INFORMATION TO USERS

This manuscript has been reproduced from the microfilm master. UMI films the text directly from the original or copy submitted. Thus, some thesis and dissertation copies are in typewriter face, while others may be from any type of computer printer.

The quality of this reproduction is dependent upon the quality of the copy submitted. Broken or indistinct print, colored or poor quality illustrations and photographs, print bleedthrough, substandard margins, and improper alignment can adversely affect reproduction.

In the unlikely event that the author did not send UMI a complete manuscript and there are missing pages, these will be noted. Also, if unauthorized copyright material had to be removed, a note will indicate the deletion.

Oversize materials (e.g., maps, drawings, charts) are reproduced by sectioning the original, beginning at the upper left-hand corner and continuing from left to right in equal sections with small overlaps.

Photographs included in the original manuscript have been reproduced xerographically in this copy. Higher quality 6" x 9" black and white photographic prints are available for any photographs or illustrations appearing in this copy for an additional charge. Contact UMI directly to order.

ProQuest Information and Learning
300 North Zeeb Road, Ann Arbor, MI 48106-1346 USA
800-521-0600

Université d'Ottawa · University of Ottawa

Performance Evaluation of FASTRAC ABR Control Algorithm With VSVD Under Long Propagation Delay

by

Kevin N. Huynh

A thesis submitted to the
School of Graduate Studies and Research
in partial fulfillment of the requirement of the degree of

Master of Applied Science

School of Information Technology and Engineering
University of Ottawa
April, 2001

Abstract

There have been many control algorithms proposed in the literature for handling the flow control of the Available Bit Rate (ABR) class of service on ATM. But few have demonstrated sound theoretical development, compliance to TM4.0, and a practical design. This thesis attempts to formalize and present ABR flow control from fundamentals to illustrate the requirements for designing a better algorithm. It explains TM4.0 ABR control rules and highlights subtleties in the standards that are commonly missed in the literature; these factors result in non-compliance and render the proposed algorithms impractical. This thesis also attempts to present various practical switch designs and their capabilities for ABR support. It applies traditional control theory to ABR flow control. It demonstrates that understanding the combination of all these three aspects is key to designing ABR control algorithms that are theoretically sound and practical. A rate thresholding explicit rate (ER) based control algorithm, FASTRAC (FAST RAte Computation), is the focus of the simulation analysis. Often literature papers presents simulation results that are improperly generated and do not observe the performance factors that matter to ABR flow control. Well designed OPNET simulation models in typical ATM network configurations are used to test the theoretical development behind FASTRAC and evaluate its performance. Long propagation delay is a major consideration to the stability of a control algorithm. Yet many algorithms simply do not attempt to address it. The effect of long propagation delay on FASTRAC is elaborately studied in a network consisting of satellite and terrestrial segments. The framework for Virtual Source/Virtual Destination (VSVD) proposed by TM4.0 outlines a network topology that breaks down an ABR loop into multiple loops, allowing network segments with long propagation delay to be controlled in separate ABR loops from network segments with shorter propagation delay. The major advantage of VSVD is to localize the impact of long propagation delay to only nodes inside the network segment where long propagation delay exists. FASTRAC is extended to support VSVD. Performance evaluation is done to verify that FASTRAC with VSVD can protect the terrestrial network segment from the effect of long propagation delay in the satellite segment.

Acknowledgements

I would like to express my sincere thanks to Dr. Oliver W. W. Yang, my research supervisor, for providing me guidance during my research. I would like to express my appreciation to Chengyu Zhu for providing help in building the No-VSVD switch models. A special thank to Michel Ouelette of Nortelnetworks. I also like to credit Nortelnetworks for providing the theoretical model of FASTRAC©, and the basic Bottleneck Configuration model in OPNET. The helpful and friendly guidance from Michel has led to a fruitful collaboration between CCNR (Computer Communication Network Group) at University of Ottawa and Nortelnetworks.

Finally, I would like to express my deepest appreciation and thanks to my wife for her understanding and emotional support throughout my study.

© The FASTRAC algorithm has been patented by Nortelnetworks.

Table of Contents

List of Figures

List of Tables

Acronyms

ABR	: Available Bit Rate
ACR	: Allowed Cell Rate
ATM	: Asynchronous Transfer Mode
BECN	: Backwards Explicit Congestion Notification
BRM	: Backwards Resource Management
BW	: Bandwidth
CBR	: Constant Bit Rate
CCR	: Current Cell Rate
CI	: Congestion Indication
DIR	: Direction Bit
EFCI	: Explicit Forward Congestion Indication
ER	: Explicit Rate
EWMA	: Exponentially Weighted Moving Average
FASTRAC	: FAST RAte Computation
FIFO	: First In First Out
FRM	: Forward Resource Management
FRTT	: Fixed Round Trip Time
GEO	: Geostationary
GFR	: Guarantee Frame Rate
ICR	: Initial cell Rate
LEO	: Low Earth Orbit
MA	: Moving Average
MACR	: Mean Allowed Cell Rate
MCR	: Minimum Cell Rate
MRM	: Minimum number of Resource Management cells
NI	: No Increase
NRM	: Number of Resource Management cells
PCR	: Peak Cell Rate

RDF	: Rate Decrease Factor
RIF	: Rate Increase Factor
RM	: Resource Management
RTT	: Round Trip Time
TM4.0	: ATM Forum's Traffic Management Specification version 4.0
TRM	: Time between Resource Management cell
QoS	: Quality of Service
UBR	: Unspecified Bit Rate
VBR	: Variable Bit Rate
VC	: Virtual Connection
VD	: Virtual Destination
VS	: Virtual source
VSVD	: Virtual Source Virtual Destination

Notations

		First Apperance (page no.)
α	control gain	20
B	high queue threshold value for queue control function	22
C	link capacity (cells/second)	18
$C_{high}(n)$	non-ABR traffic (CBR or VBR) cell arrival rate (cells/second)	18
d	number of time units of feedback delay	19
d_b	delay from bottleneck switch to source	19
d_f	delay from source to bottleneck switch	19
$f()$	queue control function	23
$M(n)$	measured ABR arrival rate (cells/second)	18
$maxRTT$	maximum round trip time for all connections (seconds)	21
N	total number of connections	25
$q(n)$	ABR queue size at time n	18
$r(n)$	fair share at time n (cells/second)	18
T_h	low queue threshold for queue control function	22
T_{max}	maximum desired service rate (cells/second)	27
$T(n)$	target service rate at time n (cells/second)	19
t_m	algorithm sampling period (seconds)	18
$w_{eff}(n)$	effective number of connections active	25
w_T	filtered effective number of connections active	25

Chapter 1: Introduction

1.1 Background of ABR

The Internet is one of the greatest technological phenomenon of the century. It will be the medium on which people get in touch with each other, learn, do business, and entertain. This vast medium therefore carries all kind of information in the form of video, audio, and data. Primarily, it is mostly data that is bursty due to the interactive nature of the Internet. At the core of the Internet are usually high capacity nodes that usually use ATM (Asynchronous Transfer Mode) as the transport layer. The ATM standard [ATMF96] defined a number of connection classes. These classes are implemented in the ATM adaptation layer and provide realtime or non-realtime variable bit rates (VBR), constant bit rates (CBR), and unspecified bit rate (UBR). CBR and VBR are intended to serve video and audio streams. UBR provides a best-effort service where no amount of bandwidth is guaranteed, and any cells may be discarded. UBR is directed at delay-tolerant applications such as file transfer and e-mail. The available bit rate (ABR) class was later conceived in order to support bursty traffic with better network utilization. It avoids the problem of overbooking the link capacity, thus increasing cost effectiveness. It is intended specificly for data applications that can adapt to time-varying bandwidth and can tolerate unpredictable end-to-end cell delays typical of WEB access. As more people use the Internet more bandwidth is required. The problem of increasing bandwidth demand should be solved with elegant solutions such as ABR rather than always increasing the link capacity. ABR is usually considered to be more cost effective for the solving the bandwidth engineering problem for bursty traffic.

The ABR class makes use of excess bandwidth in the ATM network – bandwidth that has not been committed to other traffic classes. This definition implies the need for a traffic control scheme can track changes in excess bandwidth available, and, allocate that bandwidth across ABR connections. It also implies that the ABR traffic sources are capable of receiving feedback from the network and adjusting their output rates accordingly. The ATM Forum's TM 4.0 [ATMF96] specifies the traffic management rules for ABR source, switch and destination nodes. This definition includes a protocol

for exchanging connection congestion and rate information. This protocol provides the control mechanisms. TM 4.0 does not specify the control algorithm to be employed on each of the node types. The design of the control algorithm is left up to the design community. The control algorithm should aim to satisfy the objective of the ABR service which is to minimize cell loss, and to maximize throughput at the expense of delay, while maintaining a fair distribution of the excess bandwidth to ABR connections. These qualities are governed by the feedback loop design that includes congestion detection mechanism, feedback information, and rate decrease/increase algorithms.

1.2 Literature Review of ABR Control Algorithms

Control algorithms are sets of formulas that describe quantitatively the behaviour of the control scheme. A well designed control algorithm is built based on proper and practical switch design and strict compliance to TM4.0. Many control algorithm proposals have been presented in the literature. But few have been explained with practical designs that are simple, scalable, robust, and stable. And others lack theoretical development. Some examples of such algorithms can be found in [HeBe97] which are reviewed below. Simulations have often been performed that lack real network attributes such as multiple connections with a mixture of different service categories, long propagation delay, loss cells, and degree of source burstiness [Kris97][Goca98]. Of these, long propagation delay is the major network attribute contributing to queue engineering problems and stability. Yet, it is commonly not critically analyzed. Assumptions made in the simulation are rarely specified. Performance evaluation of algorithms is not done with a set of predefined metrics in order to make any conclusive remarks about their effectiveness and stability. The need for metrics is of utmost importance in order to quantify results and compare them commensurably. Compliance to TM4.0 is sometimes ignored completely in the control algorithm. Standards compliance is crucial in traffic management because it ensures proper QoS and network stability. One non-compliant node in a network can cause the whole network to perform erratically [HuYa99]. It's also important to note assumptions made by the algorithms. Below is a survey of control algorithms [HeBe97] classified into the three characterization categories of feedback

control scheme discussed in section 2.3. The equations for these algorithms are included in Appendix A for convenience. This survey briefly highlights some of the weaknesses from a theoretical observation in algorithms presented in the literature.

1.2.1 Queue Size Thresholding Schemes [HeBe97]

These schemes detect congestion by monitoring the queue size. EQ A.1 in Appendix A describes the behaviour of a scheme that uses RM (Resource Management) cell with an unknown marking mode (neither relative rate or explicit rate). The equation is assumed to apply at the source node since RIF (Rate Increase factor) and RDF (Rate Decrease Factor) are used to calculate the ACR (Allowed Cell Rate). These two factors are source parameters. This algorithm has three operating regions:

- When the resource is underutilized causing the queue to remain empty, the source is allowed to increase its transmission rate by the product of the RIF, PCR (Peak Cell Rate), and current transmission rate of RM cells.
- In moderate loading where the queue size is non-zero but less then the bottleneck queue threshold, q_T, the source is allowed to increase its transmission rate similarly by some amount of the current service rate of RM cells at the switch.
- When the queue size exceeds q_T, the source has to reduce its transmission rate by the product of the RDF and the current service rate of RM cells, divided by the number of connections at the bottleneck switch.

Some equal sharing of switch resource is factored into this algorithm. Only one congestion bit is used to indicate the congestion state of the switch to the source. Therefore, some enhancements to the TM4.0 ABR control protocol is required to communicate to the source information about the queue size and the number of active connections residing on the bottleneck switch. Additionally, the source also has to estimate the service rate at the bottleneck switch. These special requirements from TM4.0 ABR control protocol show this algorithm has no understanding of the ABR control protocol. Also, this algorithm is expressed in continuous time, and cannot be implemented in discrete time protocols. Fairness and stability are questionable with this

single bit congestion notification scheme because the use of RIF and RDF constants does not factor in any fair resource allocation scheme.

EQ A.2 in Appendix A describes another queue size thresholding scheme, but it uses Explicit Rate notification. It employs a very simple formula for calculating the explicit rate. Under congestion, the switch decreases the current ER field value by an amount proportional to the difference between the current queue size and the queue threshold. The resultant behaviour of the algorithm is to drive the queue size towards q_T. The equation operates in discrete time where the time slot is between RM cell arrivals. The control gain, a_j, can be a queue control formula that is linear or non-linear. Lack of theoretical development on the control gain shows weaknesses. Since the explicit rate is calculated on a per-VC basis, fairness can be achieved depending on how the control gain is designed to factor in fair share allocation. Stability is doubtful with this simplistic approach.

1.2.2 Rate Thresholding Schemes [HeBe97]

Rate thresholding provides freedom of queue design at the expense of higher complexity in the control algorithm. Freedom of queue design allows portability of the control algorithm from one architecture to another. The design simply aims to achieve some target utilization, ρ_T. EQ A.3 in Appendix A describes the behaviour of the switch in detecting congestion. The switch maintains a running estimated mean allowed cell rate (MACR) from the current cell rate of each VC. Congestion is declared when utilization exceeds ρ_T, and the estimated arrival rate of connection j, $\lambda_j(t)$, exceeds the MACR(t). Control tries to keep the arrival rate from exceeding capacity and at the same time disallowing the source from exceeding the mean arrival rate at the port. It approximates min-max fairness under homogeneous ABR traffic conditions only. Although the equation is given in continuous time, it may be implemented to operate in discrete time, on the arrival of every RM cell. In this case, all time instances (t) can be replaced by (n). Intelligent marking – relative rate marking of RM cell is used for congestion notification. With single bit congestion notification it is very difficult to maintain stability. Therefore,

the switch congestion detection algorithm is heavily relied on to maintain stability. In general, MACR based traffic estimation has its limitation as demonstrated in chapter Chapter 3:. However, this algorithm shows an enhancement by factoring in per-VC arrival rate.

L. Roberts [Robe94] describes the Proportional Rate Control Algorithm (PRCA) as presented in EQ A.4 in Appendix A. This algorithm keeps an estimated mean allowed cell rate (MACR) based on the ER (Explicit Rate) field values of RM cells belonging to a connection. According to TM4.0, by default, the value of this field is the PCR if the upstream nodes did not reduce it. Therefore, it is questionable that this algorithm can maintain a proper MACR. To this extent, this algorithm may not adhere to TM4.0 compliance. The algorithm detects the onset of congestion by monitoring for the utilization to exceed some threshold. When an RM cell arrives, the ER field is updated with the minimum between the current ER field value and a fraction of the estimated input rate, if the switch is congested. Although the algorithm is presented in continuous time, the update of the ER field can only be done in discrete time.

FASTRAC [AwOu99a], owned and developed by Nortelnetworks, is a rate thresholding algorithm that demonstrates sound theoretical development, compliance to TM4.0, simplicity and practicality, and decent performance. The features of this algorithm is the main subject of investigation in this thesis, and will be detailed later.

1.3 Motivation and Approach

The motivation for this thesis is trying to collectively address some of the weaknesses of many ABR papers published in the literature. We want to approach the topic of ABR flow control in a more structured way, and explore the effect of long propagation which is a major factor in the control algorithm's performance. First, we have studied the TM4.0 document [ATMF96] in order to understand the essential details of the standard ABR control protocol. Next, we have examined several switching architectures in order to gain some practical knowledge of switch design. The theoretical basis in ABR control

has allowed us to more methodologically examine various control algorithms, and consequently, to perform an in-depth analysis of a control algorithm called FASTRAC. We have extended the design of the algorithm and implemented the simulation model in OPNET. The extended simulation model generates results that allow us to analyze, explain, and compare with the theory. Having established a firm foundation of ABR control principles through theoretical discussions, and how to evaluate performance through simulations, we focus on understanding the effect of long propagation delay on FASTRAC. Extensive simulation work is performed on VSVD to see how it improves the algorithm's performance under long propagation.

1.4 Objectives

We are interested in understanding and demonstrating the importance of integrating ABR flow control protocol described by TM4.0, practical switch designs for congestion avoidance/management, and feedback control theory in order to design good ABR control algorithms. Specificly, we want to use FASTRAC algorithm

- To demonstrate what constitutes good simulation designs,
- To explore the effect of long propagation delay,
- To design and integrate VSVD support,
- To test and verify our design by performance evaluation,
- To demonstrate that VSVD aids switch queue management in a network consisting of long propagation delay links by performance evaluation.

1.5 Thesis Contributions and Organization

The contributions of this thesis are:

- Design and implement a simulation model to demonstrate, through performance analysis, instabilities of a surveyed control algorithm from the literature.
- Evaluate the performance of FASTRAC without VSVD under long propagation delay involving satellite links.

- Modify and extend FASTRAC's implementation model by adding VSVD support. In particular, propose and add an efficient per-VC scheduler to FASTRAC's implementation model.
- Evaluate the performance of FASTRAC with VSVD through simulation.
- Compare and contrast long propagation delay performance results with or without VSVD against a competing algorithm, ERICA+ [GoCa98].
- Demonstrate good simulation designs that are aimed to validate theory and allow key performance measures for evaluating ABR algorithms on the basis of fairness, responsiveness, stability, and robustness.
- Demonstrate a better structured presentation of the ABR flow control that associates together TM4.0 ABR control protocol, practical switch designs, and feedback loop control theory.

The remainder of the thesis is organized as follows. Chapter 2 primarily present a structured model of ABR flow control by discussing TM4.0 ABR Control Protocol, VSVD network topology, and practical switch functions for supporting ABR. We also present FASTRAC and its OPNET simulation model. We discuss FASTRAC's key features and advantages over other algorithms. We also discuss how to add VSVD support into FASTRAC, including our proposed per-VC scheduler. Chapter 3 presents the work we have done to investigate instability in one of the surveyed algorithms. We also demonstrate good simulation design to test the algorithm and obtain a perspective to analyze and evaluate performance of an ABR control algorithm. Chapter 4 presents results from tests of FASTRAC's operation, to demonstrate it is theoretically sound and how it adheres to TM4.0. We continue to demonstrate good simulation design aimed to test for fairness, stability and responsiveness. Chapter 5 presents the performance evaluation of FASTRAC with and without VSVD under long propagation delay using the Satellite configuration. We also test the successful integration of VSVD into FASTRAC. We compare results with ERICA+ for obtain an appreciation of how well FASTRAC stands against a well known published algorithm. Finally, in chapter 6, we summarize conclusions on how we fulfilled our objectives and make suggestions of topics for future study.

Chapter 2: Network Operation and Modeling

This chapter presents and discusses the importance of understanding and integrating TM4.0 ABR control protocol [ATMF96], practical switch designs, and closed loop feedback control theory in order to design an effective ABR control algorithm. The essence of TM4.0 ABR control protocol most relevant to our investigation is summarized. FASTRAC [AwOu99a] is explained as an example of a well-designed ABR control algorithm while providing the information necessary for understanding how the algorithm is to be translated and implemented in OPNET.

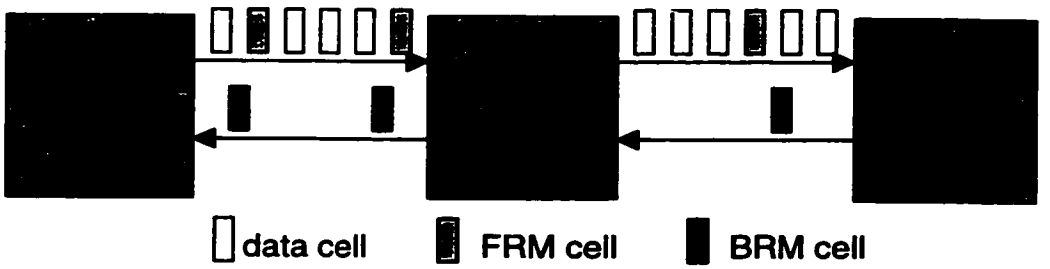


Figure 2-1: ABR Data Flow

2.1 TM4.O ABR Control Protocol [ATMF96]

The ATM Forum has defined a special cell, called a Resource Management (RM) cell to be used in the protocol for congestion and rate control. The RM cell is a special 53-byte ATM cell with a special payload to carry information about the source to the switch node and information about the switch to the source node. Figure 2-1 shows the data stream of an ABR connection. At a pre-defined interval, an RM cell is interpersed into the user data stream at the source.

This cell carries traffic parameters about the source. At a congested switch node, the switch node modifies certain congestion indication fields, discussed below for each feedback control mode, in the passed-by RM cell to indicate its congestion state. At the destination node, user cells are extracted while RM cells are looped around and transmitted back to the source node. Before these cells are transmitted their direction bit

(DIR) is flipped to indicate the backwards direction. If the destination is congested it is allowed to modify the RM cell in the same way as the switches to indicate its congestion state. When the returned RM cell reaches the source node, the source examines the RM cell's content and adjust its rate accordingly. The protocol forms an end-to-end closed-loop feedback network. RM cells moving toward the destination are known as Forward Resource Management (FRM) cells, while those turned around by the destination and heading to the source are Backward Resource Management (BRM) cells.

2.1.1 Closed-loop Feedback Control Modes

There are two categories of closed loop feedback control, employing RM cells: relative rate and explicit rate marking.

In relative rate marking, the congested switch node simply flip a congestion indication (CI) bit or No Increase (NI) bit in an FRM cell passing by to declare it is congested. Optionally, it can set the CI bit in BRM cell to more rapidly report congestion to the source. Under extreme congestion, the switch may generate an out-of-rate BRM cell and set the CI bit to report the congestion. When the CI bit is set, the source must decrease its transmission rate which is parameterized by the Rate Decrease Factor (RDF). When the NI bit is set, indicating mild congestion, no action is to be taken by the source. When no congestion is detected, the source may increase its transmission rate according to a formula involving the Rate Increase Factor (RIF). RDF and RIF are thus set by the network as part of the switch provisioning. Relative rate marking were employed in first generation ABR capable ATM switches. In the earlier days than these switches when the RM cell was not completely defined, Explicit Forward Congestion Notification (EFCI) switches were common. These switches set the EFCI bit in the ATM cell header when it is congested. The destination node detects this and is also responsible for setting the CI bit in looped back RM cells. EFCI switches are not capable of modifying the RM cells. The explanation above for the various ABR parameters involved in relative rate control shows many of the subtleties of where and how the parameters are used. These details

explain the flaws in the surveyed algorithm described in EQ A.1 in Appendix A. Second generation ABR-capable ATM switches employ explicit rate congestion notification.

In explicit rate marking, the switches and destination node can indicate its supported cell rate in the Explicit Rate (ER) field of the RM cell. These nodes are allowed to only decrease the value of the ER field when necessary. They never increase the ER value. This is another subtlety that some algorithms commonly ignore. The source is responsible for adjusting its transmission rate to the ER value of the BRM cell.

2.1.2 ABR Traffic Parameters

Relative rate and explicit rate marking requires a number of traffic parameters to be signaled by the source to the destination during connection setup. These parameters are maintained on a per-VC basis. Once the source receives permission to start transmitting, it starts scheduling cells at the Allowed Cell Rate (ACR) which is initialized to be equal to the Initial Cell Rate (ICR) parameter. Throughout the lifetime of the connection, the source must transmit at an ACR no more than the Peak Cell Rate (PCR) parameter and no less than the Minimum Cell Rate (MCR) parameter. The switch nodes are obligated to guarantee at least the MCR of bandwidth. An FRM cell is populated with the source's current cell transmission rate (i.e. ACR) in the Current Cell Rate (CCR) field, the minimum cell rate in the Minimum Cell Rate (MCR) field and its desired peak cell rate in the Explicit Rate (ER) field when it is generated at the source.

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

Table 2-1: TM4.0 ABR Source Rules

Table 2-1 summarizes the source rules for adjusting the ACR based on the traffic and configuration parameters.

When the data stream is initiated, it must start with an FRM cell to provide immediate feedback about the supported rate of the network. The protocol also provides a number of parameters that must be used by the source to time the injection of RM cells. These adjustable parameters are *Nrm* and *Trm*. An FRM cell must be sent every *Nrm* cells and must occur at least every *Trm* seconds. The *Trm* parameter is used to force the source to send an FRM cell when the data rate is too low such that the inter-cell period is longer than *Trm*. This situation is quite common with bursty traffic where the idle period might be too long. The protocol also requires that at least *Mrm* data cells must be sent between any two FRM cells to avoid just sending all FRM cells when the transmission rate is slow, starving out bandwidth for data cells. Often papers describing ABR simulations never state initial values involving ICR, *Nrm*, *Mrm*, *Trm*. An example is [Kris97].

Another very important parameter is the *FRTT* (Fixed Round Trip Time). It is accumulated during the signaling procedure as the sum of fixed RM cell processing delays and propagation delays in the round-trip. It is the minimum delay along the path and does not include any queueing delay. A large *FRTT* implies that a source might inject a large number of user cells into the network in the beginning of a connection or when RM cells are lost. *FRTT* is used to calculate *Crm* which limits the source rate in the absence of BRM cells, and used to cap the *ICR* to the maximum burst delay ratio. *FRTT* is a fundamental piece of information about the delay involved in the feedback loop that can be used to design algorithms to be more tolerant to long propagation delay. Yet, many algorithms do not use it, including all of the surveyed algorithms in section 1.2.

If a node is both a source and a destination (i.e. participating in a bi-directional call) then the node will send the FRM cell as scheduled, then send any outstanding BRM cells (turned around FRM cells for the other direction of the call), and finally send all the data cells available in the queue up to the *Nrm* count. All cells must be sent at the ACR of the source. Overall, there are 13 rules for the source node's behaviour, 5 for the switch node

and 6 for the destination node. For simplicity, time-out, re-negotiation and lossy behaviour are excluded from this study although the TM4.0 control mechanisms [sections 5.10.4, 5.10.5, 5.10.6, ATMF96] have built in safe guards that would handle those scenarios. The interest of this study is to examine the transient behaviour under controlled heavy loading. In later sections describing FASTRAC, many of the shortcomings of algorithms in standards compliance mentioned above will be addressed.

2.2 TM4.0 VSVD Loops [ATMF96]

The introduction of ABR introduces a closed loop feedback control into the network which brings with it a number of classical closed loop feedback control problems including stability. The Virtual Source Virtual Destination (VSVD) switch behaviour specified in TM4.0 introduces some tools to integrate ABR more successfully into the network. This specification calls for the splitting of an ABR connection into separately controlled ABR segments (or loops). A segment can be bounded by either a source or virtual source (VS) at one end, and either a destination or virtual destination (VD) at the other. Simple relay nodes can exist inside the segment. The boundary switches behave both as a VS and VD, hence the term VSVD switch.

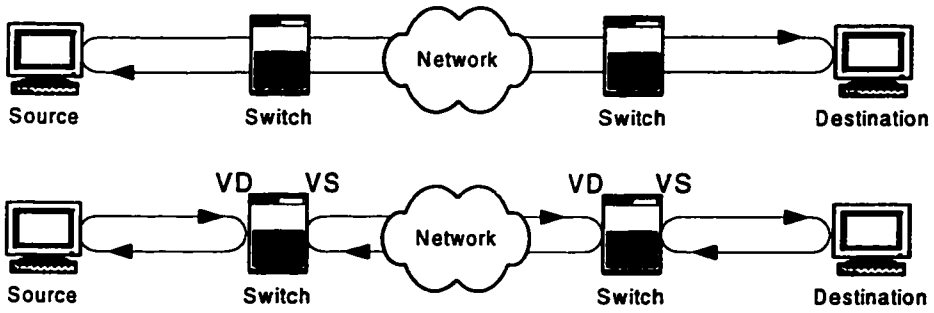


Figure 2-2: End-to-end ABR Control Loop vs VS/VD Control Loops

The VD side of the switch receives cells from the upstream segment. Data cells are relayed across to the VS and resource management (RM) cells that carry rate and congestion feedback information are turned around to the source or virtual source. The VD follows the rules of both the switch and the destination node. The VS side of the

switch behaves as a source end system. It controls the transmission rate of every virtual circuit (VC) and schedules the transmission of data cells from the upstream loop and self generated RM cells. It terminates BRM cells returned from downstream and adjusts its transmission rate based on the source rules. Generation of RM cells are based on the *Nrm*, *Trm*, and *Mrm* parameters, just like a source would behave. In effect, the end-to-end control is replaced by segment-by-segment control as illustrated in Figure 2-2. The coupling between adjacent ABR control loops has been left unspecified by the ATM Forum, and is implementation specific.

The advantage of the segment-by-segment control is that it allows the network to be broken down into smaller network segments with different characteristics. One example application that involves an end-to-end ABR circuit is isolating long latency satellite networks from terrestrial networks so as to keep the effects of long latency to within the satellite loop. In this case, the gateway switches interfacing the satellite network and the terrestrial network can implement VS/VD to isolate downstream switches from the effects of the long propagation delay satellite paths. In terrestrial/satellite networks without VS/VD, terrestrial switches that are bottleneck, must buffer cells up to the feedback bandwidth-delay product of the entire loop (including the satellite hop). With a VS/VD loop between the satellite and the terrestrial switch, the queue size due to the satellite feedback delay is confined to the satellite switch. The terrestrial switch only buffers cells that are queued due to the feedback delay of the terrestrial link to the satellite switch. Another advantage here is that the segments have shorter feedback loops that can potentially improve performance because feedback is given faster to the sources. Following the above discussion, the VS/VD capability is expected to offer several benefits including the following:

- Breaking the ABR control loop into smaller segments to improve performance.
- Choosing the ABR parameters on a per-segment basis to optimize performance.
- Creating separate control domains for administrative reasons.
- A VS/VD switch placed at the interface between a private network and a public network may act as a firewall to protect the public network.
- Allowing UBR <–> ABR or GFR <–> ABR conversion; thus providing ABR-like service to users without ABR equipment.

- A VS/VD switch with large buffers can absorb the traffic variations taking place in downstream switches/networks, thus improving the service as seen by upstream switches/networks.

In a later section, the satellite/terrestrial network configuration's performance with VSVD is examined and compared against the same network without VSVD to demonstrate the stated benefits.

2.3 Switch Functions Required for Feedback Control

Switch design plays an important role in the effectiveness of a control scheme. The switch design governs what control mechanisms can be used, and what control algorithms are possible. Each design aspect contributes to the characterization of the feedback control scheme. Understanding switch designs precludes the creation of control algorithms that are purely mathematical and cannot be practically implemented. Almost none of the algorithms surveyed presented any discussion of possible switch designs for implementing them. It is important to discuss the required or possible switch designs to obtain a sense of how practical the algorithm is and assess ease of implementation. Such information assists designers to understand the required software/hardware architecture to support the algorithm. Real-time commitment is a major factor in designing a switch that will support the advertised line rate. By understanding the required switching architecture and sampling interval, designers have an early start in assessing real-time commitments.

The switch design can be examined under three categories [Kris97]. We use these three categories to classify surveyed algorithms and FASTRAC.

a) Congestion Monitoring

Congestion Monitoring refers the mechanism under which the switch determines its congestion state. Some mechanisms that are examined in our investigation are:

Queue Fill Thresholding: The switch monitors the queue size, and declares congestion when the queue size exceeds some high threshold value. Similarly, a low threshold value can be used to detect the exit from a congestion state. Examples of algorithms using this congestion monitoring mechanism are found in section 1.2.1.

Arrival Rate Thresholding: The switch monitors the cell arrival rate, and declares congestion when the arrival rate exceeds some threshold value. An estimated cell arrival rate can be computed by keeping a count of the number of cells received in some time interval, and the rate is equal to the ratio of this cell count over the length of the time interval. The threshold value can be the available bandwidth. Arrival rate thresholding provides congestion avoidance since congestion is detected right at its onset. Additionally, this technique decouples congestion detection from the queue implementation, providing more queue design freedom. Examples of algorithms using this congestion monitoring technique are found in section 1.2.2.

b) Congestion Notification

Congestion notification refers to the mechanism the switch uses to communicate its congestion state back to the source. Some mechanisms that are seen in our investigation are:

Intelligent Marking: The switch uses the RM cell in relative rate marking mode to declare congestion on a per-VC basis by setting a congestion indication bit. The source typically reacts to congestion by making an incremental rate change. This approach requires the switch to examine each VC's resource utilization and compare it against some fair share threshold to determine if the VC should be declared congested. This kind of per-VC accounting requires sophisticated switch design. An example algorithm employing this form of congestion notification is described by EQ A.3 in Appendix A.

Explicit Rate Indication: The switch uses the RM cell in explicit rate marking mode to indicate to the source its supported rate. This approach provides rapid convergence to the supported rate compared to incremental rate change schemes that tend to oscillate around

the optimal operating point. It offers the most responsive feedback to changing network conditions. In addition to some kind of per-VC accounting required to detect congestion, this approach requires some kind of fair resource allocation scheme to determine the ER of a connection when congestion is declared. The most effective form of congestion notification, but requires the most sophisticated switch design and real-time commitment. Example algorithms employing this form of congestion notification are FASTRAC, and the ones described by EQ A.2, and EQ A.4 in Appendix A.

c) Queue Service

Queue service refers to the service provided by the switching nodes. The nature of the queue service governs what kind of congestion monitoring and scheduling can be performed.

Shared Buffer: This queue design is the simplest to implement, but lacks flexibility in the control scheme and offers poor performance. Cells from all VCs are serviced by one queue. No per-VC information is kept, thus per-VC congestion detection is not possible. Uncontrolled delays due to interaction between competing flows are created in the shared buffer that can cause delays on the processing and return of RM cells to the source. Such delay definitely results in poor control of a VC carrying ABR traffic. When combined with binary feedback notification, severe performance degradation would result [Kris97]. EQ A.1 and EQ A.2 in Appendix A are examples of algorithms that require only shared buffer. FASTRAC also requests for only a shared buffer switch architecture, but it makes improvements in other areas to enhance performance as discussed in later sections.

