = 0

2. Let NumberOfBRMsReceived = 0

3. Let BRMReceived = 0 FOR all branches

(* C. Let FRMminusBRM = FRMminusBRM − 1 *)

ELSE Discard the BRM cell

When a BRM is to be scheduled:

1. Let ER = min (ER, ER calculated by rate allocation algorithm for all branches)
2. Let LastER = ER

7.4.3 Immediate Rate Computation (Algorithm 7)

The last two algorithms can offer rapid congestion relief when an overload is detected in a branch of the multicast tree. They do not, however, account for the potential overload situation at the branch point itself: if the branch point is a switch (queuing point), the ERICA algorithm is only performed when the BRM cell is about to be scheduled on the link. In cases when the branch point is itself a switch and queuing point, the immediate rate calculation option invokes ERICA whenever a BRM is received, and not just when a BRM is being sent. Hence overload at the

branch point can be detected and indicated according to the fast overload indication option as previously described. Doing this, however, may involve some additional complexity.

The algorithm presented next is the same as *Algorithm 6* in the previous subsection, except for the addition of the ERICA invocation (italicized below).

Upon the receipt of an FRM cell:

1. Multicast FRM cell to all participating branches

(* 2. Let FRMminusBRM = FRMminusBRM + 1 *)

Upon the receipt of a BRM cell from branch i:

1. Let SendBRM = 0

2. Let Reset = 1

3. IF NOT BRMReceived_i THEN

   A. Let $\text{BRMReceived}_i = 1$

   B. Let NumberOfBRMsReceived = NumberOfBRMsReceived + 1

4. Let MER = min (MER, ER from BRM cell), MCI = MCI OR CI from BRM cell, MNI = MNI OR NI from BRM cell

5. *Let MER = min (MER, minimum ER calculated by rate allocation algorithm for all branches)*

6. IF MER $\geq$ LastER AND SkipIncrease $>$ 0 AND NumberOfBRMsReceived is equal to NumberOfBranches THEN

   A. Let SkipIncrease = SkipIncrease $-$ 1

   B. Let NumberOfBRMsReceived = 0

   C. Let BRMReceived = 0 FOR all branches

ELSE IF MER $<$ (Threshold $\times$ LastER) THEN

A. IF NumberOfBRMsReceived < NumberOfBranches THEN

1. Let SkipIncrease = SkipIncrease + 1

2. Let Reset = 0

B. Let SendBRM = 1

ELSE IF NumberOfBRMsReceived is equal to NumberOfBranches THEN

A. Let SendBRM = 1

7. IF SendBRM THEN

A. Pass the BRM with ER = MER, CI = MCI, NI = MNI to the source

B. IF Reset THEN

1. Let MER = PCR, MCI = 0, MNI = 0

2. Let NumberOfBRMsReceived = 0

3. Let BRMReceived = 0 FOR all branches

(* C. Let FRMminusBRM = FRMminusBRM − 1 *)

ELSE Discard the BRM cell

When a BRM is to be scheduled:

1. Let ER = min (ER, ER calculated by rate allocation algorithm for all branches)

2. Let LastER = ER

7.5 Algorithm Summary

Table 7.1 summarizes the description of the seven algorithms given in the previous two sections, according to the design issues discussed in section 7.2. As seen in the table, there are a number of design and implementation options. Each of these options affects the performance of the algorithm. The remainder of the chapter is devoted to the performance analysis and performance comparison of the algorithms.

Algorithm	1	2	3	4	5	6	7
FRM turn-around	Branch point	Branch point	Destination	Destination	Destination	Destination	Destination
Wait for all BRMs	No	No	No	Yes	Yes for underload	Yes for underload	Yes for underload
Condition to trigger a BRM from branch point	FRM received	FRM received and a new BRM	BRM received and a new FRM	a new FRM and all BRMs	a new FRM and either all BRMs or overload	a new FRM and either all BRMs or overload, provided no extra	a new FRM and either all BRMs or overload (including current), provided no extra
RM ratio control	through FRM	through FRM	through BRM	through BRM	through BRM in some cases	explicit	explicit
Handling non-responsive branches	unnecessary	unnecessary	unnecessary	necessary	necessary for increase	necessary for increase	necessary for increase
Interacts with switch	No	No	No	No	No	No	Yes
Per-branch accounting	None	None	None	1 bit per branch	1 bit per branch	1 bit per branch	1 bit per branch

Table 7.1: Summary of consolidation algorithm features

7.6 Performance Analysis

This section provides a performance comparison among all the presented consolidation algorithms, in a variety of configurations with bursty and non-bursty traffic, with and without variable bit rate (VBR) background, and with various link lengths, bottleneck locations, and number of leaves. A number of other configurations was also tested (see [32, 27] for some of the configurations), but only a sample of the results is shown here. In particular, configurations with a large number of leaves at varying distances in the multicast tree were simulated, and the results were consistent with those presented here.

7.6.1 Parameter Settings

Throughout our experiments, the following parameter values are used:

[1] All links have a bandwidth of 155.52 Mbps (149.76 Mbps when SONET overhead is accounted for).

[2] All point-to-multipoint traffic flows from the root to the leaves of the tree. No traffic flows from the leaves to the root, except for RM cells. The same applies for point-to-point connections.

[3] All sources are deterministic, i.e., their start/stop times and their transmission rates are known. The bursty traffic sources send data in bursts.VBR sources are on/off sources, where the on and off times are 20 ms.

[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. Initial cell rate (ICR) is also set to a high value (almost peak cell rate). These factors are set to such high values to simulate a *worst case* load situation.

[**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 source end system rule six.

[**6**] All other ABR parameters are set to their default values as specified in [43].

[**7**] The switch target utilization parameter was set at 90%. The switch measurement interval was set to the minimum of the time to receive 100 cells and 1 ms.

[**8**] The Threshold parameter used in Algorithms 5 to 7 was set to 0.95. This large value was used to illustrate the effect of the fast overload indication, RM ratio control, and immediate rate computation features of the algorithms. A lower value is recommended to be used in practice.

7.6.2 Simulation Results

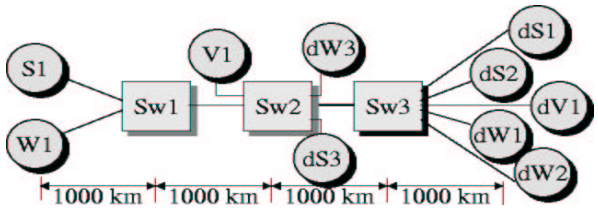


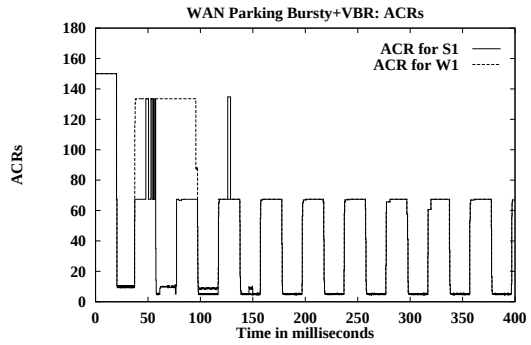
Figure 7.6: WAN parking lot configuration with bursty, persistent and VBR connections

This section discusses the performance of the seven consolidation algorithms by comparing them in a set of configurations. Two graphs are plotted for each configuration: the allowed cell rate for the all the ABR sources, and the queue lengths for *overloaded* switches.

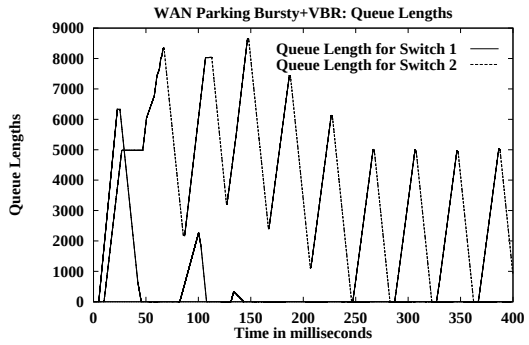
Figures 7.7 through 7.13 illustrate the performance of the seven different algorithms in a situation where there is both variable capacity and variable demand. These situations offer the toughest challenge for rate allocation algorithms [27, 31]. The configuration simulated is shown in figure 7.6 (all links are 1000 km). The source indicated by W is a bursty source, S is a persistent (infinite) source, while V is a VBR source. The VBR connection (V1 to dV1) is point-to-point, while the 2 ABR connections (bursty W1 sending to dW1, dW2, dW3; persistent S1 sending to dS1, dS2, dS3) are point-to-multipoint connections. The configuration is called a "parking lot" configuration because it resembles the situation arising when multiple streams of cars are merging to travel towards the exit.

The ACR graphs of the ABR sources for algorithms 1, 2, and 3 indicate fluctuations and inaccurate (around 140 Mbps) feedback given in the initial 150 ms. This results in large queues (>5000 cells with every VBR burst) at Switch 2 on the port going to Switch 3 (Switch 3 has negligible queues and is therefore not shown here). The high initial cell rate (ICR) and rate increase factor (RIF) [43] values are the reason for the unusually large initial ABR queues seen for all algorithms.

Algorithm 4 gives more accurate feedback, but the first correct feedback is given after around 50 ms, which results in initially large queues (as ICR is large). Algorithms 5 and 6 produce identical results to algorithm 4, as the bottleneck link is attached to the branch point. Algorithm 7, on the other hand, exhibits a very fast transient response, and gives relatively accurate feedback to both sources. The initial queues caused by high ICR, as well as the queues with every VBR burst, are much smaller. Hence, it offers the best performance as it combines the benefits of algorithm 4 with a fast transient response.

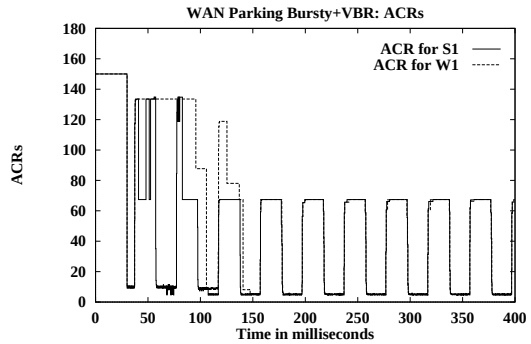


(a) Allowed Cell Rate in Mbps

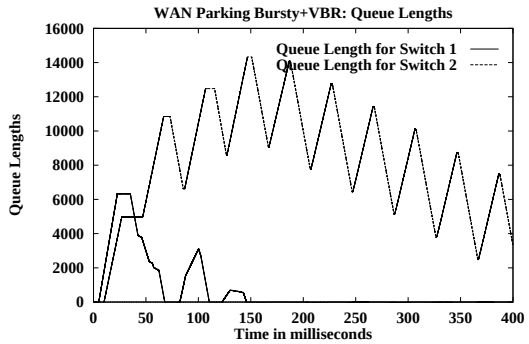


(b) ABR Queue Lengths in Cells

Figure 7.7: Simulation results for WAN parking lot configuration with bursty, persistent and VBR connections [Algorithm 1]

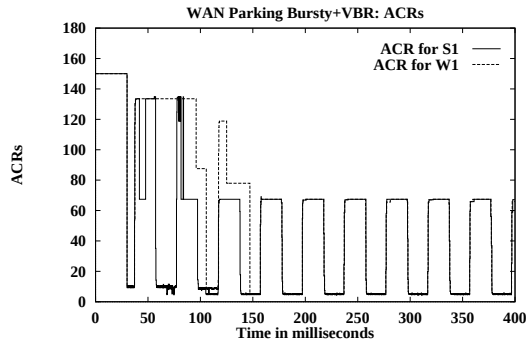


(a) Allowed Cell Rate in Mbps

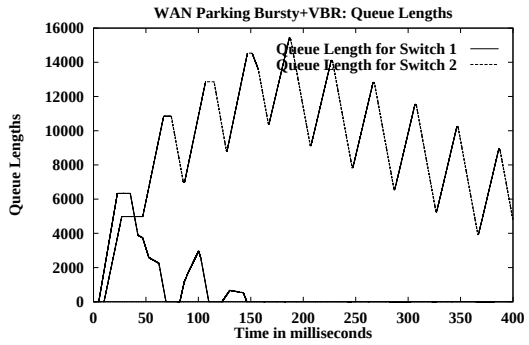


(b) ABR Queue Lengths in Cells

Figure 7.8: Simulation results for WAN parking lot configuration with bursty, persistent and VBR connections [Algorithm 2]

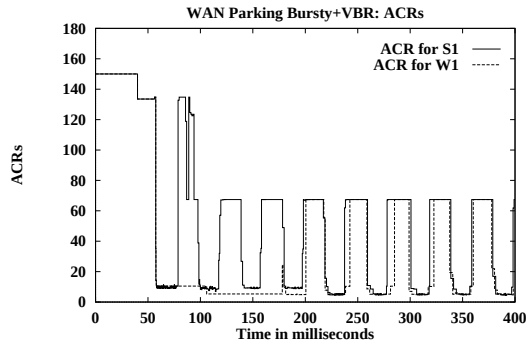


(a) Allowed Cell Rate in Mbps

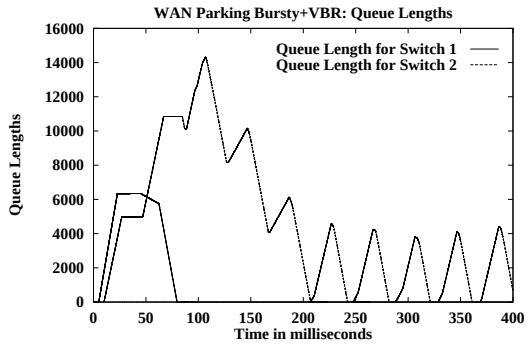


(b) ABR Queue Lengths in Cells

Figure 7.9: Simulation results for WAN parking lot configuration with bursty, persistent and VBR connections [Algorithm 3]

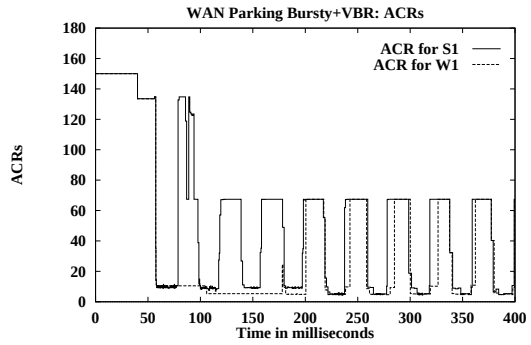


(a) Allowed Cell Rate in Mbps

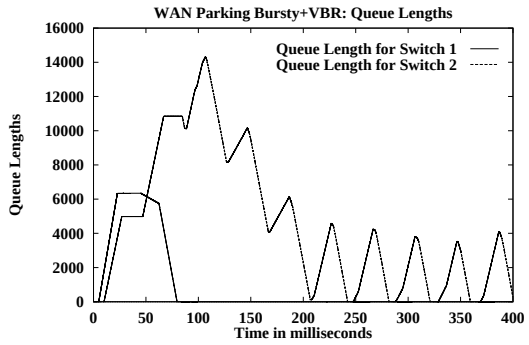


(b) ABR Queue Lengths in Cells

Figure 7.10: Simulation results for WAN parking lot configuration with bursty, persistent and VBR connections [Algorithm 4]

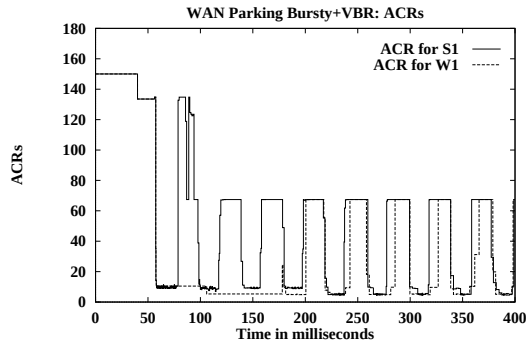


(a) Allowed Cell Rate in Mbps

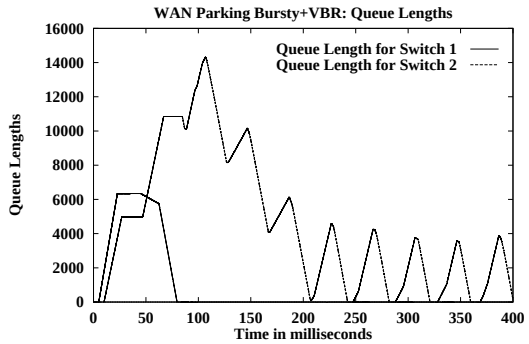


(b) ABR Queue Lengths in Cells

Figure 7.11: Simulation results for WAN parking lot configuration with bursty, persistent and VBR connections [Algorithm 5]

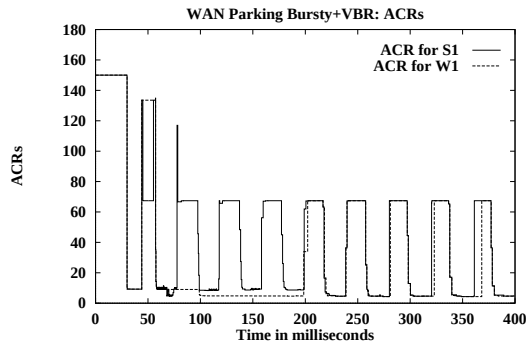


(a) Allowed Cell Rate in Mbps

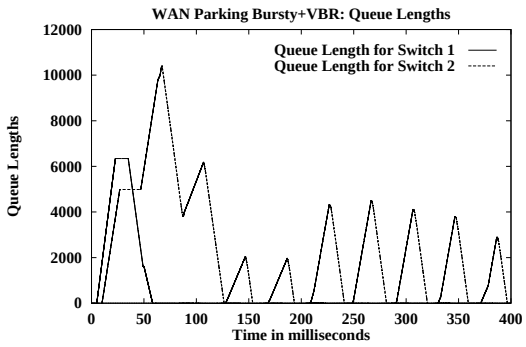


(b) ABR Queue Lengths in Cells

Figure 7.12: Simulation results for WAN parking lot configuration with bursty, persistent and VBR connections [Algorithm 6]



(a) Allowed Cell Rate in Mbps



(b) ABR Queue Lengths in Cells

Figure 7.13: Simulation results for WAN parking lot configuration with bursty, persistent and VBR connections [Algorithm 7]

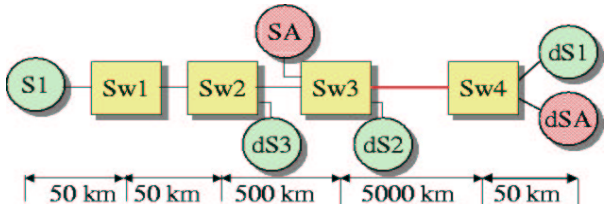
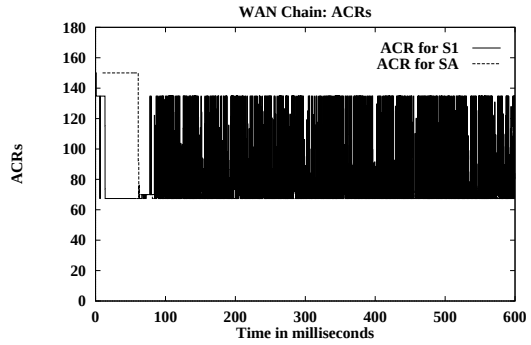


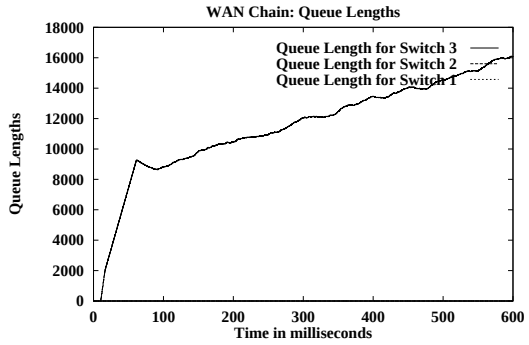
Figure 7.14: The chain configuration illustrates consolidation noise problems

The chain configuration, illustrated in figure 7.14 consists of a point-to-multipoint connection (S1 to dS1, dS2 and dS3) where one of the links on the route to the farthest leaf is the bottleneck link (shared by the point-to-point connection SA to dSA). Also the link lengths increase by an order of magnitude in each of the last two hops (all links from the end systems to the switches are 50 km).