Per-VC Accounting: This queue design requires a record to be kept of each VC passing through the switch. Each record holds information such as the rate and number of cells received for a connection. This information allows the switch to monitor each VC and track it against the fair share allocation of resources. Thus congestion can be declared and notified with per-VC information. This design does not necessarily require per-VC queueing. Although accounted for on a per-VC basis, cells from all VCs can be serviced by a shared buffer. Evidently, this design provides more flexibility in the control scheme

and improved performance. EQ A.3 and EQ A.4 in Appendix A are examples of algorithms that require per-VC accounting to operate.

Per-VC Queueing: This queue scheduling provides isolation among the cell flows of the VCs. Delay and loss behaviour can be predicted and controlled which gives rise to sustainable guaranteed quality of service. Fairness in the service can also be achieved by utilizing the appropriate service algorithm such as Fair Queueing or Weighted Round Robin. Isolation of the per-VC traffic provides the additional protection from abusive users. Although per-VC queueing offers considerable advantages over single FIFO queueing, this comes at the significant cost of switch design and implementation. Efficient buffer management and scheduling among the queues can be very costly to design and implement and even difficult if not impossible when the connections grows substantially large. Typically, per-VC accounting will suffice for any ABR control algorithm requiring per-VC information. Per-VC queueing is only necessary when the switch needs to schedule cells on a per-VC basis as with VSVD.

2.4 FASTRAC

FASTRAC [AwOu99a] is developed from the ground up using a simple control system model. This section briefly highlights how FASTRAC has good modelling of the ABR feedback loop and links TM4.0 ABR control protocol to control theory. The full derivation of the FASTRAC formulae referenced in this section is in appendix B. The FASTRAC model is based on an aggregate data flow, independent of the switch architecture. It uses arrival rate thresholding to detect congestion. Explicit rate congestion indication is employed to avoid the *beat-down* problem. Shared buffer suffices for the algorithm. Although it uses mostly the simpliest switch design, FASTRAC strives to improve from the basic principles to provide added-value enhancements such as delay sensitive gain and optional queue control functions. It is light-weight enough in implementation that it can be implemented entirely in software with modern processors to support an OC3 link. Additionally, FASTRAC is 100 percent

compliant to TM4.0, meaning it can be implemented and deployed in live networks serving the Internet. It makes proper use of TM4.0 traffic parameters.

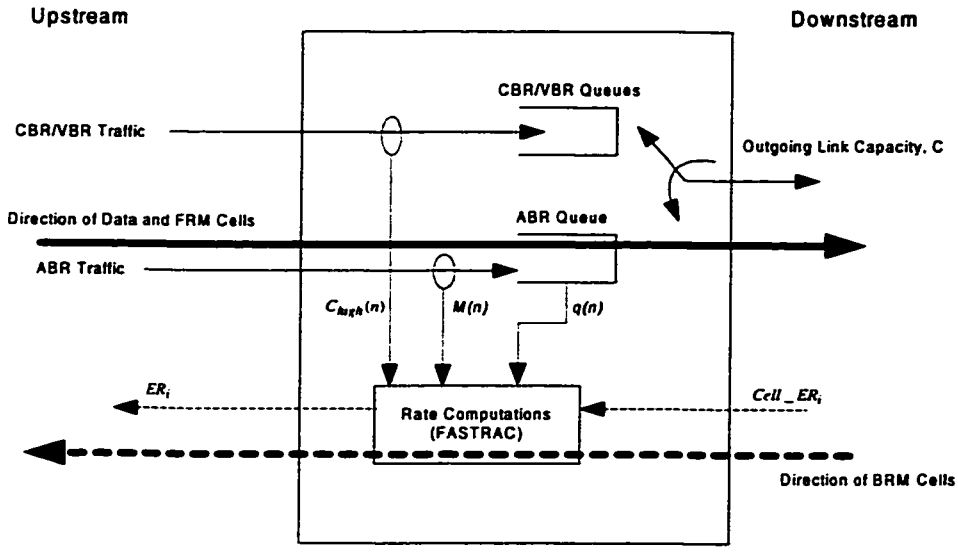


Figure 2-3: FASTRAC Model of Switch

Figure 2-3 shows a model of a switch employing FASTRAC as the ABR flow control algorithm. The input to the system consists of ABR and non-ABR (CBR and VBR) traffic whose rates are *M(n)* and $C_{high}(n)$ respectively. FASTRAC requires these rates, the ABR queue size, *q(n)*, and the output link capacity as input parameters into the algorithm. No distinction is made for per-VC flows. This simplicity of FASTRAC allows it to be implemented inexpensively and easily. It can be applied to an input, output, or input/output buffered switching architecture [AwOu99a]. For this explanation of FASTRAC, we assume a common queue for the ABR traffic since the algorithm does not specifically require per-VC queueing.

The algorithm functions in discrete time. The switch computes a fair share allocation $r(n)$ common to all ABR connections using a particular output port every t_m seconds, the algorithm's sampling clock period. The control output is computed at time $(n-1)$, n, $(n+1)$, ... , where n is understood to be multiple of t_m seconds. The fair share allocation represents the excess bandwidth on the output link. The value written into the

ER field of BRM cells passing through the output port in the backwards direction towards the source during this point in time is based on the current value of fair share. The BRM cell will reach the source after a delay of d_b seconds. The source upon receipt of a BRM cell will change its transmission rate to the ER value computed at time $t - d_b$. The effect of the source rate adaptation becomes apparent at the switch port after another delay of d_f seconds. The total time lag from the time a fair share decision is made to the time its effect is felt by the bottleneck switch is feedback delay, $d = d_b + d_f$ seconds. The rate $r(n+1)$ is stored during the next time slot $[n+1, n+2)$ in a table accessible by all BRM cells flowing through the output port. For connections bottlenecked at the output port, the computed ER based on $r(n+1)$ will be written as the new ER value in the payload of the BRM cell.

FASTRAC operates in way that always try to adjust the fair share value to achieve the target service rate, T which is typically the output link capacity minus $C_{high}(n)$. It observes an error signal representing the bandwidth mismatch between the arrival rate, $M(n)$, and the target service rate. Step 4 of section 2.4.6 describes this operation. In summarizing FASTRAC's operation, a rate mismatch due to the activation or deactivation of a number of sources produce the necessary error term $e(n) = [T(n) - M(n)]$, which can then be used to determine the correct fair rate allocation to the sources. Intuitively, we observe that FASTRAC achieves max-min fairness. If a source suddenly increases its activity, the measured ABR traffic rate $M(n)$ will rise giving a negative error $e(n) = [T(n) - M(n)]$ and resulting in a reduction in the next effective fair share $r(n+1)$; likewise if a source suddenly decreases its activity, the next effective fair share will be increased.

2.4.1 Configurable Sampling Rate

FASTRAC takes rate measurements every t_m. This controllable interval is a practicality enhancement that can be tuned depending on the performance of the switching architecture. It should be chosen to best allow an accurate measurement and yet satisfy

implementation constraints. Choosing a large number requires less processing power, but can produce slow responses. Too small of a value increases processing power and also produce faster responses. A suggested heuristic is presented in appendix B.1. Step 5 in section 2.4.6 shows values for OC3 and T3 line speed using this heuristic.

2.4.2 Addressing Stability

The concept of *stability* has a two-fold meaning. From the perspective of the goal of the control process, the stability condition is defined as the ability to produce the appropriate control signals for a wide range of input variables that maintains the system at a bounded input/bounded output stability. In this case, the environment of multiple sources simultaneously changing state and sharing the limited availability of resources must be kept *stable*. This means that when the number of active sources is fixed, system resources allocated to each source settle down to a steady state in a definite period of time. This definition also implies that, if a new source becomes active, existing active sources will adjust their resource usage so that after a brief transient period, the system settles down to a new steady state. From the perspective of implementation, stability implies that with respect to variations in resource availability due to unpredictable and physical causes, adaptation activities do not suffer from oscillations, which are undesirable because they cause both fluctuations in user-perceptible qualities, and excessive amount of adaptation attempts that may occupy too much resource to overload the system. So, in order to converge to steady state regardless of disturbances and statistical multiplexing, we need to prove that the system is stable. Appendix B.2 provides this proof.

The proof demonstrates that oscillations in the load occur if the system tries to correct more than a certain fraction of the difference between the estimated load and the target load per control interval. This is due to the periodic discrepancy between the estimated load and the actual load caused by time lags. From this, a relationship between the control gain, α, and the feedback delay, d as shown in EQ B.26 in Appendix B, is defined. This control gain is applied to the error term representing the bandwidth

mismatch in calculating the fair share allocation. The completed form of equations in calculating the fair share value is presented in step 9 of section 2.4.6.

In addition to stability requirements, it is also desired that the system responds quickly to changes in both resource availability and service requirements of the sources. In this proportional control algorithm, the control gain, α is configurable as long as the stability conditions are maintained. The dynamics of the system is affected significantly by different configurations of α. When α is large the system reaches steady state much faster but show more oscillating transients. For a smaller α the system reaches steady state slowly and has smaller oscillating transients.

2.4.3 Accounting for Control Loop Delay

FASTRAC accounts for feedback delay in formulating the control gain, α, as described in EQ B.25. The parameter FRTT that is determined for each connection during connection setup as defined by TM4.0 can be used in this equation. All of the surveyed algorithms and many others in the literature do not take advantage of such useful information available in TM4.0 ABR control protocol. The control loop delay in the algorithm at a switch is set to the maximum of all FRTTs at that switch. This is done to avoid fairness problems due to different distances from a source to the switch. If we define *maxFRTT* as the largest *FRTT* and d as the largest round-trip delay, measured in integer multiple of t_m, among all N connections routed through an output port, then

$$maxFRTT = \max\{FRTT_1, FRTT_2, ..., FRTT_N\}, \qquad \text{(EQ 2-1)}$$

and

$$d = \left\lceil \frac{maxFRTT}{t_m} \right\rceil.$$

FRTT is one of the signaled parameter during connection setup, and remains the same throughout the lifetime of the connection. *FRTT* is the sum of the fixed RM cell processing delays and propagation delays from the source to a destination and back. So

far the parameter d does not take into account the queueing delay. To obtain a more extensive control stability margin, the queueing delay at a node can be factored in the control loop parameter α. Thus, the parameter d becomes

$$d(n) = \left\lceil \frac{maxFRTT + t_q(n)}{t_m} \right\rceil \qquad \text{(EQ 2-2)}$$

where $t_q(n)$ is the time needed to transmit the queued cells $q(n)$. $d(n)$ is applied to the control gain in calculating the fair share in step 9 of section 2.4.6. By accounting for the queueing delay, the control loop parameter is more conservative during transient periods and allows for the queue to drain and the system to stabilize to steady-state while minimizing oscillations of the fair rate computation.

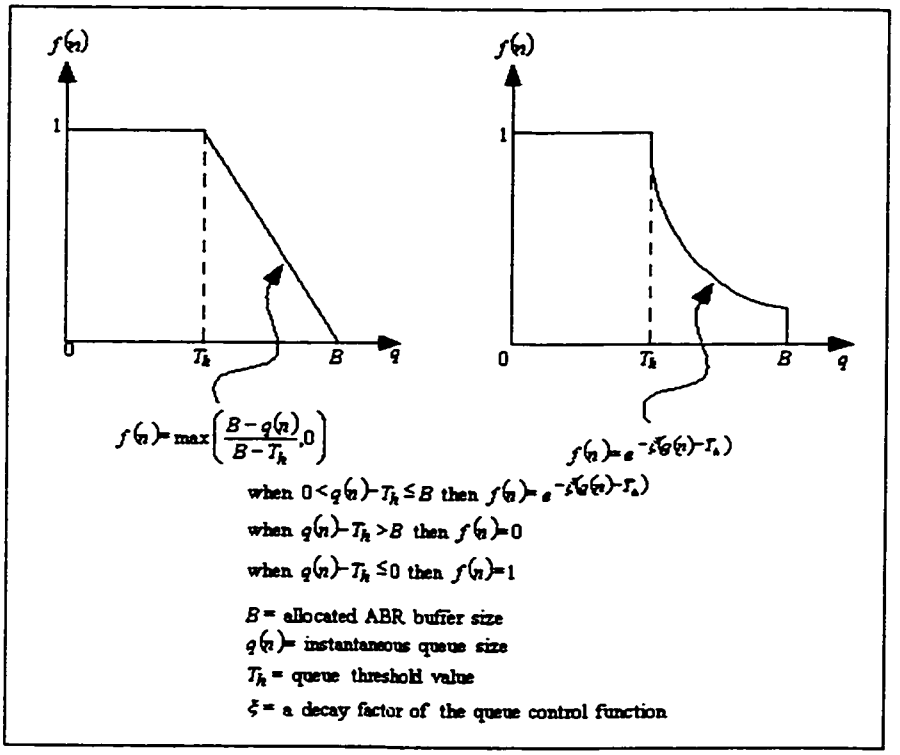


Figure 2-4: Examples of Queue Control Functions

2.4.4 Queue Size Control

FASTRAC can also integrate an optional queue size control mechanism. The queue size control mechanism is a heuristic that can be used when the network queue operates at or beyond the knee of the delay-throughput curve in Figure 2-4 where queue sizes can become large. Such overload conditions occur when the capacity, *T(n)* reduces significantly or too many connections become active suddenly and large feedback delay exists. Although, the queue size control mechanism is not required in normal operating scenarios, when used it can help to effectively control the queue size to minimize cell loss and to minimize network delays. It is an emergency mechanism to increase the FASTRAC algorithm's immunity to instability. There are two approaches to implementing the queue size control mechanism. They involve either scaling down *T(n)* or *r(n)*.

The queue size control mechanism is triggered when the number of ABR cells $q(n)$ in an output port ABR queue (assuming a single ABR queue) exceeds a queue threshold T_h. When this happens, the available ABR rate $T(n)$ is scaled down by a factor $f(n)$, with the capacity $f(n) \cdot T(n)$ used to drain the queue in order to bring the system back to its normal operating point. The function *f()* is user definable. As soon as the overload condition disappears, the queue size control mechanism is deactivated and rate allocation is determined based on the unscaled available rate $T(n)$. Similarly, the same heuristic can be applied to *r(n)*. In this case, the fair share value written to BRM cells is reduced to $f(n) \cdot r(n)$. Theoretically, scaling down *T(n)* should be more effective since it tells FASTRAC the link capacity is low, and thus FASTRAC would avoid indicating a rate increase. Step 7 and step 8 in section 2.4.6 show how a queue control function is applied to *T(n)*. Step 10 in section 2.4.6 shows how a queue control function is applied to *r(n)*.

Design of the queue control function is a separate topic of research. Examples of the function $f(n)$ are given in Figure 2-4. These examples illustrate linear and exponential decay approaches. For demonstration purpose, we applied the linear queue control

function to $T(n)$ in our study. The subject of more elaborate queue control function is not part of this study.

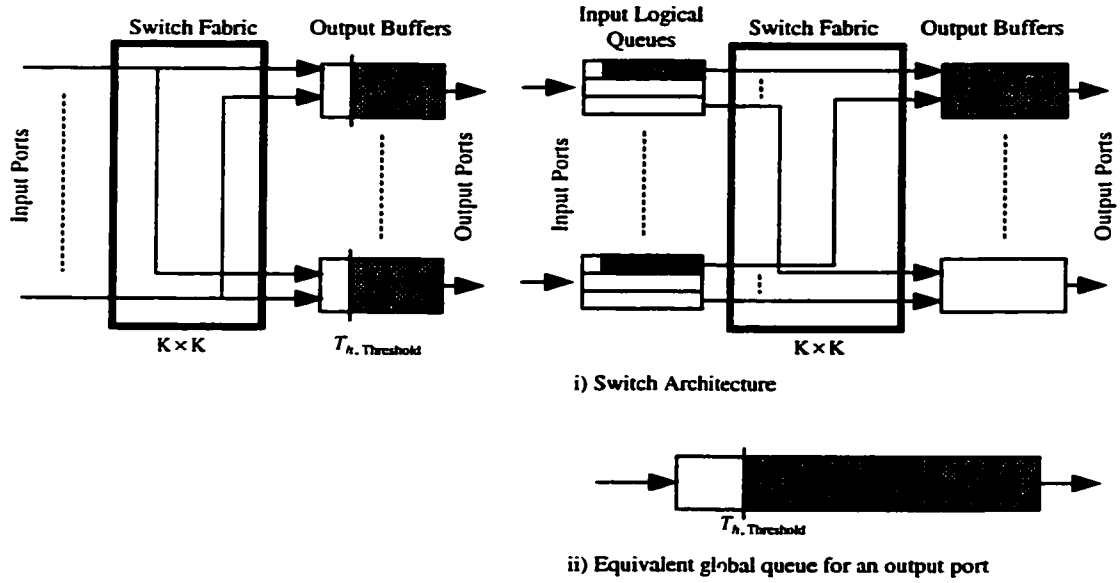


Figure 2-5 Switch architectures and queue thresholds

Another factor governing the design of the queue size control mechanism is the switch buffer architecture. For a switch with a single queue for all the ABR traffic at an output port, a single threshold T_h can be used. This design can also apply to input/output buffered switches. In this case the queue size that is of interest for a particular output port is the sum of all ABR cells that are waiting at various input logical queues, ready to be switched to a particular output port, as well as the number of ABR cells waiting at that output queue. Figure 2-5 depicts these cases. However, using an equivalent global queue may not be fair to ABR connections passing through an input port that is not overloaded. Every ABR connection therefore suffers when even one input buffer is overloaded. Another option for handling switches that have input buffering is to implement an input queue control on top of the output queue control. Let $f_{out}(n)$ be the queue control function of the output buffer. Then, during an overload period, the available output ABR capacity is scaled down as follows: $T(n) \leftarrow f_{out}(n) \cdot T(n)$. Let

$f_{p.in}(n)$ be the queue control function at an input port buffer p. Then, during overload, only ABR connections passing through the overloaded input buffer are scaled down. We can observe intuitively that this scheme eliminates the unfairness problem present in the option described earlier. Moreover, different queue control functions can be used for each input port.

2.4.5 Dynamic Tracking of the Effective Number of Active Connections

Under real world scenarios, the number of active sources changes continuously as existing sources disconnect, or become silent, or as new sources become active. FASTRAC is be able to detect the current number of active sources for the parameter N of EQ B.12 which represents the effective number of connections (or weights) in the rate computation. Dynamically changing N improves system responsiveness and increases resource utilization.

Intuitively speaking, the effective number of active sources is simply the ratio of the available capacity to the current fair share. Although this scheme is more accurate, it is "noisier" in estimating the effective number of active sources compared to the less accurate scheme of taking the ratio of measured ABR rate to fair share. Both schemes will converge to the same steady state value. Let $w_{eff}(n)$ be the effective number of active sources at time n. EQ 2-3 shows a practical version that uses a combination of both techniques.

$$w_{eff}(n) = \min\left\{\max\left(1, \frac{M^*(n)}{r(n)}, \frac{T^*(n)}{r(n)}\right), W_{Total}(n)\right\}, \qquad \text{(EQ 2-3)}$$

where

$W_{Total}(n) = \sum_{i=1}^{N} w_i(n)$ = sum of weights of all established connections at time n,
$w_i(n) = (0,1]$ is the relative weight of connection i to other connections,
$M^*(n) = \max\left(0, M(n) - \sum_{i=1}^{N} MCR_i\right)$,
$T^*(n) = \max\left(0, T(n) - \sum_{i=1}^{N} MCR_i\right)$, and
$r(n) =$ fair share rate computation.

Filtering can be done to remove high frequency noise in the estimation using a moving average, exponential weighted moving average or any suitable filtering techniques. Let's define $w_{T(n)}$ as the filtered effective number of active connections (or weights). In this case, $w_T(n) = w_{eff}(n)$. For exponential weighted moving average, $w_T(n) = (1-\theta)w_T(n-1) + \theta w_{eff}(n)$ can be used, where the gain θ maximizing performance measures was found to be within 0.25 to 0.3 [AwOu99a]. This effective weight is applied to the fair share calculation in step 9 of section 2.4.6.

2.4.6 FASTRAC Algorithm

This section describes the algorithm for implementing FASTRAC and FASTRAC with VSVD. The presentation of the algorithm in this section includes the formulae making up the FASTRAC algorithm together with all procedural information for implementing the algorithm. Only steps and values used in our investigation is presented.

Initialization (at the beginning of time interval n = 1, i.e., interval [0,1) or at time 0):

1). Measured ABR rate: $M(0) = 0$

2). Measured CBR and VBR rate: $C_{high}(0)$

3). Available capacity: $T(0) = \rho C - C_{high}(0) - \sum_i MCR_i(0)$,

where

$\sum_i MCR_i(0)$ is the sum of all MCR values of all ABR connections at time $n = 0$,

ρ = link utilization factor, and

C = link capacity.

4). $r(1) = \frac{T(0) - M(0)}{W_{Total}(0)} = \frac{\rho C - C_{high}(0) - \sum_i MCR_i(0)}{W_{Total}(0)}$,

where

$W_{Total}(n) = \sum_i w_i(n)$ = sum of the weights of all ABR connections established at time n. ($W_{Total} = N$, the total number of ABR connections, if each source has weight $w_i = 1$).

w_i is the weight assigned to an ABR connection i showing its relative priority or importance compared to other connections.

5). Sampling and control update interval,

t_m = 0.5 milliseconds for an OC3 link

t_m = 2.0 milliseconds for an T3 link

At the end of time interval n, (n=1, 2, 3, ...):

6). Estimate ABR traffic rate and CBR/VBR traffic rate at time n: $M(n)$ and $C_{high}(n)$.

7). Estimate available capacity at time n: $T(n) = [\rho C - C_{high}(n)] \cdot f(n)$

where $f(n)$ is the optional queue control function applied to $T(n)$ (described in step 8)

8). Linear queue control: Check queue threshold crossing at time n:

if $q(n) \geq T_h$ then use linear queue control function, $f(n)$ in Figure 2-4.

9). Calculate ABR fair share per unit weight:

$$r(n+1) = [r(n) + \alpha e(n)]\Big|_0^{T_{max}}$$

where

$\alpha = \frac{1}{d(n)+1}$ is the control gain, and

$e(n) = \frac{T(n) - M(n)}{w_T(n)}$ is the bandwidth mismatch error term.

$T_{max} = T(n) = \rho C - C_{high}(n)$ is the maximum available bandwidth.

$w_T(n) = (1-\theta) w_T(n) + \theta w_{eff}(n)$ is the filtered number of effective connections

with $\theta = 0.25$ in our simulation.

$w_{eff}(n)$ is defined in EQ 2-3.

$d(n)$ is defined in EQ 2-2, with $t_q(n) = \frac{q(n)}{C}$ is the queueing delay at time n.

Note that the maximum $t_q(n)$ is continuously determined and stored during each transient queue overload period (i.e., when queue builds up) and used until $q(n)$ hits zero. The operation for computing and updating the maximum $t_q(n)$ is:

if $q(n) > q(n-1)$ or $q(n) = 0$ then compute

$t_q(n)$

$q(n-1) = q(n)$

end if

Under VSVD, *q(n)* is the sum of all per-VC queues and *maxFRTT* is the maximum *FRTT* of the entire ABR flow circuit by TM4.0 definition. It is not the *maxFRTT* value of the local VSVD loop.

10). Update the ER field of the BRM cell as follows:

a) For a switch with output buffering:

Update the ER field of the arriving BRM cell of connection i:

$$ER_i = \min\left(\left[MCR_i + w_i f(n) r(n)\right]\Big|_0^{T(n)}, Cell\_ER_i\right)$$

where

$Cell\_ER_i$ is the actual ER value of the RM cell's ER field, and

$f(n)$ is an optional output port queue control function described in step 8, applied to *r(n)*.

b) For a VSVD switch with output buffering:

Update the ER field of the arriving BRM cell of connection *i* with EQ 2-6.

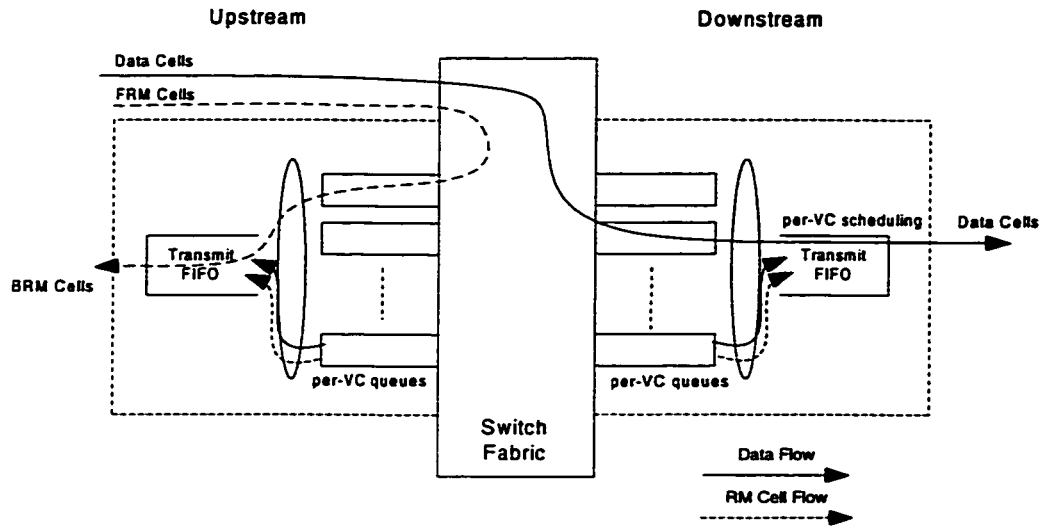


Figure 2-6: Basic VSVD Switch Architecture

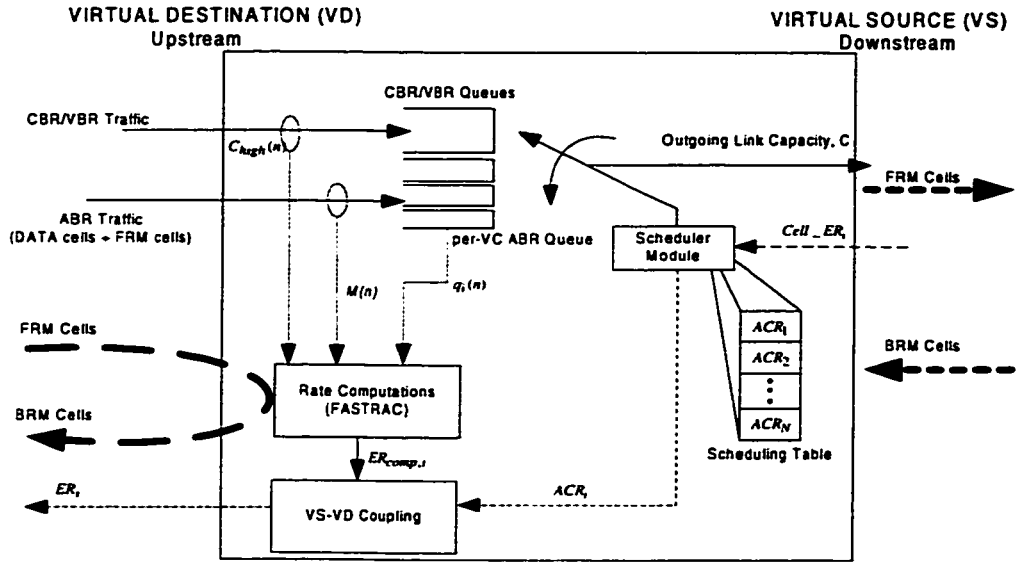


Figure 2-7: VSVD Switch Architecture with FASTRAC

2.5 FASTRAC with VSVD

The basic VS/VD switch architecture is shown in Figure 2-6. The switch sees the VD side as belonging to the upstream segment of the network, and the VS side as the downstream segment. The VD forwards the data cells from the upstream segment (or loop) to the per-VC queue at the VS of the downstream segment. The VS sends out the

data cells and generates FRM cells as specified in the source end system rules [ATMF96]. The VD terminates FRM flows and turns them into BRM cells following the destinations rules [ATMF96]. The VD continues to detect congestion and calculate fair share rate as a No-VSVD FASTRAC switch. If under congestion, it will decrease the value of the ER field in BRM cells it send upstream. The VS controls the scheduling of cells in the downstream direction. It controls its transmission rate based on the ER field information in BRM cells it receive from downstream and TM 4.0 source rules.

A more detailed view of the VSVD switch architecture with FASTRAC is shown in Figure 2-7. From this architecture, we identified and fulfilled the following work items required to add VSVD to FASTRAC.

The VS source rules require per-VC scheduling to be performed. The ACR is computed on a per-VC basis based on the ER value in the received downstream BRM cells for a virtual connection. A more sophisticated scheduling algorithm such as the one in Figure 2-8, is needed compared to the simple aggregate flow scheduler of the no-VSVD case. FASTRAC does not require per-VC accounting to function. It uses aggregate ABR arrival rate measurement, *M(n)* as an input to the control algorithm. Under VSVD, FASTRAC can continue to use the aggregate rate as an input. However, per-VC queueing is required to maintain cells of different connections separate since per-VC scheduling is required. FASTRAC continues to see the VS side as simply the output port with a link capacity, C. Fair share rate computation continues as usual. A VSVD coupling is introduced to synchronize the VD and VS side in terms of allowed cell rate.

With per-VC queueing, we define a per-VC queue control function to apply to individual per-VC queues. In this case the per-VC queue control factor, $f_i(n)$ is determined using $q_i(n)$ for VC *i*. For example,

$$f_i(n) = \max\left(\frac{B_i - q_i(n)}{B_i - T_{h,i}}, 0\right), \text{ if } q_i(n) \geq T_{h,i}, \text{ else } f_i(n) = 1, \qquad \text{(EQ 2-4)}$$

where B_i can be a per-VC high buffer threshold and $T_{h,i}$ a corresponding low threshold. Per-VC queue control isolates the transient behaviour of the VCs on the same link from each other. Thus, if VC *i* experiences a transient overload and its queue builds up, only the ER of VC *i* is reduced and the feedbacks to the remaining VCs are not affected by this temporary overload.

When a BRM cell is received at the VS side on a given connection, its ER, CI, and NI fields are used to calculate the ACR used by the VS side to shape the connection's output traffic. If the downstream switch is congested, the ACR is small. Therefore, the ABR connection may not be able to fully use the fair share of the available output rate. The coupling ensures that the ERs given by the VD side of the switch to upstream sources/VSs, never exceed the corresponding ACRs used at the VS side.

For updating the ER field of an RM cell, the following changes to FASTRAC in the are necessary,

$$ER_{comp,i} = [MCR_i + w_i f_i(n) r(n)]\Big|_0^{T(n)} \qquad \text{(EQ 2-5)}$$

and

$$ER_i = \min(ER_{comp,i}, ACR_i), \qquad \text{(EQ 2-6)}$$

where

$ER_{comp,i}$ is the computed fair share based on FASTRAC.

ER_i is the actual ER value written to BRM cells (the result of coupling).

$f_i(n)$ is defined in EQ 2-4.

The scheduler module is completely decoupled from the FASTRAC algorithm and can be designed with total freedom.

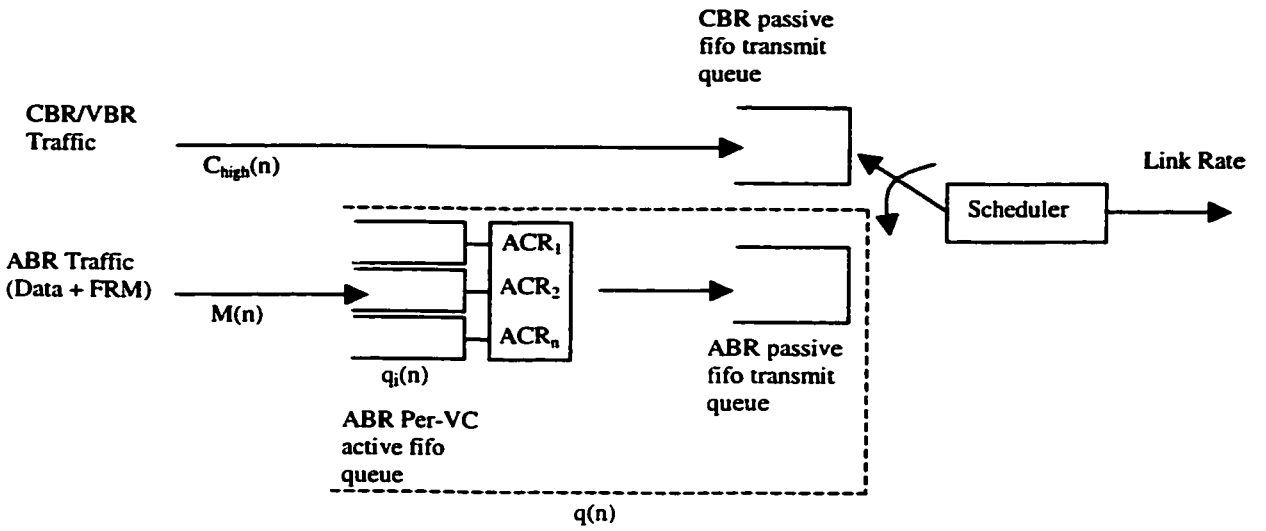


Figure 2-8: Two Stage Queueing Scheduler

One contribution of this thesis is that we proposed and implemented an efficient and robust scheduler that is very simple in implementation, as illustrated in Figure 2-8. Figure 2-8 replaces the queues and scheduler in Figure 2-7. It uses a two stage buffering technique. The first stage buffer is a set of per-VC queues which service their respective ACR_i that are stored in the scheduling table. The servicing of each queue would be done on an interrupt basis. This set of queues provide the per-VC traffic shaping required. The serviced cells are moved into a FIFO transmit queue which is serviced at link rate. This queue provides the serialization of cells from connections to maintain service fairness. This technique reduces complexity present in more elaborate techniques such as round robin servicing based on head of line (HOL) priorities [AwOu99b]. For simplicity in implementation, we implemented this scheduler because our original implementation of the proposed round robin HOL scheduler from [AwOu99b] posed many implementation challenges. This simple technique provides the highest link utilization due to simple serialization versus head of line conflict resolution techniques. The simplicity makes this scheduler robust and scalable. However, *q(n)* now is the sum of all per-VC queues plus the transmit queue. Much work has gone into the rewriting of the simulation model to incorporate the new scheduler to be described in the next section.