As seen in figures 7.15 through 7.21, this configuration is an ideal configuration for illustrating the consolidation noise problem. The problem is severe for algorithms

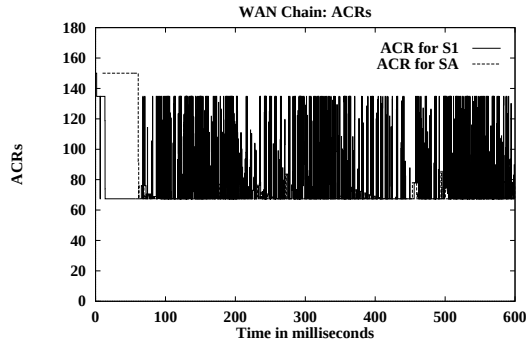


(a) Allowed Cell Rate in Mbps

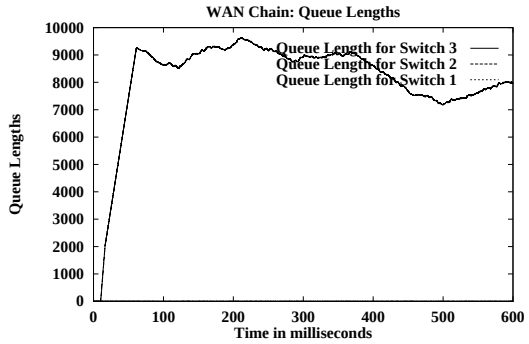


(b) ABR Queue Lengths in Cells

Figure 7.15: Simulation results for a chain configuration [Algorithm 1]

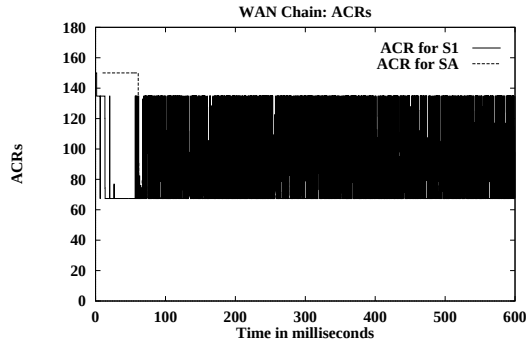


(a) Allowed Cell Rate in Mbps

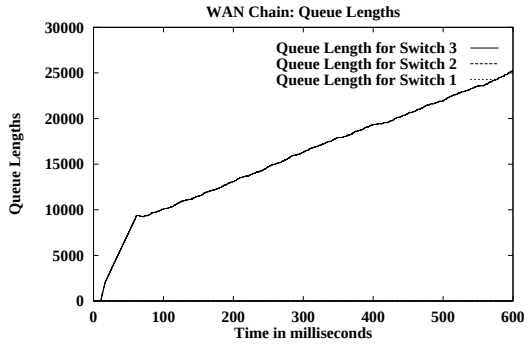


(b) ABR Queue Lengths in Cells

Figure 7.16: Simulation results for a chain configuration [Algorithm 2]

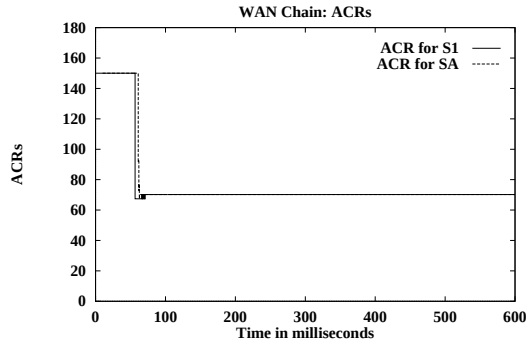


(a) Allowed Cell Rate in Mbps

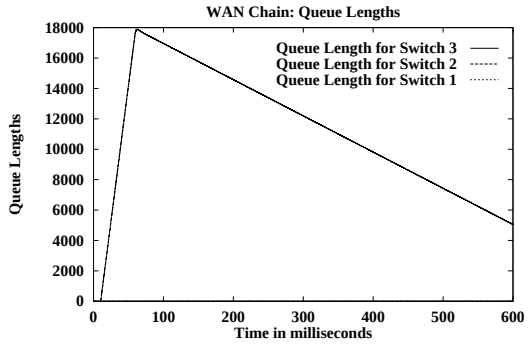


(b) ABR Queue Lengths in Cells

Figure 7.17: Simulation results for a chain configuration [Algorithm 3]

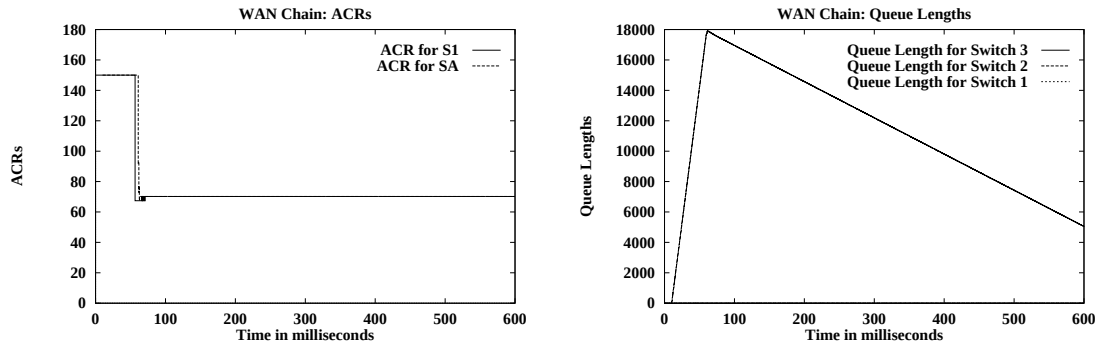


(a) Allowed Cell Rate in Mbps



(b) ABR Queue Lengths in Cells

Figure 7.18: Simulation results for a chain configuration [Algorithm 4]

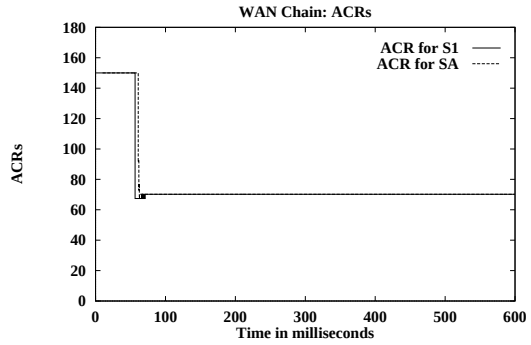


(a) Allowed Cell Rate in Mbps

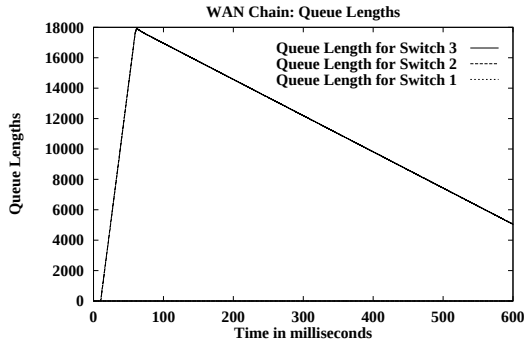
(b) ABR Queue Lengths in Cells

Figure 7.19: Simulation results for a chain configuration [Algorithm 5]

1, 2 and 3 (see figures 7.15 through 7.17), and results in rate oscillations, instability, unbounded queues, and unfairness against source SA. Switch 3 is the bottleneck in this configuration, as the link connecting Switch 3 to Switch 4 is the bottleneck link. The queues at Switch 1 and Switch 2 are negligible and hence are not visible on the graph (they are close to zero). The rate of SA remains at half of the bandwidth, while the rate of S1 continues to oscillate around a mean of about 103 Mbps. Although using a scheme such as ERICA+ leads to stability and bounded queues in this case, the persistent rate oscillations result in unacceptable performance and unfairness (*the problem can be mitigated by using small RIF values, but this slows down rate increases*). Algorithms 4, 5 and 6 (figures 7.18 through 7.20) avoid the noise completely, but suffer from a slow transient response. The rate of the source S1 only drops after around 60 ms, and by that time, large queues have built up at the switches. Algorithm 7 yields optimal performance in this case, as the rate of the source S1 immediately drops to its optimal value, as soon as the overload is detected.

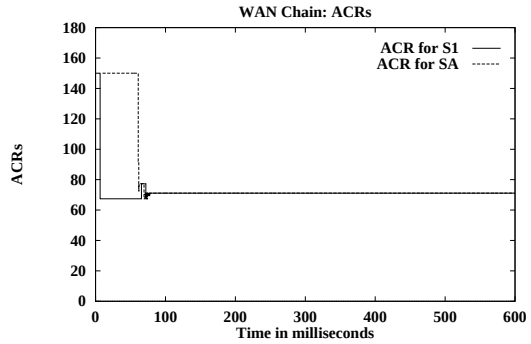


(a) Allowed Cell Rate in Mbps

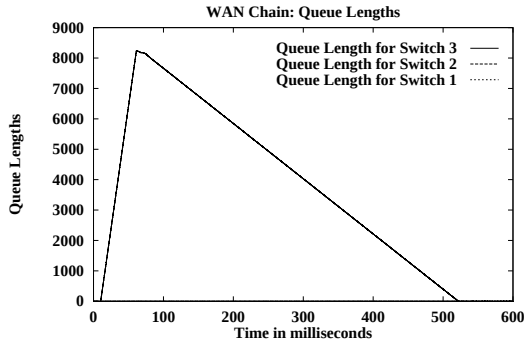


(b) ABR Queue Lengths in Cells

Figure 7.20: Simulation results for a chain configuration [Algorithm 6]



(a) Allowed Cell Rate in Mbps



(b) ABR Queue Lengths in Cells

Figure 7.21: Simulation results for a chain configuration [Algorithm 7]

Observe that algorithms 5 and 6 also yield near optimal performance (like algorithm 7) if the destination dS3 was further than dS1, as in the configuration in figure 7.22 (all links from the end systems to the switches are 50 km except for dS3 which is 8000 km away from switch 2). This result is illustrated in figures 7.23 through 7.26 which compare the results for algorithms 4, 5, 6 and 7. Here, the chain configuration is modified such that the bottleneck link is closer to the branch point at Switch2 than another leaf, namely dS3.

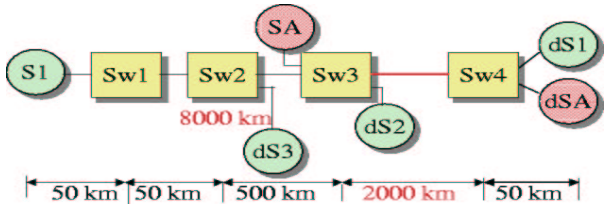
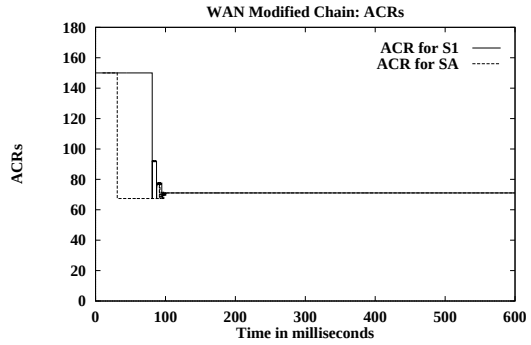
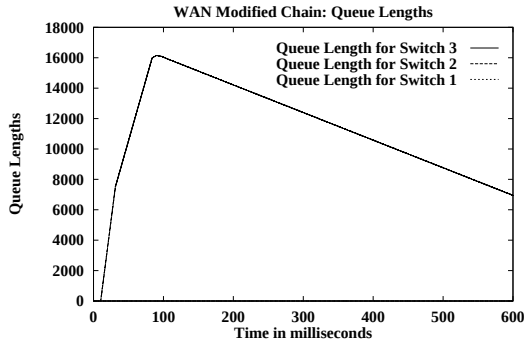


Figure 7.22: The modified chain configuration

In this case, as seen in figure 7.23, algorithm 4 wastes a long time waiting for feedback from dS3, while it has already received the bottleneck feedback from Switch 3. Algorithms 5, 6, and 7 send the feedback as soon as the overload situation is indicated by the BRM cell coming from Switch 3, and do not needlessly wait for the BRM from dS3. Hence, the 3 new algorithms perform near optimally as the rate of the source S1 goes to the optimal value after only around 20 ms for algorithms 5 and 6, and less than 10 ms for algorithm 7. The maximum queue length is also much smaller than for algorithm 4 ($>$ 16000 cells): for algorithms 5 and 6, it is around 7000 cells, and for algorithm 7, it is less than 3500 cells.

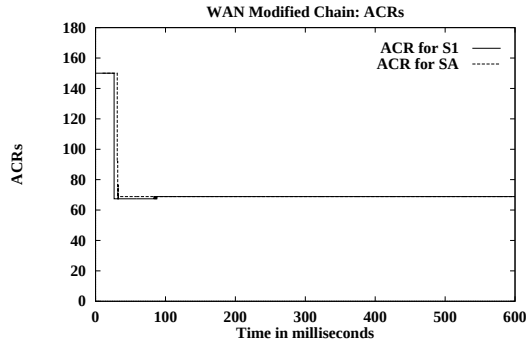


(a) Allowed Cell Rate in Mbps

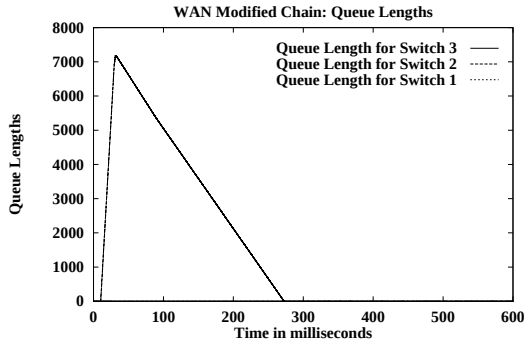


(b) ABR Queue Lengths in Cells

Figure 7.23: Simulation results for a chain modified configuration [Algorithm 4]

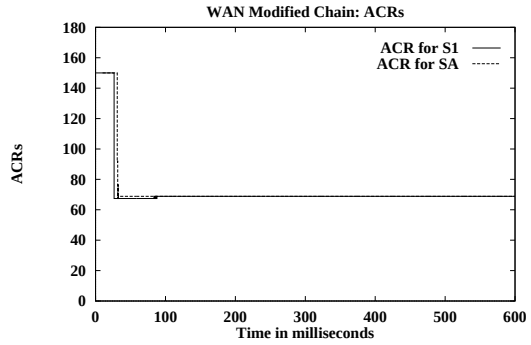


(a) Allowed Cell Rate in Mbps

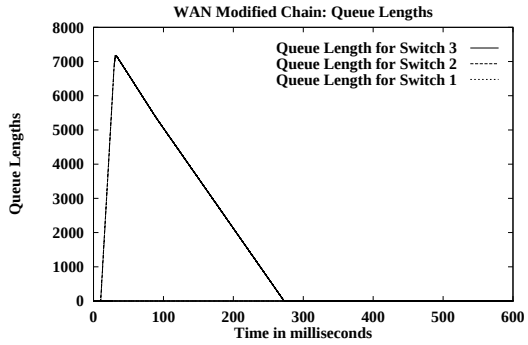


(b) ABR Queue Lengths in Cells

Figure 7.24: Simulation results for a chain modified configuration [Algorithm 5]

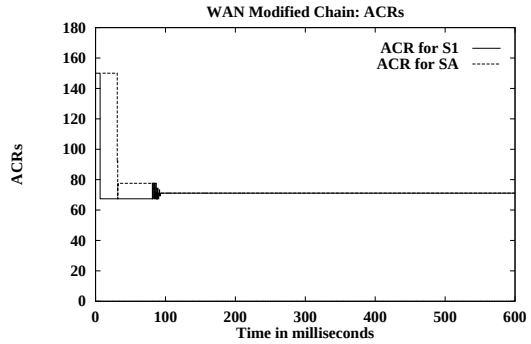


(a) Allowed Cell Rate in Mbps

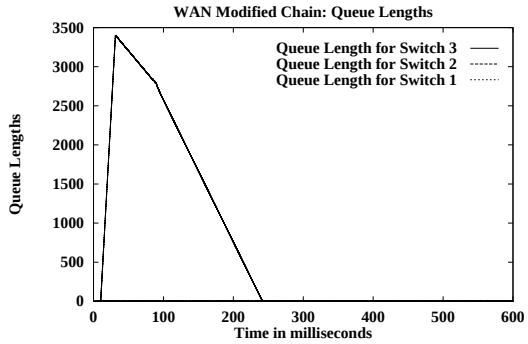


(b) ABR Queue Lengths in Cells

Figure 7.25: Simulation results for a modified chain configuration [Algorithm 6]



(a) Allowed Cell Rate in Mbps



(b) ABR Queue Lengths in Cells

Figure 7.26: Simulation results for a modified chain configuration [Algorithm 7]

We have observed a similar, but more pronounced, behavior when we simulated configurations with a larger number of leaves at varying distances and at varying levels of the multicast tree. The situation was much worse in those cases with algorithms 1, 2, and 3, which had much more severe noise problems. Algorithm 4 had an extremely slow transient response, while algorithms 5, 6, and especially 7 quickly reached the optimal values, and the queues at the switches were small.

7.7 Comparison of the Algorithms

This section summarizes the conclusions from the performance comparison of the algorithms. All the algorithms preserve the fairness and efficiency of the point-to-point congestion avoidance algorithm employed. We compare the space and time complexity, transient response, consolidation noise, algorithm overhead and scalability, and discuss the interoperability of various algorithms.

7.7.1 Implementation Complexity

In algorithms 1 and 2, the main source of complexity is that the branch point has to turn around the RM cells. This is similar to the Virtual Source/Virtual Destination (VS/VD) concept. Most studies argue that turning around RM cells has a high implementation complexity.

Algorithm 3 is definitely the simplest algorithm to implement because it does not turn around RM cells and keeps minimal per-VC accounting information. Algorithm 4 is more complex as it has to maintain the number of branches and the number of branches from which BRMs have been received, and compare those numbers. In addition, it has to maintain a bit for each output port to denote whether a BRM cell

has been received from this branch, and some responsiveness detection algorithm-related values.

Algorithms 5 and 6 are slightly more complex as they may also store the last ER sent by the branch point. Alternatively, they can use the CCR of the source, which is already stored and used by most congestion avoidance algorithms (it is used in the ERICA algorithm which we have employed in this study). Hence, the additional complexity over algorithm 4 mainly stems from the comparison of the MER value to the last ER sent or the CCR value, and maintaining the SkipIncrease counter. This additional comparison and integer register do not incur much overhead.

Algorithm 7 is more complex than algorithms 5 and 6, as it invokes the ERICA algorithm for all the branches whenever a BRM cell is received by the branch point, and not only when a BRM cell is to be sent.

7.7.2 Transient Response

Algorithm 1 exhibits a very fast transient response. Algorithms 2 and 3 also have a reasonable transient response, as, even if there are no RM cells in the network, the feedback is quickly returned on the first BRM arrival.

Algorithm 4 has a slow transient response, as it waits for feedback from all the leaves before sending BRMs. This is especially severe in cases when there are few or no RM cells already in the network, such as during startup periods and for bursty sources. Therefore, feedback can be delayed up to a function of the longest round trip times. Algorithms 5, 6 and 7 tackle this problem for overload situations. The transient response of the schemes is very fast when an overload is detected downstream (for algorithms 5 and 6), or at this branch and downstream (for algorithm 7). In such

cases, the transient response of the scheme is reasonably fast, and potential cell loss and retransmissions are alleviated.

7.7.3 Consolidation Noise

Algorithms 1, 2, and 3 suffer from severe consolidation noise problems. In particular, algorithms 1 and 3 suffer from unacceptable consolidation noise in some cases, especially with large RIF values (recall figures 7.15 and 7.17). Algorithm 2 somewhat alleviates these problems, as BRMs are not sent if no feedback has been received from any of the downstream components. However, it still exhibits considerable noise.

Algorithms 4, 5, 6, and 7 eliminate this problem by waiting for feedback from all branches. Although algorithms 5, 6, and 7 do not wait for feedback from all leaves in cases of overload, this *does not* introduce noise, as the RM cells that are sent faster than the usual cells carry overload information, which would have been conveyed by the next minimum value anyway.

7.7.4 Scalability Issues

Algorithms should be scalable in the sense that their overhead and feedback delay should not grow with the increase in the number of branch points or levels of the multicast tree.

RM cell overhead

The number of FRM cells generated by the source and the number of BRM cells received by the source should be approximately the same. Algorithm 1 generates a BRM cell at the branch point for every FRM cell it receives, thereby guaranteeing that the BRM to FRM ratio remains one. Algorithms 2 and 3 maintain a BRM to

FRM ratio of less than or equal to 1 as follows. Algorithm 2 generates a BRM for an FRM only if a BRM has been received from a leaf since the last BRM was sent by the branch point. Algorithm 3 allows a BRM to pass to the source only if an FRM cell has been received by the branch point after the last BRM cell was forwarded. Therefore, both algorithms maintain a ratio that is less than or equal to one (actually, it is strictly less than one for algorithm 2, as the first FRM cell will never be turned).

Algorithm 4 also maintains a ratio of less than or equal to one, as one BRM cell is returned when BRM cells have been received from all branches. Algorithm 5 does not guarantee that the ratio remains at 1, as RM cells carrying overload indication are allowed to quickly return to the source. Algorithms 6 and 7 fix this problem by maintaining a counter that is incremented for every extra RM cell passed, and then decremented (and the BRM cell discarded) in cases of RM cells carrying underload information, if the counter exceeds zero. Hence, over the long run, the ratio is maintained at one. The counter cannot increase indefinitely, as the rates cannot decrease indefinitely, but a maximum value can be enforced. In all cases we have examined, the counter value was always small, because ERICA quickly converged.

Delay sensitivity to the maximum number of branch points on a path (levels of the tree)

Algorithm 1 waits for an FRM cell to arrive before it can send the feedback information it has consolidated from the BRM cells. This has to be done at every branch point, leading to a delay that may increase with the number of levels of the multicast tree. Algorithm 2 suffers from the same drawback, as the algorithm also sends a BRM cell at the branch point when an FRM cell is received.

Algorithm 3 is less sensitive to the number of levels of the multicast tree. The BRM cell is passed to the source only if an FRM cell has been received since the last BRM cell was sent by the branch point. However, it is passed without additional delay.

Algorithms 4, 5, 6, and 7 can be sensitive to the multicast tree levels as BRM cells from all branches are consolidated at every branch point. However, the delay (the time between the transmission of the FRM cell at the source and the receipt of the corresponding BRM cell by the source) is mainly dependent on the round trip times from the source to the leaves at that particular time. The round trip times to the leaves can vary with time, dependent on the queuing delay of the switches on the path of the multicast tree. More than one leaf can affect that delay as BRM cells arrive asynchronously at the branch points.

7.7.5 Interoperability Issues

The various consolidation algorithms should be able to interoperate with each other if no one algorithm is standardized. Although it seems that all the algorithms can interoperate smoothly with each other, the performance of a network with point-to-multipoint VCs that branch at several branch points with different algorithms needs further study. This will be one of the areas of our future research work.

7.8 Chapter Summary

Table 7.2 shows a summary of the results of the comparison between the consolidation algorithms. Note that the main drawback of each algorithm is indicated in bold face. In terms of complexity, algorithm 3 is clearly the simplest. Algorithms 1 and 2 turn around RMs, which is an expensive operation. Algorithm 4 introduces

additional complexity to algorithm 3, as it maintains per-branch variables and performs comparisons. Algorithm 5 introduces slightly more complexity to 4; algorithm 6 introduces simple additions to 5; and algorithm 7 introduces some more operations to 6, but most of the increments are of little complexity.

Algorithm	1	2	3	4	5	6	7
Complexity	**High**	**High**	Low	Medium	Medium	Medium	>Medium
Transient Response	Fast	Medium	Medium	**Slow**	Fast for overload	Fast for overload	Fastest for overload
Noise	Hgh	Medium	**High**	Low	Low	Low	Low
BRM:FRM at Root	1	<1	≤1	≤1	**may be>1**	$lim = 1$	$lim = 1$
Delay Sensitivity	Hgh	Hgh	Low	Medium	Medium	Medium	Medium

Table 7.2: Comparison of consolidation algorithm performance

The transient response of algorithm 1 is fast, but can be erroneous. Algorithms 2 and 3 offer medium response, while algorithm 4 is clearly slow. Algorithms 5, 6, and especially 7, have a fast response when overload is detected. Consolidation noise is a problem with algorithms 1, 2, and 3, especially 1 and 3. The other algorithms overcome this problem.

As for RM cell overhead, the ratio of BRM cells received by the source to FRM cells sent by the source is maintained at unity by algorithm 1. It is less than one for algorithm 2 (at least the first FRM is not returned), and is less than or equal to one for algorithms 3 and 4. Algorithm 5 introduces additional BRM cells in case of overload, while algorithms 6 and 7 ensure the ratio is one over the long run (*lim* in the table means the limit as time goes to infinity).

Finally, the sensitivity of algorithms 1 and 2 to the number of branch points and the levels of the multicast tree is high due to the additional delay waiting for an FRM cell at each branch point, and the additional BRM cells that are turned around in the network at each level. Algorithms 3 to 7 (especially algorithm 3) are somewhat less sensitive to this.

The comparison indicates that algorithms 1 and 2 suffer from complexity and noise problems. Algorithm 3 is good, except for the consolidation noise problem which leads to unacceptable performance in some cases as seen in figure 7.17. Algorithm 4 provides reasonable performance, but has a slow transient response, which is overcome by the algorithms we proposed (5, 6 and 7). Algorithm 4 and the new algorithms are slightly more complex than algorithm 3, but this can be well worth the performance benefits gained, especially with algorithm 7. Algorithm 7 avoids congestion, while eliminating the consolidation noise problem.

CHAPTER 8

FAIRNESS AND FLOW CONTROL FOR MULTIPOINT-TO-POINT CONNECTIONS

In multipoint-to-point connections, multiple sources can concurrently send data to the same destination. A crucial concern in this case is how to define fairness within a multicast group, and among multicast groups and unicast connections. The multipoint connection can have the same identifier on each link, so senders in a group might not be distinguishable. Many ABR rate allocation algorithms implicitly assume that there is only one sender in each connection, which does not hold for multipoint-to-point connections. It is impossible for the network to determine any source-specific characteristics since all sources in the multipoint connection may use the same connection identifiers. The challenge is to develop a rate allocation algorithm without per-source operations, as these are no longer equivalent to per-connection or per-flow operations.

We give four fairness definitions for multipoint connections based on connections, senders or flows, and we discuss the tradeoffs involved. We design and simulate an

O(1) fair rate allocation scheme for point-to-point and multipoint connections. Simulation results show that the algorithm performs well and exhibits desirable properties. We discuss the main modifications necessary for any ATM-ABR rate allocation scheme to accommodate multiple sources.

8.1 Introduction

In this chapter, we focus on the traffic management issues in the case of *multiple senders*. Specifically, we investigate fairness definition and ABR flow control for multipoint-to-point connections. As shown in figure 8.1, multipoint-to-point connections require feedback regulation and rate allocation at the merge points.

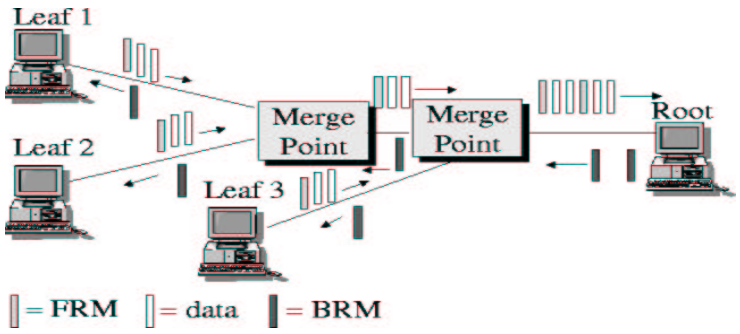


Figure 8.1: Multipoint-to-point ABR connections

Chapter 7 has shown that the source in point-to-multipoint ABR is usually controlled to the minimum rate supported by all the leaves of the multicast tree, as the leaves usually cannot tolerate losses [26, 96, 103, 106]. Therefore, the extension of fairness definitions to point-to-multipoint connections is straightforward. With multipoint-to-point and multipoint-to-multipoint connections, however, the implicit

assumption that each connection has only one source is no longer valid. The fairness definition to adopt relies on the application type and the pricing methods.

A source-based fair algorithm must give the same (or proportional) allocation to all sources bottlenecked on the same link. Source-based fairness in some switch implementations poses difficulties, since sources in the same VC cannot be distinguished (they have the same connection identifier). The challenges for rate allocation algorithms in this case include avoiding per-source accounting and avoiding estimating the number of active sources. This has to be done without adversely affecting the transient response or increasing the rate oscillations.

In this chapter, we define four methods for computing the fair allocations for multipoint-to-point virtual connections (VCs), and discuss the necessary modifications to switch schemes to give these allocations. The remainder of this chapter is organized as follows. We present our fairness definitions in section 8.2, and show their operation, merits and drawbacks with the aid of examples. We then discuss several design issues and examine how switch schemes need to be adapted to give fair allocations in section 8.3. In section 8.4, we develop rate allocation and merge point algorithms for multipoint connections, and examine their features. We analyze the performance of the algorithm in section 8.5, and conclude with a set of recommendations for rate allocation schemes to support multiple sources.

8.2 Fairness Extension for Multipoint-to-Point Connections

In this section, we define fairness for the multiple sender case, and show examples of the operation of our definitions. In addition, we discuss the merits and drawbacks of each fairness definition.

8.2.1 Fairness Definitions

Sources, VCs and flows. Before giving the fairness definitions, we need to distinguish among sources, VCs and flows. Figure 8.2 shows a configuration with 2 VCs. One of the VCs is a point-to-point VC, while the other one is a multipoint-to-point VC. The senders in the multipoint-to-point VC are indicated by dark-colored filled circles, while the sender in the point-to-point VC is denoted by the unfilled circle. At the second switch, traffic from 4 sources, but only 2 VCs, is being switched to the output port. Note, however, that the second switch can distinguish 3 input flows (the point-to-point sender and 2 flows of the multipoint-to-point connection coming on different input ports). The 2 sources whose traffic was merged at the first switch constitute a single flow at the second switch, since they cannot be distinguished downstream of their merge point. Two of the input flows that can be distinguished at the second switch belong to the same VC, while the third flow belongs to a different VC. The second switch merges the two flows of the same VC, and they become a single flow downstream of the second switch.

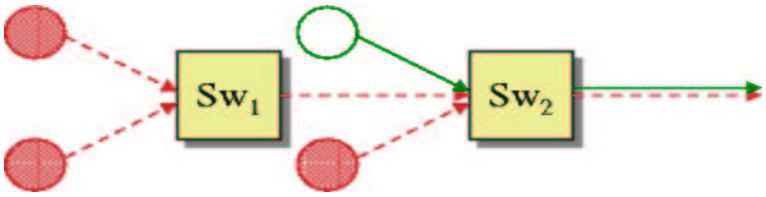


Figure 8.2: Source versus connection versus flow

Throughout the rest of the chapter, we use max-min fairness as the underlying fairness definition. However, the definitions we give apply for any underlying definition, e.g., general weighted fairness with minimum rate guarantees. We define a network *configuration* as a set of sources, destinations and switches, interconnected with links of given distances and bandwidths, and a set of virtual connections. We use the following notation:

n	denotes the number of sources in a given configuration
m	denotes the number of virtual connections in a given configuration
f	denotes the number of flows being switched to an output port of a switch
x_i	denotes the allocation given to the i^{th} source
y_j	denotes the allocation given to the j^{th} virtual connection
z_k	denotes the allocation given to the k^{th} flow
N	denotes the number of senders in a certain multipoint-to-point virtual connection
F	denotes the number of flows in a certain multipoint-to-point virtual connection at a certain output port

Source-based. Source-based fairness allocates bandwidth fairly among all sources, regardless of which VC each source belongs to. Each N-to-one connection is treated the same as N one-to-one connections.

Definition: *Source-based* fairness divides bandwidth fairly among active sources as if they were sources in point-to-point connections, ignoring group memberships. The allocation vector $\{x_1, x_2, \ldots, x_n\}$ is determined by applying the underlying fairness definition for all active sources x_i. □

With source-based fairness, VCs that have a larger number of concurrently active senders get more bandwidth than VCs with less concurrent senders on the same link. The resource allocation may be unfair *among the VCs*.

VC/source-based. Fair bandwidth allocation can be performed for all VCs, and allocations to the sources in the same VC can be fair *within* the VC.

Definition: *VC/source-based* fairness first gives fair bandwidth allocations among the VCs, and then fairly allocates the bandwidth of each VC among its sources.
The allocation vector $\{y_1, y_2, \ldots, y_m\}$ is determined by applying the underlying fairness definition for all active VCs y_j.
The allocation vector $\{x_1, x_2, \ldots, x_N\}$ for each of the VCs is determined by applying the underlying fairness definition for all active sources x_i in the VC, based upon the capacity available for the VC. □

Flow-based. A third possibility is *flow-based* fairness. Intuitively, each VC coming on an input port (link) is considered a separate flow. Hence, two VCs coming on the same input port are considered two separate flows, and traffic coming from two *different* input ports on the same VC (and being merged at the switch) is also considered as two separate flows. *The key advantage of this technique is that a switch can easily distinguish the flows.*

Definition: *Flow-based* fairness gives fair allocations for each active flow, where a flow is a VC coming on an input link. Formally, we define the number of flows for an output port as the sum of the number of active VCs sending to this output port, for each of the input ports of the switch:
$NumFlows_j$, $j \in OutputPorts =$
$\forall i, i \in InputPorts, \sum_i$ Number of active VCs coming on port i and being switched to port j
The allocation vector $\{z_1, z_2, \ldots, z_f\}$ is determined by applying the underlying fairness definition for all active flows z_k being switched to the output port. □

For example, assume that at a certain switch, traffic is coming from three different input ports (ports 1, 2, and 3). The traffic is being switched to the same output port

(port 4). One of the input ports (port 2) has two VCs sending to port 4, while each of the other two ports (ports 1 and 3) has only one VC sending to port 4 (may be the same VC, but different senders). Then the number of flows at port 4 would be considered as 2 (port 2), plus 1 (port 1), plus 1 (port 3), equals four. Note that the flow-based allocation is "local" since there is no global notion of flow (two flows can merge and become a single flow). We will later see that the flow-based fairness definition suffers from a major drawback.

VC/flow-based. The flow-based definition can be also be adopted within each VC in the VC-based approach. We call this VC/flow-based fairness.

Definition: *VC/flow-based* fairness first gives fair bandwidth allocations among the VCs, and then fairly allocates the bandwidth of each VC among its flows.
The allocation vector $\{y_1, y_2, \ldots, y_m\}$ is determined by applying the underlying fairness definition for all active VCs y_j.
The allocation vector $\{z_1, z_2, \ldots, z_F\}$ for each of the VCs is determined by applying the underlying fairness definition for all active flows z_k in the VC, based upon the capacity available for the VC. □

8.2.2 Examples

We illustrate multipoint fairness with the aid of two examples. We postpone the discussion of how to design and implement the rate allocation algorithms to section 8.3. The first example illustrates a downstream bottleneck situation, while the second one shows an upstream bottleneck, to illustrate the allocation of capacity left-over by connections bottlenecked elsewhere.

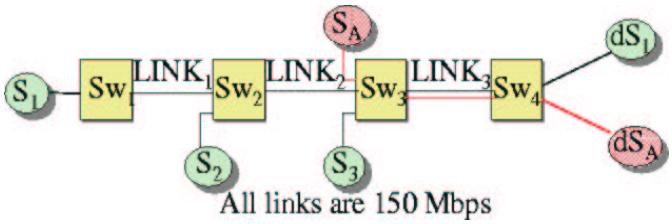


Figure 8.3: Example multipoint-to-point configuration with a downstream bottleneck

Example 1

Figure 8.3 illustrates a configuration with two VCs: one of the VCs is a multipoint-to-point VC with three senders and one receiver, and the other is a point-to-point VC. Sources S_1, S_2, and S_3 are sending to destination dS_1, and source S_A is sending to destination dS_A. All links are approximately 150 Mbps (after SONET overhead is accounted for). Clearly, all four sources are sharing a bottleneck link ($LINK_3$) between $Switch_3$ and $Switch_4$. *The aim of this example is to show the division of the 150 Mbps capacity of the bottleneck link among the sources.*

Source-based Definition. In this case, we disregard which sources belong to which connections, and simply treat this as a regular "four sources on a single bottleneck" situation. Applying the max-min fairness definition among sources, the allocations computed are:

$$\{S_1, S_2, S_3, S_A\} \leftarrow \{37.5, 37.5, 37.5, 37.5\}$$

Each of the four sources is allocated $\frac{1}{4} \times 150 = 37.5$.

Observe, however, that on $LINK_3$, the multipoint-to-point VC is getting 3 times as much bandwidth as the point-to-point VC. If there were 100 concurrent senders in the multipoint-to-point VC, the VC obtains 100 times as much bandwidth as the

point-to-point VC. In essence, the bandwidth allocated to a multipoint-to-point VC with N concurrent senders all bottlenecked on a certain link would be N times the bandwidth for a point-to-point VC bottlenecked on that same link, and N/K times that for a K-sender multipoint-to-point VC bottlenecked on the same link.

VC-based Definition: VC/Source. If a VC-based definition is adopted, we are essentially dividing up the fair allocation computation process into two phases. In the first phase, we ignore the number of senders in each VC, and simply apply the max-min fairness computation to the VCs. In the second phase, we consider each multipoint-to-point VC separately and divide up its allocation max-min fairly among the senders in that VC. This process is repeated for each multipoint-to-point VC.

According to this definition, the allocation vector for the example above would be:

$$\{S_1, S_2, S_3, S_A\} \leftarrow \{25,\ 25,\ 25,\ 75\}$$

This is because both of the VCs are bottlenecked at $LINK_3$, so each VC is allocated half of the available bandwidth ($\frac{1}{2} \times 150 = 75$). Then, for the multipoint-to-point VC, $LINK_3$ is again the bottleneck, so each of the three sources gets one third of the bandwidth allocated to this VC ($\frac{1}{3} \times 75 = 25$).

Flow-based Definition. Recall that a flow was defined as a VC coming on an input port. Hence, the number of flows switched to $LINK_3$ is three, and each of the flows gets one third of the bottleneck bandwidth ($\frac{1}{3} \times 150$). This bandwidth is then divided equally among the two flows detected at the output port of $Switch_2$, producing the allocation vector:

$$\{S_1, S_2, S_3, S_A\} \leftarrow \{25,\ 25,\ 50,\ 50\}$$

The allocation suffers from the "beat-down problem" commonly observed with bit marking switches. We discuss this further in section 8.2.3.

VC-based Definition: VC/Flow. If we use the VC-based approach, but instead of fairly dividing the bandwidth among the senders in the same VC, we fairly divide the bandwidth among the flows in the VC, we obtain the following allocation vector:

$$\{S_1, S_2, S_3, S_A\} \leftarrow \{18.75,\ 18.75,\ 37.5,\ 75\}$$

This is because the bandwidth is divided max-min fairly among the two VCs at $Switch_3$, giving 75 Mbps to each VC. For the multipoint-to-point VC, $Switch_3$ divides the 75 Mbps equally among the two flows in that VC, so the flow originating from S_3 is allocated $\frac{1}{2} \times 75 = 37.5$ Mbps. $Switch_2$ divides the 37.5 Mbps that $Switch_3$ had allocated to the flow consisting of S_1 and S_2 equally among these two sources, each obtaining $\frac{1}{2} \times 37.5 = 18.75$ Mbps.