2.6 OPNET Implementation Model

The simulation model is implemented in OPNET (Optimized Network Engineering Tools) [MIL3]. OPNET is a communication network simulation software for PC and UNIX environments. It allows network models to be described in a hierarchical graphical environment of networks, nodes, modules, and processes where C code is used to describe procedural information. Its simulation kernel and analysis tools allow simulations to be run on models for performance analysis. All simulation models in our research are implemented with OPNET version 6.0. Simulations were run on a Suns Microsystems's Ultra 10.

The ABR/VBR source models and the basic No-VSVD Edge Switch model was provided by Nortelnetworks with the Bottleneck configuration. The No-VSVD Edge Switch model is modified to produce the VSVD Edge Switch model and the VSVD Core Switch Model. The network configurations simulated are constructed from re-useable links, sources and switches. The switches and internal components are designed to be modular and configurable. We made numerous small changes to the original implementation model to allow components to be tunable and provide the require statistics collection for performance evaluation. Switches can be tuned in many ways including gain factor, sampling rate, maxFRTT, number of connections. traffic parameters (ICR, MCR, PCR), and ABR parameters (NRM, MRM, TRM). Sources are configurable to generate traffic with varying levels of burstiness or persistence, and have settable start/stop time. Statistic collection built into the model provides data about transient queue size, arrival rate, link utilization, ACR, fair share, queue control function, and dynamic source tracking.

This section walks through the implementation to briefly show the many subcomponents working together to allow a switch to perform ABR flow control and VSVD. Most importantly it provides a view that links the FASTRAC algorithm in theory to actual implementation.

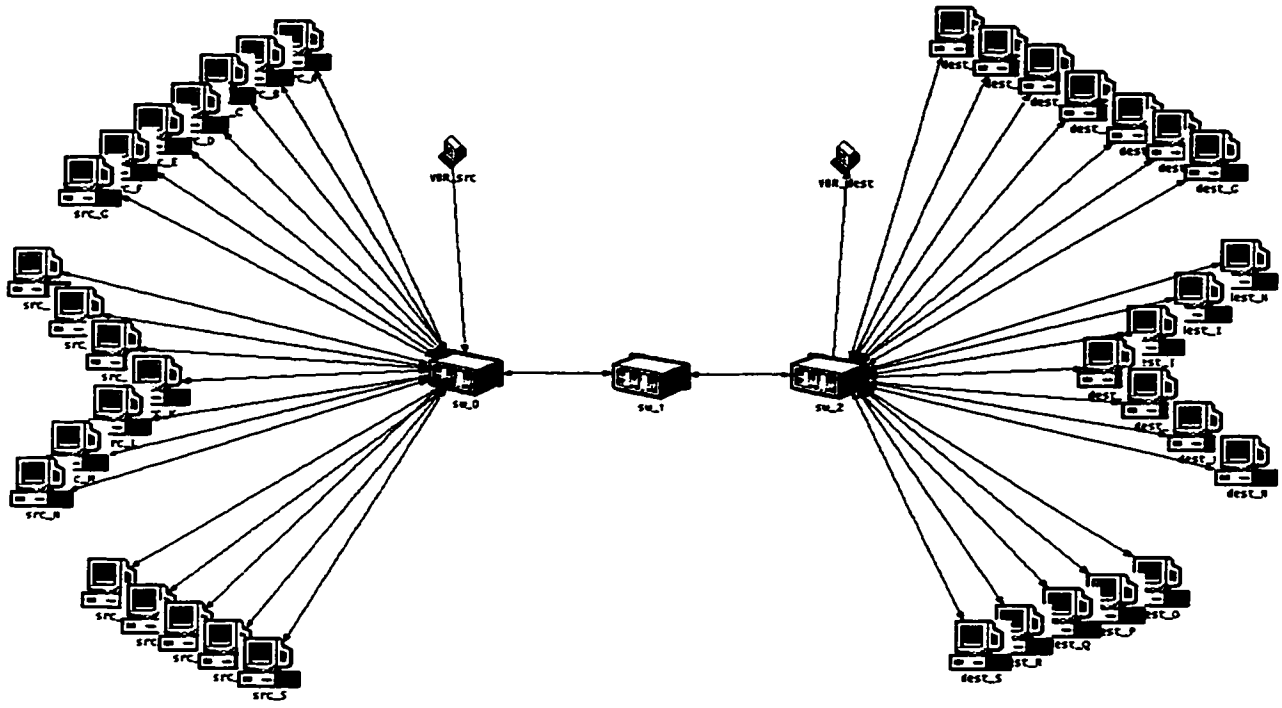


Figure 2-9: OPNET Network Model

2.6.1 Network Model

Figure 2-9 shows the OPNET network model. This is the generic network model that consists of all of the network components that are used in the simulated Bottleneck and Satellite network configuration. The model is built to handle up to 19 ABR source-destination pairs and 1 VBR source-destination pair. The ABR sources generate data and RM cells and implement the TM4.0 source rules for RM cell insertion and termination as described in section 2.1.

In the Bottleneck configuration, switch 2 (sw_1) is removed from the network, and the two switches left use the No-VSVD edge switch model. In Satellite configuration without VSVD, the edge switches, sw_0 and sw_2 use the No-VSVD edge switch model, and sw_1 uses the No-VSVD core switch model. In Satellite configuration with VSVD, all switches use the VSVD switch model. All switch models will be described in the following section under Node Models.

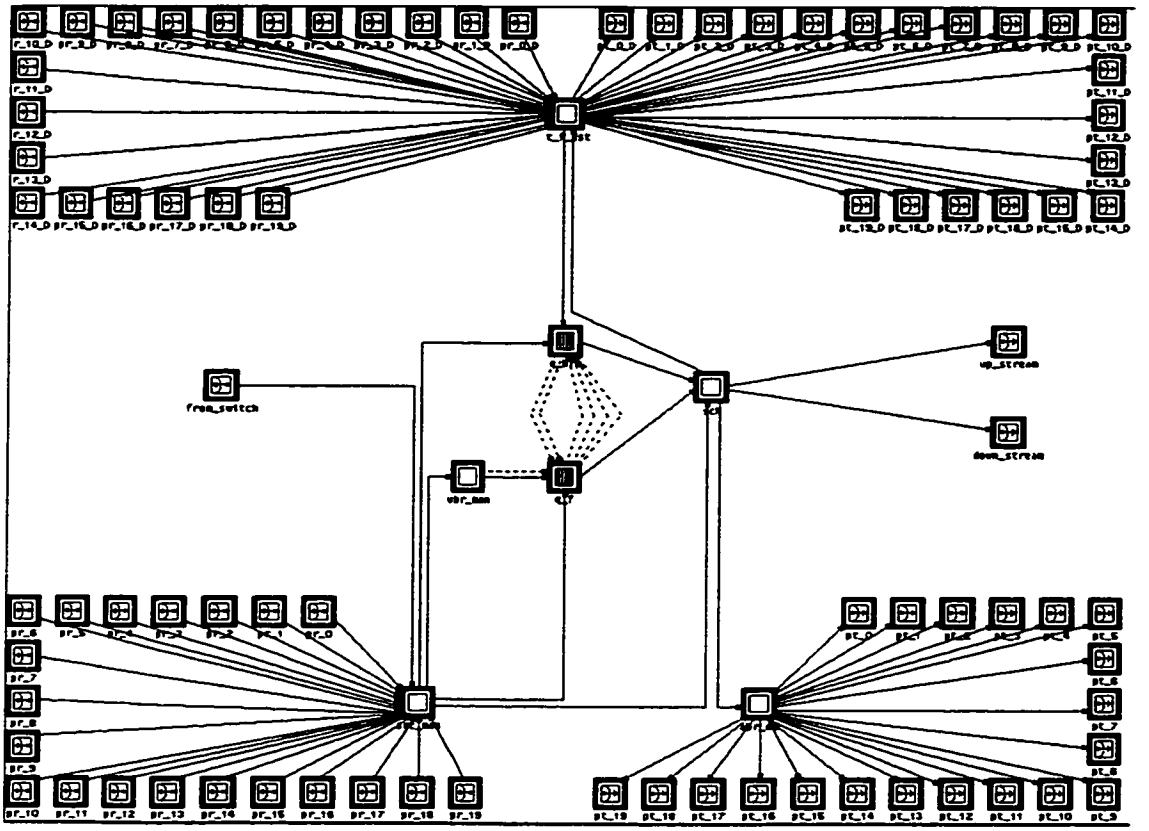


Figure 2-10: OPNET No-VSVD Edge Switch Model

2.6.2 Node Models

2.6.2.1 No-VSVD Edge Switch Model

Model name: TM_ER_Switch_edge_noVSVD

Figure 2-10 shows the OPNET No-VSVD Edge Switch model. This switch is designed to interface with another switch, sources, and destinations. The switch consists of the following modules that collectively implement the functional components described in Figure 2-3.

Source Manager (src_man)

- Interfaces with both ABR and VBR sources
- Directs VBR traffic from source to VBR Manager
- Directs ABR data and FRM cells from source to Forward Queue

- Directs ABR BRM cells from downstream switch to BRM Queue
- Attaches routing information to each cell based on a configurable table that allows control over routing on a per-VC basis

VBR Manager (vbr_man)

- Forwards VBR traffic to Forward Queue
- Collects statistic for $C_{high}(n)$, and communicate it to Forward Queue

Forward Queue (q_f)

- Consists of 2 sub-queues (1 for VBR and 1 for ABR traffic)
- Schedules cells in weighted round robin
- Forwards cells to Scheduler to send out appropriate link
- Collects statistics for $M(n)$, $T(n)$, ABR queue size and link utilization
- Communicates $M(n)$, $C_{high}(n)$, queue control factor, ABR queue size to BRM Queue
- Implementation for queue control function, and communicate scale factor to BRM Queue

BRM Queue (q_brm)

- Fair share calculation
- Updates ER field of BRM cells received either from downstream switch or destination interfaces
- Tracks number of effective source
- Forwards cells to Scheduler to send out appropriate link

Scheduler (sch)

- Decides which interface to forward cells to
    - If forward ABR and VBR traffic that reached the destination switch send to Destination Demux; else send downstream
    - If BRM cells that reached the appropriate source switch send to ABR Source Demux; else send upstream

ABR Source Demux (abr_dm)

- Based on the VCI demultiplex the traffic to appropriate source

Destination Demux (t_f_dst)

- Interfaces with destinations
- Multiplex BRM traffic from destinations and send to BRM Queue for processing
- Based on VCI demultiplex ABR or VBR traffic to appropriate destinations

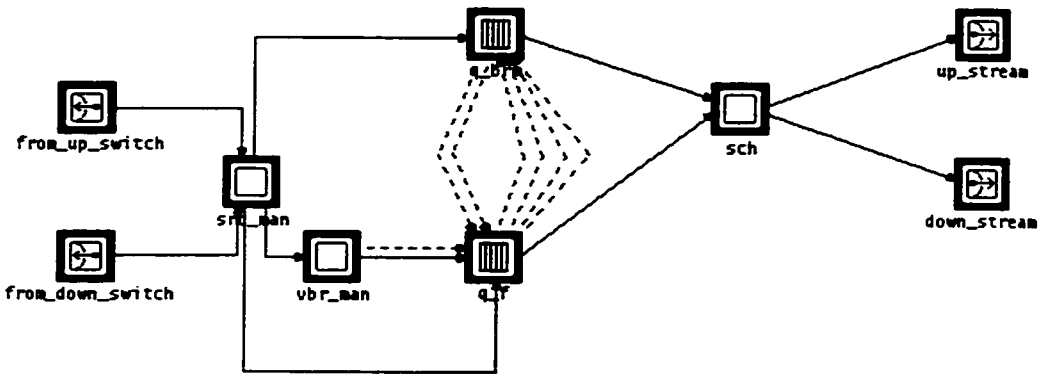


Figure 2-11: OPNET No-VSVD Core Switch Model

2.6.2.2 No-VSVD Core Switch Model

Model name: NoVSVD_TM_ER_Switch

Figure 2-11 shows the OPNET No-VSVD Core Switch model. This switch is a simplified version of the edge switch where the capability of interfacing with sources and destinations are removed for simplicity. The edge switch is only able to interface with one other switch. But, the core switch is able to interface with two switches. The following module is modified.

Source Manager (src_man)

- Directs VBR traffic from source to VBR Manager
- Directs ABR data and FRM cells from source to Forward Queue
- Directs ABR BRM cells from downstream switch to BRM Queue

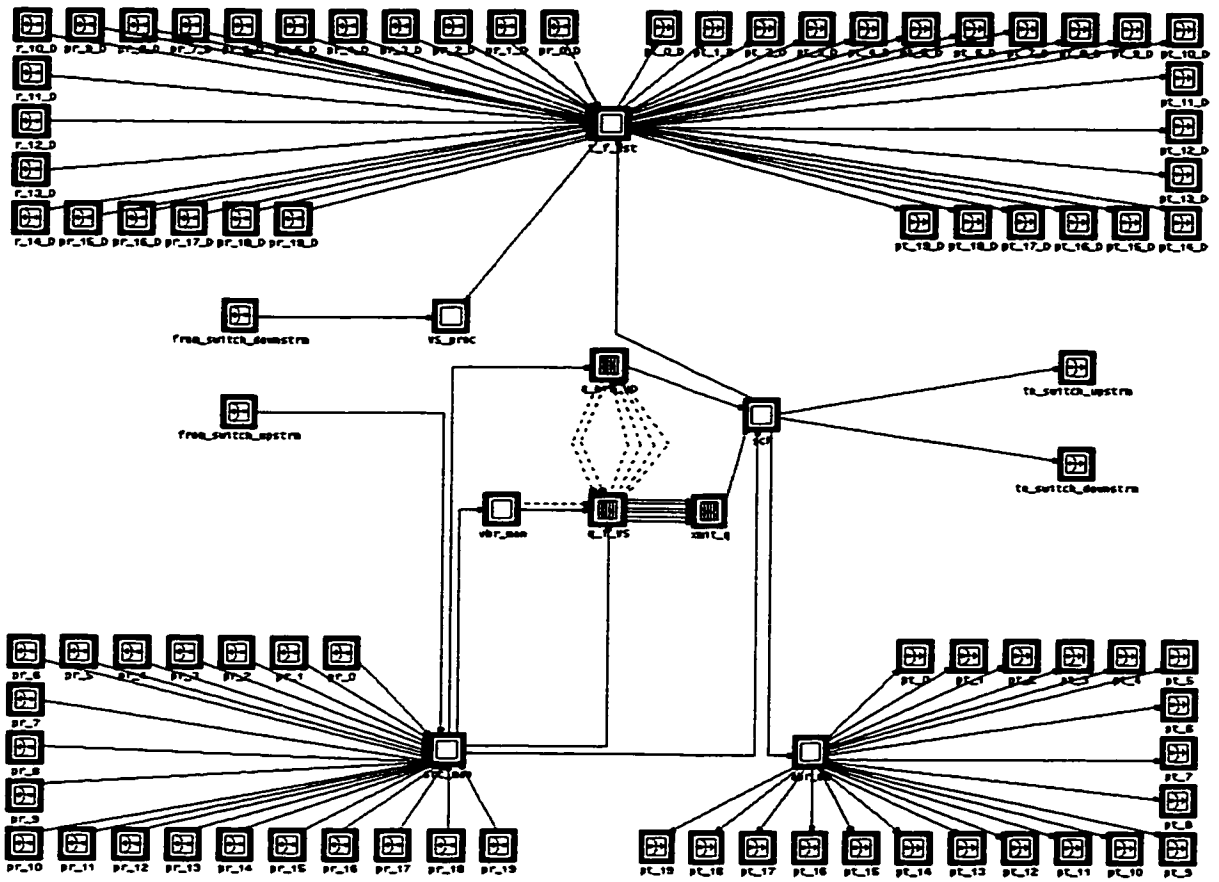


Figure 2-12: OPNET VSVD Switch Model

2.6.2.3 VSVD Switch Model

Model name: TM_ER_Switch_VSVD_fifo_xmit

Figure 2-12 shows the OPNET VSVD switch model. This switch is built from the No-VSVD edge switch model. It can interface with two other switches, sources and destinations. It consists of components that provide the functions of a VS and VD. The following modules are modified or added. These modules collectively implement the functional components described in Figure 2-7 and Figure 2-8.

Source Manager (src_man)

- Interfaces with both ABR and VBR sources
- Directs VBR traffic from upstream switch or source to VBR Manager

- Directs ABR data cells from upstream switch or source to Virtual Source Forward Queue
- Directs ABR FRM cells from upstream switch or source to Virtual Destination BRM Queue
- Attaches routing information to each cell based on a configurable table that allows control over routing on a per-VC basis

Virtual Source Forward Queue (q_f_VS)

- Consists of ABR per-VC sub-queues
- Each per-VC sub-queue autonomously schedules and send cells at the VC's ACR
- Forwards ABR cells to Scheduler to serialize for transmission
- Forwards VBR cell directly to Scheduler without storing to buffer
- Collects statistics for $M(n)$, $T(n)$, ABR queue size and link utilization
- Communicates $M(n)$, $C_{high}(n)$, queue control factor, ABR queue size to Virtual Destination BRM Queue
- Implementation for queue control function, and communicate scale factor to Virtual Destination BRM Queue
- FRM cell generation according to TM4.0 source rules

Virtual Destination BRM Queue (q_brm_VD)

- Fair share calculation
- Performs VSVD coupling
- Updates ER field of BRM cells received either from downstream switch or destination interfaces
- Tracks number of effective source
- Forwards cells to Scheduler to send out appropriate link

Virtual Source Processor (VS_proc)

- Provides BRM cell processing, acting as VS
- Obtains ER value from BRM cells in local VS-VD loop from downstream switch or destination

- Calculates ACR based on TM 4.0 source rules
- Updates ACR table
- Destroys BRM cells

Transmit Queue (Xmit_q)

- Consists of 2 sub-queues (1 for VBR and 1 for ABR)
- Maintains FIFO order of cells from VCs scheduled to transmit
- Weighted round robin servicing of sub-queues at line speed
- Forwards all cells to Scheduler to transmit out appropriate link

Chapter 3: Instability Analysis of a Rate Thresholding Scheme

This chapter provides a simulation analysis of a surveyed rate thresholding scheme to demonstrate its instability when it is put under the proper simulation model design that ensures TM4.0 compliance that we believe the author did not do (primarily TM4.0 source rule #10). The lessons learned from this simulation work partly created motivation for this thesis. The work presented in this chapter [HuYa99] represents our initial investigation of ABR Congestion Control. This initial experience on OPNET and analysis allows us to tackle more sophisticated simulation work with FASTRAC and VSVD.

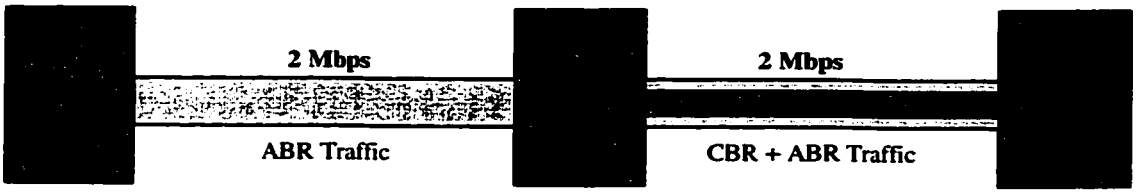


Figure 3-1: Simple ABR Network

3.1 OPNET Simulation Model

OPNET is used to implement the simulation model. The simulated network consisting of three nodes is in Figure 3-1. One unidirectional ABR connection is set up from the source node to the destination node. The source generates persistent traffic (i.e. non-bursty traffic that consumes all the available bandwidth). The links between the nodes are 2 Mbps. But the link between the switch and destination node is shared with a CBR connection from a different source node. The CBR connection consumes 1 Mbps of bandwidth when it is setup and active, causing the sharp bandwidth change stimulus. The adjustable round trip time (RTT) of the network is modeled by a propagation delay attribute on each of the links.

The control algorithm utilizes rate thresholding, intelligent marking (RM cell in ER mode), and per-VC queueing. TM4.0 ABR ER mode control mechanisms described in section 2.1.1 are implemented in the network with some simplification made. The

simplification includes: (1) only in-rate RM cells are generated, (2) switches are not allowed to generate out-of-rate backwards explicit congestion notification (BECN), and (3) no protocol support for loss and timeout situations. The simplication does not affect the desired tests to be performed. It simply remove ABR control mechanism support for extended idle periods common in very bursty traffic. The ABR parameters (explained in section 2.1.2) used are: MCR = 1 Mbps, PCR = 2 Mbps, ICR = 1.5 Mbps, NRM = 32, TRM = 100 ms, and MRM = 2.

The control algorithm under test is based on Krishnan's work [Kris97], as captured in EQ 3-1. The switch performs congestion avoidance using rate thresholding. An estimate of the arrival rate is calculated on the arrival of every RM cell. The estimated arrival rate is equal to the number of cells received during the inter-RM cell arrival period divided by the length of this period. If the ratio of the estimated arrival rate to the available bandwidth exceeds a settable threshold, ρ_T, congestion is declared. When in congested state, a fair share rate is calculated and inserted into the ER field of the RM cell. An estimated mean allowed cell rate (MACR) is maintained for the input port. The fair share amount is equal to a fraction of the MACR. On the reception of a returned RM cell, the source reduces its current ACR to the value of the ER field if this value is lower than the current ACR. Otherwise, it will increase its ACR by RIF*PCR.

$$MACR(t) = (1-\alpha)\cdot MACR(t^-) + \alpha \cdot CCR_j(t)$$

$$ER_j(t^+) = \begin{cases} \beta \cdot MACR(t) & \text{if } \rho > \rho_T \\ PCR_j(t) & \text{if } \rho \le \rho_T \end{cases} \qquad \text{(EQ 3-1)}$$

MACR(t)	the estimated mean allowed cell rate
α	exponential averaging weight
$CCR_j(t)$	current value of the CCR field for connection j
$ER_j(t)$	current value of the ER field for connection j
ρ_T	utilization threshold
ρ	utilization of resource at switch
β	rate decrease factor

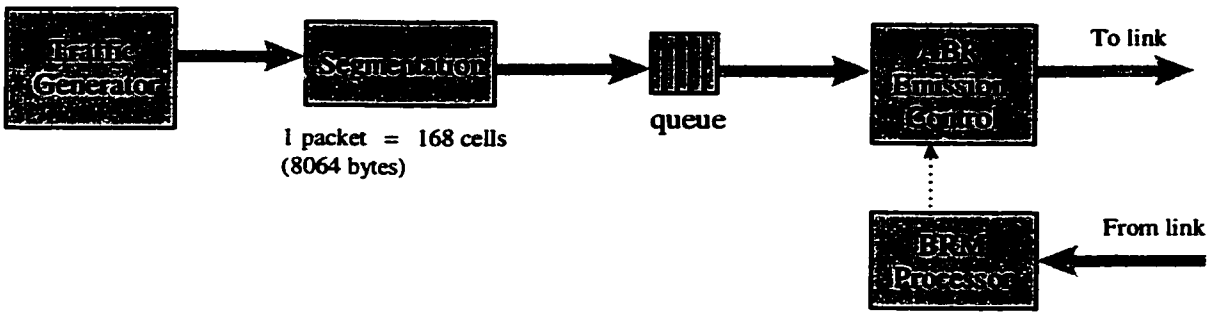


Figure 3-2: Block Diagram of Source Node

The ABR source node, the most complicated, implements most of the TM 4.0 source rules. These functions are distributed across cooperating modules as shown in Figure 3-2. The traffic generator produces packets that are then segmented into cells. The traffic generator module can be easily configured to produce bursty traffic or can be replaced by a practical traffic model. The ABR emission control process schedules the transmission of cells at the ACR. The BRM processor receives BRM cells, and extracts the ER value in them with which it adjusts the ACR appropriately.

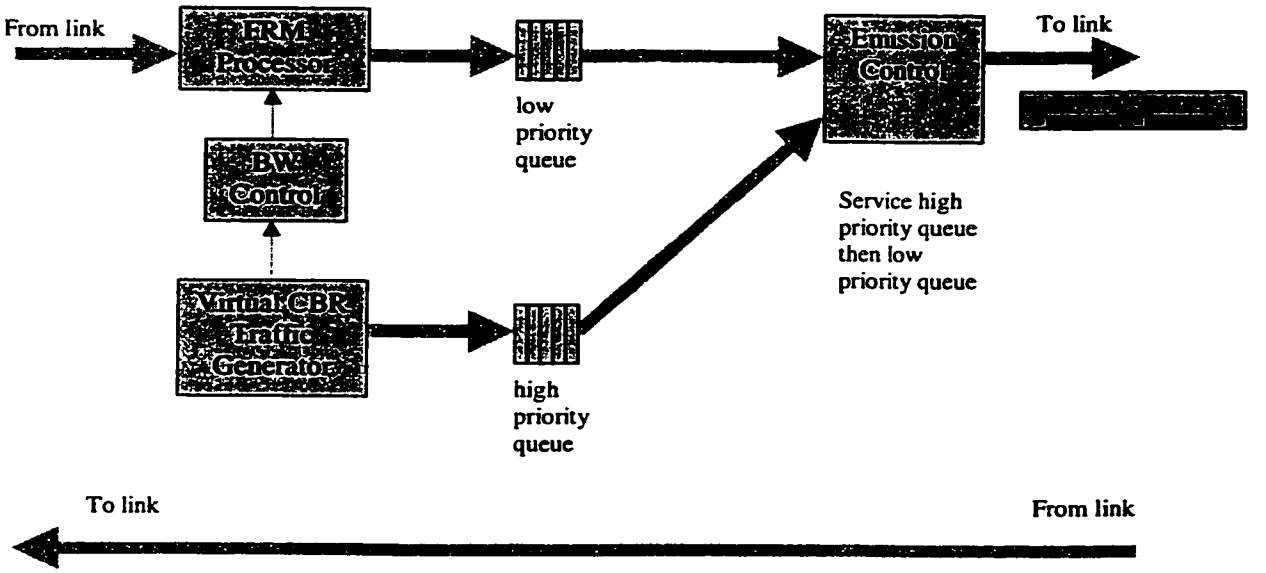


Figure 3-3: Block Diagram of Switch Node

The switch node, shown in Figure 3-3, primarily implements most of the control algorithm. The amount of free or available bandwidth is accounted by the BW control process by subtracting any bandwidth reserved for CBR connections from the overall link bandwidth. This free bandwidth is divided up equally to all ABR connections to cater their MCR requirements. The FRM processor provides the responsibilities of estimating the arrival rate, rate thresholding, and changing the ER field in passing-by FRM cells. Data cells are passed directly through by the FRM processor.

The virtual CBR traffic generator produces interruptions in available BW at the output link shared between CBR and ABR connections by periodically setting up and taking down one CBR connection. The result is a series of square pulses of bandwidth consumption. These pulses introduce deterministic sharp changes in BW available for the ABR connections. ABR traffic is stored in a low priority passive FIFO queue and CBR traffic is stored in a high priority passive FIFO queue. The emission control process schedules the transmission of CBR and ABR traffic out the link using weighted round robin servicing of the queues where the higher priority is placed on the CBR queue. The destination node simply discards data cells and turns FRM cells around as BRM cells.

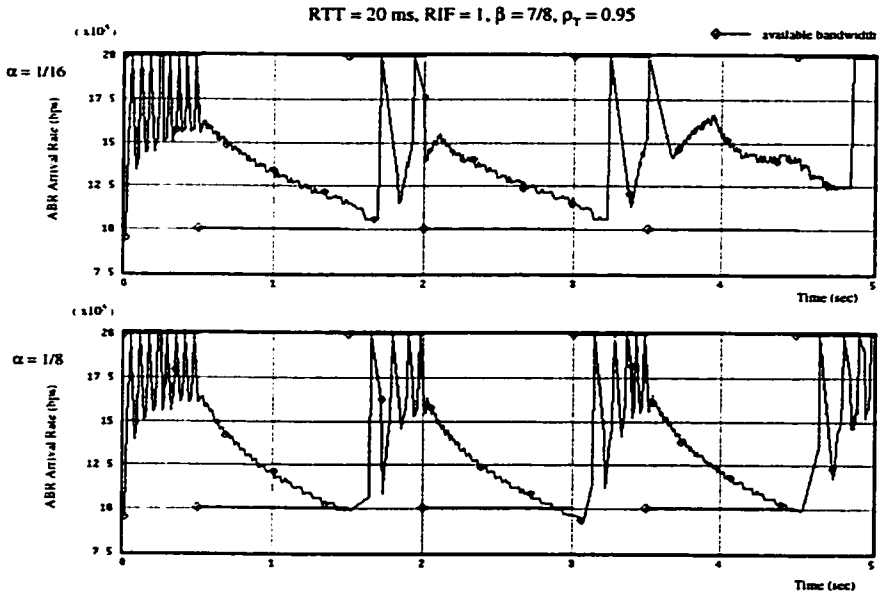


Figure 3-4: ABR Arrival Rate for Various α

3.2 Performance Evaluation

Three main activities are used to evaluate the performance of this algorithm. Parameter tuning of the algorithm is performed to examine the algorithm's sensitivity. TM4.0 compliance test is demonstrated. And the effect of long propagation delay is observed. Figure 3-4 shows the effect of α on the arrival rate to give a pictorial indication of how well the algorithm controls the source transmission rate.

The horizontal line segments overlaid on the ABR arrival rate curves show the available bandwidth at instances in time. At zero second, the available bandwidth is 2 Mbps. The CBR call starts at 0.5 second and consumes 1 Mbps of bandwidth for 1 second and repeats every 0.5 second, producing square unit pulses of bandwidth loss to the ABR connection. At 0.5 second, the available bandwidth is reduced to 1 Mbps. The curve for the larger α is able to decrease its rate faster. According to EQ 3-1, the larger α is the more weight is placed on the current value of CCR, and the faster the decay is on the weight of the previous CCRs, thus following the changing CCRs closer. When congestion occurs, the source decreases its transmission rate to a fraction of the MACR. A larger α would mean that the congestion state would be exited faster when there is a sharp drop in bandwidth since the transmission rate would be reduced to the new bandwidth level faster. However, getting out of congestion state quicker gives the source a chance to increase its rate earlier and enter congestion state more frequently. The observation is the presence of more oscillations in the arrival rate.

Oscillations are present even when the whole 2 Mbps of bandwidth is available because of the congestion avoidance algorithm described in EQ 3-1. Oscillations like these give the end user a perception of poor level of QoS is provided by the network. Since ρ_T is set to 0.95 and RIF = 1, the switch is desired to be quite highly utilized. This means that the switch does not remain in congested state for long where the ER value of passed RM cells are reduced according to EQ 3-1. According to TM4.0, if the switch is not in congestion state it is *not* allowed to modify the content of the ER field which by default is set to any rate up to the PCR by the source (TM4.0 source rule #10). In our experiment, we set the ER field to the PCR which is typically what most implementation would do.

Therefore, when the congestion state is exited the source immediately adjusts its transmission rate to the PCR as indicated in the unmodified ER field and congestion is declared when the cells transmitted at the PCR arrives at the switch. The result is the congestion state is entered and exited continuously. This is a sign that the algorithm originally designed did not factor in the TM4.0 source rule #10. The frequency of the oscillations is lower after the rise in bandwidth at 1.5 second. This effect is caused by the processing of fewer RM cells since the source is transmitting at a lower rate. Thus there are fewer instances of feedback communication. The effect of propagation delay is also evident in these graphs. It causes a time delay in the response of the arrival rate to the changing bandwidth.

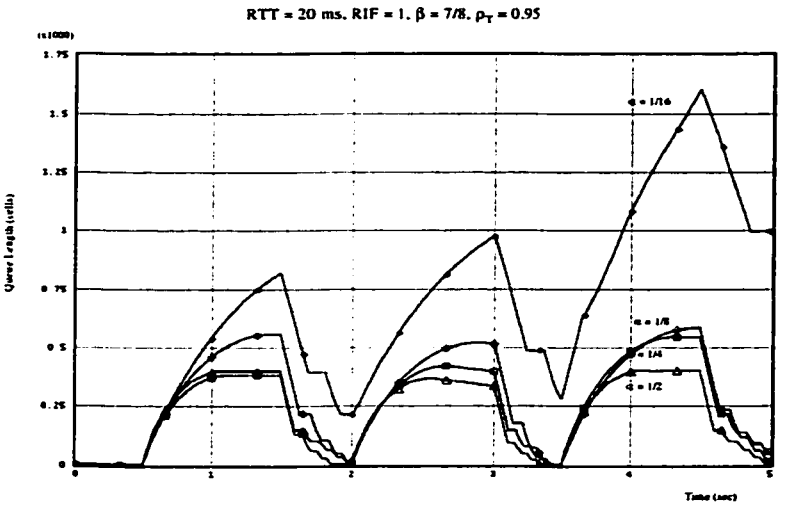


Figure 3-5: Transient Queue Length for Various α

The transient queue length is observed in Figure 3-5. Despite oscillations, Figure 3-4 shows that the larger α is the better the arrival rate is at adapting to bandwidth changes. This produces smaller queue lengths. For a larger α, the queue length reaches zero several times, resulting in under utilization and causing lower throughput. For $\alpha = 1/16$ (the author's suggested value), the queue growths out of control. The smallest queue requirement is observed for $\alpha = 1/2$. This value of α represents a decay rate of 50 percent on the weight of the MACR calculation for each CCR going backwards in time.