Example 2

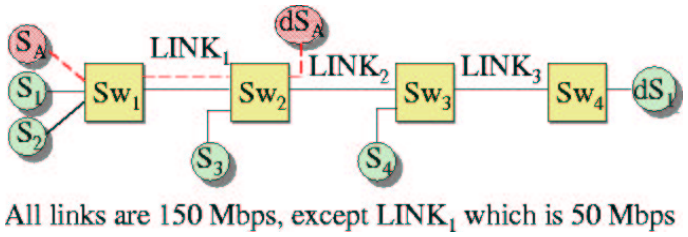


Figure 8.4: Example multipoint-to-point configuration with an upstream bottleneck

Figure 8.4 illustrates a configuration with two VCs: one of the VCs is a multipoint-to-point VC with four senders and one receiver, and the other is a point-to-point VC. Sources S_1, S_2, S_3 and S_4 are sending to destination dS_1, and source S_A is sending

to destination dS_A. All links are approximately 150 Mbps (after SONET overhead is accounted for), except for the link between $Switch_1$ and $Switch_2$ ($LINK_1$) which is only 50 Mbps. Clearly, sources S_1, S_2 and S_A are bottlenecked at $LINK_1$, while sources S_3 and S_4 are bottlenecked at $LINK_3$. *The aim of this example is to illustrate the allocation of the capacity left over by sources bottlenecked on $LINK_1$ to the sources bottlenecked on $LINK_3$.*

Source-based Definition. $\{S_1, S_2, S_3, S_4, S_A\} \leftarrow \{16.67,\ 16.67,\ 58.33,\ 58.33,\ 16.67\}$

This is because each of sources S_1, S_2 and S_A is allocated one third of the bandwidth of $LINK_1$. At $LINK_3$, the $50 \times \frac{2}{3} = 33.33$ Mbps used by sources S_1 and S_2 is subtracted from the available bandwidth, and the remaining capacity (116.67 Mbps) is equally divided upon sources S_3 and S_4.

VC-based Definition: VC/Source. $\{S_1, S_2, S_3, S_4, S_A\} \leftarrow \{12.5,\ 12.5,\ 62.5,\ 62.5,\ 25\}$

This is because each of the VCs is allocated half of the bandwidth on $LINK_1$, and this bandwidth is divided equally among S_1 and S_2 of the multipoint VC. On $LINK_3$, the remaining capacity ($150 - 25 = 125$ Mbps) is divided max-min fairly among the sources within the multipoint-to-point VC.

Flow-based Definition. $\{S_1, S_2, S_3, S_4, S_A\} \leftarrow \{16.67,\ 16.67,\ 41.67,\ 75,\ 16.67\}$

This is because $Switch_3$ detects two flows on $LINK_3$, and allocates half of the capacity to each flow (hence, source S_4 is allocated half of $LINK_3$ bandwidth). $Switch_1$ divides the 50 Mbps equally among the three flows sharing $LINK_1$ (each of S_1, S_2

and S_A gets $\frac{1}{3} \times 50 = 16.67$). $Switch_2$ divides the 75 Mbps (that $Switch_3$ had allocated to the flow emerging from it) equally among the flow from S_3 and the flow from $Switch_1$, but detects that one of the flows (that from $Switch_1$, i.e., S_1 and S_2) is only using 33.33 Mbps, so it allocates the remaining $75 - 33.33 = 41.67$ Mbps to source S_3.

VC-based Definition: VC/Flow. $\{S_1, S_2, S_3, S_4, S_A\} \leftarrow \{12.5, 12.5, 50, 75, 25\}$

$Switch_1$ divides the available 50 Mbps equally among the two VCs (giving each 25 Mbps), and divides the bandwidth of the multipoint-to-point VC equally among the two flows in that VC (each getting 12.5 Mbps). $Switch_3$ divides the bandwidth fairly among the two flows, allocating 75 Mbps to the flow from S_4 and 75 Mbps to the flow from $Switch_2$. $Switch_2$ detects that the flow from S_1 and S_2 is only using 25 Mbps, so it allocates the remaining 50 Mbps to the other flow (source S_3).

8.2.3 Merits and Drawbacks of the Different Definitions

The examples above illustrate that fairness based upon the concepts of source, VC, and flow can give different allocations. Appropriate fairness definitions are affected by application types and pricing issues.

Flow and VC/flow. The flow-based method is not max-min fair if we view an N-to-one connection as N one-to-one connections, since the same flow can combine more than one source. More importantly, the flow-based and VC/flow-based methods inherently suffer from the beat-down problem. The "beat-down problem" means that sources whose flow travels a larger number of hops are allocated less bandwidth than those traveling a smaller number of hops, even if both flows have the same bottleneck.

In flow-based fairness for multipoint connections, sources whose flow traverses a larger number of merge points are allocated less bandwidth than those traversing a smaller number of merge points. This is because once the flows of two sources are merged at a switch, they are considered a single flow at all downstream switches. "Example 1" above clearly shows that sources 1 and 2 are allocated less bandwidth because their flows traverse a larger number of merge points than source 3.

One may, however, argue that it may be desirable to favor sources traversing a smaller number of merge points, since these are more likely to encounter less bottlenecks. For example, if a user in New York city is fetching some web pages from a server several hops away (say in Germany), he expects to wait longer than if he is fetching web pages within his same local network. Thus, although flow-based fairness is unfair to sources whose traffic is merged multiple times, it may be acceptable in many practical situations. The VC/flow-based fairness is max-min fair with respect to VCs, but within the same VC, it favors sources whose traffic goes through a smaller number of merge points. Hence, the beat-down problem is less pronounced than in the flow-based case.

Source and VC/source. Source-based fairness entirely ignores membership of sources to connections, and divides the available bandwidth fairly among the currently active sources. If pricing is based upon sources, this mechanism is good, since allocation is fair among sources.

If pricing is based on connections, however, a VC with a hundred concurrent senders should not be allocated hundred times the bandwidth of a point-to-point connection bottlenecked on the same link. Source-based fairness is clearly unfair if this is the pricing method adopted, and VC/source-based fairness is superior. The

application type also has a significant impact on this choice, since the application type determines the probability of having a large number of concurrent senders.

Summary and Conclusions. The above discussion shows that each type of fairness has its own merits and drawbacks, and the choice of the type of fairness to adopt relies on the application type, and pricing methods used. We prefer *source-based* fairness, as it is a natural extension of point-to-point fairness definitions. In addition, it is the simplest to implement (refer to section 8.3), and does not suffer from beat-down problems as with flow-based solutions. The inter-connection unfairness discussed above is not a major problem since the number of concurrent senders in the multipoint connection is usually small in typical applications (e.g., one speaker at a time in an audio conference). Pricing can be based on sources in this case. VC/source-based fairness is appropriate for applications with a large number of concurrent senders, which price bandwidth by connection/session. The next section discusses the design and implementation of algorithms to compute the various types of fair allocations.

8.3 Multipoint Algorithm Design Issues

There are several ways to design multipoint-to-point ABR flow control algorithms. Each method offers a tradeoff in complexity, scalability, overhead, and response time. In this section, we examine the design of multipoint rate allocation algorithms with VP merging, VC merging, different types of accounting, and we discuss scalability issues.

- **Delay Sensitivity.** Some merge point algorithms wait for an FRM cell to be received before sending feedback. What are the implications of this on the scalability of the scheme? Will the feedback delay grow with the number of

levels of merge points? If each merge point must wait for the next FRM cell, the time to return a BRM cell can increase with the number of levels of the tree, which is an undesirable property. This is also dependent on the FRM and BRM cell rates, and their relationships during transient phases. Schemes that return the BRM cell received from the root to the leaves (which have sent FRM cells to the merge point since the last BRM cell was passed), are less sensitive to number of merge points.

- **Merging.** With *VC merge* implementations (refer to section 2.9.2), it is impossible to distinguish among the cells of different sources in the same multipoint-to-point VC (since the same VPI/VCI fields are used for all the cells of a VC on the same hop). Hence, switch traffic management algorithms must not rely on being able to determine the number or rates of *active sources* with VC merge (number and rates of active VCs and number and rates of active flows can still be determined). With *VP merge*, however, the VCI field is used to distinguish among cells of different sources in the same multipoint-to-point VC on the same hop. Hence, it is possible to determine the number and rates of *active sources* in such implementations, and perform necessary per-source accounting operations. (This may, however, incur additional complexity and reduce scalability.)

- **Accounting.** All switch traffic management algorithms need to use some registers for storing the values they need to compute the rate allocations. Some of these values are stored for each input port, and some for each output port. Other algorithms use per-VC accounting, per-source accounting, or per-flow

accounting. With multipoint-to-point VCs, per-VC accounting, per-source accounting, and per-flow accounting are no longer equivalent (they are equivalent for point-to-point scenarios). This leads to a set of interesting problems. For example, some algorithms store the value of the current cell rate (CCR) indicated in FRM cells, and later use it for rate computation, while others use the CCR value from BRM cells directly in rate computation to avoid O(N) storage. But the CCR value is actually per-source (and the sources cannot be distinguished with VC merge). We have shown in [29] that using the CCR field from BRM cells can cause unfairness to upstream sources. Further, some algorithms attempt to measure the source rates, or distinguish overloading and underloading sources (e.g., MIT scheme, UCSC scheme, and basic ERICA). These schemes may be unstable in the multiple sender case. In general, *per-source accounting is infeasible with VC merge, while per-VC accounting must account for the VC as a whole (even if its traffic is coming from different ports), and per-flow accounting must distinguish both input ports and VCs.*

- **Bi-level operation.** For point-to-point and point-to-multipoint connections, and for multipoint-to-point connections with source-based fairness, the switch computes the rate allocations it can support based on available ABR capacity, and indicates these allocations in the BRM cells (only if they are less than the allocations computed by downstream switches, as indicated in the ER field of BRM cells). This suffices for these situations since the algorithm operates at the source level only, and all sources at a bottleneck are allocated equal rates. With VC/source, flow, and VC/flow-based fairness, however, switches may need to first estimate the maximum available capacity for the VC/flows (possibly using

the ER field in BRM cells coming from downstream, and the rate allocation algorithm applied at the link level), and then sub-divide this capacity among the senders/flows. Per-VC or per-flow accounting may be used for the sub-division operation. We call this bi-level operation. Alternatively, weights can be used to scale the rate allocations, but the weights depend on the number of senders/flows in the multipoint VC/port, which is a variable quantity.

8.4 The Algorithm

We first discuss the rate allocation algorithm, and then the merge point algorithm. Then we discuss some design issues.

8.4.1 Rate Allocation Algorithm

Rate allocation algorithms are employed at every network switch to compute and indicate the appropriate feedback to the sources. The algorithm we discuss is based upon the ERICA+ rate allocation algorithm [73]. However, we eliminate all the steps that required per-VC accounting in ERICA+ as explained in section 5.6. The reason for this is that rate algorithms perform per-VC accounting as if it were *per-source* accounting. Per-source accounting must be avoided for compatibility with VC merge switches and for scalability.

As explained in chapter 2 and 5, the algorithm uses a measurement interval to measure the quantities required for computing the rate allocation. At the end of every interval, the algorithm averages some of the quantities measured, and uses these quantities to give the appropriate feedback to the sources in the following interval. The algorithm also uses the current cell rate (CCR) of the sources, as indicated in

the FRM cells. In addition, it keeps track of the maximum explicit rate indicated to all sources sending to this port during each interval.

In the pseudo-code below, there are two options that are not necessary for the algorithm, but help reduce rate fluctuations in some cases (especially when the measurement interval value is very small). The first option (which we label option 1) does not use the most current CCR value from FRM cells, but uses the maximum of the CCR values seen in FRMs in the current interval. This option is useful when there are multiple sources in the same VC, as explained in the next subsection. The second option (option 2) uses exponential averaging for the maximum ER given in the previous interval to smooth out variations.

The algorithm executes for each output port: when an FRM cell is received, when a BRM cell is received, and at the end of each measurement interval. The algorithm is O(1) and its complexity is independent of the number of connections and the number of sources. Since the calculations of the input rate, target capacity and overload factor are the same as in the ERICA+ algorithm, we only briefly outline these here.

FRM cell is received for VC j:

(current cell rate)$_j$ $\leftarrow$ CCR field from the FRM cell

Or as an option (option 1: maximum CCR option):

IF (first FRM in interval)$_j$ = TRUE THEN

(current cell rate)$_j$ $\leftarrow$ CCR field from the FRM cell

(first FRM in interval)$_j$ $\leftarrow$ FALSE

ELSE

(current cell rate)$_j$ $\leftarrow$ maximum (CCR field from the FRM cell, (current cell

rate)$_j$)

END

BRM cell is to be sent out for VC j:

IF (overload factor $> 1+\delta$) THEN

ER $\leftarrow$ (current cell rate)$_j$/overload factor

ELSE

ER $\leftarrow$ maximum ((current cell rate)$_j$/overload factor, maximum ER in previous interval)

END

ER $\leftarrow$ minimum (target capacity, ER)

maximum ER in current interval $\leftarrow$ maximum (ER, maximum ER in current interval)

ER in BRM cell $\leftarrow$ minimum (ER, ER in BRM cell)

End of measurement interval:

target capacity $\leftarrow$ exponential average of (across intervals) of link capacity minus CBR and VBR capacity, scaled for queues to drain by using a fractional function (refer to [73])

input rate $\leftarrow$ exponential average (across intervals) of total ABR input cells being switched to this output port

overload factor $\leftarrow$ input rate/target capacity

$\forall j$ (first FRM in interval)$_j$ $\leftarrow$ TRUE

maximum ER in previous interval $\leftarrow$ maximum ER in current interval

Or as an option (option 2: averaging the maximum ER in previous interval option):

maximum ER in previous interval $\leftarrow$

(1-α) $\times$ maximum ER in current interval + α $\times$ maximum ER in previous interval

maximum ER in current interval $\leftarrow 0$

Notes:

1. The input rate, target capacity, overload factor, maximum ER in current interval and maximum ER in previous interval are computed and stored for each output port. The "first FRM in interval" (if used) and the "current cell rate" are stored for each VC for each output port.

2. In our simulations, the parameter δ is set to 0.1, and the parameter α is also set to 0.1. These are the recommended values for these parameters.

3. The "averaging of maximum ER in previous interval" option (option 2) slightly reduces rate oscillations in some cases. It is not essential if its implementation complexity is high.

4. The maximum CCR option (option 1) also reduces rate oscillations in cases of extremely small averaging interval values ($<$ 200 μs for rates about 10 Mbps per source). It is also unnecessary. Exponentially averaging the maximum CCR values across intervals might further improve the performance. The next subsection discusses the usage of CCR in more detail.

Reference [73] gives a proof that this algorithm converges to the max-min fair rates for a single bottleneck case. The main idea of the proof is that the algorithm is fair because it allocates all sources bottlenecked at the same link the exact same

rates. In addition, the algorithm converges to rates that result in an overload factor value close to one, because the rates are scaled by the overload factor.

8.4.2 Merge Point Algorithm

This algorithm is the same as the multipoint-to-point algorithm developed by Ren and Siu in [105]. The algorithm is employed at every merge point where cells from different sources in the same multipoint-to-point VC are being merged and follow the same path to the destination. We first give the pseudo-code for the algorithm, and then discuss some properties of the algorithm.

A flag (can be one bit) called Ready is maintained for each of the *flows* being merged. The flag indicates that an FRM cell has been received from this flow after a BRM cell had been sent to it.

Upon the receipt of an FRM cell from branch i:

1. Forward FRM cell to the outgoing link

2. Let Ready_i = TRUE

Upon the receipt of a BRM cell from the root:

FOR ALL upstream branches DO

IF Ready_i = TRUE THEN

Send a copy of the BRM to branch i

Let Ready_i = FALSE

END

END

When a BRM cell is about to be scheduled:

Perform the rate allocation algorithm as described in the previous section

Reference [105] gives a proof by induction on the number of levels of the multipoint tree to show that this algorithm gives fair allocations for multiple sources if the rate allocation algorithm employed gives max-min fair allocations.

8.4.3 Rate Allocation Design Issues

As previously mentioned, rate allocation algorithms for multipoint-to-point (or multipoint-to-multipoint) connections may not be able to distinguish cells from different sources in the same VC. Thus they cannot: (1) use the number of established connections as an indication of the number of sources, (2) measure or estimate the rate of each source, (3) distinguish between overloading and underloading sources, or compute the number of overloading sources, (4) estimate the effective number of active sources. Such techniques are used in many of the popular point-to-point switch schemes, such as the MIT scheme [16] and the UCSC scheme [79].

Most switch schemes also use the current cell rate of the sources in the computation of the explicit rate. **Algorithms which use the CCR values noted from backward RM cells are not fair for multipoint-to-point connections**. This is because it may be impossible to determine which source the RM cell belongs to. The CCR value in the BRM cells at the merge point may not capture upstream bottleneck information for *any* of the flows whose traffic is being merged, since it may actually be the CCR of a downstream source whose bottleneck rate is high. We explain this next.

Lemma 1: *Algorithms which use the CCR values noted from backward RM cells are not fair for multipoint-to-point connections:* $ER = f(CCR_{BRM})$ *is not necessarily max-min fair.*

Proof Sketch: The proof is by counter-example. We give a case where an algorithm using CCR_{BRM} gives unfair allocations. Suppose a multipoint-to-point VC has two sources, one of which has a bottleneck rate of 58 Mbps, and the other has a bottleneck rate of 16 Mbps, and the two sources are being merged at a switch. Figure 8.4 shows an example where at $Switch_2$, S_1 (and S_2) of rate 16 Mbps and S_3 of rate 58 Mbps are being merged (we will simulate this case in sections 8.5.2 to 8.5.2). The source which is bottlenecked at 16 Mbps (say S_1) shares its bottleneck link with a point-to-point connection (S_A to dS_A). At the merge point, BRM cells of the higher rate source (the 58 Mbps source) are more frequently sent to *all* the sources in this VC being merged with a high ER value (since the CCR is assumed to be 58 Mbps). This can result in over-allocation to the lower rate source(s) being merged, and unfairness to the point-to-point connection. □

Hence, algorithms that use the CCR value for rate computation must use the value of the CCR indicated in FRM cells for computation when a BRM cell is received. This is the most up-to-date value of CCR, since the CCR in the BRMs may be stale after traveling all the way to the destination and back. The CCR value in the FRM cells at the merge point captures upstream bottleneck information for one of the flows whose traffic is being merged. *The FRM cells of the sources being merged, however, may still be indistinguishable at the merge point. In the remainder of this section, we argue that this does not affect the convergence and steady state behavior of the algorithm.*

Lemma 2: *Algorithms which use the CCR values noted from forward RM cells can compute statistically fair allocations for multipoint-to-point connections.*

Proof Sketch:

Since the guaranteed fairness is statistical, the proof is also statistical. Assume that there are two flows S_{low} and S_{high} being merged. We will briefly examine the situation when the forward CCR used to compute the ER for a flow is not the CCR corresponding to that flow.

CASE 1:

When computing the ER for S_{low}, if the CCR of S_{high} is used, then the ER computed for S_{low} will be too high. But S_{low} is bottlenecked upstream of the merge point (otherwise its bottleneck rate will not be less than that for S_{high}, since S_{low} and S_{high} merge at the merge point and never split after that), so the ER given to S_{low} at the merge point will be overwritten by upstream switches.

CASE 2:

For the case when the CCR of S_{low} is used to compute the ER for S_{high}, first consider the algorithm with the maximum CCR option. The only situation when the ER for S_{high} is calculated based upon the CCR for S_{low} is when only FRM cells of S_{low} have been seen since the beginning of the current interval. (Note that if no FRM cells have been seen at all, the CCR value used is the maximum seen in the previous interval, which will be the CCR of the higher rate source S_{high} unless S_{high} is sending at a very low rate, in which case the scheme should *not* allocate it high rates: see the discussion in [82] for more details on handling low rate sources.) Since S_{high} has a higher rate, it has a higher frequency of FRM cells, so it becomes highly improbable for this to hold.

This argument can be extended for the algorithm without the maximum CCR option. In this case, instead of the smaller CCR being used when only FRM cells

from the lower rate source have been seen so far in this interval, it is the last FRM cell received that determines the CCR used. But, again, since the higher rate source has a higher FRM rate, it is statistically unlikely for the smaller CCR to be used. The maximum ER in the previous interval term ensures that if the small CCR is in fact used, the source is allocated at least as much as other VCs going to the same output port, which ensures fairness and fast convergence. □

8.4.4 Merge Point Design Issues

There are a number of ways to implement multipoint-to-point merge point algorithms. Each method offers a tradeoff in complexity, scalability, overhead, response time and steady state behavior.