A maximum boundary on queue length is a major consideration to the usefulness of an algorithm. It provides an estimate of buffer requirement in designing a switch to the algorithm.

The effect of β on the transient arrival rate has been found to be inversely proportional to the effect of α on the transient arrival rate. The effect of β on the transient queue length is also inversely proportional to the effect of α on the transient queue length. Although the author recommended value $\beta = 7/8$ produces the highest throughput, the queue seems to grow out of control.

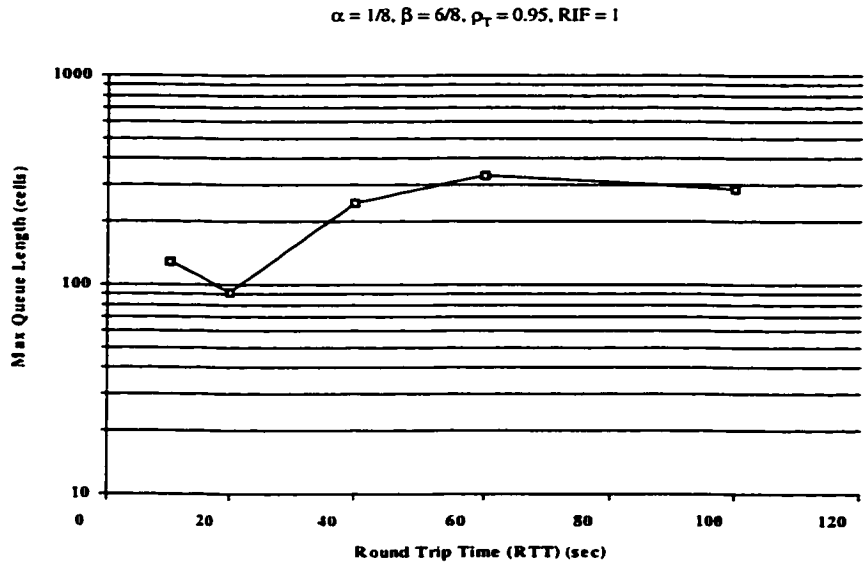


Figure 3-6: Maximum Queue Length vs RTT

Theoretically, long RTTs cause longer delays in the feedback loop which cause the source to adjusts its transmission rate less timely to the changing network conditions. When there is a sharp drop in available bandwidth, the source will continue to transmit at a higher rate longer before reducing its rate. When there is a sharp rise in available bandwidth, the source will continue to transmit at the lower rate longer before increasing its rate. The net result should be larger maximum queue lengths. The theoretical

expectation is experimentally fulfilled for RTT = 20, 40 and 60 seconds as shown in Figure 3-6. For RTTs outside this range, the experimental results deviate from theory and become unpredictable. This leads us to a conclusion that given the set of values for α, β, and ρ_T the control algorithm is predictable only under certain bounds. The optimal operating point for these parameter values is at RTT = 20 seconds. Further investigation is necessary to make any conclusion on the predictability of the effect of RTT for this control algorithm.

3.3 Remarks

The simulation analysis provides an in-depth look at the control algorithm's practicality. From this, a number of questions have been raised regarding the theoretical development behind the algorithm. Although some general conclusions are made regarding the α and β parameters of the control algorithm. The suggested constant values for them proved ineffective at certain values of α, β, and RTT. The questions raised are what are the guidelines for choosing them and what is their significance in the control algorithm. The results also show that understanding TM4.0 ABR traffic management mechanisms is essential to design an effective control algorithm. For a network of ER-based switches, a RIF factor of one is suggested to promote faster convergence [JaFa96]. As demonstrated in the simulation results, a RIF factor of one combined with TM4.0's source rule #10 causes tremendous oscillations. Extra attention to this subtlety is required when designing a control algorithm. The algorithm also does not factor in propagation delay. As demonstrated, propagation delay greatly affects the engineering of the switch node's queue size. Engineering aspects such as maximum queue length, throughput and stability were exploited. These aspects provided a good basis for evaluating the algorithm.

In this chapter, instabilities are demonstrated on a surveyed control algorithm when it is put under test in a simulation model that adheres to TM4.0 ABR control protocol. The results conclude that understanding of the operation involved in ABR flow control and effectively modeling it using control theory is key to successfully design an ABR control algorithm that is practical in implementation and performance.

Chapter 4: Performance Evaluation of Bottleneck Configurations

In this chapter, we shall investigate a scenario sufficient to test the response of the FASTRAC algorithm under homogeneous traffic condition. It is used to demonstrate capabilities of FASTRAC and to verify the theoretical development in section 2.4 in order to highlight the prudent effort put into the FASTRAC design to integrate TM4.0 compliance, practical switch design, and feedback loop control theory. Performance is evaluated based on fairness, stability, and responsiveness which together are key elements making up the apparent ABR quality of service to the user. The performance analysis will also demonstrate the good simulation design where assumptions and initial conditions are stated and results are justified.

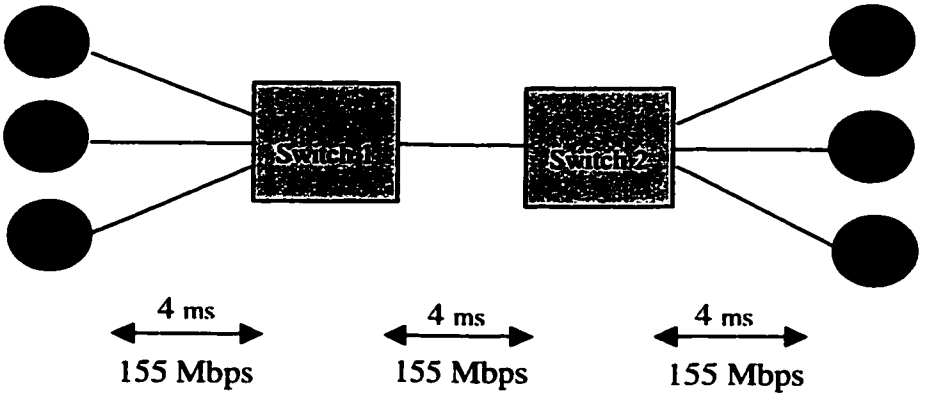


Figure 4-1: Bottleneck Configuration

4.1 Bottleneck Configuration

The Bottleneck configuration [AwOu99a] allows us to study the effect of a bottleneck in bandwidth occurring on a node where a set of VCs of equal propagation delay passes through. This simple two node network, connected by OC3 links, with three persistent sources (non-bursty sources that will use up all allocated bandwidth) is shown in Figure 4-1. Both switches in this configuration use the No-VSVD Edge Switch model described in section 2.6.2.1. Switch 1 is the bottleneck switch. Switch 2 is the destination switch. Sources are labeled sA, sB, and sC with corresponding destinations labeled dA, dB, and

dC. All tests performed on this configuration have ABR source parameters for RM cells set to default values (NRM = 32, TRM = 100 ms, MRM = 2). The desired link utilization, ρ of all links is at 0.95.

Configuration parameters:

Number of sources: 3

MaxFRTT = [2 * (4 + 4 + 4) ms] = 24 ms

Switch Parameters:

Dynamic source tracking: on

Control Gain α: 1/[d(n)+1]

Source	ICR (Mbps)	PCR (Mbps)	MCR (Mbps)	Start Time (second)	Stop Time (second)
A	50	155	10	0	1.5
B	40	155	30	0.4	0.8
C	45	155	50	0	1.5

Table 4-1: Source Traffic Parameters for Bottleneck Configuration

Dynamic source tracking is enabled because the number of active connections varies with time in all experiments using this configuration. The control gain is set to the value derived in Appendix B.2 for stability condition. The characteristics of the sources are described in Table 4-1. Cell rates are arbitrarily chosen to test FASTRAC. Source B has a delayed start and early finish to create a change in the number of active sources and also to generate a change in ABR bandwidth request.

4.2 Analysis of FASTRAC's Operation

In this section we observe the performance curves of FASTRAC's response to a bottleneck and verify FASTRAC's operation. All simulations performed have the optional queue control function disabled to allow only the FASTRAC algorithm to be

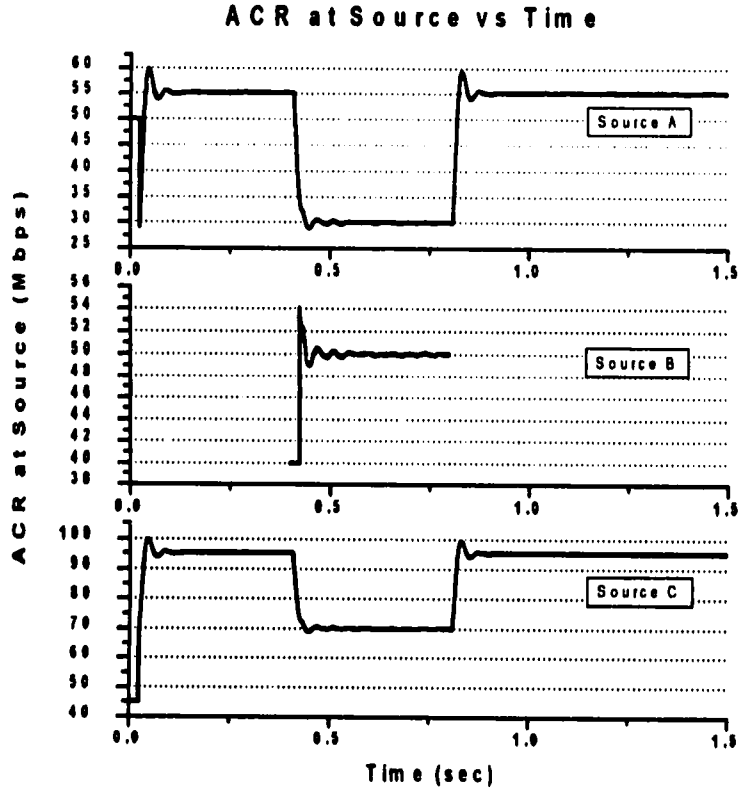


Figure 4-2: Bottleneck Configuration: ACR at Source

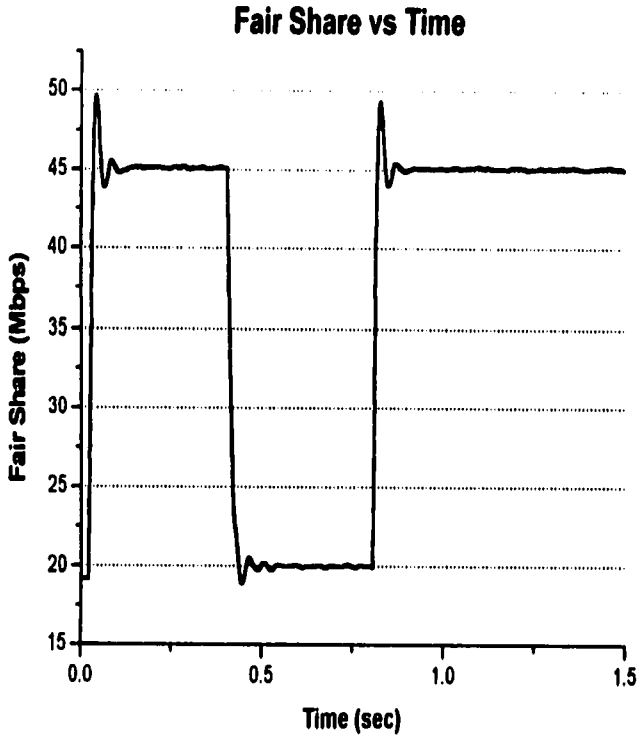


Figure 4-3: Bottleneck Configuration: Fair Share at Switch 1

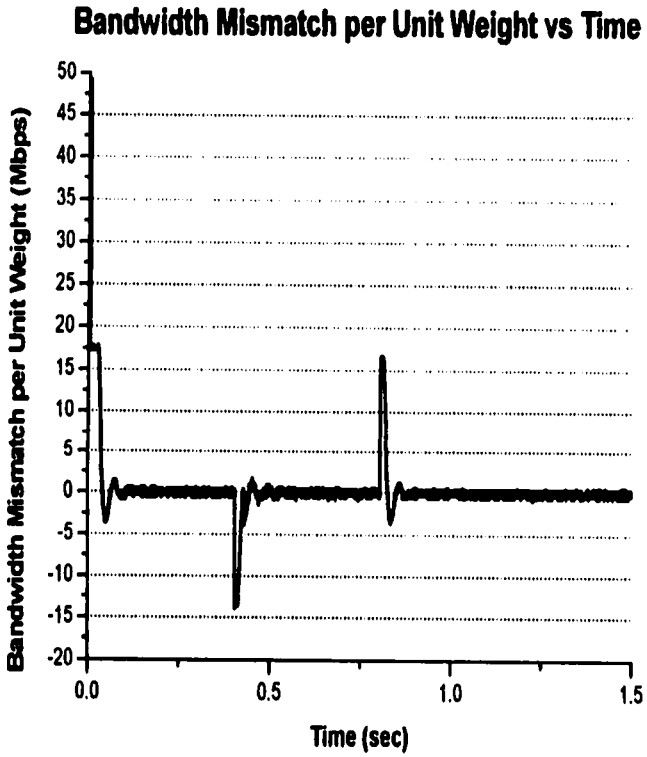


Figure 4-4: Bottlneck Configuration: Bandwidth Mismatch at Switch 1

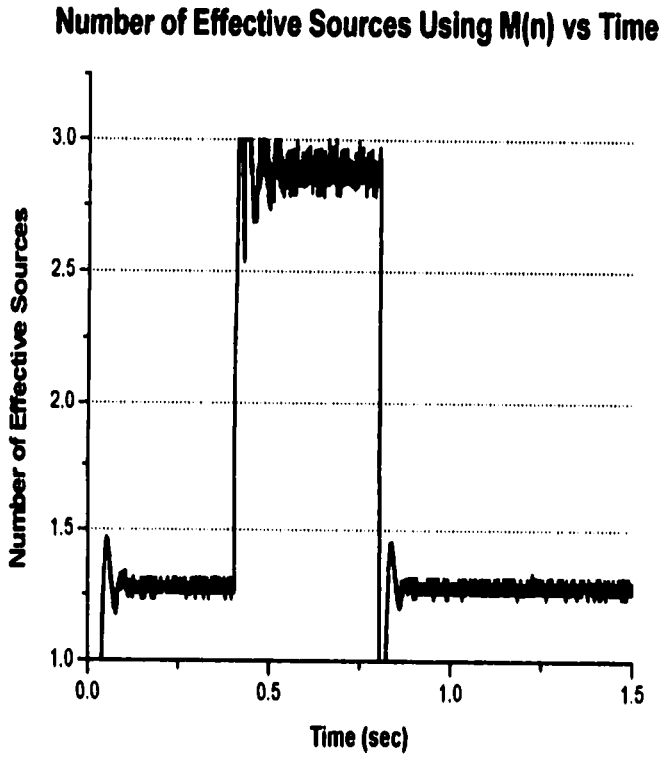


Figure 4-5: Bottleneck Configuration: Number of Effective Sources Using M(n)

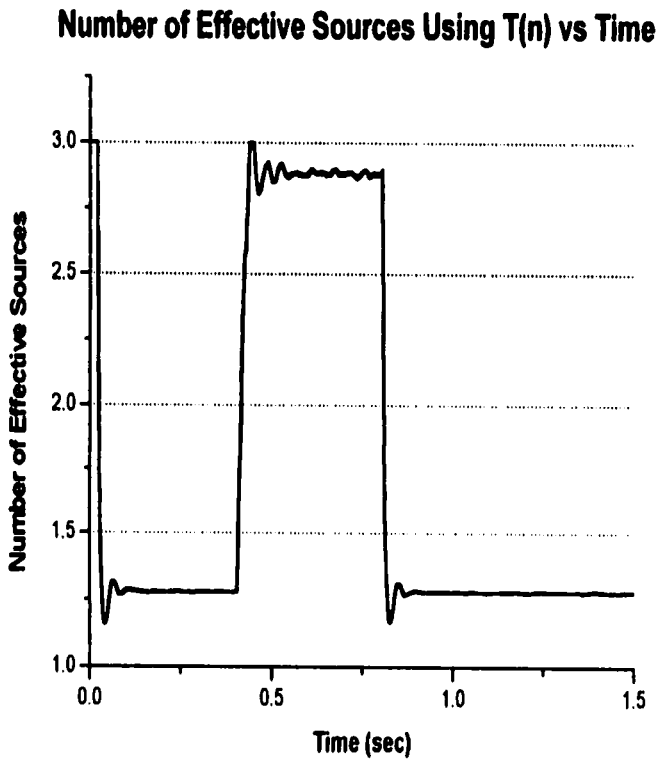


Figure 4-6: Bottleneck Configuration: Number of Effective Sources Using T(n)

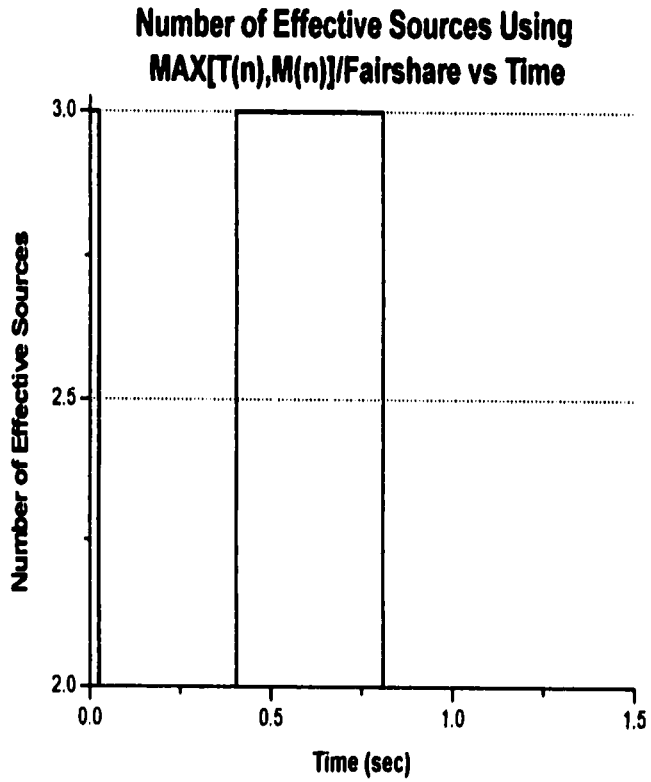


Figure 4-7: Bottleneck Configuration: Number of Effective Sources Using MAX[M(n), T(n)]/Fair share

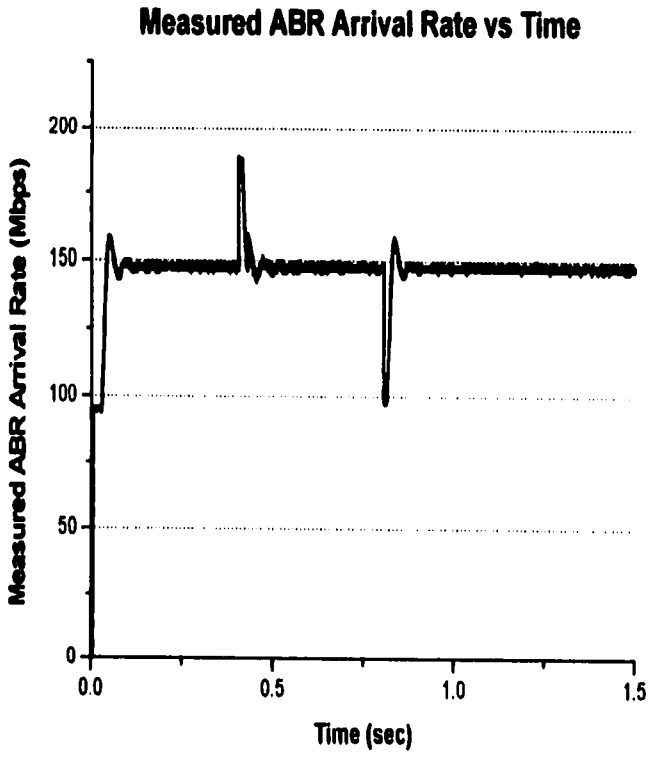


Figure 4-8: Bottleneck Configuration: Measured ABR Arrival Rate at Switch 1

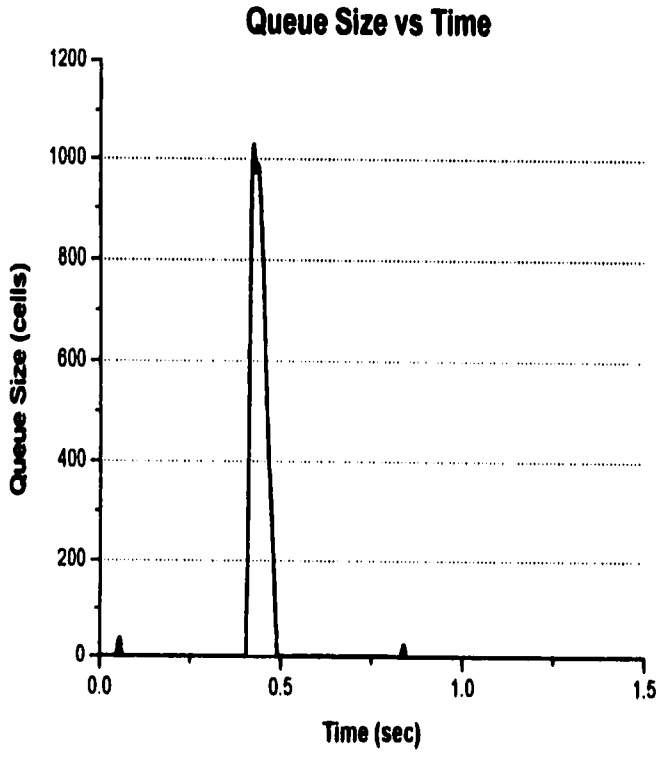


Figure 4-9: Bottleneck Configuration: Queue Size at Switch 1

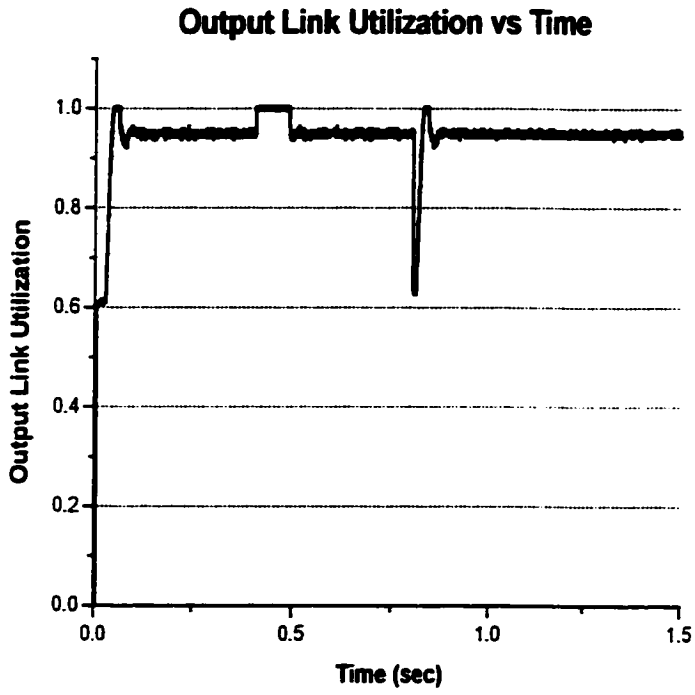


Figure 4-10: Bottleneck Configuration: Output Link Utilization at Switch 1

studied. All observations made below are at the bottleneck switch where we can study FASTRAC's response to congestion.

In this scenario, sources A and C start at zero second time, and source B starts at 0.4 second and ends at 0.8 second. Based on step 9 in section 2.4.6, before 0.4 second and after 0.8 second, the fair share (error term) should be 43.9 Mbps (ie. [(0.95 * link capacity) – (MCR_A + MCR_C)]/number of active sources = [(0.95 * 155.52) – (10 + 50)]/2). Therefore, the ACR for source A should be 53.9 Mbps (ie. MCR_A + fair share = 10 Mbps + 43.9 Mbps). Likewise, the ACR for source C should be 93.9 Mbps. From Figure 4-2 and Figure 4-3, observe the ACR of a source is equal to its MCR + fair share. The fair share is the excess BW determined by the error term calculation in on step 9 in section 2.4.6. Between 0.4 and 0.8 second, the fair share is 19.25 Mbps. The corresponding ACRs are 29.25 Mbps, 49.25 Mbps, and 69.25 Mbps for source A, B, and C respectively. These results demonstrate fairness (ie. equal division of excess

bandwidth among the sources) and MCR guarantees. These aspects are key performance requirements for ABR service since the purpose of ABR to divide up excess bandwidth to connections of the ABR service category. The sharp spikes on the curves at points where there is a sharp change in traffic flow provides a view of how quick FASTRAC converges to steady state. These spikes are directly affected by the control gain. Increasing the control gain would increase the rate of change in the curves, at the expense of more oscillations.

Alternatively, another way to look at FASTRAC's response is to observe the error term as shown in Figure 4-4. This figure shows the mismatch between the service rate and the arrival rate divided by the total weights (i.e. the total number of active connections). A positive value means that there is excess bandwidth of that value is available to each ABR connection. A negative value indicates the amount of bandwidth each connection must release because their total arrival rate is exceeding the total available bandwidth. The difference on the time axis where the mismatch value begins to be non-zero until it returns back to zero gives an indication of how long FASTRAC require to bring the changing traffic flow back into a controlled state. The long but narrow spikes show that FASTRAC's response time is very short in order to correct the large amount of bandwidth mismatch.

Dynamic source tracking (described in section 2.4.5) performed by Switch 1 is demonstrated in Figure 4-5, Figure 4-6, and Figure 4-7. Both techniques of using $M(n)$ or $T(n)$ are shown to be equally effective in determining the number of active sources because the differences in the level of noise in the estimation between the two is small. Since the two schemes converge to the same value in steady state, taking the MAX and applying an EWMA filter on their estimated values provide faster convergence and less noisy estimations. Dynamic source tracking is also part of the FASTRAC algorithm.

FASTRAC demonstrates an excellent response in the Bottleneck configuration. From a practical observation, both input and output links at the bottleneck switch are highly utilized as shown in Figure 4-8 and Figure 4-10. The queue size presented in Figure 4-9

is in perfect control. At the onset of congestion, the queue grows to a maximum of roughly 1000 cells, and is brought back down to zero within 0.1 second. To give an indication of how effective and smooth FASTRAC is, remember that during 0.1 second time 36,555 cells can enter the switch at OC3 rate. However, the queue size does not show any signs of those cells and no oscillations. Without ABR, the two original connections would not have been able to adapt to a lower available transmission rate when source B started transmitting. Also, they would not have been able to regain use of the full link bandwidth when source B became inactive. Under this scenario, FASTRAC demonstrates to be very responsive, fair, stable, and efficient to traffic rate changes.

4.3 Optional Queue Control Function

In this section we observe how the optional linear queue control function in Figure 2-4 can be used to maintain the queue size of the bottleneck switch to a certain level when congestion occurs. All configuration parameters described in section 4.1 are used except that the queue control function is enabled.

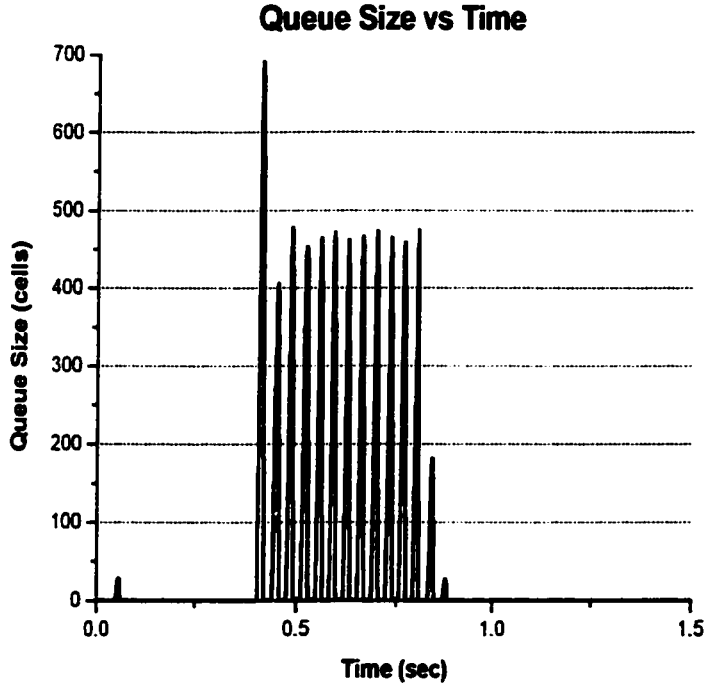


Figure 4-11: Bottleneck Configuration: Queue Size with Queue Control at Switch 1

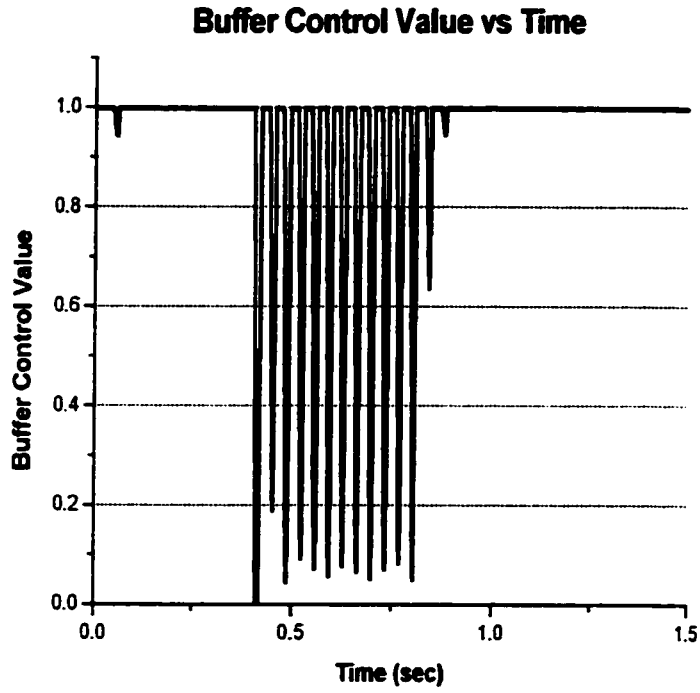


Figure 4-12: Bottleneck Configuration: Queue Control Function at Switch 1

Figure 4-11 shows the transient queue size of the bottleneck switch when linear output queue control function is applied to scale down *T(n)*. The goal of the experiment was to limit the queue size to 500 cells. However, this is a difficult task. T_h has to be set down to 0 in order to achieve this. Based on these values, the available capacity, *T(n)*, is unaffected at a queue size of 0 cells and is scaled to 50 % when the queue size is 250 cells, and 0 % when the queue size is 500 cells, with a linear relationship. However, at the onset of congestion (0.4 second), the queue control was only able to reduce the queue size from approximately 1000 cells to 700 cells. The queue control took some time to have an effect due to propagation delay experienced by the BRM cells. To understand the oscillations we need to look at the transient behaviour of the queue control function as shown in Figure 4-12. The queue control function value is close to unity when the queue size is close to zero, and approaches a value of zero when the queue size approaches the value *B*. The value of the queue control function changes constantly between the two

extreme points. An upside down picture of the queue control function should be very similar to the graph of the queue size.

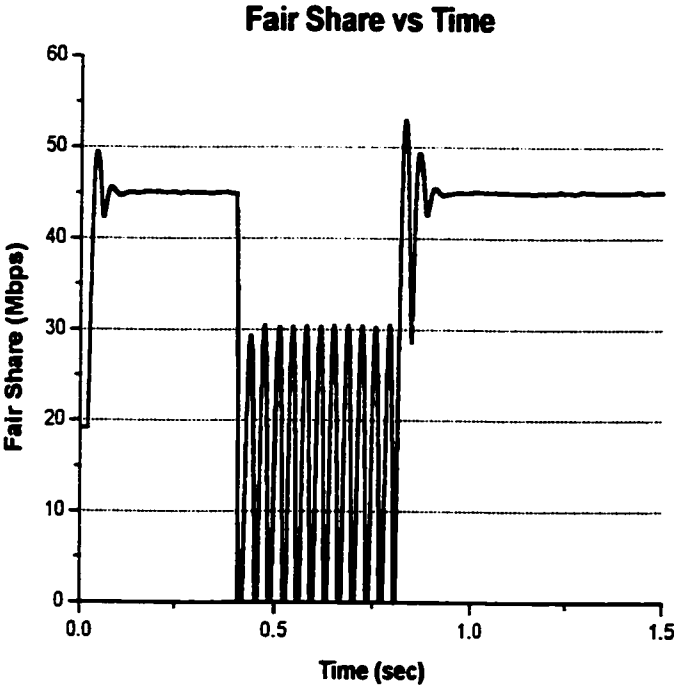


Figure 4-13: Bottleneck Configuration: Fair share with Queue Control at Switch 1

The queue control function affects the available capacity in calculating the error term. Therefore, the same oscillations should be exhibited in the fair share value as illustrated in Figure 4-13. The results show that on average the queue control function has to bring the fair share value about 5 Mbps below its original value without the effect of queue control during congestion in order to maintain the maximum queue size below 500 cells.

4.4 Remarks

We were able to demonstrate FASTRAC's responsiveness, fairness, stability, and efficiency to traffic rate changes because of good simulation design and the good ABR control model that FASTRAC is built on. Results are traceable back to the control model and are conclusive. Also, various experiments we have performed showed that it is very difficult to choose T_h and B parameters for the optional queue control function to obtain

the desired maximum queue size. The linear queue control function is not aggressive enough. Better performance is expected with an exponential decay queue control function. A good hypothesis to test is the queue control function when made dependent on the gain factor converges more quickly, and with smaller oscillations. The current proposal of integrating an optional queue control function into FASTRAC shows some motivation for practicality. Having a queue control function would definitely ease the process of engineering the switch to obtain the desired buffer requirement. The topic of designing a more sophisticated queue control function is outside the scope of this study.

Chapter 5: Performance Evaluation of Satellite Configurations

In this chapter we shall investigate the effect of long propagation delay on FASTRAC and FASTRAC with VSVD. Long propagation delay is typically associated with a satellite hop in the VC's span. This hop causes long delay in the feedback loop which increases the value of *d(n)* in EQ 2-2. This increased delay affects the control gain, α in step 9 of section 2.4.6. Section 5.1 describes the satellite network that is used in this chapter. The satellite configuration allows us to demonstrate the benefit of having VSVD when there is one long propagation segment in the network. Section 5.2 presents the performance results when the satellite network is set up as the LEO (Low Earth Orbit) configuration. Section 5.3 presents the performance results when the satellite network is set up as the GEO (Geostationary) configuration. The LEO and GEO configurations differ from each other only in the amount of propagation delay. Results presented in the satellite configuration are used to demonstrate particular objectives. Many scenarios with various randomness were used to test FASTRAC with VSVD.

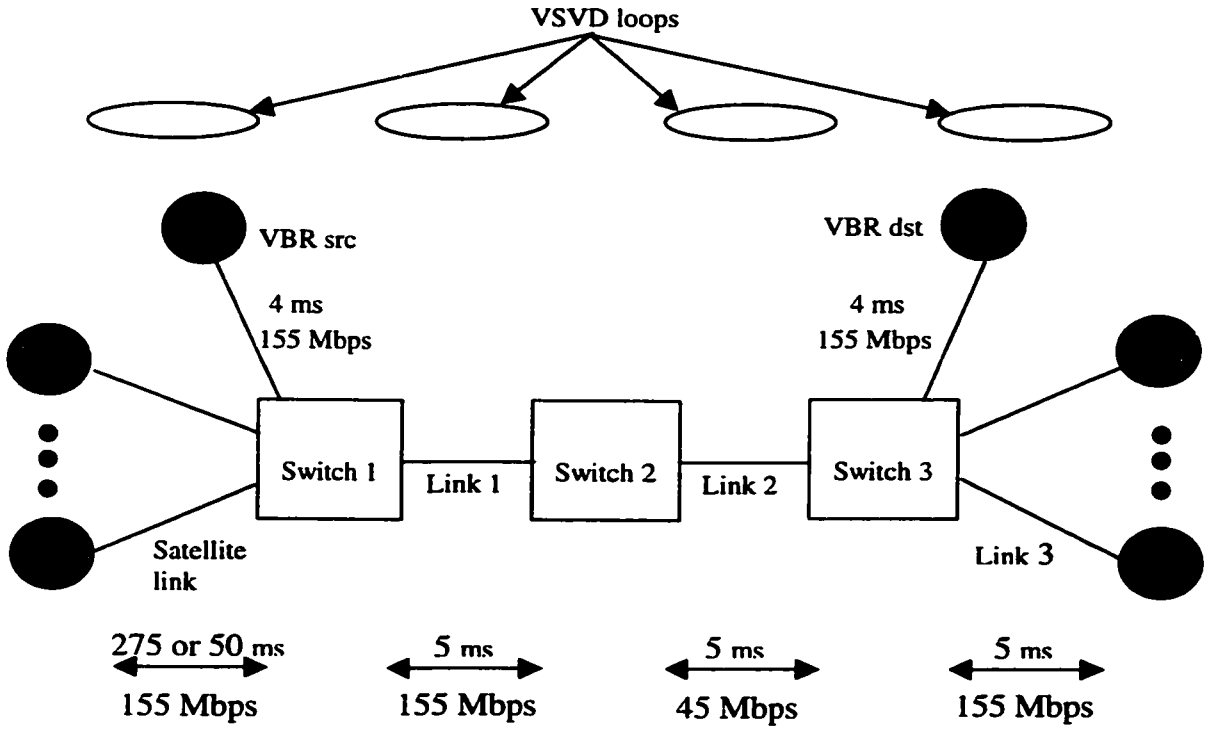


Figure 5-1: Satellite Configuration

5.1 Satellite Configuration

The satellite configuration is used as a test bed for long propagation delay and VSVD. It consists of satellites that are connected via long propagation delay satellite links to a satellite switch which is connected the terrestrial network as shown in Figure 5-1. The sources (sA to sE) represent the satellites. Switch 1 is the satellite switch. Switch 2 is the bottleneck switch that has a bottleneck link 2 with 45 Mbps capacity. The destinations (dA to dE) are connected to Switch 3. All links, accept link 2, are 155 Mbps. The VBR source and destination pair is configured inactive by default. Two configurations can be created from the satellite configuration by changing the propagation delay of the satellite link. The LEO configuration has a satellite link delay of 50 ms. The GEO configuration has a satellite link delay of 275 ms. The satellite configuration is found in [GoCa98], where it is used to test the performance of VSVD with switches employing ERICA+[1]. We chose to build the same configuration to allow us to compare results obtained from using FASTRAC with results obtained from using ERICA+. The maxFRTT parameter for FASTRAC is 130 ms (2 * (50 + 5 + 5 + 5) ms) or 580 ms for the LEO and GEO configuration respectively. All tests performed on the satellite configuration have ABR source parameters for RM cells set to default values (NRM = 32, TRM = 100 ms, MRM = 2). The desired link utilization, ρ, of all links are set 0.95.

In the satellite configuration without VSVD, the edge switches, Switch 1 and Switch 3, use the No-VSVD edge switch model, and Switch 2 uses the No-VSVD core switch model. In the satellite configuration with VSVD, all switches use the VSVD switch model. Refer to section 2.6.2.1 for switch model details.

Under VSVD, the configuration is segmented into 4 VSVD loops as shown. The *maxFRTT* parameter in the delay, *d(n)*, is kept the same in all loops. According to TM4.0, this value is accumulated by measuring the delay of response at connection setup and applies to the entire ABR circuit.

---

[1] ERICA+ uses a combination of rate thresholding and queue size thresholding, similar to FASTRAC with the queue control option.

In the previous chapter we demonstrated that FASTRAC is a stable and fair algorithm. However, our initial investigation using the satellite configuration with the original control gain derived from the stability analysis performed in appendix B.2 produced results that did not converge to a steady state. Oscillations occur when there are large sudden large bandwidth changes under long propagation delay present in the satellite configuration. In this chapter we use a modified control gain, suggested by Nortelnetworks' FASTRAC team. This control gain is designed to be able to adapt to large bandwidth changes under long propagation delay.

5.2 LEO Configuration Results

The LEO configuration allows us to demonstrate the benefit of VSVD. Long propagation delay means that when a bottleneck occurs the queue would typically grow in size by the bandwidth-delay product until the control algorithm's action takes effect after a period equal to the feedback delay. This queue build up would occur at the bottleneck switch regardless of where the long propagation delay link is in the VC span. From knowledge of VSVD (explained in details in section 2.2), we understand that VSVD breaks the ABR feedback loop into smaller segments. We stipulate that the net effect of applying VSVD to this configuration is the localization of the effect of long propagation delay on queue size to the switch that receives traffic directly from the long propagation delay link. The following sub-sections present performance results to demonstrate this stipulation and test FASTRAC with VSVD support.

5.2.1 VSVD vs No VSVD

We analyze and compare here the performance of FASTRAC with and without VSVD. The optional queue control function is disabled to avoid modifying FASTRAC's response with an optional enhancement feature. Also, dynamic source tracking is disabled to remove further estimation errors inherent with it. We are interested in

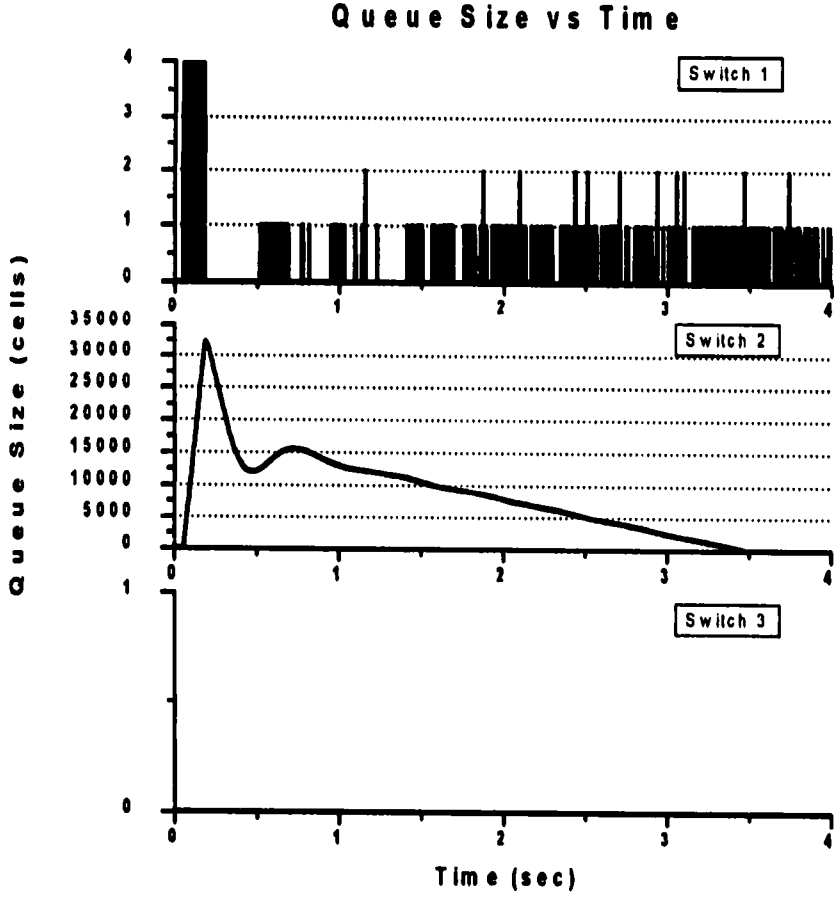


Figure 5-2: LEO Configuration with No-VSVD: Queue Size

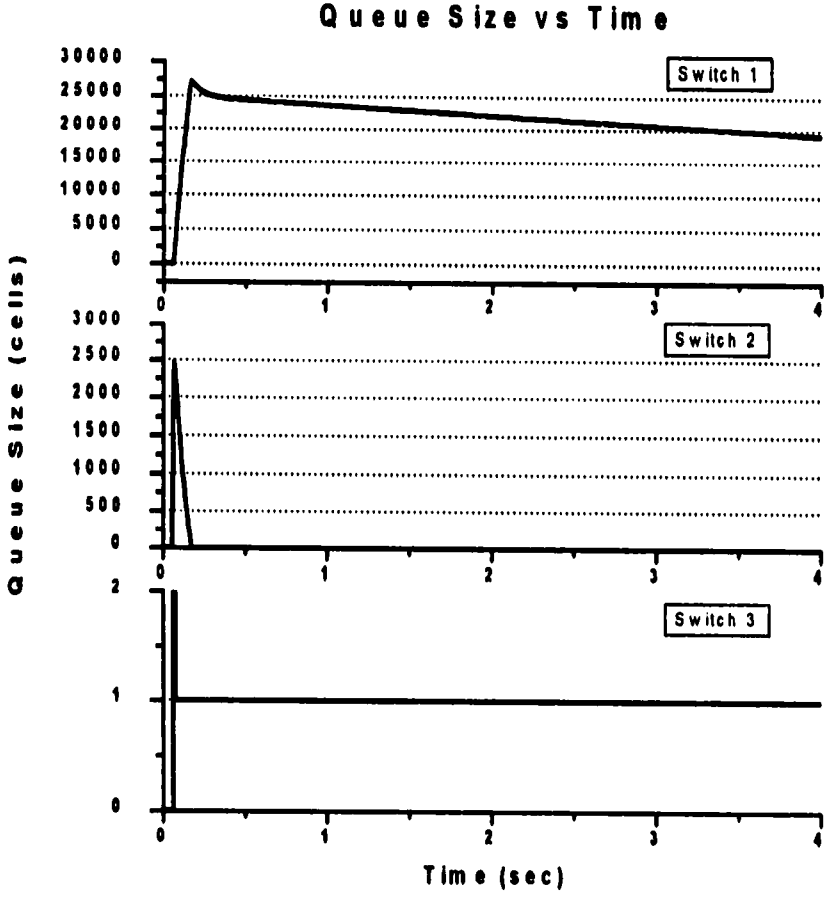


Figure 5-3: LEO Configuration with VSVD: Queue Size

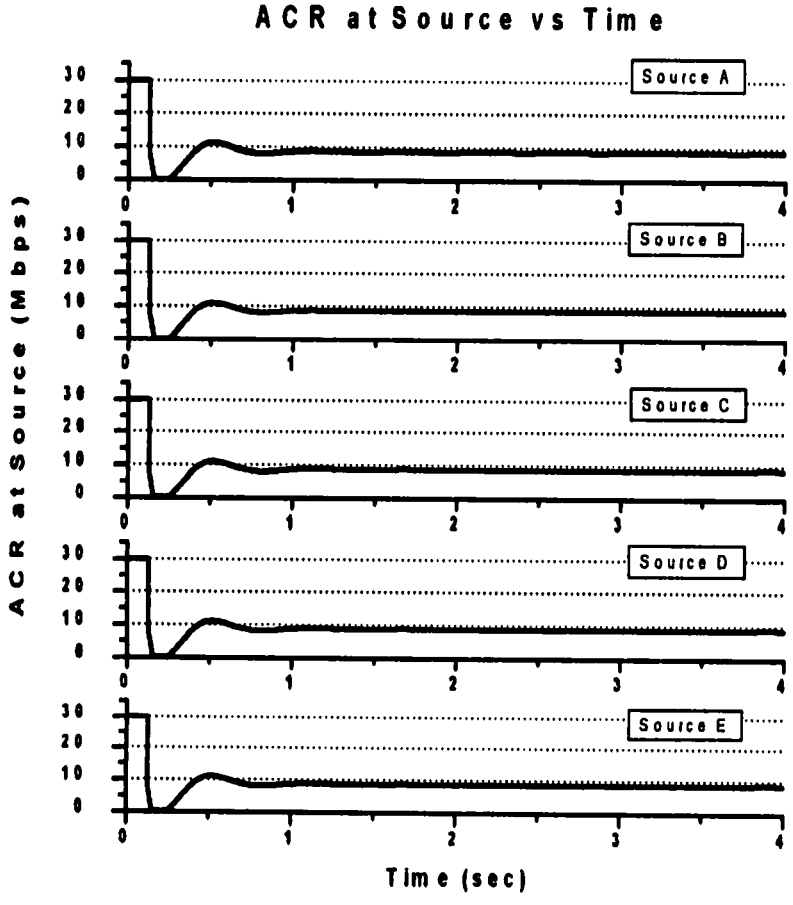


Figure 5-4: LEO Configuration with No-VSVD: ACR at the Sources

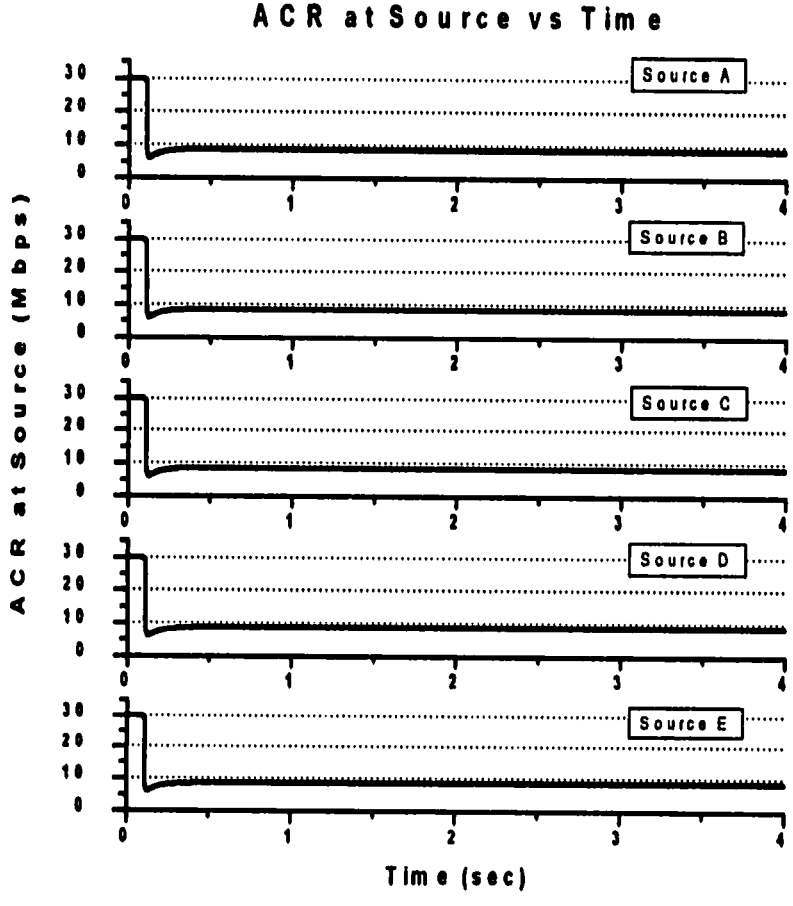


Figure 5-5: LEO Configuration with VSVD: ACR at the Sources

observing the behaviour of the basic FASTRAC algorithm with and without VSVD. The configuration parameters we used are as follows:

Configuration parameters:

Number of sources: 5

Switch Parameters:

Queue control function: off

Dynamic source tracking: off

Control Gain α: 1/[2d(n)+1]

Source	ICR (Mbps)	PCR (Mbps)	MCR (Mbps)	Start Time (second)	Stop Time (second)
A to E	30	155	0.1	0	4

Table 5-1: Source Traffic Parameters for LEO Configuration

The characteristics of the sources are described in Table 5-1.

With the specified source traffic parameters, the onset of congestion occurs at Switch 2 where the 150 Mbps of combined traffic from the five sources reaches the 45 Mbps link. This congestion causes the queues in Switch 2 to build up until FASTRAC reduces the source transmission rate. Under VSVD, the peak queue size is in the satellite switch rather than in the terrestrial bottleneck switch, as shown in Figure 5-3 versus Figure 5-2. VSVD demonstrates the ability to isolate the effect of the longest propagation delay link from the rest of the network. The queue requirement due to long propagation delay is shifted to the satellite switch only. Also note that the maximum queue sizes are also smaller due to the shorter feedback loops. The rate of decrease in the queue size of the switch that experiences the most congestion (ie. Switch 1 for VSVD case, Switch 2 for No-VSVD case) is slower for the VSVD case. This is caused by the smaller error term since Switch 1 has a much higher output link capacity (ie. $T(n)$) than Switch 2). Following this observation, we can conclude that the congested switch will take longer to return to a manageable queue level with VSVD. The large queue requirement makes the

switch more costly and harder to scale. However, the purpose of VSVD is to only localize the congestion closer to the source of it, providing easier network engineering. A queue control function can be used to help Switch 1 in the VSVD case to control queue growth.

The maximum queue size without VSVD for the bottleneck switch is approximately 7 percent higher compared to ERICA+ (shown in Figure C- 1 (b)). The maximum queue size with VSVD for Switch 1 and Switch 2 are approximately 15 percent lower compared to ERICA+ (shown in Figure C- 1 (c) and (d)). However, the queue drain rate is much sharper for the satellite switch with ERICA+ under VSVD. With ERICA+, Switch 1's queue under the VSVD configuration and Switch 2's queue under the No-VSVD configuration are completely drained in 1 second. The faster response is possible at the expense of more oscillations in the queue, evident in both the satellite and bottleneck switches whether or not VSVD is present. More oscillations in the queue means that there are more oscillations in the ACR, thus giving the client a perception instability in throughput.

Figure 5-4 and Figure 5-5 show the allowed cell rate at the sources. Overall, fairness among the sources is maintained in either case with or without VSVD. Without VSVD, the sources continue to transmit at the ICR for a longer period of time because the RTT for an RM cell is 130 ms (ie. [50 + 5 + 5] ms * 2). When the bottleneck switch detects the congestion, it must more aggressively control the ACR due to a larger error term. This is evident by the almost zero ACR value it publishes to the sources in Figure 5-4. With VSVD, the RTT for feedback information is 110 ms (ie. RTT of the satellite link + RTT of link 1 = [50 + 5] ms * 2). The shorter RTT allows the network to reduce the ACR more gracefully. The network reaches a steady state ACR of the bottlenecked link much faster and efficiently. Once in steady state, there are also fewer oscillations, in fact no oscillations, in the ACR. FASTRAC exhibits steady state behaviour by converging to a constant ACR. ERICA+ operates more in the transient domain, throttling the ACR to maintain buffering control as shown in Figure C- 2. Without VSVD, ERICA+ produces