In the above algorithm, a BRM cell is returned to a source for every one or more FRM cells it sends. Thus the BRM to FRM cell ratio at the source is less than or equal to one. In steady state, the ratio is likely to approach one, since the FRM rate and BRM rate will be similar. This is an important property of ABR flow control that should be maintained for multipoint-to-point connections. The BRM to FRM ratio *in the network* is also one in this case. (If FRM cells are turned around at merge points as in [104], the same FRMs can be turned around at another merge point or the destination, creating BRM cells that eventually get discarded in the network.)

Also observe that in this scheme, since the merge point does not need to turn around every FRM cell, the overhead of the algorithm is reduced. However, the scheme needs to duplicate BRM cells. With the new advances in multicast ATM switch architectures, this operation can be quite efficient.

The algorithm we use returns a BRM cell received from the root to the branches which have sent FRM cells to the merge point since the last BRM cell had been passed. This makes the scheme less sensitive to the number of levels of merge points, as compared to those schemes which turn around FRM cells (such as the scheme in [104]). This is because schemes turning around FRMs have to wait for an FRM to be received at every merge point, so their response time increases with the number of levels in the tree. In addition, the ER value returned by such schemes may be incorrect if no BRM cells have been received since the last one was sent, leading to rate oscillations and possibly large queue lengths.

8.5 Performance Analysis

This section provides a simulation analysis of the multipoint algorithm described in the previous two sections. Only a few simple experiments are shown here; more stringent tests have been conducted, and the preliminary results are consistent with those presented next.

The results are presented in the form of four graphs for each configuration:

(a) Graph of allowed cell rate (ACR) in Mbps versus time for each source

(b) Graph of ABR queue lengths in cells versus time at the bottleneck port of each switch

(c) Graph of link utilization versus time for each of the main (backbone) links (those that connect two switches to each other)

(d) Graph of number of cells received versus time for each destination

8.5.1 Parameter Settings

Throughout our experiments, the following parameter values are used:

1. Except where otherwise indicated (in sections 8.5.2 to 8.5.2), all links have a bandwidth of 155.52 Mbps (149.76 Mbps after SONET overhead is accounted for).

2. All multipoint-to-point traffic flows from the leaves to the root of the tree. No traffic flows from the root to the leaves, except for RM cells. Point-to-point connections are also unidirectional.

3. Except in section 8.5.2 where we experiment with the source parameter rate increase factor (RIF), we have set RIF to 1/32 in our simulations. We do not, however, expect the performance of the algorithm to be significantly influenced by the value of RIF, as seen in section 8.5.2.

4. 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 scheme from the rate reductions due to TBE.

5. All other ABR parameters are set to their default values [43].

6. A dynamic queue control function is used to scale the available capacity and achieve a constant queuing delay in steady state [73]. The "target delay" parameter specifies the desired queuing delay. A value of 1.5 ms was used. An inverse hyperbolic function is used. The hyperbolic function curve parameters

used were $a = 1.15$ and $b = 1$. The queue drain limit factor is set to 0.5 (which means that up to 50% of the link capacity can be used to drain queues).

7. A fixed time measurement interval is used to measure and average the input rate and available capacity, and to note the maximum allocation given (and possibly the maximum CCR value in FRM cells). The interval is set to 5 ms in all experiments except those in section 8.5.2.

8. Since we do not implement VC merge in our switches, we only use one cell long packets. Our next study will implement VC merge and examine its effect.

9. All sources are deterministic, i.e., their start/stop times and their transmission rates are known.

10. Simulation time is two seconds.

11. The simulations use both the maximum CCR option and exponentially averaging the maximum ER option as discussed in section 8.4.1. We have simulated all our configurations without using either option, and with each option separately, and the differences were insignificant. We do not show these results here for space considerations. In particular, the results when neither of the two options is enabled, and with extremely small measurement intervals (as with the simulations in section 8.5.2) showed that the algorithm still rapidly converges to the optimal allocations, and that the oscillations (though they do slightly increase) were *not* significantly more than the results we show in section 8.5.2.

8.5.2 Simulation Results

In this section, we discuss a *sample* of our simulation results. We mainly use two configurations, and experiment with different link lengths, initial cell rates of the sources, rate increase factor values, and lengths of the measurement interval.

Downstream Bottleneck Configuration

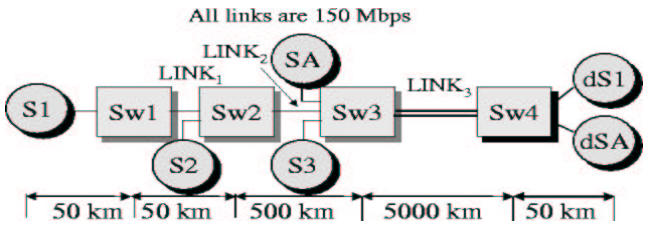


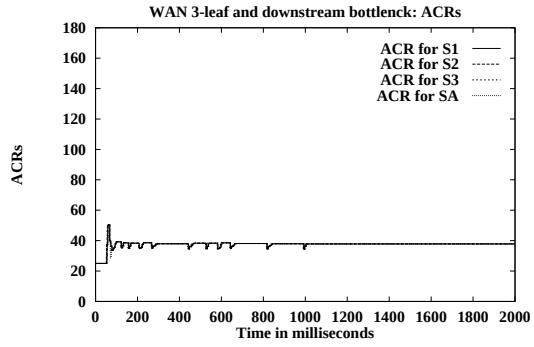
Figure 8.5: Example multipoint-to-point configuration with a downstream bottleneck

Figure 8.5 illustrates a configuration with two VCs: one of the VCs is a multipoint-to-point VC with three sources and one destination, and the other is a point-to-point VC. Sources S_1, S_2, and S_3 are sending to destination dS_1, and source S_A is sending to destination dS_A. All links are 149.76 Mbps (OC-3 links after SONET overhead is accounted for), and their lengths are as shown in the figure. Clearly, all four sources are sharing a bottleneck link ($LINK_3$) between $Switch_3$ and $Switch_4$. This experiment shows the division of the capacity of this bottleneck link among all sources in both types of connections.

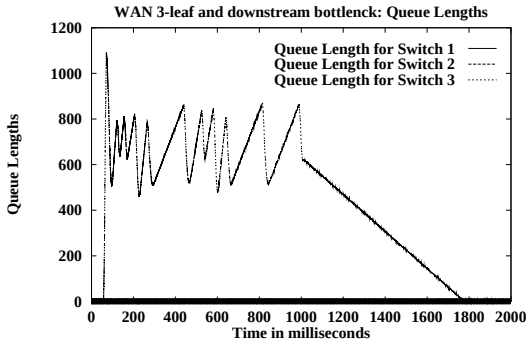
Applying the fairness definition among sources, the optimal allocations should be:

$\{S_1, S_2, S_3, S_A\} \leftarrow \{37.5, 37.5, 37.5, 37.5\}$

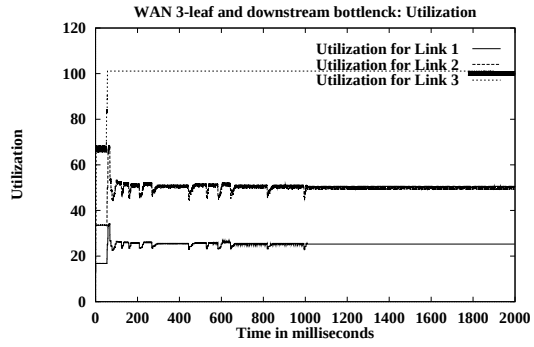
Each of the four sources is allocated $\frac{1}{4} \times 150 = 37.5$.



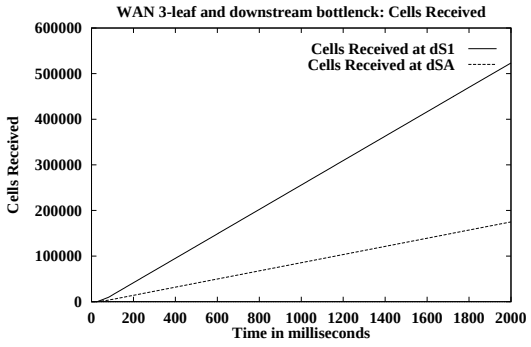
(a) Allowed Cell Rate in Mbps



(b) ABR Queue Lengths in Cells



(c) Link Utilization



(d) Cells Received

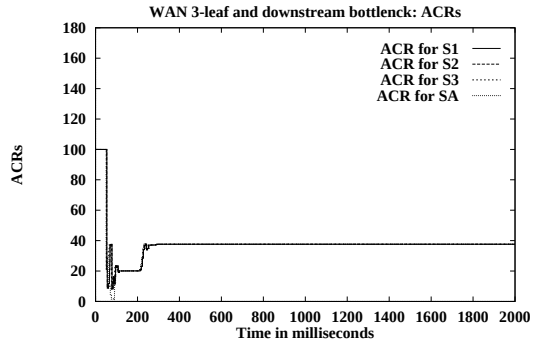
Figure 8.6: Simulation results for a WAN multipoint-to-point configuration with a downstream bottleneck (long LINK3, low ICR)

Figure 8.6 illustrates the results of simulating the above configuration. The sources start with an ICR value of 25 Mbps, which is below their optimal allocation. Clearly, all sources rise to their optimal rates quickly (figure 8.6(a)), and the queues are small (figure 8.6(b)). The bottleneck link ($LINK_3$) is fully utilized (figure 8.6(c)). $LINK_2$ is 50% utilized (since only 2 of the 4 sources utilize it) and $LINK_1$ is only 25% utilized (1 out of 4 sources).

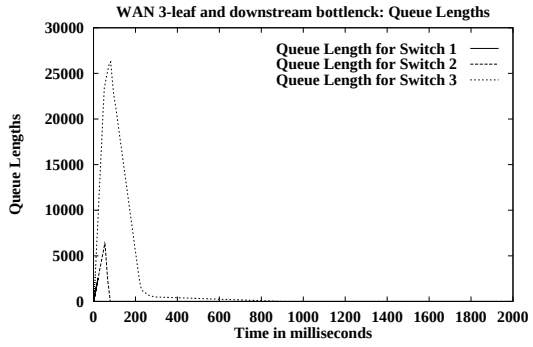
Observe that with source-based fairness, VCs that have a larger number of concurrently active sources get more bandwidth than VCs with less concurrent sources on the same link. This can be clearly seen from the slope of the cells received graph (figure 8.6(d)) for dS_1 and dS_A. Clearly dS_1 has a slope that is three times as large as that for dS_A. After 2 seconds, the ratio of cells received at dS_A to dS_1 is around 175000 to 520000, which is exactly 1 to 3. Thus the resource allocation is not fair among the VCs.

Figure 8.7 illustrates the results of simulating the same configuration (figure 8.5) when all sources start at a high ICR value. The ICR for all sources here is 100 Mbps. This creates an initial overload on $LINK_3$ of $\frac{400}{150} = 2\frac{2}{3}$. The algorithm recovers from this situation and all sources converge to the correct value of approximately 37.5 Mbps (figure 8.7(a)). The queues at $Switch_3$ start dropping after approximately one round trip (figure 8.7(b)).

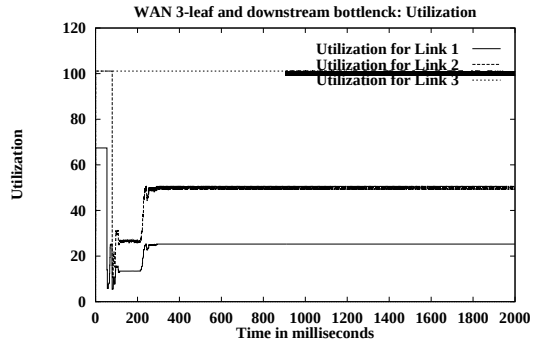
Figure 8.8 shows the results for the same configuration, but different sources start at different ICR values. Sources S_1 and S_3 start at an ICR of 65 Mbps, while sources S_2 and S_A start at 10 Mbps. Notice that the sum of the source rates for all sources is 150 Mbps, so the initial load value is close to 1. The rates for sources S_1 and S_3 are



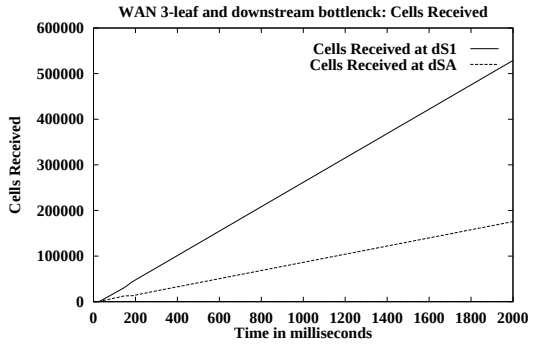
(a) Allowed Cell Rate in Mbps



(b) ABR Queue Lengths in Cells

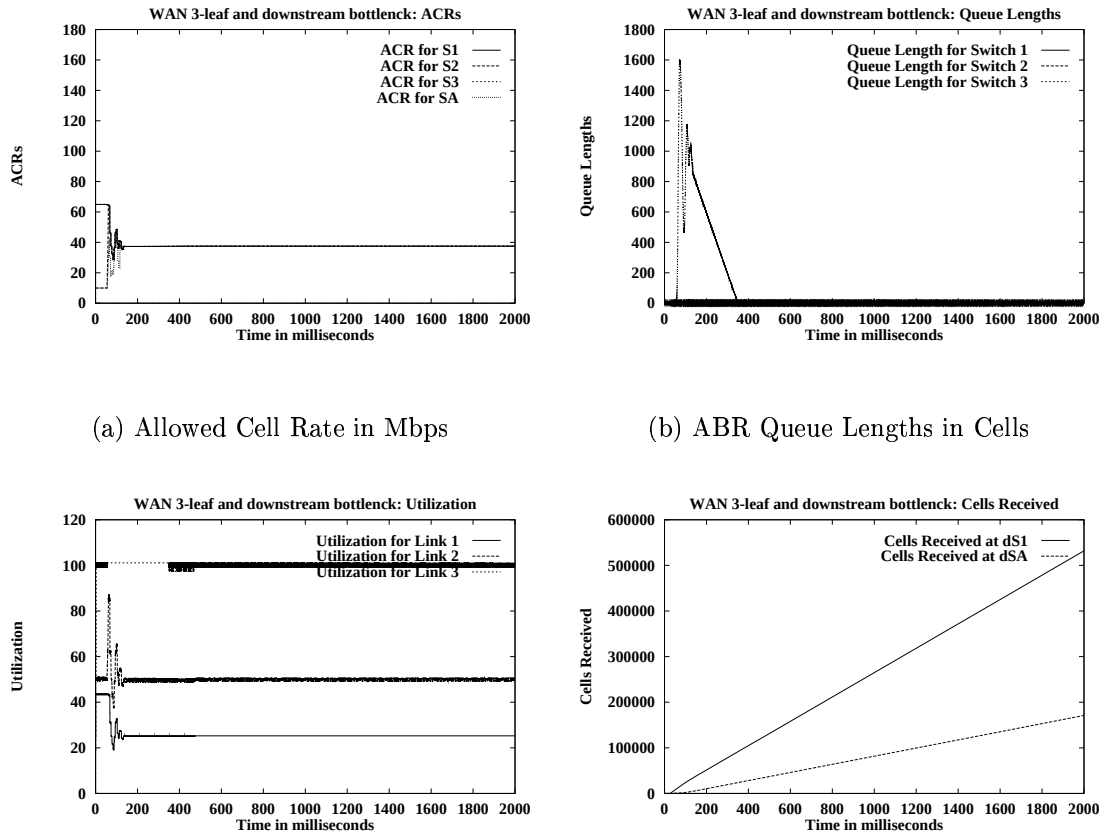


(c) Link Utilization



(d) Cells Received

Figure 8.7: Simulation results for a WAN multipoint-to-point configuration with a downstream bottleneck (long LINK3, high ICR)



(a) Allowed Cell Rate in Mbps

(b) ABR Queue Lengths in Cells

(c) Link Utilization

(d) Cells Received

Figure 8.8: Simulation results for a WAN multipoint-to-point configuration with a downstream bottleneck (long LINK3, different ICRs)

quickly reduced, while those of sources S_2 and S_A quickly rise, as seen in figure 8.8(a). The queues are also quite small (figure 8.8(b)).

Upstream Bottleneck with Heterogenous Links Configuration

Figure 8.9 illustrates the same configuration as in figure 8.4, where all links are approximately 150 Mbps, except for the link between $Switch_1$ and $Switch_2$ ($LINK_1$) which is **only 50 Mbps**. The link lengths are as shown in the figure.

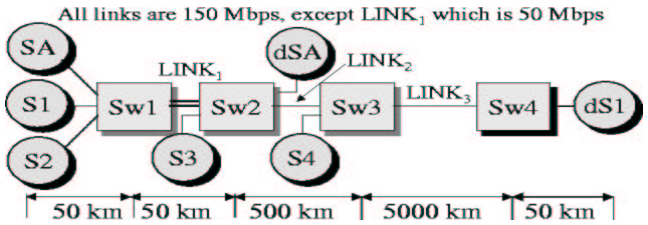


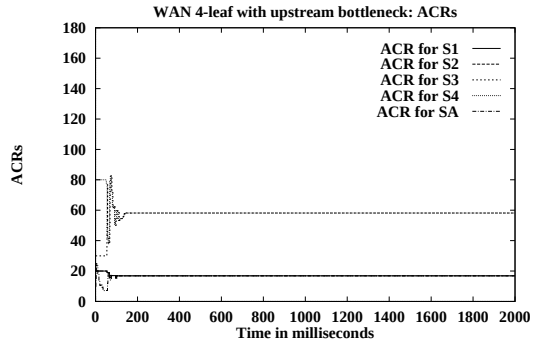
Figure 8.9: Example multipoint-to-point configuration with an upstream bottleneck

Recall that the allocation vector according to the source-based fairness definition is:

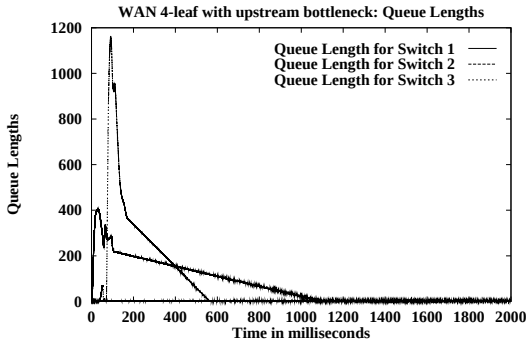
$$\{S_1, S_2, S_3, S_4, S_A\} \leftarrow \{16.67, 16.67, 58.33, 58.33, 16.67\}$$

Figure 8.10 illustrates the results for this configuration. Sources S_1 and S_2 start at an ICR of 20 Mbps. Source S_3 starts at 30 Mbps and source S_4 starts at 80 Mbps. Source S_A starts at 10 Mbps.

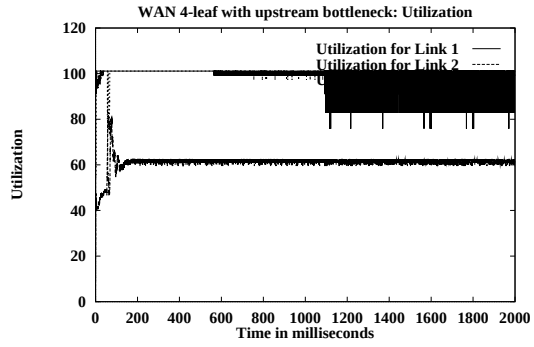
As seen in figure 8.10(a), sources S_1, S_2 and S_A converge to about 16.67 Mbps, while sources S_3 and S_4 converge to about 58.33 Mbps. The queues are bounded to reasonable values (figure 8.10(b)) and utilization of the bottleneck links ($LINK_1$ and $LINK_3$) are close to 100% (figure 8.10(c)). Destination dS_A gets much less throughput than dS_1 (figure 8.10(d)), since source S_A is bottlenecked on a 50 Mbps link with 2 other sources. After 2 seconds, the ratio of the throughputs for destinations dS_A to dS_1 is approximately 80000 to 700000 which is 0.11. The slopes of the two lines also have the same ratio. This is close to the optimal value since 16.67/149.76 = 0.11.



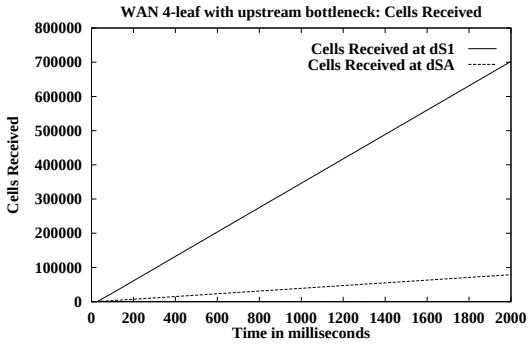
(a) Allowed Cell Rate in Mbps



(b) ABR Queue Lengths in Cells



(c) Link Utilization



(d) Cells Received

Figure 8.10: Simulation results for a WAN multipoint-to-point configuration with an upstream bottleneck (long LINK3)

Effect of Large Rate Increase Factor Values

The rate increase factor determines the maximum increase when a BRM cell indicating underload is received. If the RIF is set to a fraction less than one, the maximum increase at each step is limited to RIF $\times$ the peak cell rate for the VC. Setting RIF to small values is a more conservative strategy that controls queue growth and oscillations, especially during transient periods. It, however, may slow down the response of the system when capacity suddenly becomes available leading to underutilization.

Figure 8.11 illustrates the results for the configuration of figure 8.9 when the rate increase factor (RIF) is set to its maximum possible value, which is 1. Part (a) of the figure shows that the rates do not oscillate more than the corresponding figure with a small RIF value (figure 8.10(a)). The queues in figure 8.11(b) are also similar to those in figure 8.10(b).

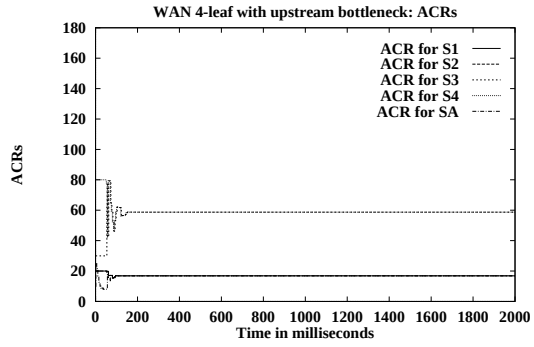
Effect of Extremely Short Measurement Intervals

As discussed in section 8.4.1, extremely short measurement intervals can cause the algorithm to suffer from oscillations. To examine this effect, we have simulated the algorithm with a measurement interval of 200 μs. Recall that in the upstream bottleneck configuration (shown in figure 8.9), the optimal rates for sources S_1, S_2 and S_A are 16.67 Mbps, and those for sources S_3 and S_4 are 58.33 Mbps. This implies that, in steady state, RM cells for sources S_1, S_2 and S_A arrive every:

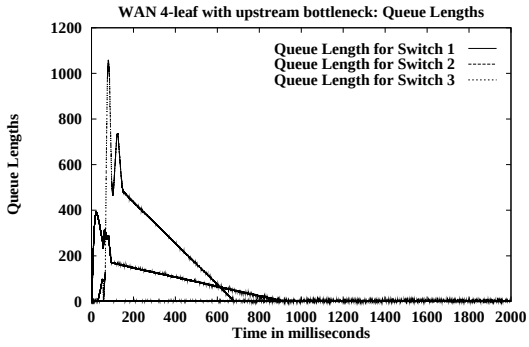
$$\frac{Nrm \times bits/cell}{ACR} = \frac{32 \times 53 \times 8}{16.67\ M} = 813.92\ \mu s$$

For sources S_3 and S_4, RM cells arrive every:

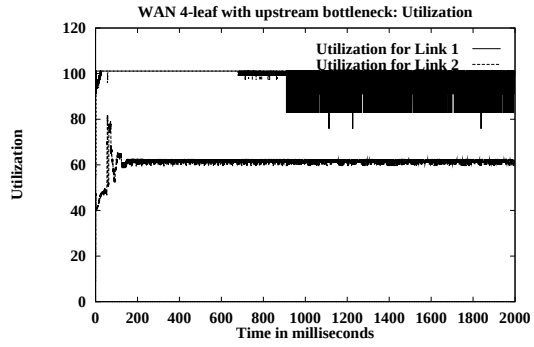
$$\frac{Nrm \times bits/cell}{ACR} = \frac{32 \times 53 \times 8}{58.33\ M} = 232.61\ \mu s$$



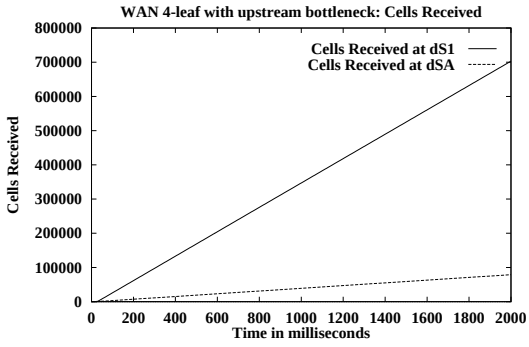
(a) Allowed Cell Rate in Mbps



(b) ABR Queue Lengths in Cells



(c) Link Utilization



(d) Cells Received

Figure 8.11: Simulation results for a WAN multipoint-to-point configuration with an upstream bottleneck (long LINK3, large RIF)

where a source sends an FRM cell every Nrm cells, and the default value of Nrm is 32.

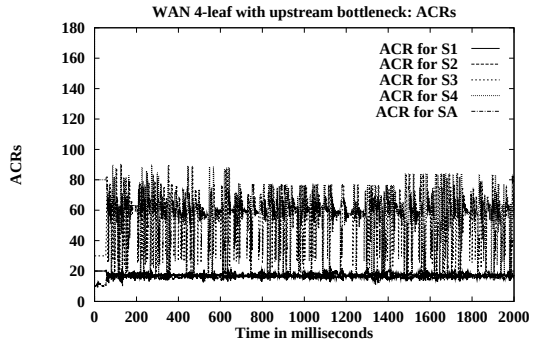
Setting the measurement interval to 200 μs means that RM cells for S_3 and S_4 might not be received every measurement interval, and that RM cells for S_1, S_2 and S_A might not be received for 4 consecutive measurement intervals.

In order to receive at least one FRM cell from the highest rate source in a certain interval, the interval length should be $> \frac{Nrm}{ACR\ maximum}$. This condition is likely to hold for reasonably long intervals, unless *all* sources are sending at very low rates, in which case the overload factor will be low and their rates will increase if they have data to send.

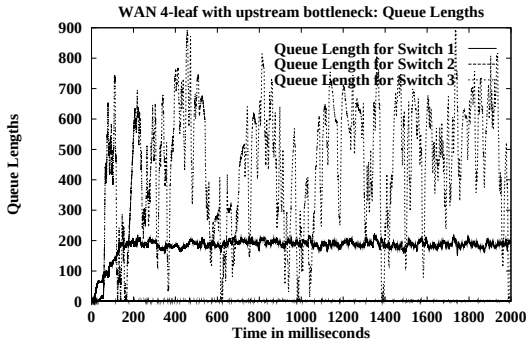
Figure 8.12 illustrates the results for the configuration of figure 8.9. Clearly, the short averaging interval causes more oscillations, but the rates of the sources still converge to their fair rates. Also observe that the number of cells received for both connections is the same as in figure 8.10(d). Increasing the value of the parameter α (in section 8.4.1) can reduce the oscillations.

8.6 Chapter Summary

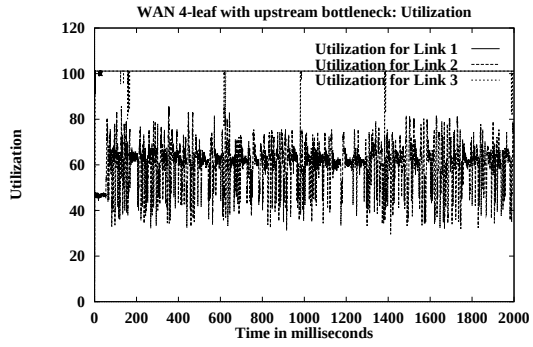
There are several issues to be resolved in ATM multipoint communication, including devising a scalable method for merging traffic from multiple senders, and resolving traffic management issues. ATM-ABR should be supported, as ABR provides good performance for both data and real-time applications, because of its loss and rate guarantees and delay control, as well as its efficient utilization of bandwidth and buffering.



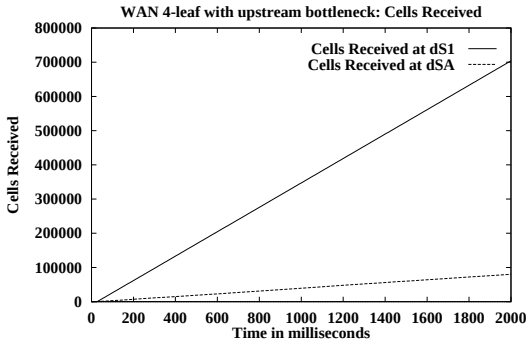
(a) Allowed Cell Rate in Mbps



(b) ABR Queue Lengths in Cells



(c) Link Utilization



(d) Cells Received

Figure 8.12: Simulation results for a WAN multipoint-to-point configuration with an upstream bottleneck (long LINK3, short interval)

Traffic management for multipoint connections may be implemented differently in VC merge and VP merge implementations. VP merge uses the VCI field to distinguish among different sources in the same multipoint VC, while VC merge does not distinguish sources, and implements packet-level buffering at the merge points.

Four different types of fairness can be defined for multipoint-to-point connections:

1. **Source-based fairness** divides bandwidth fairly among active sources as if they were sources in point-to-point connections, ignoring group memberships.

2. **VC/source-based fairness** first gives fair bandwidth allocations at the VC level, and then fairly allocates the bandwidth of each VC among the active sources in this VC.

3. **Flow-based fairness** gives fair allocations for each active flow, where a flow is a VC coming on an input link. Formally,

   $NumFlows_j,\ j \in OutputPorts =$

   $\forall i, i \in InputPorts, \sum_i$ Number of active VCs coming on port i

   and being switched to port j

4. **VC/flow-based fairness,** which first divides the available bandwidth fairly among the active VCs, and then divides the VC bandwidth fairly among the active flows in the VC.

Modifications are necessary for switch algorithms to implement each of the four types of fairness. For source-based fairness (the simplest), algorithms operating with VC merge should not perform source-level accounting, and must only use information supplied in the RM cells, in addition to aggregate measurements of load, capacity

and queuing delays. VC/source, flow and VC/flow fair allocations require bi-level operation or use of weights [33, 30].

All source-based switch algorithms operating in VC merge switches need to avoid distinguishing among sources in the same VC. Key lessons learned from this study include (refer to section 8.4.3 for supporting arguments):

1. Source-level accounting should not be performed in multipoint rate allocation algorithms. For example, measuring the rates for each source, or distinguishing overloading and underloading sources cannot be performed. If such accounting is performed at the VC level or the flow level, an additional mechanism to divide VC or flow bandwidth among sources is necessary.

2. Estimating the effective number of active sources in order to divide the available capacity among them is very difficult in multipoint connections, since it is impossible to distinguish among sources in the same multipoint VC with VC merge implementations.

3. The only information a multipoint rate allocation algorithm can use is the information supplied in RM cells, in addition to *aggregate* measurements of load, capacity and queuing delays.

4. CCR values from BRM cells should not be used in computing rate allocations for sources in multipoint connections, since the CCR value can be that of another source that does not go through the switch performing the computation. That source may have a much higher bottleneck rate, and using its CCR can result in unfairness.

5. CCR values from FRM cells can be used to compute rate allocations for sources in multipoint connections, even though the CCR used to compute the rate for a source may not actually be the CCR value of the source. This does not create problems due to the properties of the merged flow (see section 8.4.1 for a more detailed explanation). The maximum CCR value seen in an interval can be used instead of the CCR of the source. Exponential averaging of the maximum CCR seen or maximum ER given may further improve the performance of the algorithm.

6. Merge point algorithms should avoid changing the BRM to FRM ratio at the source or inside the network, to maintain the rate of feedback that the source requires, and avoid excessive overhead in the network. Scalability of the scheme is also affected by these ratios. Excessive complexity, noise, and response time can also be avoided by returning the BRM cells coming from the root, instead of turning around the RM cells at the merge points (refer to section 8.4.4 for supporting arguments).

We have given and simulated an O(1) algorithm for computing source-based fair allocations for multipoint-to-point and point-to-point connections. The algorithm uses simple aggregate measurements and maximum CCR values from FRM cells during successive intervals to perform rate computation. The algorithm exhibited very good behavior for the configurations tested. More extensive performance analysis is crucial to examine the fairness, complexity, overhead, transient response, delays, and scalability tradeoffs in multipoint algorithm design. Extending multipoint-to-point schemes for multipoint-to-multipoint connections can be performed by combining

point-to-multipoint algorithms (such as those developed in [26]) with the multipoint-to-point algorithm.

While our four fairness definitions are already included as baseline text in the living list of the ATM Forum traffic management working group, it is essential to continue this work to define the desirable forms of fairness, and extend current switch traffic management algorithms for multipoint connections. Extensive performance analysis is also crucial to examine the fairness, complexity, overhead, transient response, delays, and scalability tradeoffs involved. Multipoint-to-multipoint connections can be supported by combining branch point algorithms (e.g., [26]) with merge point algorithms (e.g., [29]).

This work is also related to the fairness issues in Internet multicast protocols. Again the fairness, reliability and congestion control method to adopt is application specific. The inter-group and intra-group fairness notions discussed in this chapter need to be precisely defined and measured for Internet traffic, and protocols need to be developed to provide multicast services that achieve fairness. This topic is currently gaining considerable attention at the IETF, with the increasing popularity of group collaboration applications.

CHAPTER 9

SUMMARY AND OPEN ISSUES

Computer networks have become an integral part in our everyday lives. They are affecting our business, communication, and entertainment, and every aspect of our lives. Corporations extensively use the Internet and intranets for communication and information. If the Internet is to become ubiquitous, the services it provides must be reliable and timely. Therefore, powerful traffic management techniques are the key to the development of these services and the success of the Internet.

In this research, we have studied traffic management for efficient group communication services. We have developed fairness definitions and flow control algorithms for ABR unicast and multicast connections, including connections with multiple concurrent and indistinguishable senders. We have also developed an architecture and framework for multiplexing virtual connections on virtual paths. Finally, we have conducted sensitivity analysis for ABR parameters, and performed extensive simulations of various configurations and traffic patterns. We have focused on the fairness, efficiency, link utilization and buffer requirements of the developed schemes. We have also emphasized the stability and transient response for feedback consolidation algorithms for connections with multiple receivers.

In this chapter, we give a few concluding remarks on the applications, and future extensions of this research.

9.1 Key Results

Several issues arise in flow and congestion control of unicast and multicast connections. In this dissertation, we have:

- Investigated the effect of ABR parameter values on network performance
- Proposed and studied several variations on ABR rate allocation schemes
- Proposed an architecture for connecting enterprise sites using ABR VPCs
- Developed an algorithm for rate allocation of multiplexed ABR VCCs on ABR VPCs
- Designed and compared feedback consolidation algorithms for ABR point-to-multipoint connections
- Developed fairness definitions for multipoint rate allocation
- Developed rate allocation algorithms for multipoint-to-point connections

Our results have indicated that the ABR service performs well in ATM backbones for connecting enterprise sites. ABR controls transient congestion, and efficiently utilizes network buffers and bandwidths. Weighted fair allocations with minimum rates can be designed for point-to-point as well as multipoint connections with multiple concurrent senders and receivers. With the proper implementation, ABR can have low cell loss and delay. Feedback consolidation can be designed to control noise and achieve a fast transient response.

9.2 Open Issues and Future Work

This dissertation has provided insight into the design of flow and congestion control techniques for unicast and multicast traffic. Using this understanding of the dynamics of traffic management and group communication, both for ATM networks and for the Internet, we can tackle many of the remaining open problems. We discuss a few of those problems next.

9.2.1 Signaling and Forwarding for ATM Multipoint Connections

As discussed in chapter 2, ATM does not currently support multipoint-to-multipoint connections directly. The MARS architecture [2] for IP over ATM uses the point-to-point and point-to-multipoint VCs supported by UNI 3.1 signaling to forward packets within a multicast group cluster and uses a multicast router to go outside a cluster. The protocol, however, incurs high state management overhead, leading to scalability problems [11, 116, 10, 114]. A true multipoint-to-multipoint service needs to be defined for ATM, solving the problems of (1) signaling and connection management, and (2) forwarding.

A special server should not be required to handle forwarding and to also avoid the scalability problems of VC meshes. A single VC (shared tree) can be used for a multicast group consisting of multiple senders and multiple receivers. This will improve scalability because the number of VC connections is independent of the number of endpoints. The duplication required is kept to a minimum.

Shared tree ideas can be borrowed from IP, making use of the techniques of UNI 4.0 signaling. End systems can be oblivious to group memberships. Member-initiated

joins of the multicast group should be allowed. A possible approach is to use core-based so that cores route signaling messages for the group. Cells can also be forwarded by switches on a spanning tree, to allow packets to follow the shortest path along the shared tree by emulating reverse path forwarding [52]. Solutions to this problem should make use of the experiences gained in solving this problem for the Internet.

Finally, further study is needed for the cell interleaving problem in forwarding/reassembling packets, discussed in chapter 2. As previously mentioned, the problem occurs when cells from different senders are merged and interleaved on the links of a multipoint connection (implemented as a shared tree), so the AAL5 at the receiver cannot assemble the data. VC merge seems among the most promising approaches to solve this problem. The pros and cons of this scheme need to be examined more carefully. Buffer requirements and delays incurred with different workloads should also be modeled.

9.2.2 Renegotiation and Heterogeneity

In the resource reservation protocol (RSVP) recently proposed for IP, different receivers in a multicast group can be heterogeneous. This means that receivers can specify different quality of service (QoS) requirements [8, 28, 5, 21, 40]. When a virtual connection includes receivers with different QoS requirements, the VC is referred to as a "variegated VC." In addition, receivers are allowed to dynamically change their QoS requirements throughout the connection lifetime (due to the periodic renewal of reservations, or "soft state"). Group membership also changes throughout connection lifetime.

ATM does not currently allow different destinations in a multicast group to have different QoS requirements, or different senders to specify different traffic characteristics. Furthermore, renegotiation is currently foreseen to be supported in ATM by tearing down the ATM connection and setting up a new one, which is clearly inefficient. We believe that ATM must adapt to the dynamic and varying needs of receivers by directly supporting their requirements [25, 76, 125].

Novel techniques are required for the following: (1) allowing signaling for variegated VCs, (2) renegotiating parameters during the connection lifetime, (3) defining a service where not all receivers experience the same QoS, but experience precisely the QoS they specify, and (4) defining a service where not all receivers experience the same QoS, not because of their requirements, but because they need a best-effort service where lost data need not be retransmitted.

Problem 1: When different receivers have different QoS requirements, the signaling scheme needs to allow receiver-initiated QoS requests to scale well to a large number of receivers. One way to facilitate renegotiation is to separate connection establishment from QoS negotiation, as in the Internet. A light-weight signaling protocol can use intelligent encoding of the most important elements in the signaling cell to avoid segmentation and reassembly of signaling messages [56, 20]. The QoS negotiation can later be done on the same channel as the data (in-band) to speed up parallel requests. This allows receivers to send their possibly varying QoS requests.

Problem 2: The signaling rules of ATM are proposed to be modified to allow renegotiation of parameters. Connection admission control procedures must be streamlined to enable this to be accomplished with minimum overhead. One way to do this is to allow propagation of resource reservation requests during the connection lifetime

and perform a subset of the connection admission control functions (light-weight signaling) at that point. If connection establishment is separated from QoS negotiation, as explained above, this makes the QoS negotiation and renegotiation functions uniform. A timer or counter mechanism must be used to ensure that changes do not occur at a high rate, but finding an appropriate value for this timer is not straightforward.