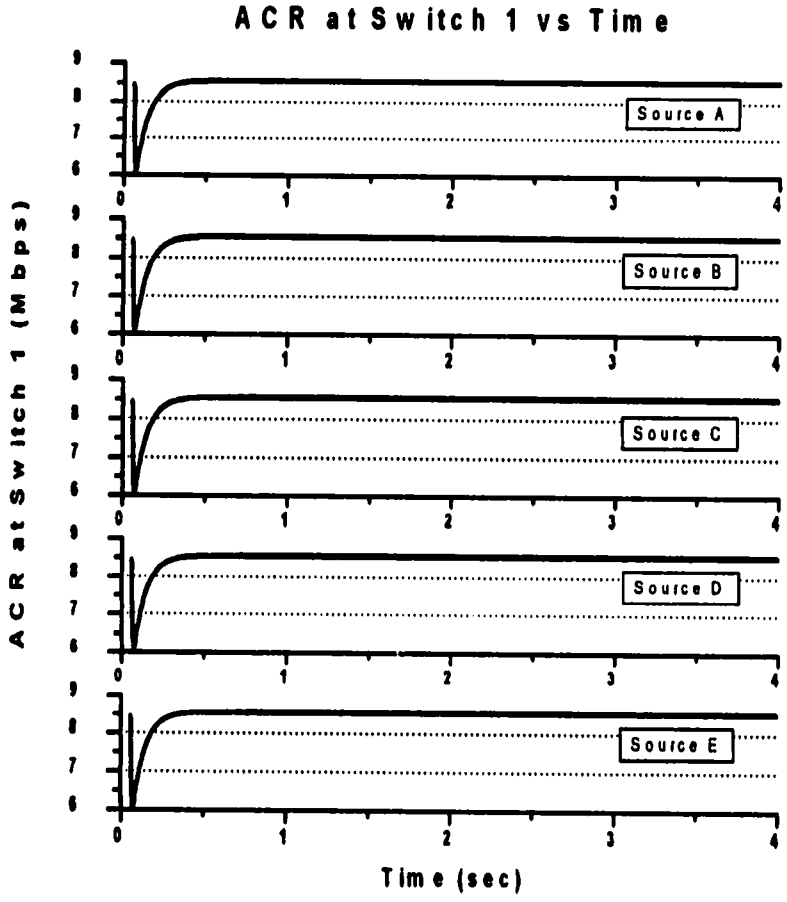


Figure 5-6: LEO Configuration with VSVD: ACR at Switch 1

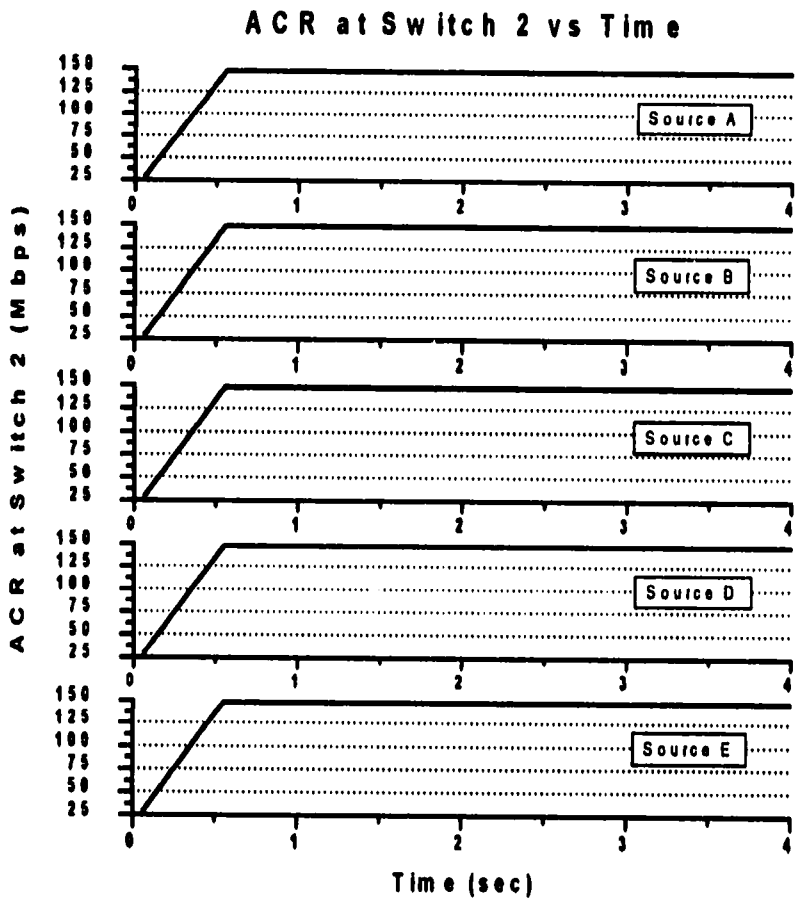


Figure 5-7: LEO Configuration with VSVD: ACR at Switch 2

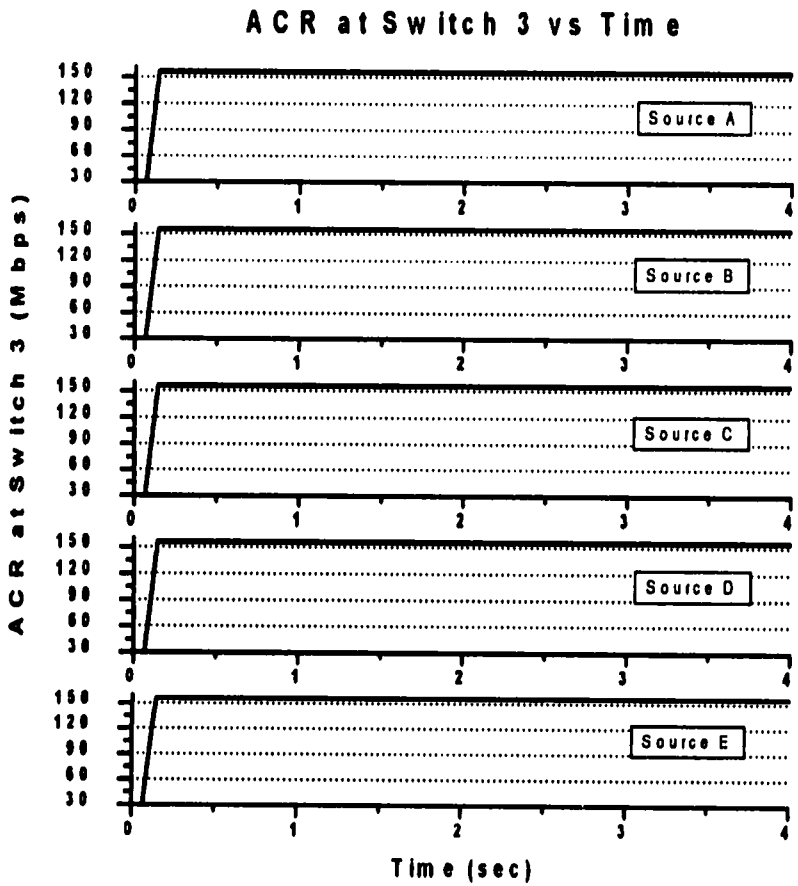


Figure 5-8: LEO Configuration with VSVD: ACR at Switch 3

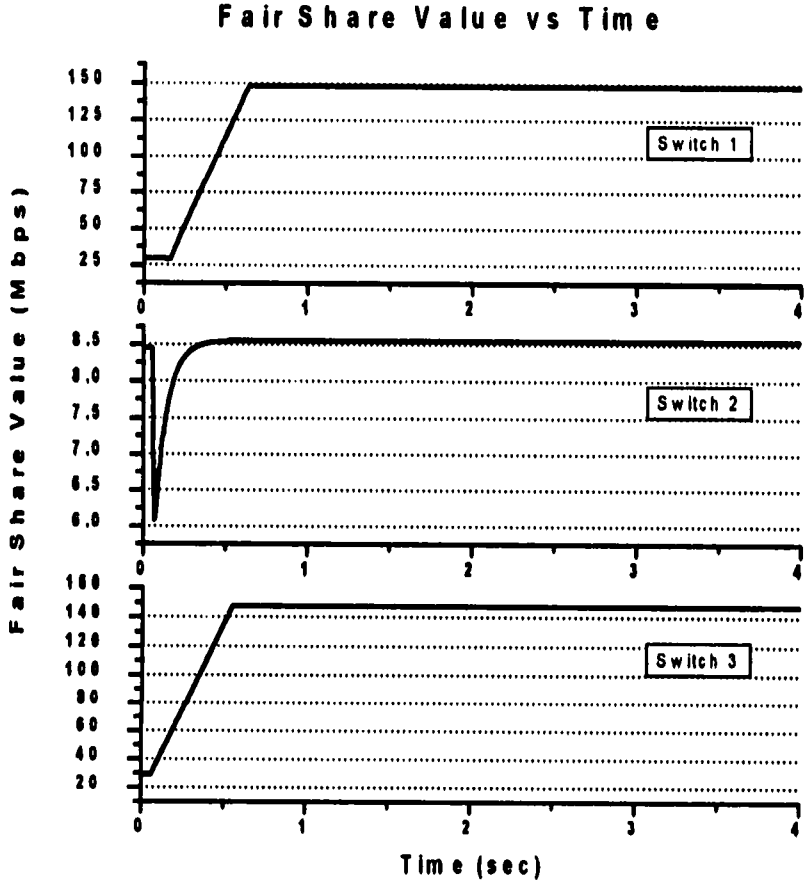


Figure 5-9: LEO Configuration with VSVD: Fair Share

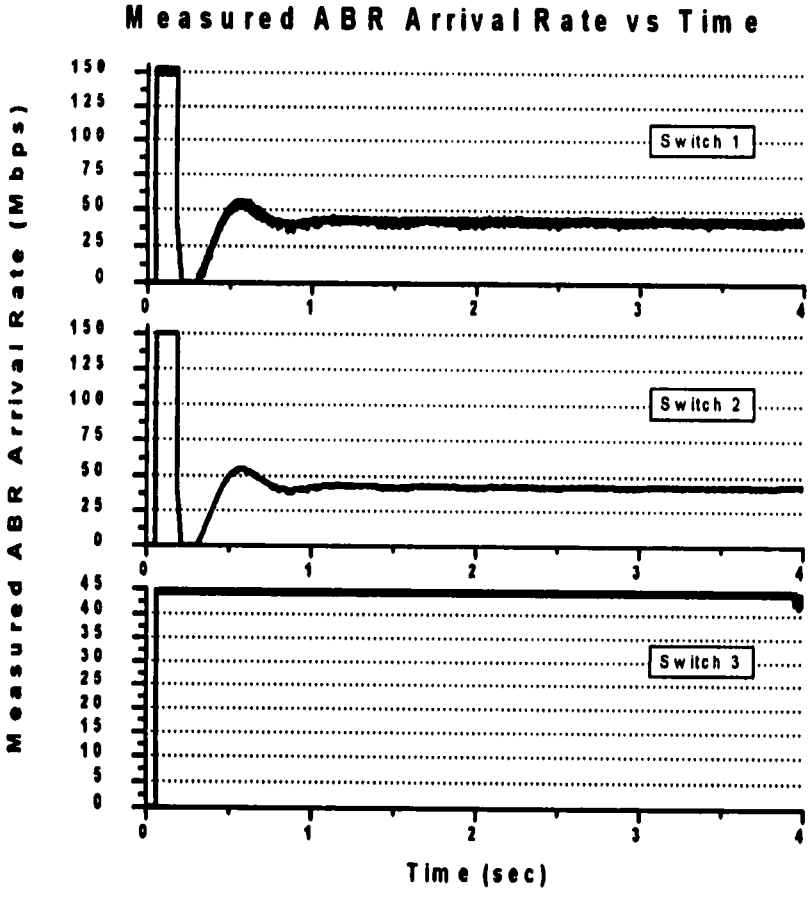


Figure 5-10: LEO Configuration with No-VSVD: Measured ABR Arrival Rate

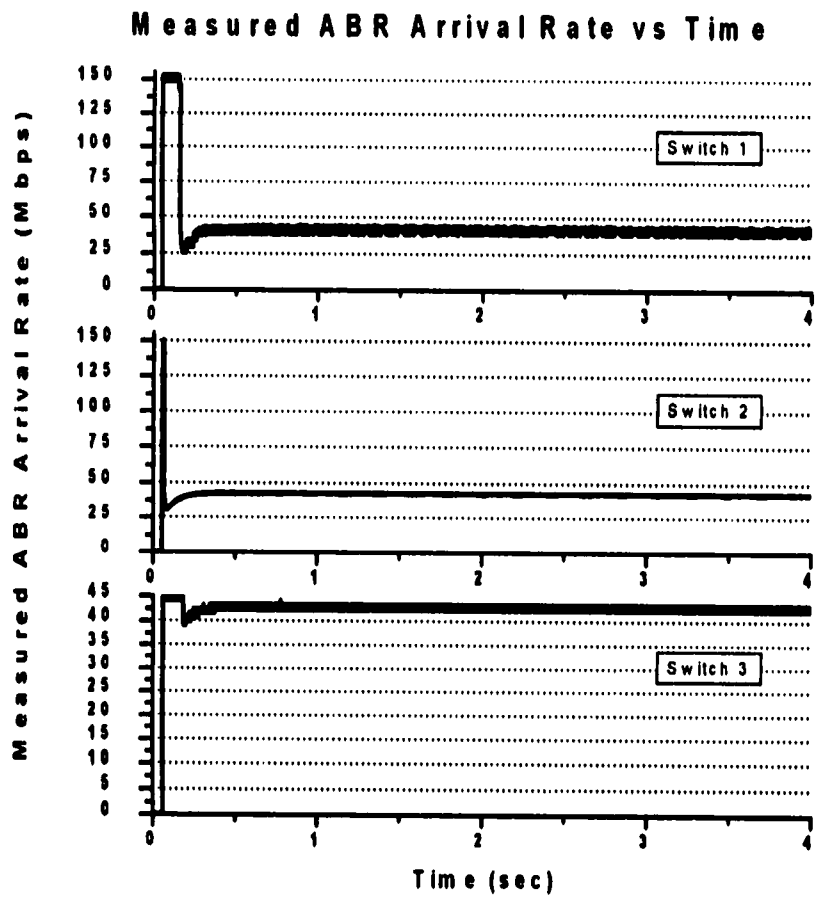


Figure 5-11: LEO Configuration with VSVD: Measured ABR Arrival Rate

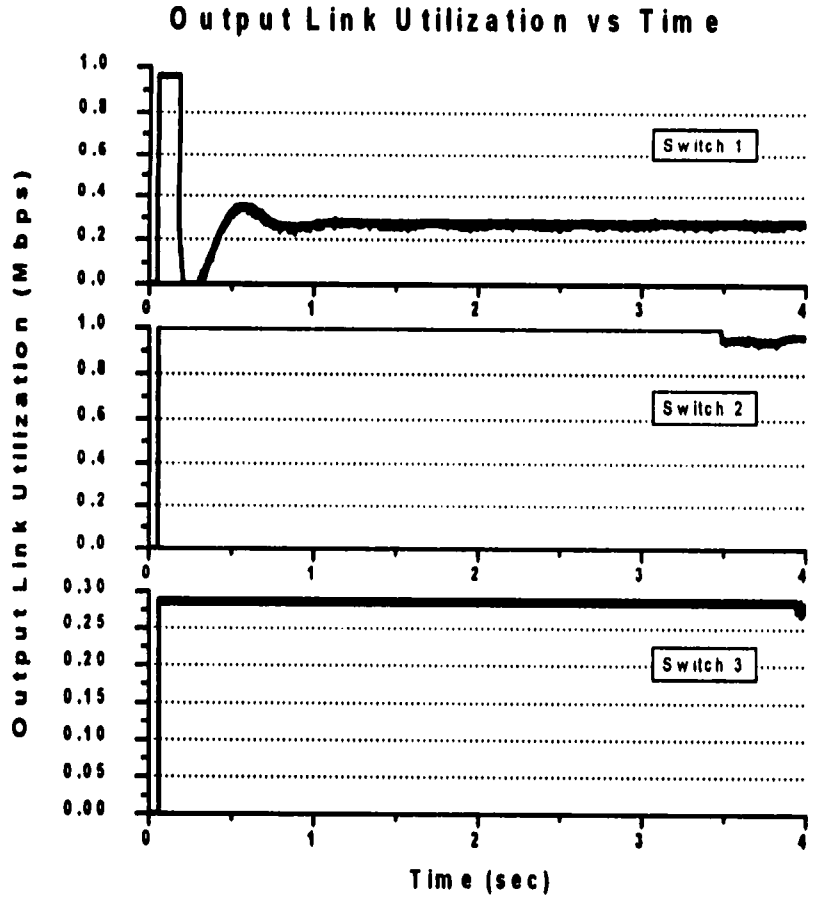


Figure 5-12: LEO Configuration with No-VSVD: Output Link Utilization

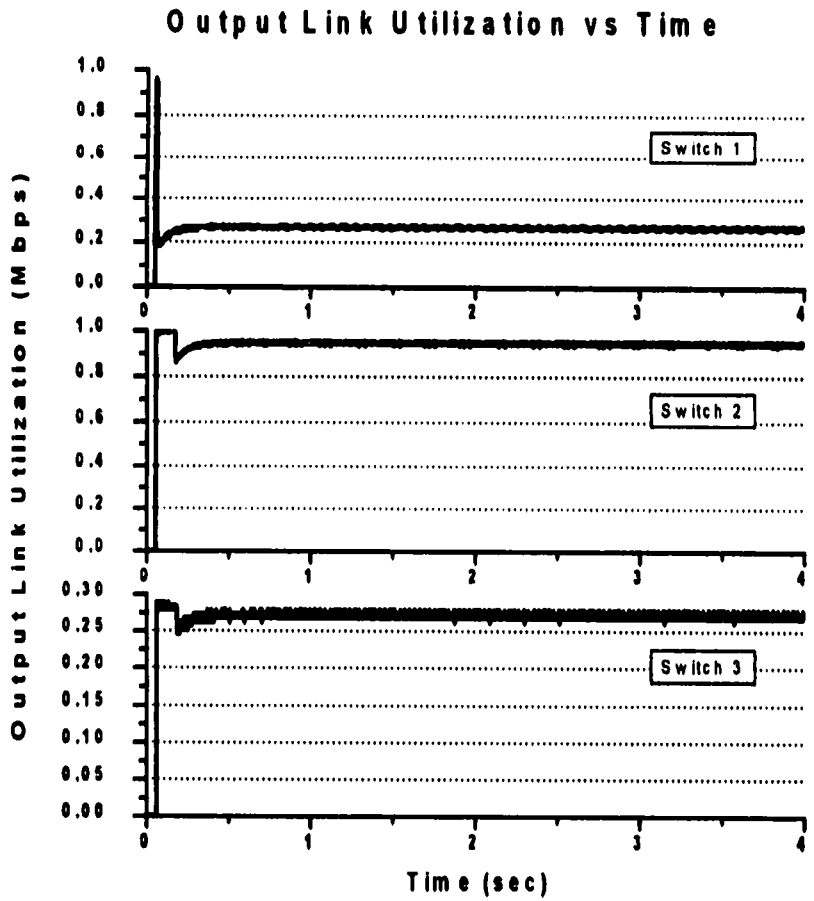


Figure 5-13: LEO Configuration with VSVD: Output Link Utilization

ACR curves that are saw-tooth in nature as shown in Figure C- 2 (a). With VSVD, the ACRs undergo tremendous sharp changes at very high frequency as shown in Figure C- 2 (b). Although being more effective in restricting the maximum queue size, the transient behaviour of ERICA+ may give a poorer perceived quality of service from the user's perspective.

The effect of the VSVD coupling is demonstrated in the ACRs at the switches in Figure 5-6, Figure 5-7, and Figure 5-8. The FASTRAC algorithm only sees congestion occurring when the arrival rate exceeds the service rate. Since there is no congestion at the destinations and Switch 3, the ERs communicated back to their corresponding upstream switches (ie. Switch 3 and Switch 2 respectively) are at the maximum value equal to the PCR after the initial climb from the ICR as shown in Figure 5-7 and Figure 5-8 as the ACRs of the respective upstream switches. Also note the effect the slower link has on communicating the ER back to Switch 2. A longer period exists for the ACR of Switch 2 to approach the PCR value in Figure 5-7. Congestion exists at Switch 2 due to the bottleneck link is detected by the FASTRAC algorithm, and the computed fair share is communicated back to Switch 1 as shown in the ACR of Switch 1. The dip in the ACR of Switch 1 shows that it is told by Switch 2 (ie. Switch 2's computed fair share) to reduce its ACR. Figure 5-9 of the computed fair share of Switch 2 has the same dip indicating the need to publish to the upstream switch to reduce its ACR. Notice that Switch 1 does not see congestion and advertise a high fair share value as shown in Figure 5-9. However, VSVD coupling of EQ 2-6 forces the ACR of Switch 1 to be communicated back to the source rather than its own computed fair share.

The reduced oscillations in ACR with VSVD result in almost no oscillations in curves for the measured ABR arrival rate and output link utilization as they converge to steady state. This result is seen when comparing Figure 5-10 to Figure 5-11 for measured ABR arrival rate and Figure 5-12 to Figure 5-13 for output link utilization. The output link utilization with VSVD reaches steady state faster. It is slightly higher for Switch 1 and lower for Switch 2 when comparing the VSVD case to No-VSVD case. This result is directly attributed to where the queue build-up (ie. congestion) occurs. The queue build-up

occurs in Switch 1 in the VSVD case, thus the output link utilization would be at its maximum as the switch strives to empty its queue. The measured ABR arrival rate demonstrates a more consistent acceptance of traffic into the network. Overall effect of VSVD is a more consistent throughput is achieved.

Due to larger and higher frequency oscillations in ACRs when using ERICA+, the link utilization for all links (shown in Figure C- 3) are lower compared to results using FASTRAC. Without VSVD, FASTRAC produces utilization that is roughly 12 % higher for link 1 and 23 % higher for link 2. With VSVD, FASTRAC produces utilization that is roughly 6 % higher for link 1 and 14 % higher for link 2. We can conclude that FASTRAC achieves higher and more consistent throughput with or without VSVD when compared to ERICA+. Nevertheless, the more responsive queue management of ERICA+ makes it more immune to heavy loading.

5.2.2 VSVD with Delayed Sources

In this section we evaluate the responsiveness of FASTRAC with VSVD to sudden sharp gain and loss of available bandwidth. The configuration parameters we used are as follows:

Configuration parameters:

Number of sources: 5

Switch Parameters:

Queue control function: off

Dynamic source tracking: on

Control Gain α: 1/[2d(n)+1]

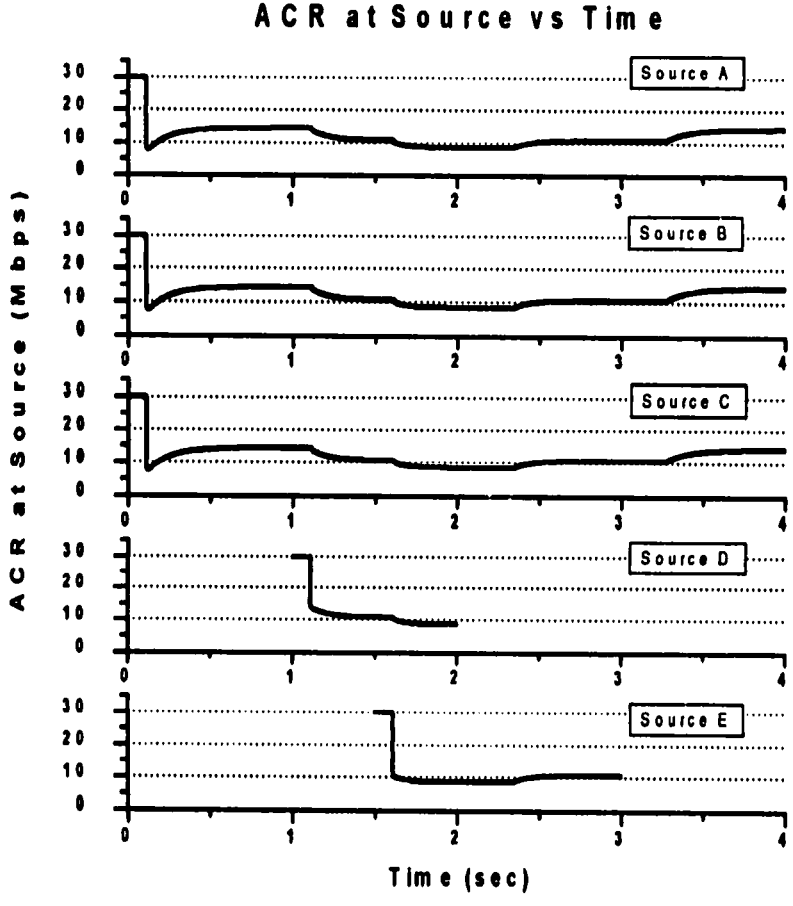


Figure 5-14: LEO Configuration with Delayed Sources: ACR at the Sources

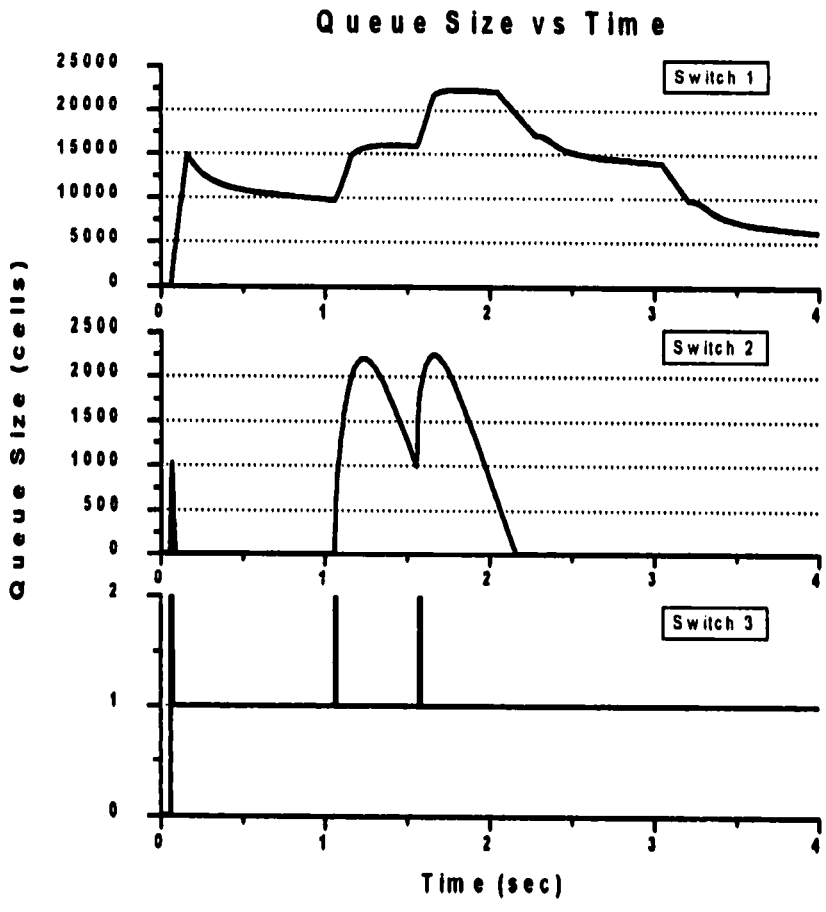


Figure 5-15: LEO Configuration with Delayed Sources: Queue Size

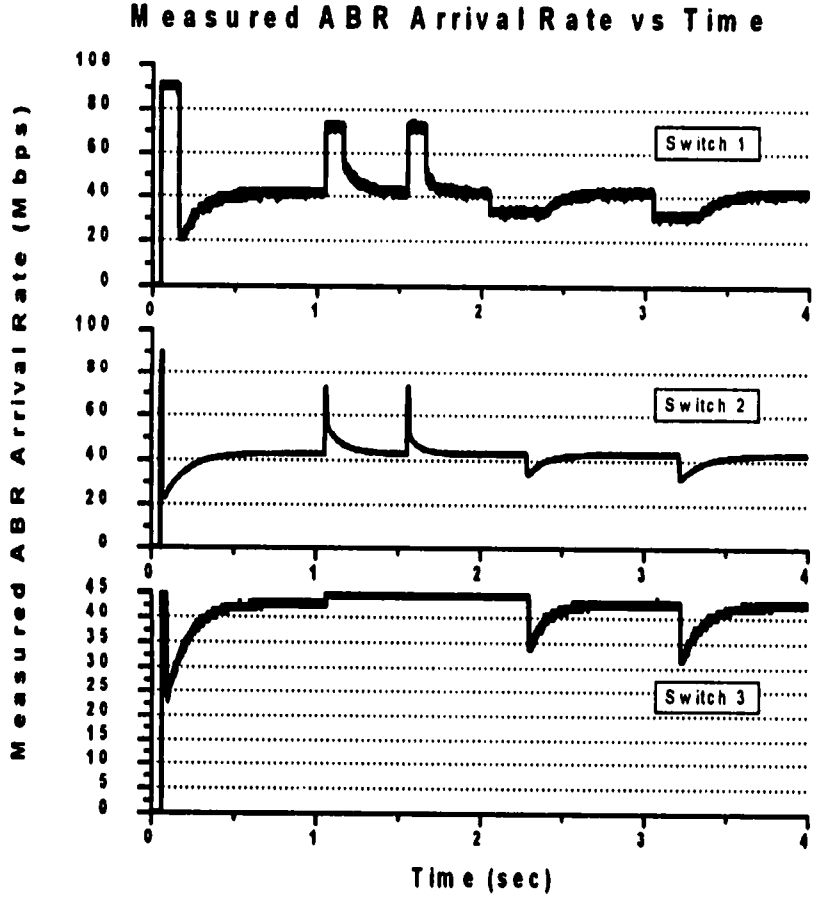


Figure 5-16: LEO Configuration with Delayed Sources: Measured ABR Arrival Rate

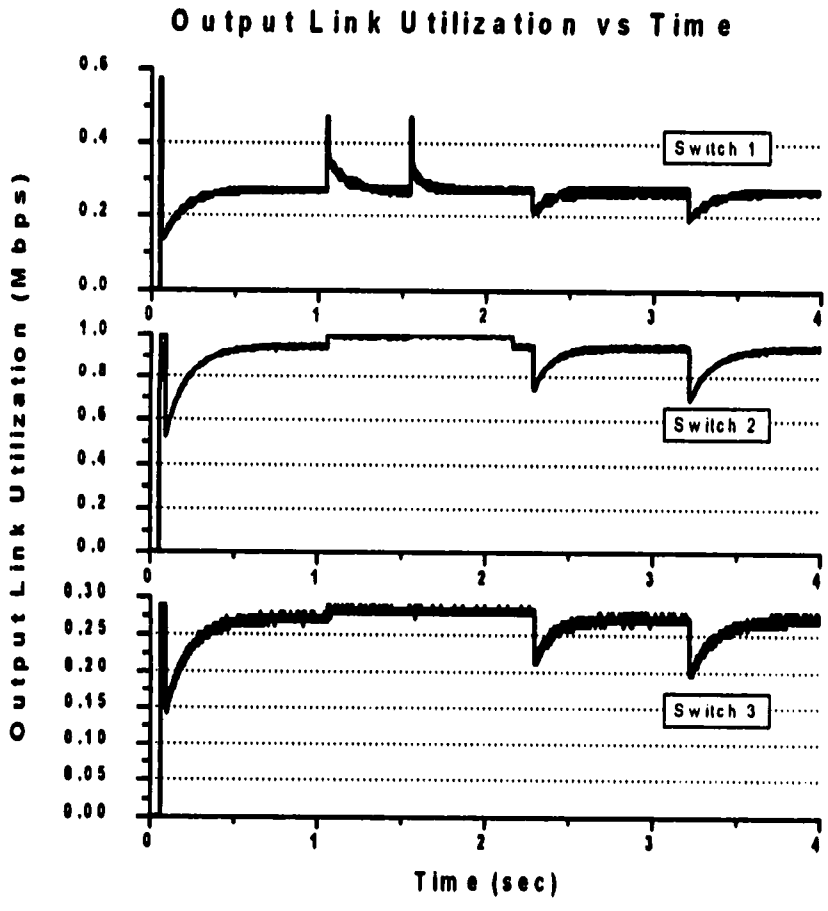


Figure 5-17: LEO Configuration with Delayed Sources: Output Link Utilization

Source	ICR (Mbps)	PCR (Mbps)	MCR (Mbps)	Start Time (second)	Stop Time (second)
A to C	30	155	0.1	0	4
D	30	155	0.1	1	2
E	30	155	0.1	1.5	3

Table 5-2: Source Traffic Parameters for Delayed Sources Experiment

The characteristics of the sources are described in Table 5-2.

The start time of sources can be delayed, and we stagger the start and stop time of some sources to produce the sharp bandwidth changes. The dynamic source tracking mechanism is enabled in order for FASTRAC to determine the number of active sources which changes when sources become active and inactive. Queue control function is disabled because we do not want its queue shaping characteristics to affect results in this experiment.

To demonstrate FASTRAC's ability of handling dynamic changes in ABR rate we observe transient behaviour when sources come active and inactive. We use the same LEO configuration but activate sources D and E at 1 and 1.5 second respectively. This allows us to observe FASTRAC's response to bandwidth loss for sources A to C. By stopping these sources at 2 and 3 second allows us to look at FASTRAC's response to bandwidth gain for sources A to C. Figure 5-14 shows the ACR at the sources. Note how the existing original sources are told to reduce the ACR when the delayed sources become active. The new sources themselves are also told to reduce the ACR from its ICR immediately. The original sources are also instructed to regain bandwidth quite timely when the delayed sources become inactive. There are no oscillations in the ACR curves when they react to the bandwidth changes demonstrating a good convergence to steady state. This graph also demonstrates the effectiveness of the dynamic source tracking mechanism. The similarity among the ACR curves signifies fairness is maintained.

Figure 5-15 shows the queue size of the switches. The queues of the terrestrial switches (Switch 2 and Switch 3) are maintained in perfect control. Observe the consistency in how FASTRAC reduces the queue size each time a delayed source becomes active, evident at Switch 2's queue. The maximum queue size of the satellite switch is smaller compared to when all five sources started at the same time since there are fewer sources active at the beginning. FASTRAC is able to drain the queue much better when there are fewer sources becoming active at once. The observation signifies that the queue control function must be a function of the number of active connections. And there have been studies of queue dimensioning that suggest some relationship to number of active connections.

Figure 5-16 shows the measured ABR arrival rate. In this figure, observe the peaks when new sources become active and drops when sources become inactive to indicate the rise and drop in incoming traffic respectively. Figure 5-17 shows the output link utilization. To see how effective FASTRAC is in maintaining steady throughput, observe how the output link utilization is recovered after the instance in time where the peaks and drops of the incoming traffic occur.

5.2.3 Effect of VBR Background Traffic

In this section we observe the response of FASTRAC with and without VSVD when a background VBR traffic is present. The VBR traffic basically reduces the available bandwidth to ABR connections and at the same time makes the available bandwidth noisy. Queue control function is not required and is disabled. Dynamic source tracking is also disabled since there are no transient changes in the number of active connections in this experiment. The configuration parameters we used are as follows:

Configuration parameters:

Number of sources: 5

Switch Parameters:

Queue control function: off

Dynamic source tracking: off

Control Gain α: 1/[2d(n)+1]

Source	ICR (Mbps)	PCR (Mbps)	MCR (Mbps)	Start Time (second)	Stop Time (second)
A to E	30	155	0.1	0	4
VBR	Hurst parameter: 0.75 Max Rate: 42 Mbps Mean Rate: 20 Mbps Standard Deviation: 4			0	4

Table 5-3: Source Traffic Parameters for VBR Background Traffic Experiment

The characteristics of the sources are described in Table 5-3.

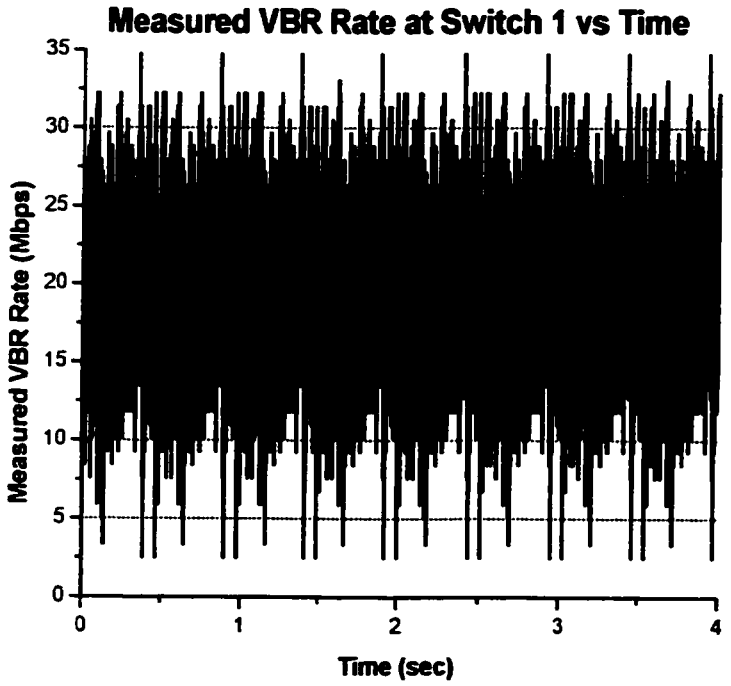


Figure 5-18: VBR Traffic

To observe the behaviour of FASTRAC under more realistic network traffic, VBR traffic is introduced as background traffic into the configuration. Just like the ABR sources, a VBR source connected to Switch 1, injects traffic that is destined to a destination connected to Switch 3. Figure 5-18 shows the VBR traffic that enters Switch 1. The switches service VBR traffic before it serves ABR traffic in a round robin fashion. The VBR source is configured to generate self-similar traffic to mimic self-similarity of internet traffic. The Hurst parameter when configured between 0.5 to 1 exhibits self-similarity in the stochastic process. Simulations with different seeds were run. They generated results similar to what is observed in this section.

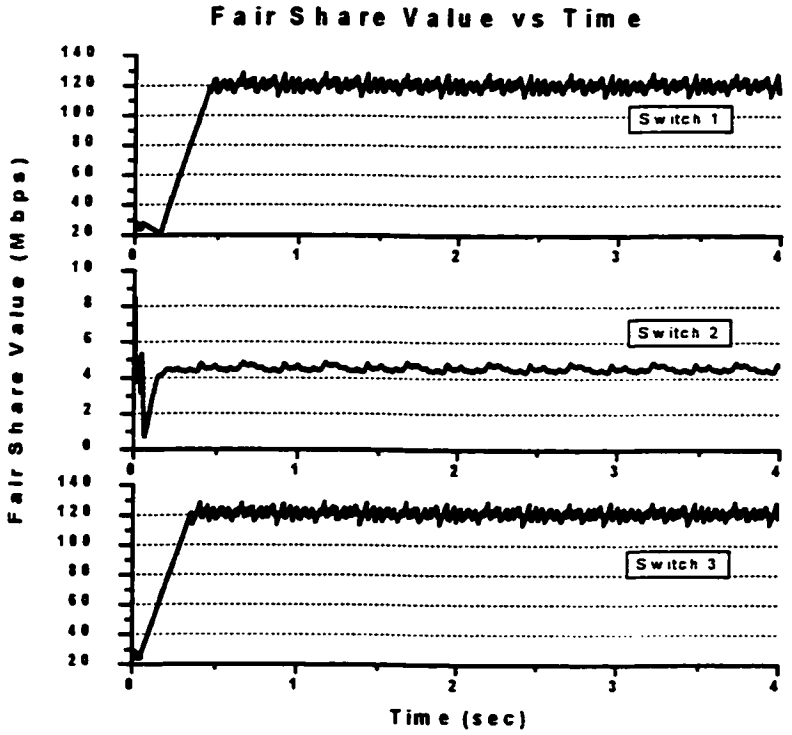


Figure 5-19: LEO Configuration with VSVD and VBR Background Traffic: Fair share

From our experiment, we observed that the ACRs at the sources with and without VSVD when VBR background traffic is present are very much the same as the ones without background VBR traffic in Figure 5-4 and Figure 5-5. They are omitted for conciseness purpose. The only difference is that they converge at a lower ACR value of roughly 4.5 Mbps (ie. 22.75 Mbps available bandwidth/5 sources) since the VBR traffic on average

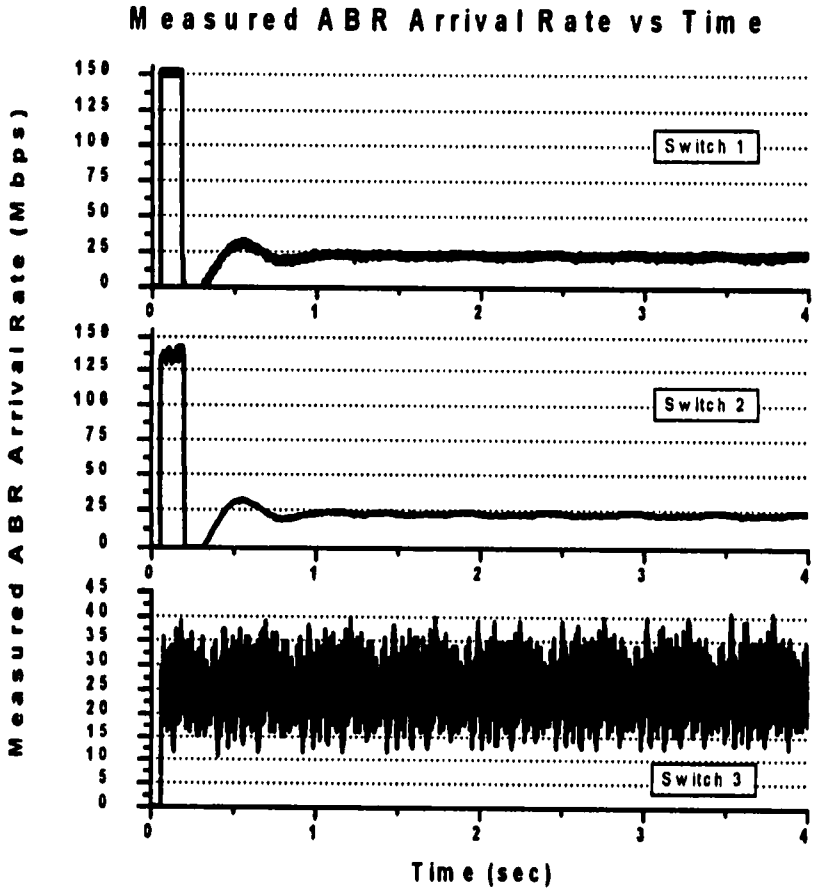


Figure 5-20: LEO Configuration with No-VSVD and VBR Background Traffic: Measured ABR Arrival

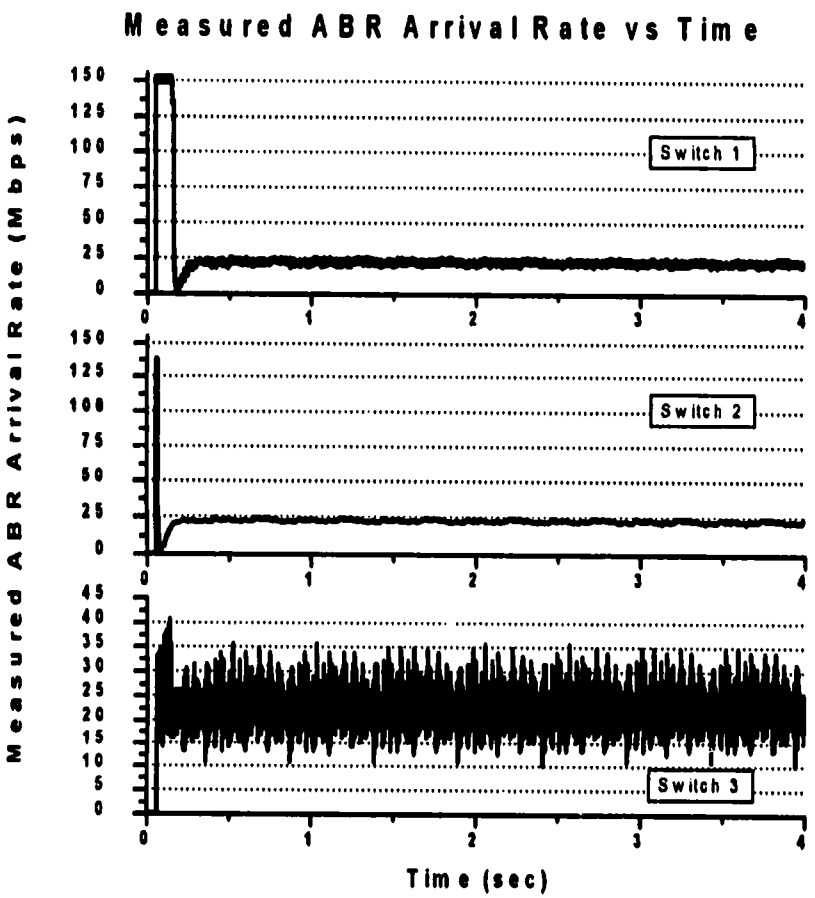


Figure 5-21: LEO Configuration with VSVD and VBR Background Traffic: Measured ABR Arrival Rate

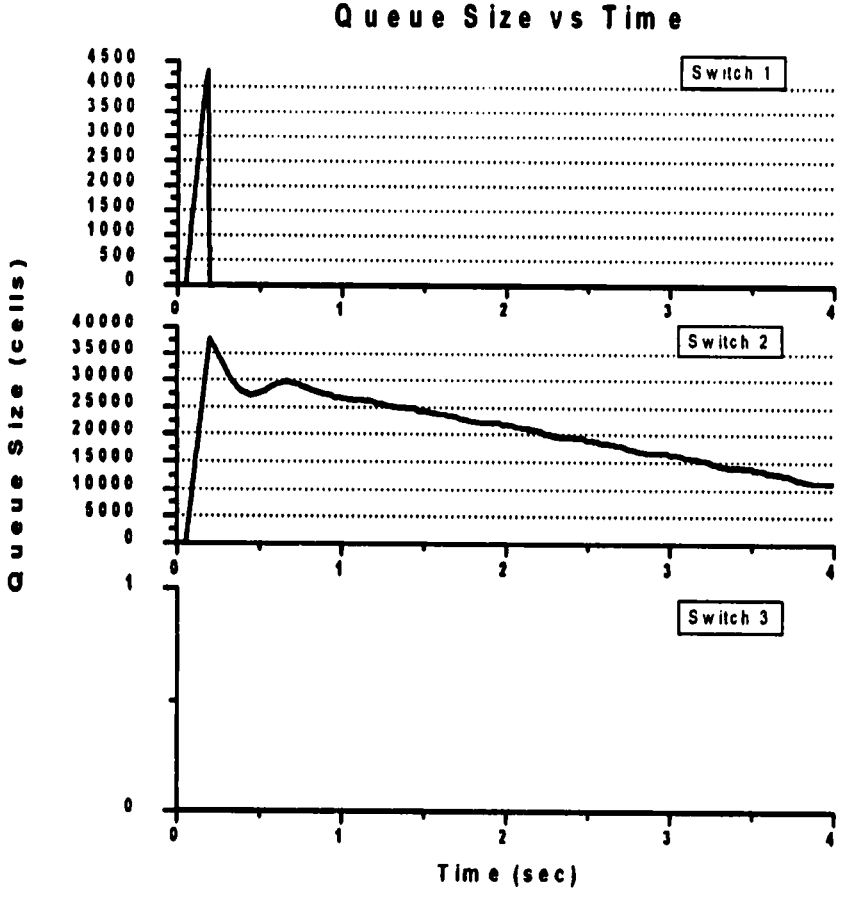


Figure 5-22: LEO Configuration with No-VSVD and VBR Background Traffic: Queue Size

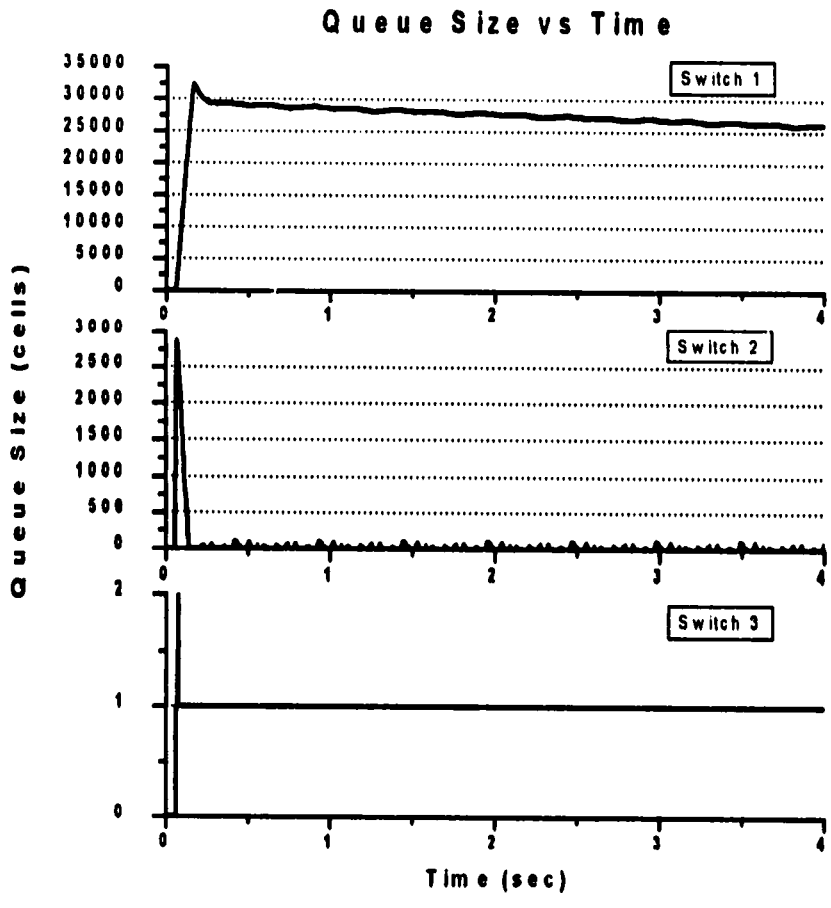


Figure 5-23: LEO Configuration with VSVD and VBR Background Traffic: Queue Size

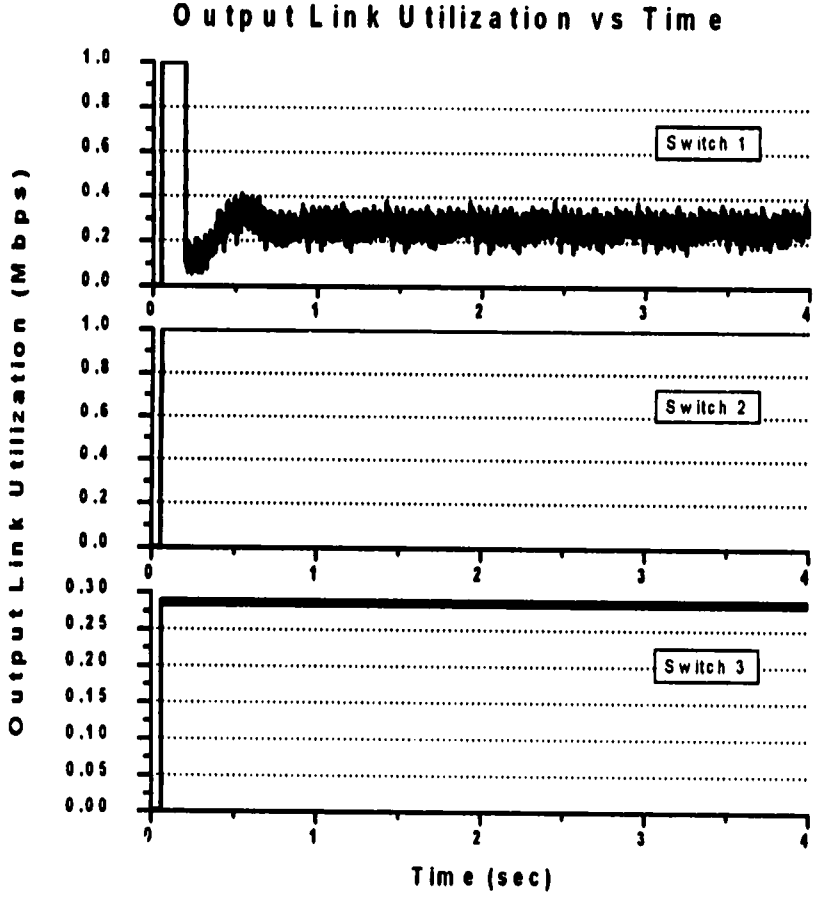


Figure 5-24: LEO Configuration with No-VSVD and VBR Background Traffic: Output Link Utilization

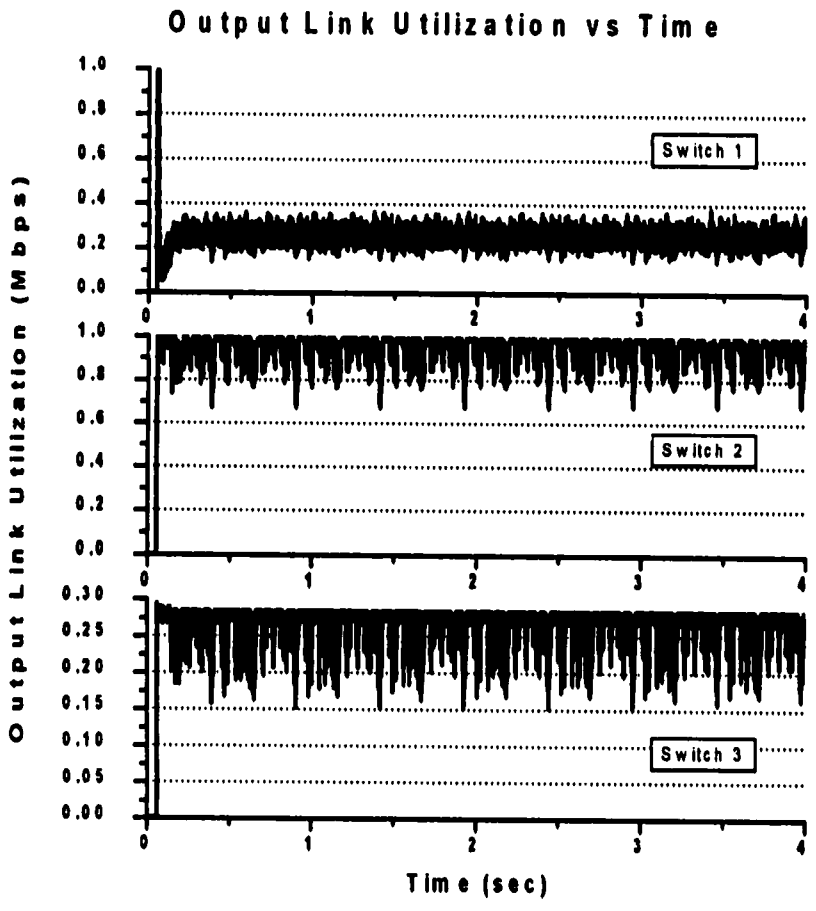


Figure 5-25: LEO Configuration with VSVD and VBR Background Traffic: Output Link Utilization

consumes 20 Mbps of a total bottleneck capacity of 42.75 Mbps. Our results demonstrate that FASTRAC is very immune to noise such as VBR background traffic.

The VBR bandwidth consumption reduces $T(n)$ in calculating the error term in step 9 of section 2.4.6 which in turn affects the fair share calculation as shown in Figure 5-19. The most obvious evidence of the VBR disturbances is illustrated in the fair share of Switch 2, the bottlenecked switch. The fair share converges to an average of 4.5 Mbps which equates to 22.5 Mbps of ABR capacity for the 5 ABR sources. The remainder 20 Mbps is used by VBR traffic. Likewise, the fair share of the other 2 switches are also affected by the VBR bandwidth consumption.

The earlier discussion of the ACR curves being relatively unaffected by the background VBR traffic naturally extends to the measured ABR arrival rate. The measured ABR arrival rate aligns with the ACR curves, averaging roughly at 22.5 Mbps under steady state as shown in Figure 5-20 and Figure 5-21. The shorter delay feedback loops of VSVD also demonstrates to better reduce the effect of transient noise as illustrated in the measured ABR arrival rate at Switch 3 by the thinner oscillation band.

There is a tremendous amount of difference in the ability to control the queue size when VBR traffic is present for the configuration without VSVD as captured in Figure 5-2 and Figure 5-22. Definitely there is a big increase in the queue size and the drain rate at the bottleneck switch when VBR background traffic is present. Also, Switch 1 also experiences a surge in the ABR queue size at the onset of the network congestion that is not present when there is no VBR background traffic. With VSVD, the general queue behaviour does not differ much as shown in Figure 5-23, compared to Figure 5-3. The queue at Switch 2 is also unaffected by the VBR traffic. However, there is an increase in queue size at the satellite Switch 1. This is expected since the goal of VSVD is to isolate the effect of congestion closer to the source of it. The elevated queue size of Switch 1 is due to the additional loading by VBR background traffic. The sum of the ICRs of all ABR sources already consumes all of the service capacity of Switch 1. We are seeing the results of overbooking the ICRs by roughly 20 Mbps. If we reduce the ICR of each

source by 4 Mbps, we would probably obtain the same queue size as the case without VBR background traffic. Nonetheless, a queue control function can be deployed at Switch 1 to safeguard against growing the queue size to high.

Figure 5-24 and Figure 5-25 display the utilization for various links with and without VSVD active. Without VSVD, the congestion is present in Switch 2 evident by its large queue size. Thus the output link utilization at Switch 2 is also very high since the switch is constantly busy trying to empty the queue. There are more oscillations in the link due the more adaptive nature of FASTRAC with VSVD. However, the amount of the utilization of the links are quite the same. In general, output link utilization remains very good with or without VSVD when VBR background traffic is present.

5.3 GEO Configuration Results

In this section we want to observe FASTRAC with VSVD under extremely long propagation delay of 275 ms on the satellite link to see how robust FASTRAC is. The optional queue control function is disabled since we are not interested in its aid. Dynamic source tracking is disabled since the number of active sources remains the same throughout the experiment. All experiments are conducted on FASTRAC with VSVD. As proven in section 5.2.1, FASTRAC without VSVD does not perform well under long propagation delay. Therefore, there is no need to study it under extremely long delay propagation. The configuration parameters used are as follows:

Configuration parameters:

Number of sources: 5

Switch Parameters:

Queue control function: off

Dynamic source tracking: off

Control Gain α: $1/[2d(n)+1]$

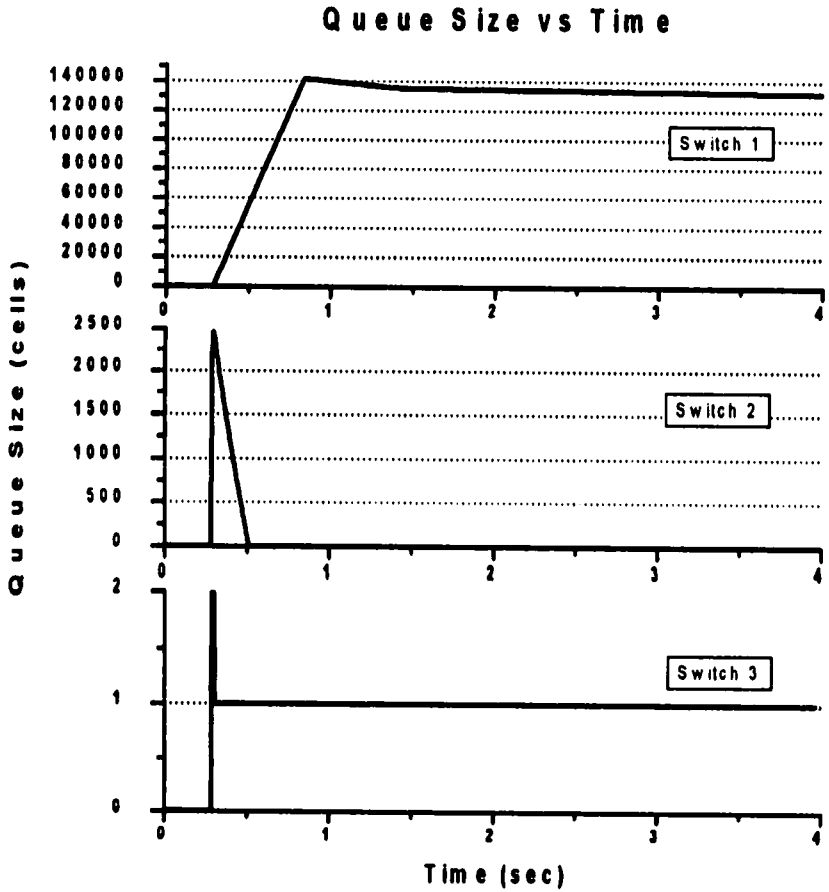


Figure 5-26: GEO Configuration with VSVD: Queue Size

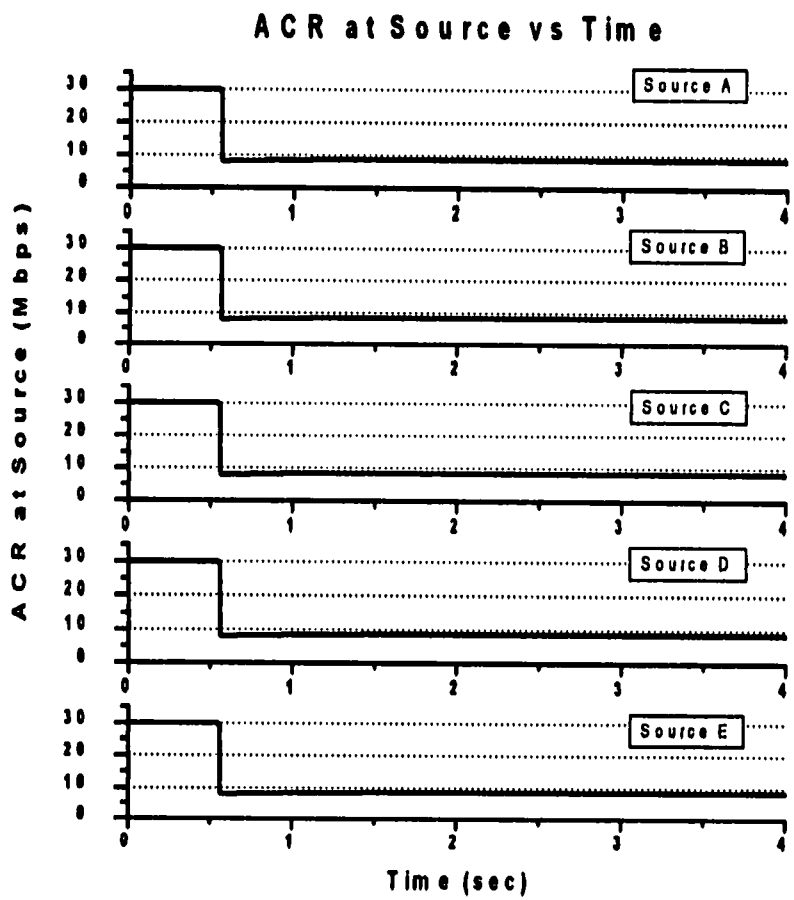


Figure 5-27: GEO Configuration with VSVD: ACR at the Sources

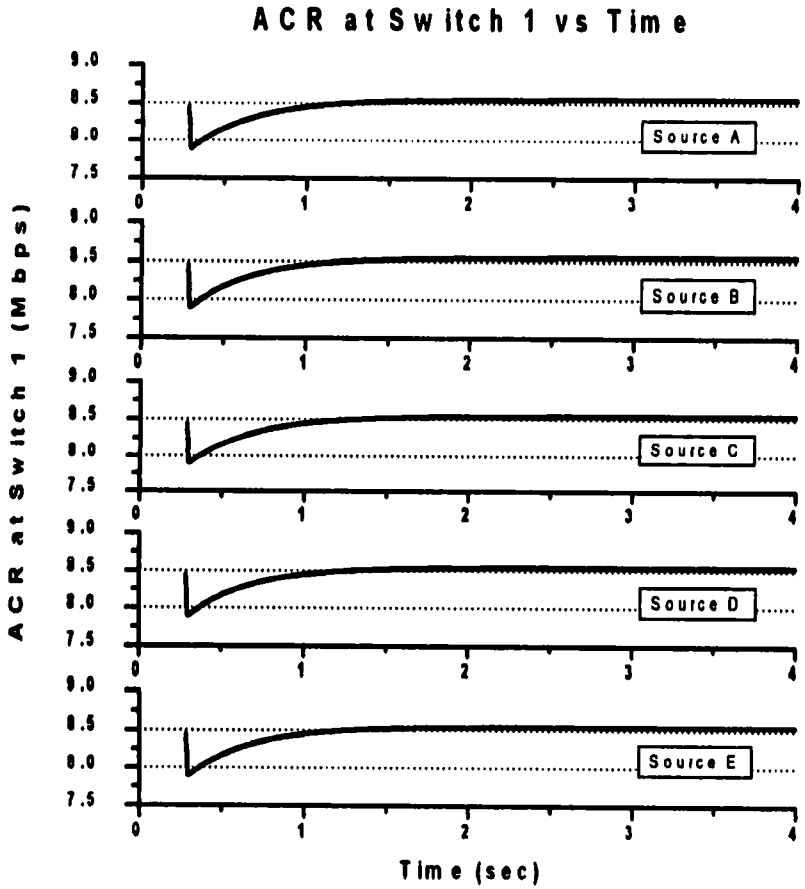


Figure 5-28: GEO Configuration with VSVD: ACR at Switch 1

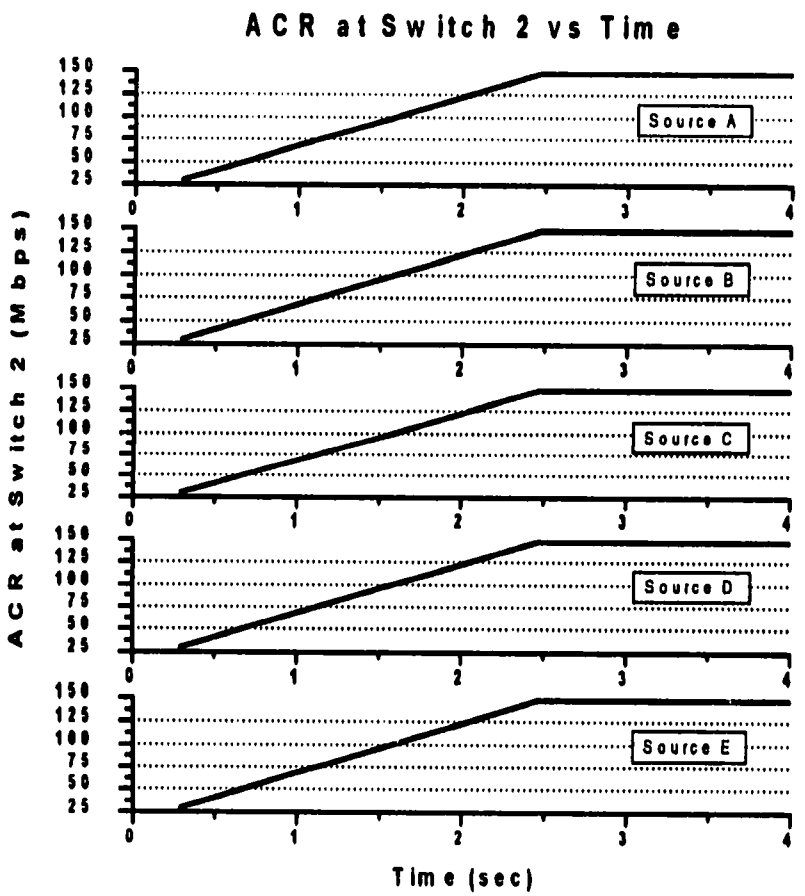


Figure 5-29: GEO Configuration with VSVD: ACR at Switch 2

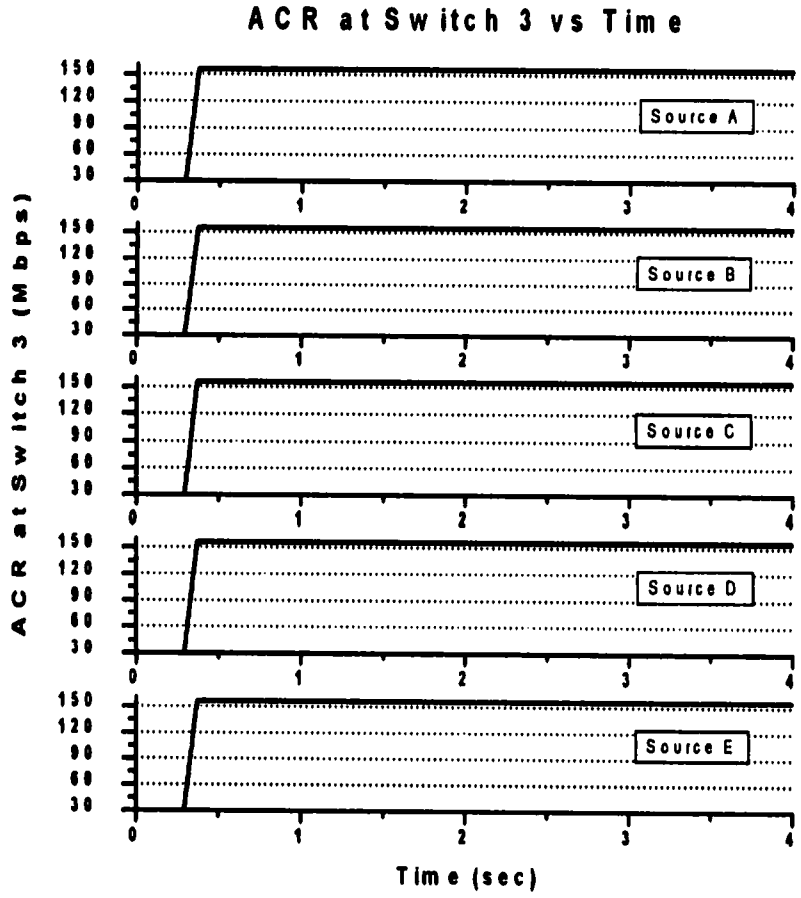


Figure 5-30: GEO Configuration with VSVD: ACR at Switch 3

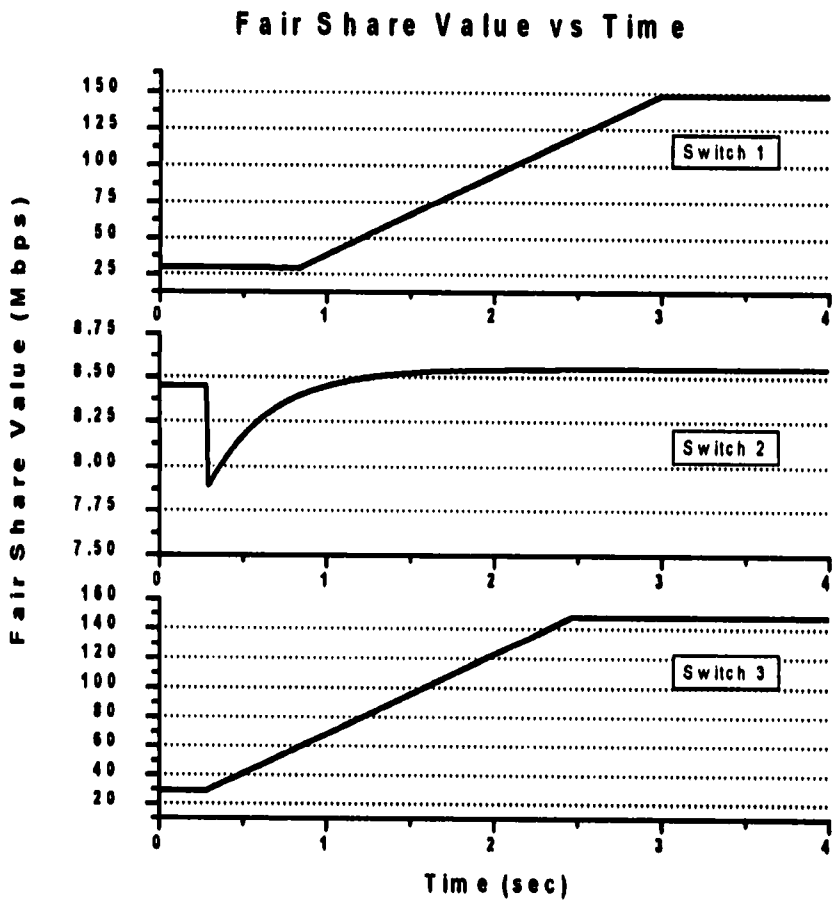


Figure 5-31: GEO Configuration with VSVD: Fair share

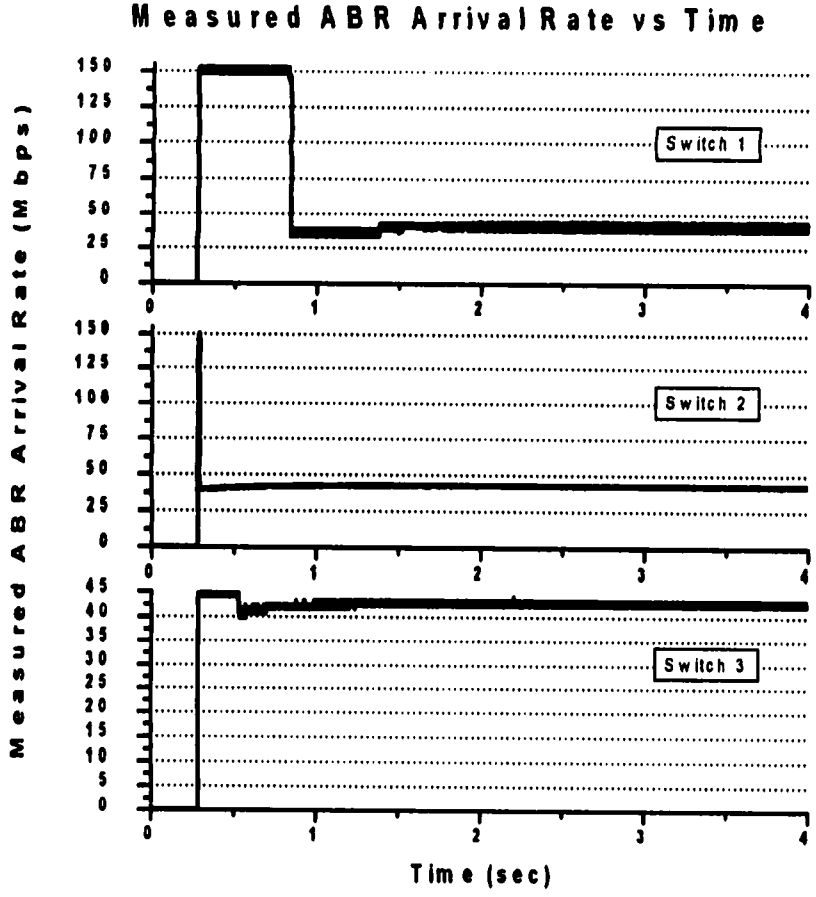


Figure 5-32: GEO Configuration with VSVD: Measured ABR Arrival Rate

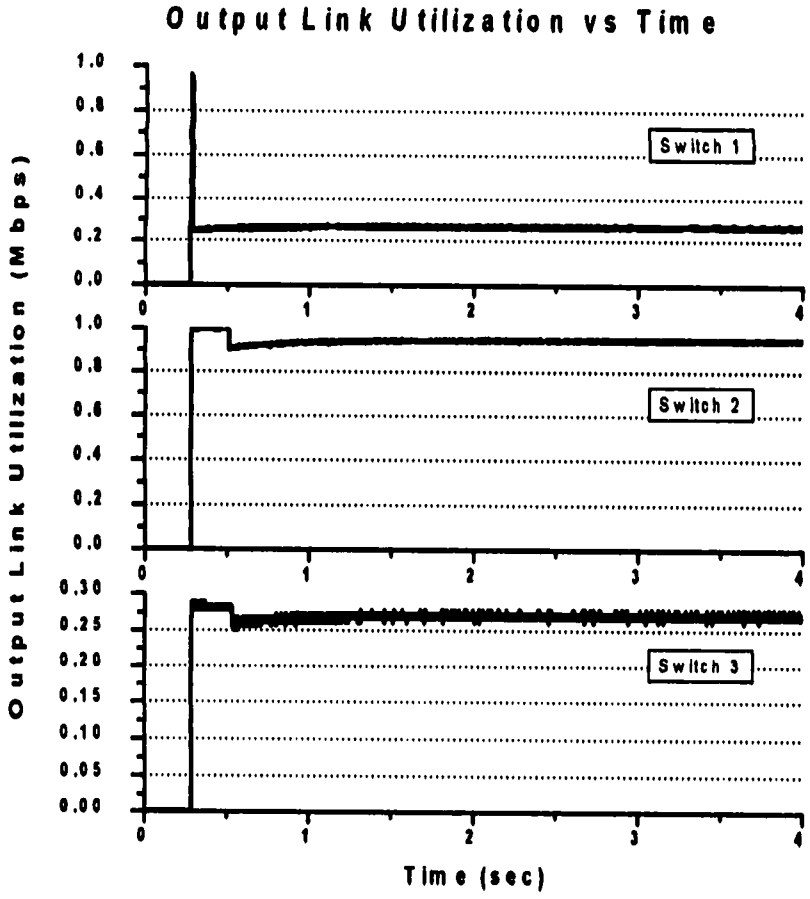


Figure 5-33: GEO Configuration with VSVD: Output Link Utilization

Source	ICR (Mbps)	PCR (Mbps)	MCR (Mbps)	Start Time (second)	Stop Time (second)
A to E	30	155	0.1	0	4

Table 5-4: Source Traffic Parameters for GEO Configuration

The characteristics of the sources are described in Table 5-4.

Comparing Figure 5-3 and Figure 5-26 demonstrates that VSVD is able to protect the terrestrial network from the longer propagation delay of the satellite link. The queue size of the terrestrial network switches in the GEO configuration are the same as the ones in the LEO configuration. The longer propagation delay of the satellite link increases the maximum queue size of the satellite switch only. With VSVD, the engineering of the terrestrial network is not affected while only the satellite switch needs to be re-engineered and re-configured in order to better accommodate the longer propagation delay of the satellite link.

The enormous queue size of Switch 1 of roughly 140,000 cells is matched in the results using ERICA+ (shown in Figure C- 4). Like in the LEO configuration, the queue drain rate with ERICA+ is much faster. Nevertheless, it is the maximum queue size that is an important factor in engineering the switch to avoid cell loss. Similar results to ERICA+ under very large propagation delay shows that FASTRAC is very capable. Note also that the sources are persistence, meaning that these tests are using extreme boundary conditions. Results should be better with bursty sources.

Figure 5-27 shows the ACR curves at the sources in the GEO configuration are exactly the same as the one in the LEO configuration of Figure 5-5, except that they are stretched out along the time axis more. The longer propagation delay of the satellite link causes the source ACRs to adjust roughly 225 ms later. Figure 5-28 to Figure 5-30 shows the ACR measurements at the switches. These ACRs converge to the same values as in the LEO configuration which are shown in Figure 5-6 to Figure 5-8. The only difference is that the switches react slower due to the delayed arrival of cells at Switch 1 because of

the longer propagation delay. The results are more stretched ACR curves along the time axis. Also note that the start of the ACR curves begins at a later time because cells take longer to arrive due to the longer propagation delay. The fair share curves exhibit the same behaviour of being stretched along the time axis as shown in Figure 5-31.