Problem 3: In the most straightforward approach, the most stringent requirements are reserved. As with RSVP, requests are propagated only until they merge with a more stringent request. To allow actual heterogeneity on some branches, we can make use of (A) hierarchical encoding, and (B) intelligent drop policies [50] to dynamically provide different receivers with different QoS. With video traffic, using techniques such as interlacing (used in GIF), progressive and hierarchical encoding (used in JPEG and MPEG), and intelligent scheduling and drop policies, can produce data at different rates to different receivers in the same multicast group. For example, the receiver that can only receive at the slowest rate can receive only the highest priority traffic, while the receiver that can receive at the highest rate can receive all levels of the encoded traffic. This idea is not limited to voice and video traffic: some data applications can tolerate loss, such as stock market price updates, and can make use of variegated VCs, where some receivers may get more frequent updates than others. The encoder needs to be given information on the levels to use, and merging of requests has to be done. Preliminary studies include [95, 133].

Problem 4: For flow-controlled best effort traffic, we have observed many situations when giving the minimum allocation specified by all the branches might lead to underutilization of links. This occurs when bottlenecks exist on other branches of the tree. This is the disadvantage of taking the minimum of the rates indicated by all the

leaves of the multicast tree through flow/congestion control feedback mechanisms. If data applications can tolerate loss, better link utilizations can be achieved by providing appropriate quality of service to different receivers. One technique that has been mentioned above is using hierarchical encoding. On more bottlenecked branches, the less significant information can be dropped, using intelligent buffer allocation techniques that distinguish among flows. This ensures a higher link utilization for non-bottlenecked links.

Various techniques for allowing different receivers to experience different quality of service include: (A) destination set grouping [18, 4]: receivers desiring (approximately) the same QoS are grouped together in the same connection, (B) definition of an index dependent on the cell loss tolerated by each receiver, and allowing the source to send at a rate that is a function of the index [77], (C) implementing virtual source/virtual destination (VS/VD) at every fork [47].

9.2.3 ABR End System Parameters for Multipoint Connections

Setting ABR source parameters for multipoint connections has been briefly examined in [107, 7, 58], but the results were not incorporated in the ATM traffic management specifications [43, 68] because more studies need to be conducted to determine the best method and guidelines to compute their values. The main factor that complicates the setting of these parameters for multipoint connections is the possible existence of widely varying round trip times from the source to the different receivers, and the possible existence of a bottleneck on a distant branch. Thus a more conservative approach in the setting of the parameters might be preferred. This may, however, adversely affect the efficiency experienced by the connections.

The following ABR end system parameters are especially critical: the transient buffer exposure (TBE), initial cell rate (ICR), count of missing resource management cells (CRM), and rate increase factor (RIF). Formulae are needed to compute the values of these parameters, to prevent transient overloads, while ensuring high link utilization. The proposed formulae should be tested in problem situations, such as long delay links, rate retaining sources, and sudden overload and underload situations. The source end system rules 5, 6, 8 and 9 [43] for multipoint connections need to be studied.

Multipoint connections may suffer from initial overallocation until feedback is received from all the distant leaves. This problem can be overcome by correctly setting the transient buffer exposure (TBE) parameter, and the correct calculation of the initial cell rate (ICR) parameter. The following formula was proposed to calculate the optimal value of ICR for multipoint connections [107]:

$$ICR = min(ICR \text{ allowed by the network}, \frac{TBE}{\text{maximum } FRTT} - \frac{RIF \times TBE}{2})$$

Note that the value of ICR is a function of the TBE value, the longest round trip time (FRTT) value, and the rate increase factor (RIF). But if the calculation of ICR depends on the round trip time, a problem arises: should the ICR change when nodes join or leave the group to account for the longest RTT for all destinations? The RTT of the farthest leaf reduces the ICR value according to the proposed formula. What happens when that farthest leaf leaves the multicast group? The root should not need to be notified every time a leaf joins or leaves the group, otherwise the signaling rules would imply that ABR multipoint communication will not scale [58]. An alternative formula for computing ICR can make use of information on buffer sizes and using probabilistic estimates.

A related issue is the setting of the ABR rate increase factor (RIF) parameter, which controls the amount of source rate increase during underload. Many studies [107] argue that transient queues can be mitigated by setting RIF to a small value (1/FRTT is recommended, but again this depends on the round trip time). Small RIF values may be useful in cases of distant bottlenecks that are multiple branch points away. Such small values have the adverse effect of slowing the rise to the optimal rates, resulting in underutilization. This tradeoff needs further investigation.

9.2.4 Reliable Multicast for Internet Terrestrial and Satellite Networks

Developing a reliable transport protocol for multicast connections has been an active research area in the past few years. Reliable multicast transport protocols aim at providing congestion and flow control in a scalable and secure manner. The Internet Research Task Force (IRTF) has been working on these issues and an IETF working group has recently been formed to formalize this work. The toughest problems in devising a reliable transport protocol for multicast connections include:

- The implosion problem for positive acknowledgments (ACKs) (or negative acknowledgments, NAKs, if used). The number of acknowledgments the sender receives should not be proportional to the number of receivers. This is very similar to the feedback consolidation problem discussed in chapter 7.

- Computing the correct timeout values given that different receivers have widely varying round trip times.

- Multicast group and address maintenance.

- Routing support.

- Fair resource allocation among contending flows.
- Security.

One of the problems we have started investigating is the design of a reliable multicast protocol for the Internet, with special emphasis on satellite networks. The proper method of reliable multicasting is dependent upon the class type and distribution requirements of a given application [94]. The underlying network technology and topological nature of a network are also critical factors in determining the proper reliable multicast approach.

Reliable multipoint transport protocols can be sender-initiated or receiver-initiated. In sender-initiated protocols, positive acknowledgments (ACKs) are sent by the receivers, and the sender must bear much of the complexity associated with reliable data transfer, such as maintaining state information for each of the receivers. This approach is the same as what is used in point-to-point transport. Receiver-initiated protocols, on the other hand, shift this complexity to the receivers by using negative acknowledgments (NAKs) [123]. This usually scales better. Some approaches combine both ACKs and NAKs. Most of the protocols attempt to satisfy atomicity (all or none of the receivers receive the message) and termination (the result of the message multicast can be determined in finite time) [25].

Several techniques for reliable multicast transport have been proposed by several institutions around the world [42, 53, 93, 51, 86, 23, 99]. Techniques for reliable multicast have to be adapted to satellite networks, due to the high error rate, high latency and bandwidth asymmetry in satellite networks. Forward error correction (FEC) techniques are particularly suited to satellite networks to minimize the reverse NAK traffic, and the retransmission. In [138], the authors propose an extension to a

reliable multicast protocol that minimizes the traffic from the receivers to the sender (NAK traffic) by providing a tuning capability to selectively request retransmissions. This is especially important when bandwidth from the Earth station to the satellite is limited, as is typically the case. Other techniques, such as using different receiver groups [4, 12] and using layering [133] are also useful for satellite multicast communication. More research needs to be done to investigate which of these techniques works best with the satellite link characteristics outlined above.

9.2.5 Internet Differentiated Services and Explicit Congestion Notification, and their Extension for Multicast Sessions

As discussed in chapter 6, Internet differentiated services provide an alternative to per-flow processing and per-flow state in network backbones [6]. Differentiated services in large core networks can interface with integrated services with the resource reservation protocol (RSVP) [8] in peripheral stub networks, to provide end-to-end QoS. The specification of the expedited and assured forwarding per-hop behavior is currently being finalized. These behaviors provide an inexpensive service differentiation mechanism through tagging of packets. The specification and performance of the services themselves, and the extension of such a framework to handle multicast connections is an area for future research.

Another traffic management mechanism proposed for the Internet with TCP is the explicit congestion notification (ECN) mechanism [101]. As explained in section 2.4.1, TCP currently uses packet loss as an indication of congestion. When congestion is detected, the TCP slow start mechanism is triggered. With ECN, the end systems do not have to wait until packets are lost to detect congestion and reduce their rates;

rather they examine the ECN bit in the packet headers. The ECN bit is set when the buffer occupancy as high, as defined by the buffer management mechanism [41]. The condition for setting the ECN bit, as well as the end system response to ECN bits still needs further study. In addition, the operation of the ECN mechanism with multicast connections is not clear at this point, and is left as a topic for future research.

BIBLIOGRAPHY

[1] Y. Afek, Y. Mansour, and Z. Ostfeld. Phantom: A simple and effective flow control scheme. In *Proceedings of the ACM SIGCOMM*, August 1996.

[2] G. Armitage. Support for multicast over UNI 3.0/3.1 based ATM networks. Request for Comments 2022, November 1996.

[3] A. Arulambalam, X. Chen, and N. Ansari. Allocating fair rates for available bit rate service in ATM networks. *IEEE Communications Magazine*, 34(11), November 1996.

[4] S. Bhattacharyya, J. Kurose, D. Towsley, and R. Nagarajan. Efficient rate-controlled bulk data transfer using multiple multicast groups. In *Proceedings of IEEE INFOCOM*, April 1998.

[5] A. Birman, V. Firoiu, R. Guerin, and D. Kandlur. Provisioning of RSVP-based services over a large ATM network. Technical Report RC 20250 (10/27/95) Computer Science, IBM research division, T. J. Watson Research Center, Yorktown Heights, New York, October 1995.

[6] S. Blake, D. Black, M. Carlson, E. Davies, Z. Wang, and W. Weiss. An architecture for differentiated services. RFC 2475, December 1998.

[7] F. Bonomi and K. W. Fendick. Rate-based flow control framework for the available bit rate ATM service. *IEEE Network Magazine*, 9(2):25–39, March/April 1995.

[8] R. Braden, L. Zhang, S. Berson, S. Herzog, and S. Jamin. Resource ReSerVation Protocol (RSVP). RFC 2205, September 1997. http://info.internet.isi.edu:80/in-notes/rfc/files/rfc2205.txt.

[9] R. Braudes and S. Zabele. Requirements for multicast protocols. Request for Comments 1458, May 1993.

[10] T. Braun, S. Gumbrich, and H. Stuttgen. Comparison of concepts for IP multicast over ATM. In *Proceedings of IEEE ATM Workshop, San Francisco*, August 1996.

[11] M. Buddhikot and C. Papadopoulos. Washington university workshop on integration of IP and ATM: Overview of session 3: Multicast support in ATM+IP networks. http://www.arl.wustl.edu/arl/workshops/atmip/proceedings.html, 1996.

[12] J. Byers, M. Luby, M. Mitzenmacher, and A. Rege. A digital fountain approach to reliable distribution of bulk data. In *Proceedings of the ACM SIGCOMM*, pages 56–67, September 1998.

[13] D. Cavendish and M. Gerla. Rate based congestion control for many-to-many multicast ABR traffic. In *Proceedings of SPIE '98 Conference on Performance and Control of network systems II*, volume 3530, pages 122–130, November 1998.

[14] D. Cavendish, S. Mascolo, and M. Gerla. Rate based congestion control for multicast ABR traffic. In *Proceedings of IEEE GLOBECOM*, volume 2, pages 1114–1118, November 1996.

[15] A. Charny. An algorithm for rate allocation in a cell-switching network with feedback. Master's thesis, Massachusetts Institute of Technology, May 1994.

[16] A. Charny, D. Clark, and R. Jain. Congestion Control with Explicit Rate Indication. In *Proceedings of ICC'95*, page 10 pp, June 1995.

[17] T. M. Chen, S. S. Liu, and V. K. Samalam. The available bit rate service for data in ATM networks. *IEEE Communications Magazine*, 34(5):12, May 1996.

[18] S. Y. Cheung, M. H. Ammar, and X. Li. On the use of destination set grouping to improve fairness in multicast video distribution. In *Proceedings of IEEE INFOCOM*, volume 2, pages 553–560, April 1996.

[19] Y-Z Cho and M-Y Lee. An efficient rate-based algorithm for point-to-multipoint ABR service. In *Proceedings of IEEE GLOBECOM*, volume 2, pages 790–795, November 1997.

[20] I. Cidon, A. Gupta, T. Hsiao, A. Khamisy, A. Parekh, R. Rom, and M. Sidi. OPENET: An open and efficient control platform for ATM networks. In *Proceedings of IEEE INFOCOM*, volume 2, April 1998.

[21] E. Crawley, L. Berger, S. Berson, F. Baker, M. Borden, and J. Krawczyk. A framework for integrated services and RSVP over ATM. Work in progress, draft-ietf-issll-atm-framework-03.txt, April 1998.

[22] S. Deering. Host extensions for IP multicasting. Request for Comments 1112, August 1989.

[23] D. DeLucia and K. Obraczka. Multicast feedback supression using representatives. In *Proceedings of IEEE INFOCOM '97*, April 1997.

[24] D. DeLucia and K. Obraczka. A multicast congestion control mechanism for reliable multicast. In *Proceedings of IEEE ISCC*, 1998.

[25] C. Diot, W. Dabbous, and J. Crowcroft. Multipoint communication: A survey of protocols, functions and mechanisms. *IEEE Journal on Selected Areas in Communications*, 15(3):277–290, April 1997.

[26] S. Fahmy et al. Feedback consolidation algorithms for ABR point-to-multipoint connections. ATM Forum/97-0615, July 1997. http://www.cis.ohio-state.edu/~jain/amtf/a97-0615.htm.

[27] S. Fahmy et al. Performance analysis of ABR point-to-multipoint connections for bursty and non-bursty traffic with and without VBR background. ATM Forum/97-0422, April 1997. http://www.cis.ohio-state.edu/~jain/amtf/a97-0422.htm.

[28] S. Fahmy and R. Jain. *Handbook of Communications Technologies: The Next Decade*, chapter "Resource ReSerVation Protocol (RSVP)". CRC Press, July 1999.

[29] S. Fahmy, R. Jain, R. Goyal, and B. Vandalore. A switch algorithm for abr multipoint-to-point connections. ATM Forum/97-1085R1, December 1997. http://www.cis.ohio-state.edu/~jain/atmf/a97-1085.htm.

[30] S. Fahmy, R. Jain, R. Goyal, and B. Vandalore. Fairness for ABR multipoint-to-point connections. In *Proceedings of SPIE '98 Conference on Performance and Control of network systems II*, volume 3530, pages 131–142, November 1998. http://www.cis.ohio-state.edu/~jain/papers/spie98.htm.

[31] S. Fahmy, R. Jain, R. Goyal, B. Vandalore, and S. Kalyanaraman. Design and evaluation of feedback consolidation for ABR point-to-multipoint connections in ATM networks. *Computer Communications*, 22(12):1085–1103, July 1999. http://www.cis.ohio-state.edu/~jain/papers/pt2mpt.htm.

[32] S. Fahmy, R. Jain, R. Goyal, B. Vandalore, S. Kalyanaraman, S. Kota, and P. Samudra. Feedback consolidation algorithms for ABR point-to-multipoint connections. In *Proceedings of IEEE INFOCOM*, volume 3, pages 1004–1013, March 1998. http://www.cis.ohio-state.edu/~jain/papers/cnsldt.htm.

[33] S. Fahmy, R. Jain, R. Goyal, B. Vandalore, S. Kota, and P. Samudra. Fairness for ABR multipoint-to-point connections. ATM Forum/97-0832, September 1997. http://www.cis.ohio-state.edu/~jain/atmf/a97-0832.htm.

[34] 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 the First International Workshop on Satellite-based Information Systems*, pages 108–115, Rye, New York, November 1996. http://www.cis.ohio-state.edu/~jain/papers/wosbis.htm.

[35] S. Fahmy, R. Jain, S. Kalyanaraman, R. Goyal, and B. Vandalore. On determining the fair bandwidth share for ABR connections in ATM networks. In *Proceedings of the IEEE International Conference on Communications (ICC) '98*, volume 3, pages 1485–1491, June 1998. http://www.cis.ohio-state.edu/~jain/papers/neff.htm.

[36] S. Fahmy, R. Jain, S. Rabie, R. Goyal, and B. Vandalore. Quality of service for internet traffic over ATM service categories. *Computer Communications, special issue on enterprise networks*, 1999. http://www.cis.ohio-state.edu/~jain/papers/enterprs.htm.

[37] K. Fendick, S. Mithal, and K. K. Ramakrishnan. Clarifications to the specifications of existing ATM service classes. ATM Forum/96-1628, 1996.

[38] K. W. Fendick. Evolution of controls for the available bit rate service. *IEEE Communications Magazine*, 34(11):35–39, November 1996.

[39] W. Fenner. Internet group management protocol, version 2. IETF Internet Draft: draft-ietf-idmr-igmp-v2-06.txt, January 1997.

[40] V. Firoiu and D. Towsley. Call admission and resource reservation for multicast sessions. In *Proceedings of the IEEE INFOCOM*, volume 1, pages 94–101, March 1996.

[41] S. Floyd and V. Jacobson. Random early detection gateways for congestion avoidance. *IEEE/ACM Transactions on Networking*, 1(4):397–413, August 1993.

[42] S. Floyd, V. Jacobson, S. McCanne, C-G Liu, and L. Zhang. A reliable multicast framework for light-weight sessions and application level framing. In *Proceedings of ACM SIGCOMM '95*, pages 342–356, October 1995.

[43] The 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.

[44] M. W. Garrett. Service architecture for ATM: from applications to scheduling. *IEEE Network*, 10(3):6–14, May/June 1996.

[45] E. Gauthier, J.-Y. Le Boudec, and P. Oechslin. SMART: A many-to-many multicast protocol for ATM. *IEEE Journal on Selected areas in Communications*, 15(3):458–472, April 1997.

[46] N. Golmie, Y. Chang, and D. Su. NIST ER switch mechanism (an example). ATM Forum/95-0695, June 1995.

[47] R. Goyal, X. Cai, R. Jain, S. Fahmy, and B. Vandalore. Per-vc rate allocation techniques for ATM-ABR virtual source virtual destination networks. GLOBECOM '98, November 1998. http://www.cis.ohio-state.edu/~jain/papers/globecom98.htm.

[48] R. Goyal, R. Jain, S. Fahmy, and B. Vandalore. Providing rate guarantees to TCP over the ATM GFR service. In *Proceedings of LCN'98*, October 1998. http://www.cis.ohio-state.edu/~jain/papers/lcn98.htm.

[49] R. Goyal, R. Jain, S. Kalyanaraman, S. Fahmy, and B. Vandalore. Improving the performance of TCP over the ATM-UBR service. *Computer Communications*, 21(10):898–911, July 1998. http://www.cis.ohio-state.edu/~jain/papers/cc.htm.

[50] R. Goyal, R. Jain, S. Kalyanaraman, S. Fahmy, and B. Vandalore. Improving the performance of TCP over the ATM-UBR service. *Computer Communications*, 1998. http://www.cis.ohio-state.edu/~jain/papers/cc.htm.

[51] M. Grossglausser. Optimal deterministic timeouts for reliable scalable multicast. In *Proceedings of the IEEE INFOCOM*, volume 3, pages 1425–1432, March 1996.

[52] M. Grossglausser and K. K. Ramakrishnan. SEAM: Scalable and efficient ATM multipoint-to-multipoint multicasting. In *Proceedings of the IEEE INFOCOM*, April 1997.

[53] A. Gupta, W. Howe, M. Moran, and Q. Nguyen. Resource sharing for multi-party real-time communication. In *Proceedings of the IEEE INFOCOM*, volume 2, pages 1230–1237, March 1996.

[54] G. Hasegawa, H. Ohsaki, M. Murata, and H. Miyahara. Performance evalulation and parameter tuning of TCP over ABR service in ATM networks. *IEICE transactions on communications*, E79-B(5):668–683, May 1996.

[55] F. Hellstrand and A. Veres. Simulation of TCP/IP router traffic over ATM using GFR and VBR.3. ATM Forum/98-0087, February 1998.

[56] G. Hjalmtysson and K. K. Ramakrishnan. UNITE - An architecture for lightweight signaling in ATM networks. In *Proceedings of IEEE INFOCOM*, volume 2, April 1998.

[57] C. Huitema. *Routing in the Internet.* Prentice Hall, Inc., April 1995.

[58] D. Hunt. Open issues for ABR point-to-multipoint connections. ATM Forum/95-1034, August 1995.

[59] V. Jacobson. Congestion avoidance and control. In *Proceedings of the ACM SIGCOMM*, pages 314–332, August 1988.

[60] J. M. Jaffe. Bottleneck flow control. *IEEE Transactions on Communications*, COM-29(7):954–962, July 1981.

[61] R. Jain. A timeout-based congestion control scheme for window flow-controlled networks. *IEEE Journal on Selected Areas in Communications*, SAC-4(7):1162–1167, October 1986.

[62] R. Jain. A delay-based approach for congestion avoidance in interconnected heterogeneous computer networks. *Computer Communications Review*, 19(5):56–71, October 1989.

[63] R. Jain. Congestion control in computer networks: Issues and trends. *IEEE network magazine*, pages 24–30, May 1990.

[64] R. Jain. Myths about congestion management in high-speed networks. *Internetworking: Research and experience*, 3:101–113, 1992.

[65] R. Jain. Congestion control and traffic management in ATM networks: Recent advances and a survey. *Computer Networks and ISDN Systems*, 28(13):1723–1738, November 1996. http://www.cis.ohio-state.edu/~jain/papers/cnis.htm.

[66] R. Jain, S. Fahmy, S. Kalyanaraman, and R. Goyal. ABR switch algorithm testing: A case study with ERICA. ATM Forum/96-1267, October 1996. http://www.cis.ohio-state.edu/~jain/amtf/a96-1267.htm.

[67] R. Jain, S. Fahmy, S. Kalyanaraman, R. Goyal, F. Lu, and S. Srinidhi. More strawvote comments: TBE vs queue sizes. ATM Forum/95-1661, December 1995. http://www.cis.ohio-state.edu/~jain/atmf/a95-1661.htm.

[68] R. Jain, S. Kalyanaraman, S. Fahmy, R. Goyal, and S. Kim. Source behavior for ATM ABR traffic management: An explanation. *IEEE Communications Magazine*, 34(11):50–57, November 1996. http://www.cis.ohio-state.edu/~jain/papers/src_rule.htm.

[69] R. Jain, S. Kalyanaraman, S. Fahmy, R. Goyal, and B. Vandalore. The ERICA Switch Algorithm for ABR Traffic Management in ATM Networks. Submitted to IEEE/ACM Transactions on Networking, May 1999. http://www.cis.ohio-state.edu/~jain/papers/erica.htm.

[70] R. Jain, S. Kalyanaraman, S. Fahmy, and F. Lu. Out-of-rate RM cell issues and effect of Trm, TOF, and TCR. ATM Forum/95-0973R1, August 1995. http://www.cis.ohio-state.edu/~jain/atmf/a95-0973.htm.

[71] R. Jain, S. Kalyanaraman, S. Fahmy, and F. Lu. Straw-vote comments on TM 4.0 R8. ATM Forum/95-1343, October 1995. http://www.cis.ohio-state.edu/~jain/atmf/a95-1343.htm.

[72] 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. http://www.cis.ohio-state.edu/~jain/atmf/a95-1660.htm.

[73] R. Jain, S. Kalyanaraman, R. Goyal, S. Fahmy, and Ram Viswanathan. ERICA switch algorithm: A complete description. ATM Forum/96-1172, August 1996. http://www.cis.ohio-state.edu/~jain/amtf/a96-1172.htm.

[74] R. Jain, S. Kalyanaraman, and R. Viswanathan. The OSU Scheme for Congestion Avoidance in ATM Networks: Lessons Learnt and Extensions. Performance Evaluation Journal, 1997.

[75] R. Jain, K. K. Ramakrishnan, and D. M. Chiu. Congestion avoidance in computer networks with a connectionless network layer. Digital Equipment Corporation, Technical Report, DEC-TR-506, August 1987, 17 pp. Also in C. Partridge, Ed., Innovations in Internetworking, Artech House, Norwood, MA, 1988, pp. 140-156.

[76] M. Jeffrey. Scope, concepts and issues for the new multiway BOF. ATM Forum/96-0628, June 1996.

[77] T. Jiang, M. Ammar, and E. Zegura. Inter-receiver fairness: a novel performance measure for multicast ABR sessions. In *Proceedings of the ACM SIGMETRICS*, pages 202–211, June 1998.

[78] T. Jiang, E. W. Zegura, and M. Ammar. Improved consolidation algorithms for point-to-multipoint ABR service. In *Proceedings of IEEE Workshop on ATM*, pages 195–201, May 1998.

[79] L. Kalampoukas, A. Varma, and K. K. Ramakrishnan. An efficient rate allocation algorithm for ATM networks providing max-min fairness. In *Proceedings of the 6th IFIP International Conference on High Performance Networking*, September 1995.

[80] S. Kalyanaraman. *Traffic Management for the Available Bit Rate (ABR) Service in Asynchronous Transfer Mode (ATM) Networks.* PhD thesis, The Ohio State University, August 1997.

[81] S. Kalyanaraman, R. Jain, S. Fahmy, R. Goyal, and Seong-Cheol Kim. Performance and buffering requirements of internet protocols over ATM ABR and UBR services. *IEEE Communications Magazine*, 36(6):152–157, June 1998. http://www.cis.ohio-state.edu/~jain/papers/ieee-mag.htm.

[82] S. Kalyanaraman, R. Jain, R. Goyal, S. Fahmy, and S-C Kim. Use-it or lose-it policies for the available bit rate (ABR) service in ATM networks. *Journal of Computer Networks and ISDN Systems*, 30(24):2293–2308, December 1998. http://www.cis.ohio-state.edu/~jain/papers/uili.htm.

[83] J. B. Kenney. On forwarding EFCI beyond an ABR VP destination. ATM Forum/97-0386, April 1997.

[84] J. B. Kenney. Solution for problem of resetting EFCI at ABR VP source. ATM Forum/97-0184, February 1997.

[85] J. B. Kenney. Traffic management draft specifications. ATM Forum/TM 4.1, July 1999.

[86] A. Koifman and S. Zabele. RAMP: A reliable adaptive multicast protocol. In *Proceedings of the IEEE INFOCOM*, volume 3, pages 1442–1451, March 1996.

[87] A. Koike, H. Kitazume, H. Saito, and M. Ishizuka. On end system behavior for explicit forward congestion indication of ABR service and its performance. *IEICE transactions on communications*, E79-B(4):605–610, April 1996.

[88] S. Komandur and D. Mosse. SPAM: A data forwarding model for multipoint-to-multipoint connection support in ATM networks. In *Proceedings of the sixth international conference on computer communications and networks (ICCCN '97)*, pages 383–388, September 1997.

[89] R. Krishnan. Rate based control schemes for ABR traffic - design principles and performance comparison. In *Proceedings of the IEEE global telecommunications conference (GLOBECOM)*, volume 2, pages 1231–1235, November 1996.

[90] D. Lee, K. K. Ramakrishnan, W. M. Moh, and A. U. Shankar. Protocol specification using parameterized communicating extended finite state machines- a case study of the ATM ABR rate control scheme. In *Proceedings of ICNP*, pages 208–217, October 1996.

[91] D. Lee, K. K. Ramakrishnan, W. M. Moh, and A. U. Shankar. Performance and correctness of the ATM ABR rate control scheme. In *Proceedings of the IEEE INFOCOM*, volume 2, pages 785–794, March 1997.

[92] X. Li, S. Paul, and M. Ammar. Multi-session rate control for layered video multicast. In *Proceedings of SPIE '99 Conference on Multimedia Computing and Networking*, San Jose, CA, 1999.

[93] J. C. Lin and S. Paul. RMTP: A reliable multicast transport protocol. In *Proceedings of the IEEE INFOCOM*, volume 3, pages 1414–1424, March 1996.

[94] A. Mankin, A. Romanow, S. Bradner, and V. Paxson. IETF criteria for evaluating reliable multicast transport and application protocols. RFC 2357, June 1998.

[95] S. McCanne, V. Jacobson, and M. Vetterli. Receiver-driven layered multicast. In *Proceedings of ACM SIGCOMM '96*, pages 117–130, August 1996.

[96] W. M. Moh. On multicasting ABR protocols for wireless ATM networks. In *Proceedings of the International Conference on Network Protocols (ICNP) '97, Atlanta*, pages 24–31, 1997.

[97] W. M. Moh and Y. Chen. Design and evaluation of multipoint-to-point multicast flow control. In *Proceedings of SPIE '98 Conference on Performance and Control of network systems II*, volume 3530, pages 143–154, November 1998.

[98] W. M. Moh, Y. Chen, and B. Niizawa. Branch-point algorithms for multicasting ATM ABR protocols. In *Proceedings of IEEE Workshop on ATM*, May 1998.

[99] Y. Ofek and B. Yener. Reliable concurrent multicast from bursty sources. In *Proceedings of the IEEE INFOCOM*, volume 3, pages 1433–1441, March 1996.

[100] H. Ohsaki, M. Murata, and H. Miyahara. Robustness of rate-based congestion control algorithm for ABR service class in ATM networks. In *Proceedings of the IEEE global telecommunications conference (GLOBECOM)*, volume 2, pages 1097–1101, November 1996.

[101] K. Ramakrishnan and S. Floyd. A proposal to add explicit congestion notification (ECN) to IP. RFC 2481, January 1999.

[102] 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. http://www.cis.ohio-state.edu/~jain/atmf/a95-1599.htm.

[103] W. Ren, K-Y Siu, and H. Suzuki. On the performance of congestion control algorithms for multicast ABR service in ATM. In *Proceedings of IEEE ATM Workshop, San Francisco*, August 1996.

[104] W. Ren, K-Y Siu, and H. Suzuki. Multipoint-to-point ABR service in ATM networks. In *Proceedings of the International Conference on Communications, ICC, Montreal*, June 1997.

[105] W. Ren, K-Y Siu, H. Suzuki, and M. Shinohara. Multipoint-to-multipoint ABR service in ATM. *Computer Networks and ISDN Systems*, 30(19):1793–1810, 1998.

[106] L. Roberts. Rate based algorithm for point to multipoint ABR service. ATM Forum/94-0772R1, November 1994.

[107] L. Roberts. Point-to-multipoint ABR operation. ATM Forum/95-0834, August 1995.

[108] A. Romanow and S. Floyd. Dynamics of TCP traffic over ATM networks. *IEEE Journal on selected areas in communications*, 13(4):633–641, May 1995.

[109] S. Rosenberg, M. Aissaoui, K. Galway, and N. Giroux. Functionality at the edge: Designing scalable multiservice ATM networks. *IEEE Communications Magazine*, 36(5):88–99, May 1998.

[110] H. Saito, K. Kawashima, H. Kitazume, A. Koike, M. Ishizuka, and A. Abe. Performance issues in public ABR service. *IEEE Communications Magazine*, 34(11):40–48, November 1996.

[111] P. Samudra. UNI signaling 4.0. ATM Forum/95-1434R14, July 1996.

[112] K. Siu and T. Tzeng. Intelligent congestion control for ABR service in ATM networks. *Computer Communication Review*, 24(5):81–106, October 1995.

[113] K-Y Siu and H-Y Tzeng. Congestion control for multicast service in ATM networks. In *Proceedings of the IEEE GLOBECOM*, volume 1, pages 310–314, 1995.

[114] M. Smirnov. Scalable and efficient multiprotocol IP multicast over ATM. In *Proceedings of the SPIE Symposium*, November 1997.

[115] E. M. Spiegel. Multipoint connection support in PNNI 2.0. *ATM Forum Perspectives, IEEE Network*, Nov/Dec 1997.

[116] R. Talpade, G. Armitage, and M. Ammar. Experience with architectures for supporting IP multicast over ATM. In *Proceedings of IEEE ATM Workshop, San Francisco*, August 1996.

[117] K. Teraslinna. Comment on straw vote for traffic management specification V4.0. ATM Forum/95-1519R1, December 1995.

[118] K. Teraslinna. New work proposal: Traffic aggregation services. ATM Forum/96-0168R2, April 1996.

[119] K. Teraslinna. Proposal for informative appendix on VPC compliance. ATM Forum/97-0714, September 1997.

[120] K. Teraslinna. Study proposal for VC to VP traffic aggregation. ATM Forum/97-0624, June 1997.

[121] K. Teraslinna. Proposal to move VCC to VPC multiplexing and conformance to TM work plan. ATM Forum/98-0131, February 1998.

[122] K. Teraslinna. Proposed appendix on VCC to VPC multiplexing and conformance. ATM Forum/98-0132, February 1998.

[123] D. Towsley, J. Kurose, and S. Pingali. A comparison of sender-initiated and receiver-initiated reliable multicast protocols. *IEEE Journal on Selected Areas in Communications*, 15(3):398–406, April 1997.

[124] D. Tsang and W. Wong. A new rate-based switch algorithm for ABR traffic to achieve max-min fairness with analytical approximation and delay adjustment. In *Proceedings of the IEEE INFOCOM '96*, pages 1174–1181, March 1996.

[125] J. Turner, Q. Bian, and K. Shiomoto. Dynamic flow switching: A new communication service for ATM networks. In *Proceedings of IEEE INFOCOM*, April 1998.

[126] J. S. Turner. Extending ATM networks for efficient reliable multicast. In *Proceedings of Workshop on Communication and Architectural Support for Network-based Parallel Computing*, February 1997.

[127] H-Y Tzeng and K-Y Siu. On max-min fair congestion control for multicast ABR service in ATM. *IEEE Journal on Selected Areas in Communications*, 15(3):545–556, April 1997.

[128] B. Vandalore, S. Fahmy, R. Jain, R. Goyal, and M. Goyal. A definition of general weighted fairness and its support in explicit rate switch algorithms. In *Proceedings of the Sixth International Conference on Network Protocols 1998 (ICNP '98)*, pages 22–30, October 1998. http://www.cis.ohio-state.edu/~jain/papers/icnp98_bv.htm.

[129] B. Vandalore, S. Fahmy, R. Jain, R. Goyal, and M. Goyal. QoS and multipoint support for multimedia applications over the ATM ABR service. *IEEE Communications Magazine*, pages 53–57, January 1999. http://www.cis.ohio-state.edu/~jain/papers/multabr.htm.

[130] B. Vandalore, R. Jain, R. Goyal, and S. Fahmy. Dynamic queue control functions for ATM ABR switch schemes: Design and analysis. *Journal of Computer Networks*, 1999. http://www.cis.ohio-state.edu/~jain/papers/cnis_qctrl.htm.

[131] R. Venkateswaran, C. S. Raghavendra, X. Chen, and V. Kumar. A scalable dynamic multicast routing algorithm in ATM networks. In *Proceedings of the International Conference on Communications, ICC'97, Montreal*, volume 3, June 1997.

[132] R. Venkateswaran, C. S. Raghavendra, X. Chen, and V. Kumar. Support for multiway communications in ATM networks. ATM Forum/97-0316, April 1997.

[133] L. Vicisano, J. Crowcroft, and L. Rizzo. TCP-like congestion control for layered multicast data transfer. In *Proceedings of IEEE INFOCOM '98*, volume 3, pages 996–1003, March 1998.

[134] H. Wang and M. Schwartz. Comparison of multicast flow control algorithms over combined wireless/wired networks. In *Proceedings of the 3rd International Workshop on Mobile Multimedia Communications, Princeton*, September 1996. URL: http://www.ctr.columbia.edu/~whycu/res.html.

[135] H. Wang and M. Schwartz. Performance analysis of multicast flow control algorithms over combined wired/wireless networks. In *Proceedings of IEEE INFOCOM, Japan*, volume 3, pages 1038–1045, April 1997.

[136] H. Wang and M. Schwartz. Achieving bounded fairness for multicast and TCP traffic in the Internet. In *Proceedings of ACM SIGCOMM '98*, September 1998.

[137] I. Widjaja and A. I. Elwalid. Performance issues in VC-merge capable switches for IP over ATM networks. In *Proceedings of the IEEE INFOCOM*, volume 1, pages 372–380, March 1998. Also ATM Forum/97-0675, July 1997.

[138] T. Wong, T. Henderson, S. Raman, A. Costello, and R. Katz. Policy-based tunable reliable multicast for periodic information dissemination. In *Proceedings of the third International Workshop on Satellite-based Information Services*, 1998.

[139] N. Yin. Analysis of a rate-based traffic management mechanism for ABR service. In *Proceedings of the IEEE global telecommunications conference (GLOBECOM)*, volume 2, pages 1076–1082, November 1995.

[140] N. Yin and M. G. Hluchyj. On closed-loop rate control for ATM cell relay networks. In *Proceedings of the IEEE INFOCOM '94*, pages 99–108, June 1994.

[141] T. Yokotani, T. Ichihashi, and M. Ishizaka. Congestion control of multicast connnections in ATM networks. In *Proceedings of the 20th conference on local computer networks*, pages 38–47. IEEE Computer Society TC on Computer communications, October 1995.

ACRONYMS

AAL	ATM Adaptation Layer
ABR	Available Bit Rate
ACK	Acknowledgment
ACR	Allowed Cell Rate
ADTF	ACR Decrease Time Factor
ATM	Asynchronous Transfer Mode
B-ISDN	Broadband Integrated Services Digital Network
BECN	Backward Explicit Congestion Notification
BRM	Backward Resource Management cell
CAC	Connection Admission Control
CBR	Constant Bit Rate
CBT	Core-Based Tree
CCR	Current Cell Rate
CDF	Cutoff Decrease Factor
CDV	Cell Delay Variation
CDVT	Cell Delay Variation Tolerance
CI	Congestion Indication
CLP	Cell Loss Priority
CLR	Cell Loss Ratio
CRM	Missing RM-cell Count
CWND	TCP Congestion Window
DES	Destination End System
DVMRP	Distance Vector Multicast Routing Protocol
ECN	Explicit Congestion Notification
EFCI	Explicit Forward Congestion Indication
EOM	End Of Message
EPD	Early Packet Discard
ER	Explicit Rate
ERICA+	Explicit Rate Indication for Congestion Avoidance+
FIFO	First In First Out
FRM	Forward Resource Management cell
FRTT	Fixed Round-Trip Time
FTP	File Transfer Protocol
GCRA	Generic Cell Rate Algorithm

GEO	Geosynchronous Earth Orbit
GFR	Guaranteed Frame Rate
ICMP	Internet Control Message Protocol
ICR	Initial Cell Rate
IETF	Internet Engineering Task Force
IGMP	Internet Group Management Protocol
ION	Internetworking Over Non-broadcast multiple access
IP	Internet Protocol
IRTF	Internet Research Task Force
ISP	Internet Service Provider
ISSLL	Integrated Services over Specific Link Layers
ITU	International Telecommunications Union
LAN	Local Area Network
LANE	Local Area Network Emulation
LEO	Low Earth Orbit
LIJ	Leaf-Initiated join
MBS	Maximum Burst Size
MARS	Multicast Address Resolution Server
maxCTD	Maximum Cell Transfer Delay
MCR	Minimum Cell Rate
MCS	Multicast Server
MFS	Maximum Frame Size
MOSPF	Multicast extensions to Open Shortest Path First
MPLS	Multiprotocol Label Switching
MPOA	Multiprotocol Over ATM
Mrm	Controls RM bandwidth allocation
NAK	Negative acknowledgment
NI	No Increase
Nrm	Number of cells between FRM cells
PCR	Peak Cell Rate
PIM	Protocol-Independent Multicast
PIM-DM	Protocol-Independent Multicast-Dense Mode
PIM-SM	Protocol-Independent Multicast-Sparse Mode
PNNI	Private Network to Network Interface
PPD	Partial Packet Discard
QoS	Quality of Service
RDF	Rate Decrease Factor
RED	Random Early Detection
RFC	Request For Comments
RIF	Rate Increase Factor
RIJ	Root-Initiated join
RM	Resource Management
RSVP	Reservation Protocol

RTT	Round-Trip Time
SCR	Sustained Cell Rate
SES	Source End System
SSTHRESH	Slow Start Threshold
TBE	Transient Buffer Exposure
TCP	Transmission Control Protocol
TCR	Tagged Cell Rate
Trm	Upper Bound on Inter-FRM Time
UBR	Unspecified Bit Rate
UDP	User Datagram Protocol
UNI	User to Network Interface
UPC	Usage Parameter Control
VBR-nrt	Non Real-Time Variable Bit Rate
VBR-rt	Real-Time Variable Bit Rate
VCC	Virtual Channel Connection
VCI	Virtual Channel Identifier
VPC	Virtual Path Connection
VPI	Virtual Path Identifier
VPN	Virtual Private Network
VS/VD	Virtual Source/Virtual Destination
WAN	Wide Area Network
WWW	World Wide Web