With ERICA+, the ACR curves under the GEO configuration (shown in Figure C- 5) are also similar to the ones under the LEO configuration except that they are stretched out more on the time axis. The behaviour is exactly what is observed with FASTRAC. The temporal stretching causes the peaks in these curves to be farther apart which suggests that throughput would be lower because the low points in the ACR curves extends over long periods of time. Compared this with FASTRAC suggests that FASTRAC would have higher throughput than ERICA+ under long propagation delay.

The behaviour inside the network observed through the ACR curve at each switch is very much the same as in the LEO configuration, except for temporal stretching. This leads to conclude that the input and output behaviour of the network should be similar. Figure 5-32 and Figure 5-33 show the measured ABR arrival rate and output link utilization at each switch. These curves are the same as in the LEO configuration, again with only temporal stretching.

Earlier conclusion that FASTRAC should have higher throughput than ERICA+ is backed up by observing the measured ABR arrival rate and output link utilization. ERICA+ shows many stretched out low points in the output link utilization under the GEO configuration (shown in Figure C- 6), suggesting poorer throughput compared to the LEO configuration. Under the GEO configuration, since FASTRAC exhibits no oscillations in the output link utilization curves under steady state operation except for temporal stretching concludes by cross comparison with ERICA+ that FASTRAC should have better throughput.

5.4 Remarks

We performed various experiments to show how VSVD provides better queue management at the network level when a network segment with long propagation delay is present. VSVD localizes the queue build up to the switch at the long propagation delay segment of the network rather than at the bottlenecked switch. However, the queue build up is not necessarily reduced. Maximum queue size reduction should be possible if a queue control function is applied. The investigation of queue control function for VSVD is outside the scope of this study, and is left for future work. The LEO configuration served adequately as a test configuration to verify the successful integration of VSVD support into FASTRAC. Results using ERICA+ [GoCa98] under the satellite configuration was a good point of reference to verify results from FASTRAC. FASTRAC's results are very competitive with ERICA+'s. Even under the robustness test using the GEO configuration, results are comparable. FASTRAC seems to be less aggressive in reducing queue build up than ERICA+. However, FASTRAC provides more consistent and stable throughput. This is quite explainable because ERICA+ uses a combination of rate thresholding and queue size thresholding while FASTRAC with VSVD did not have the optional queue control (which is similar to queue size thresholding) enabled.

Although we employed a more suitable control gain in the satellite configuration, FASTRAC is still less aggressive than ERICA+ in maintaining low queue size. The modified control gain of 1/(2d+1), which still falls in the stability boundary of $(0 < \alpha < 2/(d+1))$ [ref: Appendix B.2], actually reduces the amount of adjustment FASTRAC can make to correct the bandwidth mismatch. Oscillations in the load occur if the system tries to correct more than a certain fraction of the difference between the estimated load and the target load per control interval. This is due to a periodic discrepancy between the estimated load and the actual load caused by time lags. Nortel Network's FASTRAC team is currently working on the theoretical development behind this control gain. Therefore, the proof is not available to include into this thesis.

Chapter 6: Conclusion

This thesis presents ABR and VSVD from both a theoretical and practical perspective. We presented TM4.0 ABR Control Mechanisms and provided detailed background information about how ABR works and the protocol proposed to provide flow control for ABR traffic. Our discussion on switch design and characterization of the feedback control scheme provided the toolset to systematically classify control algorithms and quickly assess their practicality in implementation. We analyzed a handful of algorithms found in the literature, and demonstrated the diversity of control algorithms, their good qualities, and most importantly their deficiencies. A slight deficiency or non-compliance to TM4.0 can produce completely unusable algorithms since one misbehaving ABR node can affect the entire network. Through performance analysis of one of the surveyed algorithms, we presented an example of poor compliance to TM4.0 ABR control protocol. Therefore, it is also very important to design control algorithms that are resistant to adverse network behaviour.

Our in-depth study of FASTRAC explores many aspects of ABR control techniques. We demonstrate that FASTRAC is a very simple ABR control algorithm compared to many of the ones proposed in the literature, yet it is completely compliant to TM4.0 ABR control rules. Simplicity improves robustness and scalability. Simple rate thresholding and shared buffering are employed in the FASTRAC model. Yet, the algorithm factors in background traffic, measurement or estimation error, control loop delay, and link rate dependent sampling rate. Increased control is also offered by additional queue control functions. Dynamic tracking of the number of active sources allows FASTRAC to handle switched connections and maximize link utilization when sources become inactive.

The simplicity of FASTRAC, allowed us to add VSVD to it with ease and elegance. Although requiring per-VC queueing to maintain cells of different VCs separate, FASTRAC with VSVD still functions with aggregate cell rates. VSVD coupling is easily integrated into FASTRAC's rate allocation calculation. Since FASTRAC is decoupled

from scheduling, any scheduling algorithm can be used. Our proposed two stage queueing scheduler demonstrates to maximize link utilization by avoiding head of line conflict resolution, instead it maintains fairness through serialization of cells in a second stage transmit queue.

Our well-designed simulation scenarios targeted testing particular aspects of the FASTRAC. We stated and discussed all configuration parameters and assumptions to assist interpreting and judging results. Many literature papers do not state assumptions and provide detailed configuration parameters, making interpreting the results very difficult. We exploited engineering aspects such as maximum queue length, throughput and stability. These aspects provide a good basis for evaluating the algorithm. We used the Bottleneck configuration to provide a comprehensive test of FASTRAC against ABR control theory. Results from this experiment demonstrate how FASTRAC work and its compliance to TM4.0, responsiveness, efficiency, stability, and min-max fairness. We also demonstrated enhancements to FASTRAC such dynamic tracking of number of effective sources and queue control function. Results show how FASTRAC accurately track the number of effective sources. The simple linear queue control function shows how difficult queue control can be. Results show that correct dimensioning of the thresholds is very difficult to achieve. Thresholds could be governed by a number of factors alluded to such as the control algorithm gain and the number of connections.

The ABR control principles and simulative demonstration up to this point establish the firmer foundation, compared to many literature papers, to study ABR under long propagation delay. We used the Satellite configuration as a test bed for long propagation delay and VSVD. We compared FASTRAC's results with ERICA+'s results. ERICA+'s results gave us a good point of reference to interpret and accept FASTRAC's performance results. We used the LEO configuration with and without VSVD to show the advantages and disadvantages of VSVD. The queue requirement of the terrestrial network is completely protected from the satellite network which requires larger queue size due to longer propagation delay. Overall, FASTRAC competes quite well with ERICA+ in buffering requirement. FASTRAC requires slightly smaller buffers, but is

slower in draining the queue, possibly giving ERICA+ a credit to better immunity to heavy loading. VSVD improves the responsiveness of FASTRAC because of the shorter feedback loops. With or without VSVD, ERICA+ makes more abrupt changes in ACR while FASTRAC exhibits much more smoother transitions. This nature provides better steady state behaviour and more consistent throughput. We illustrated VSVD coupling and min-max fairness by the results collected. We examined the responsiveness of FASTRAC under VSVD by adding delayed sources to the configuration. FASTRAC proves to effectively consume excess bandwidth when there are sharp changes in ABR traffic volume due to an ABR source becoming active/inactive. We characterized the links with more realistic loading by adding VBR background traffic. With or without VSVD, FASTRAC is resistant to background traffic noise. The shorter feedback loop delay given by VSVD enhances FASTRAC background noise tolerance. We raised the boundary condition test by increasing the long propagation delay in the GEO configuration. This scenario tests robustness and reliability. FASTRAC remains poised at protecting the queues of the terrestrial network from the longer long propagation delay in the GEO satellite network. Results show consistency and are also comparable with ERICA+'s results.

6.1 Future Work

To render FASTRAC as a completely practical algorithm under the presence of satellite links (very long propagation delay), a number of future study topics mentioned below needs to be addressed. Our investigation shows that the dimensioning of thresholds for optional linear queue control functions is not quite deterministic. Furthermore, although queue sizes are comparable to other algorithms such as ERICA+, they are still too large to be practical. Queue control functions should help reduce the buffer requirement. Queue control functions for both FASTRAC with or without VSVD should be studied in more details. Further tests of robustness would be to introduce data and RM cell loss and connection timeouts. Robustness is an attribute that is very important to network operators. We have been studying persistent sources that only provide boundary condition testing of the algorithm. Adding bursty internet traffic to sources in the

satellite configuration allows us to study more real world behaviour. Bursty traffic should also reduce the buffer requirement. We have been concentrating more to focused experiments designed to test various aspects of the control algorithm such as effect of long propagation delay to observe stability and robustness, sharp bandwidth changes to observe responsiveness, background traffic to observe noise tolerance, and multiple sources to observe fairness and scalability. A new satellite configuration having sources with different propagation delays and paths through the network would increase the randomness of cell delay. This scenario adds more realism to tests.

References

[ATMF94] ATM Forum, Traffic Management Subworking Group, Test Configurations Sub-sub Working Group, "Test Configurations for Fairness and Other Tests", July 1994.

[ATMF96] ATM Forum, "ATM Forum Traffic Management Specification, Version 4.0", April 1996.

[ATMF97] ATM Forum, Traffic Management Subworking Group, ATM FORUM Document Number: 97-1086R1, "Per-VC Rate Allocation Techniques for ABR Feedback in VS/VD Networks", February 1997.

[AwOu99a] Aweya, J., Ouellette, M., and et al "A Simple, Scalable, Provably Stable, Explicit Rate Computation scheme for Flow Control in Computer Networks", *Nortelnetworks Technical Report # 10-270-99*, September 1999.

[AwOu99b] Aweya, J., Ouellette, M., and et al, "Virtual Source/ Virtual destination (VS/VD) feature for ABR Service", *Nortelnetworks Technical Report # 10-263-99*, May 1999.

[AwOu01a] Aweya, J., Ouellette, M., and et al "Discrete-Time Analysis of a Rate Control Mechanism", *Performance Evaluation: An International Journal (Elseiver Science.)*, Vol 43, Issue 2-3, pp. 63-94, February 2001.

[AwOu01b] Aweya, J., Ouellette, M., and et al "A Simple, Scalable, Provably Stable, Explicit Rate Computation scheme for Flow Control in Computer Networks", To be published in *International Journal of Communication Systems (John Wiley & Sons)*, February 2001.

[ChJa89] Chiu, D., Jain, R., "Analysis of Increase/Decrease Algorithms for Congestion Avoidance in Computer Networks", *Computer Networks and ISDN Systems*, vol. 17, no. 1, pp. 1-14, June 1989.

[DiSt90] DiStefano, J.J., Stubberud, A. R., and et al, "Feedback and Control Systems", *Schaum's Outline Series, McGraw-Hill*, New York, 1990.

[GoCa98] Goya, R., Cai, X., and et al, "Per-VC Rate Allocation Techniques for ATM-ABR Virtual Source Virtual Destination Networks", *Proceedings of Globecom '98*, November 1998.

[FaJa99] Fahmy, S., Jain, R., and et al, "ABR Engineering: Roles and Guidelines for Setting ABR Parameters", *Journal of Computer Networks*, February 1999.

[Haye94] Hayes, J.F., "Modeling and Analysis of Computer Communications Networks", *Plenum Press*, New York, 1994.

[HeBe97] Hernandez-Valencia E., Benmohamed L., and et al, "Rate Control Algorithms for ATM ABR Service", *European Transaction on Telecommunications*, vol.8, no. 1, pp. 7-20, January-February 1997.

[HuYa99] Huynh, K., Yang, O., "ABR Controllability under Sharp Bandwidth Changes and Long Propagation Delay", *Opnetwork99 Conference Paper*, September, 1999.

[JaKa96] Jain R., Kalyanaraman S., and et al, "Source Behavior for ATM ABR Traffic Management: An Explanantion", *IEEE Communications Magazine*, pp. 50-57, November 1996.

[Kris97] Krishnan R., "Rate-based Control Schemes for ABR Traffic – Design Principles and Performance Comparison", *Computer Networks and ISDN Systems*, vol. 29, pp. 583-593, 1997.

[MIL3] MIL 3 Inc., "OPNET Modeler Manuals", *OPNET Version 6*, 1999.

[PiSc96] Pitts, J.M., Schormas, J.A., "Introduction to ATM design and Performance", *John Wiley & Sons Ltd.*, West Sussex, 1996.

[Robe94] Roberts L., "Enhanced PRCA (Proportional Rate-Control Algorithm)", *ATM_Forum/94-0735(R1)*, 1994.

[Schw96] Schwartz, M., "Broadband Integrated Networks", *Prentice Hall*, New Jersey, 1996.

[SuOk98] Sudo T., Okuda M., and et al, "Analysis of ABR Behavior over ATM-Based Broadband Access Networks", *IEEE Transactions in Communications*, vol. E81-B, No. 2, pp. 402-408, February 1998.

[SuTz94] Sui K., Tzeng H., "Adaptive Proportional Rate Control Algorithm for ABR Service in ATM Networks", *UC Irvine Technical Report*, no. 94-07-01, 1994.

[VaFa98] Vandalore, B., Fahmy, S., and et al, "A Definition of General Weighted Fairness and its Support in Explicit Rate Switch Algorithms", *Proceedings of ICNP'98*, pp. 22-30, October 1998.

[ZhYa96] Zhang, H., Yang O., and et al "Adaptive rate-based congestion control in ATM switching networks", *Computer Syst. Sci. & Eng.*, vol 6, pp. 361-367, 1996.

[ZhYa97] Zhang, H., Yang, O., and et al, "Design of robust congestion controllers for ATM networks", *Proc. IEEE INFOCOM'97*, pp. 302-309, 1997.

[ZhYa98] Zhang, H., Yang, O., and et al, "A Hop-by-hop Flow Controller for a Virtucal Path", *Computer Networks*, vol. 32 (2000), pp.99 - 119.

Appendix A: Surveyed Algorithms

This appendix describes the equations of surveyed algorithms from [HeBe97]. A generic feedback loop is shown in Figure A- 1 is used to describe the surveyed control algorithms. The round trip time denoted by τ primarily consists of propagation delay. The service rate at the switch node is denoted by μ. q_T represents the bottleneck queue threshold.

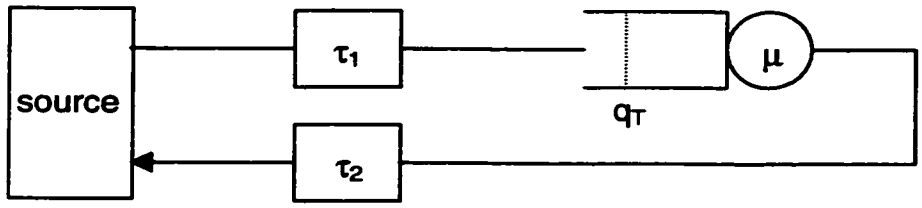


Figure A- 1: Generic ABR Feedback Loop

$$\frac{d}{dt}ACR(t) = \begin{cases} RIF \cdot PCR \cdot \min[\frac{ACR(t-\tau)}{Nrm}, \frac{\mu}{Nrm}] & \text{if } q(t-\tau_2) = 0 \\ RIF \cdot PCR \cdot \frac{\mu}{Nrm} & \text{if } 0 < q(t-\tau_2) < q_T \\ -\frac{RDF \cdot \mu}{N_{conn} \cdot Nrm} & \text{if } q(t-\tau_2) > q_T \end{cases} \quad \text{(EQ A.1)}$$

μ service rate at switch

N_{conn} number of active VCs at bottleneck queue

Source: [HeBe97], equation 7

$$ER(n+1) = ER(n) - a_i[q(n) - q_T] \qquad \text{(EQ A.2)}$$

a_i	gain for connection i
ER(n)	the explicit rate at time slot n
q(n)	the queue of the switch at time slot n

Source: [HeBe97], equation 11

$$MACR(t) = (1-\alpha)\cdot MACR(t^-) + \alpha \cdot \lambda_j(t)$$

$$u_j(t^+) = \begin{cases} +1 & \text{if } \rho \geq \rho_T \text{ and } \lambda_j(t) < MACR(t) \\ -1 & \text{if } \rho \geq \rho_T \text{ and } \lambda_j(t) > MACR(t) \end{cases} \qquad \text{(EQ A.3)}$$

MACR(t)	the estimated mean allowed cell rate
α	exponential averaging weight
$u_j(t)$	congestion state where (+1 is no congestion, - 1 is congestion)
ρ_T	utilization threshold
ρ	utilization of resource at switch
$\lambda_j(t)$	estimated current arrival rate of connection j

Source: [HeBe97], equation 10, [SuTz94]

$$MACR(t) = \alpha \cdot MACR(t^-) + (1-\alpha) \cdot ER_j(t)$$

$$ER_j(t^+) = \begin{cases} \beta \cdot MACR(t) & \text{if } \rho \geq \rho_T \text{ and } ER_j(t) > MACR(t) \\ ER_j(t) & \text{if } \rho \geq \rho_T \text{ and } ER_j(t) < MACR(t) \end{cases} \quad \text{(EQ A.4)}$$

MACR(t)	the estimated mean allowed cell rate
α	exponential averaging weight
$ER_j(t)$	explicit rate of connection j
ρ_T	utilization threshold
ρ	utilization of resource at switch
β	rate decrease factor

Source: [HeBe97], equation 16, [Robe94]

Appendix B: Derivation of FASTRAC Formulae

This appendix provides a summary of the derivation of FASTRAC Formulae found in [AwOu99a].

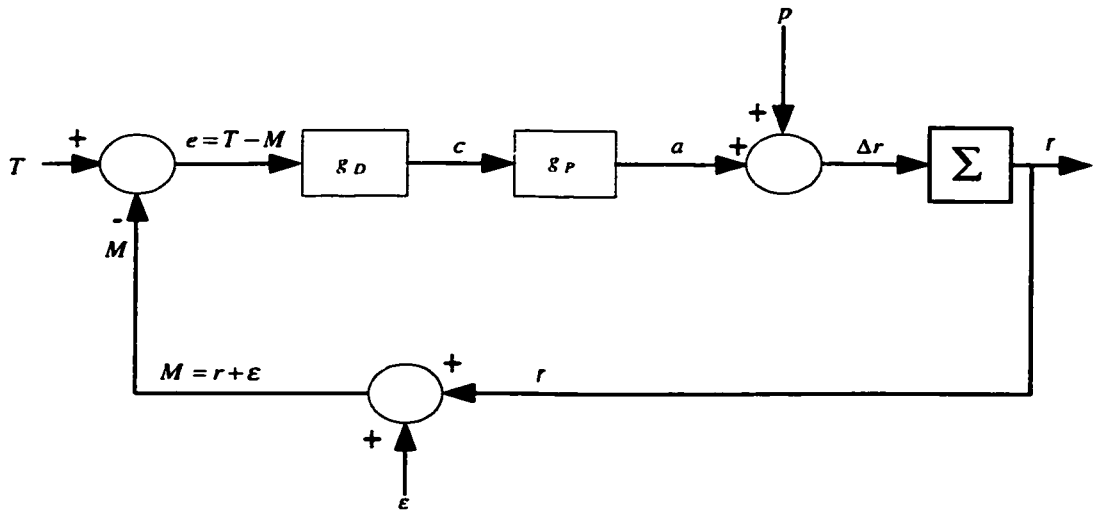


Figure B- 1: FASTRAC Control Model

Figure B- 1 shows an abstraction of the FASTRAC control loop for a traffic flow through a switch. The input to the control process is the target service rate, T. This value is typically the output link capacity minus $C_{high}(n)$. The process output r is detected by the measuring device, which because of sampling distributions, measurement inaccuracies, or both, adds an error ε to the true value r. The value M available for comparison with the desired level of operation T is thus the sum of the true value and the measurement error. This value represents the estimated ABR arrival rate. The error signal e is given by

$$e = T - M = T - r - \varepsilon \qquad \text{(EQ B- 1)}$$

This error signal represents the bandwidth mismatch between the arrival rate and the target service rate. The controller produces a control signal, c, that is a specified function g_D, of e. The control signal c reacts with the physical properties of the process being controlled to produce a controlled change a in the process output. g_P is used here to represent the functional relationship between a and c. The output is also influenced

by the effects of uncontrollable variables p. Uncontrolled variables include internal system disturbances. Since a is defined as an instantaneous adjustment and p as an instantaneous perturbation, each affects the rate of change of output $\dot{r}$ (Δr). The actual output r is the integration of these rates of changes over time. Assume a slotted time system. The nth sampling period corresponds to the time t in the range $n \cdot t_m \leq t < (n+1) \cdot t_m$. The system difference equation is derived as follows

$$e(n) = T(n) - M(n) = T(n) - r(n) - \varepsilon(n). \qquad \text{(EQ B- 2)}$$

Using the *proportional-control* rule with $g_D = \alpha$ being the proportionality constant,

$$c(n) = \alpha e(n) = \alpha T(n) - \alpha r(n) - \alpha \varepsilon(n). \qquad \text{(EQ B- 3)}$$

α is the control gain of the control process. Therefore Since there is no delay in the control process the desired adjustment $c(n)$ is completed within the sampling period. Therefore,

$$a(n) = c(n), \qquad \text{(EQ B- 4)}$$

and g_p is simply an identity function,

$$g_p = \delta(n). \qquad \text{(EQ B- 5)}$$

Likewise, the control decision rule is simply

$$g_D = \alpha\delta(n). \qquad \text{(EQ B- 6)}$$

Therefore,

$$a(n) = c(n) = \alpha e(n) = \alpha[T(n) - r(n) - \varepsilon(n)].$$

In all further discussion of process controllers the rapid-response feature will be assumed. Assuming the perturbations are addictive, the difference between $r(n+1)$and $r(n)$ must be the sum of $a(n)$ and $p(n)$. Thus,

$$\Delta r = r(n+1) - r(n) = a(n) + p(n) \qquad \text{(EQ B- 7)}$$

or

$$r(n+1)=r(n)+a(n)+p(n). \qquad \text{(EQ B- 8)}$$

Substitution of (3) into (7) and rearranging yields

$$r(n+1)-(1-\alpha)r(n)=\alpha T-\alpha\varepsilon(n)+p(n) \qquad \text{(EQ B- 9)}$$

which is the desired system difference equation. Thus, the process output at instant $(n+1)$ is equal to the sum of the process output at the previous sampling instant n, the adjustment just made, and the net effect of all perturbations caused by factors external to the system which occurred during the period $n\cdot t_m \le t < (n+1)\cdot t_m$.

From the above discussion, the time evolution equation for a delayless system is

$$r(n+1)=r(n)+\alpha[T-M(n)] \qquad \text{(EQ B- 10)}$$

In practice, the traffic load is never negative. However, $r(n)$ could become negative in the above equation, so in the implementation of the control algorithms, the $r(n)$ values are bounded to realistic values as indicated below, where T_{max} is an upper bound on $r(n+1)$. T_{max} represents the maximum desired service rate of the output port. Therefore,

$$r(n+1)=[r(n)+\alpha[T-M(n)]]\Big|_0^{T_{max}}. \qquad \text{(EQ B- 11)}$$

The control model described above concentrates on one single source in the flow control process. For N sources transmitting data to the bottlenecked node, we redefine

$$r(n+1)=r(n)+\alpha\left[\frac{T-M(n)}{N}\right] \qquad \text{(EQ B- 12)}$$

where we assume that the N sources share the effort in reducing the rate mismatch between T and M. Intuitively, the above equation then gives the following per-source rate control model

$$r(n+1)=r(n)+\alpha[T'-M'(n)] \qquad \text{(EQ B- 13)}$$

where $T' = T/N$ is an equivalent per-source target load and $M'(n) = M(n)/N$ a per-source estimated traffic load.

The control model can be extended to account for feedback control delays. That is the time lag after which the control decisions (i.e., rate allocations) take effect at the bottlenecked node. Let d be this time interval, in units of t_m. If RTT is the round-trip time from the source to destination and back, then $d \leq RTT$ depending on the location of the bottlenecked node from the source. We have $d = RTT$ as the extreme case if the destination is directly attached to the bottlenecked node. For a single source transmitting data to the bottlenecked node, the system difference equation for a proportional controller with a feedback delay of d time intervals is

$$r(n+1) - r(n) + \alpha r(n-d) = \alpha T - \alpha \varepsilon(n-d) \qquad \text{(EQ B- 14)}$$

or

$$r(n+d+1) - r(n+d) + \alpha r(n) = \alpha T - \alpha \varepsilon(n). \qquad \text{(EQ B- 15)}$$

Rewriting the above equation as

$$r(n+d+1) = r(n+d) + \alpha [T - r(n) - \varepsilon(n)]$$

shows that the rate allocation $r(n+d+1)$, which will take effect at the bottleneck node at time $(n+d+1)$, is computed based on the rate allocation $r(n+d)$ (which will take effect at time $(n+d)$) and the current error $e(n) = [T - r(n) - \varepsilon(n)]$.

B.1 Sampling Rate Heuristic

An engineering heuristic must be put in place to choose the value of t_m. As a guideline, we require t_m to be sufficiently large to produce a rate resolution of 0.5 to 1% of the target link rate. Thus the sampling interval t_m is determined as

$$t_m = \frac{1}{(\phi/100)\rho_t C} \text{ seconds} \qquad \text{(EQ B- 16)}$$

where $\phi/100$ is a rate resolution factor (in the range of 0.5 to 1%), ρ_t is the target link utilization factor (e.g. 90%), and C is the link capacity (in cells/seconds). For an OC3 link, t_m = 0.5ms to 1ms which is a reasonable compromise between the computational burden and resulting system performance. This allows easy implementation at high speed where the computations can even be done in software.

Filtering is another technique that be applied to remove high frequency fluctuations, at the expense of more sophisticated buffering design. The filtering can be done using a moving average (MA) filter, an exponentially weighted moving average (EWMA) filter, or any other suitable filtering technique. However, we only suggest applying filters in measuring rates of less bursty background (CBR/VBR) traffic to help to improve performance. Filtering the ABR rate is outside of the current scope of the FASTRAC model.

B.2 Stability Analysis

The stability of the flow control process can be determined by the stability analysis of the above $(d+1)$th order difference equation with constant real coefficients. The Routh-Hurwitz test can be applied to EQ B-15 for fixed values of d and α. Here we establish the relationship between d and α that guarantees the stability of the $r(n)$ process in the absence of noise $\varepsilon(n)$ (i.e., $\varepsilon(n)=0$ for all n). The constraint $0 \le r(n) \le T_{max}$ will also not be used. The stability of the flow control problem is determined by analyzing the roots of the denominator $(F_d(z))$ of the z-transform $F(z)=\sum_{n=0}^{\infty} r(n)z^n$. The denominator $F_d(z)$ is given by the polynomial equation

$$F_d(z)=z^{d+1}-z^d+\alpha,\ 0<\alpha<1. \qquad \text{(EQ B- 17)}$$

The system is stable if the roots of $F_d(z)=0$ lie inside the unit circle in z space. According to theorems in digital control theory, in order for the system to be stable, all roots of the characteristics need to be within the stability boundary, which is the unit

circle. In other words, for any root z, we need $|z|<1$. The location of these roots relative to the unit circle can be determined by using a bilinear transform and then applying the Routh-Hurwitz test [DiSt90]. The stability of a linear discrete-time system expressed in the z-domain can also be determined using the s-plane methods developed for continuous systems (e.g., Routh-Hurwitz). The following bilinear transformation of the complex variable z into the new complex variable h given by the expression

$$h=\frac{z+1}{z-1}$$

transforms the interior of the unit circle in the z-plane onto the left half of the h-plane [DiSt90]. Therefore the stability of a discrete-time system with characteristic polynomial $F_d(z)$ can be determined by examining the locations of the roots of

$$\hat{F}_d(h)=F_d(z)\big|_{z=(h+1)/(h-1)}=0$$

in the h-plane, treating h like s and using s-plane techniques to establish stability properties. The bilinear transformation results in the relation

$$\hat{F}_d(h)=(h+1)^{d+1}-(h+1)^d(h-1)+\alpha(h-1)^{d+1}=0. \qquad \text{(EQ B- 18)}$$

The roots of this polynomial equation must lie in the left-half h-plane for stability. Expanding and grouping like powers of h gives

$$\hat{F}_d(h)=\sum_{k=0}^{d+1}\frac{d!}{k!(d+1-k)!}\left[2k+(-1)^k(d+1)\alpha\right]h^{d+1-k} \qquad \text{(EQ B- 19)}$$

The Routh-Hurwitz test is performed on arrays of numbers generated by the coefficients of the polynomial equation

$$\hat{F}_d(h)=c_0h^{d+1}+c_1h^d+\ldots+c_dh+c_{d+1}=0. \qquad \text{(EQ B- 20)}$$

The entries for the first two rows of the Routh array are, therefore, given by

$$G_k=2k+(-1)^k(d+1)\alpha. \qquad \text{(EQ B- 21)}$$

The entries in the first two rows of the Routh-Hurwitz array for EQ-18 are

$$G_0 = \alpha \qquad \text{(EQ B- 22)}$$

and

$$G_1 = 2-(d+1)\alpha. \qquad \text{(EQ B- 23)}$$

For stability, both of these must be of the same sign. Therefore, since $\alpha > 0$ and $d > 0$, the condition on G_1 requires that

$$0 < 2-(d+1)\alpha$$

or

$$\alpha < \frac{2}{d+1} \qquad \text{(EQ B- 24)}$$

For $\alpha < 0$, however, $G_1 < 0$ requires that $\alpha > 2/(d+1) > 0$, which is impossible for negative α. This confirms the nonnegativity of α and also indicates that the leading entries of all rows in the Routh array must be positive. EQ B-23 is the desired relationship between α and d that is required for the flow control process to be stable. The discussion above shows that even when the process of estimating the load is perfect (i.e., no errors), oscillations in the load occur if the system tries to correct more than a certain fraction of the difference between the estimated load and the target load per control interval. This is due to a periodic discrepancy between the estimated load and the actual load caused by time lags.

For multiple sources with different feedback delays, $d_1 < d_2 < ... < d_N$, it is easily inferred from the above discussion that setting $\alpha < 2/(d_N + 1)$ ensures fairness and global stability of the flow control process. Because, intuitively if we consider the per-source flow control discussion above, the only single value of α that ensures global stability is $\alpha < 2/(d_N + 1)$, since

$$\frac{2}{d_N + 1} < \frac{2}{d_{N-1} + 1} < ... < \frac{2}{d_1 + 1}.$$

Thus, in our flow control algorithm, the value of α used is

$$\alpha < \frac{2}{d+1}, \quad d = \max(d_1, d_2, ..., d_N) \qquad \text{(EQ B- 25)}$$

The value of d is derived based on the parameter RTT that accounts for the propagation delay, queueing delay and processing delay. Thus, for N connections through the bottlenecked node we have

$$d = \left\lceil \frac{maxRTT}{t_m} \right\rceil, \; maxRTT = \max\{RTT_1, ..., RTT_N\} \qquad \text{(EQ B- 26)}$$

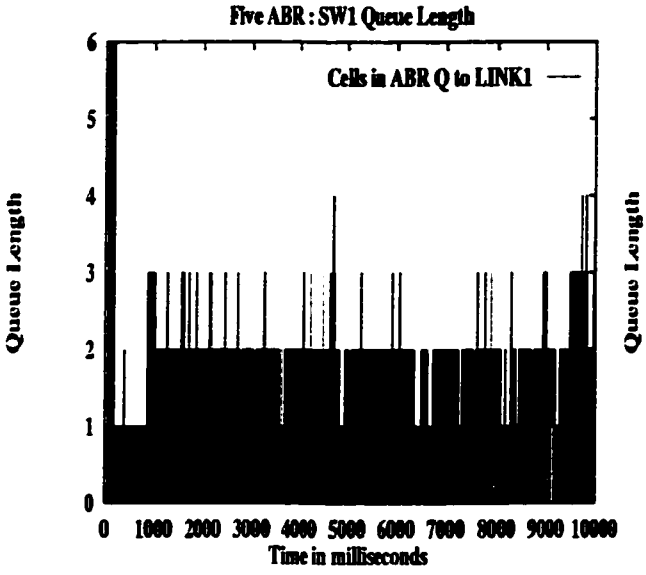
The stability constraint given by EQ B.23 is necessary but may not be sufficient for all values of d [AwOu99a]. Note that the above stability analysis was done assuming a noise-free environment (i.e., $\varepsilon(n) = 0$). To account for system noise and disturbances that might be encountered during the course of the flow control process, we select α to be

$$\alpha = \frac{1}{d+1} \qquad \text{(EQ B- 27)}$$

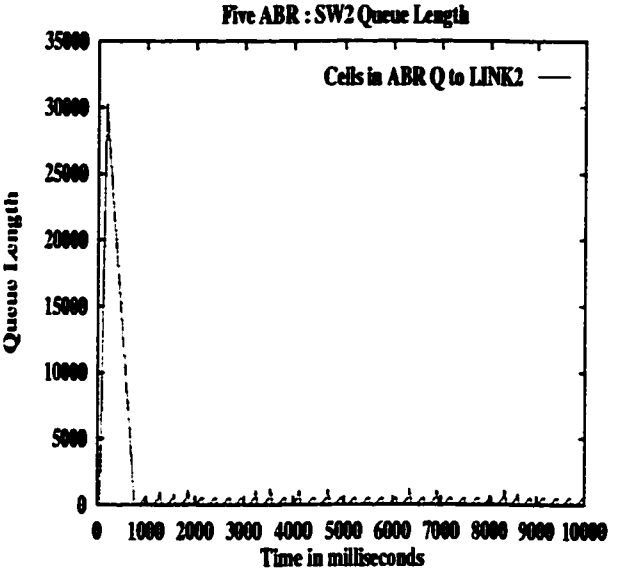
which is within the stability bound stated above (i.e., $0 < \alpha < 2/(d+1)$).

Appendix C: Performance Results For ERICA+

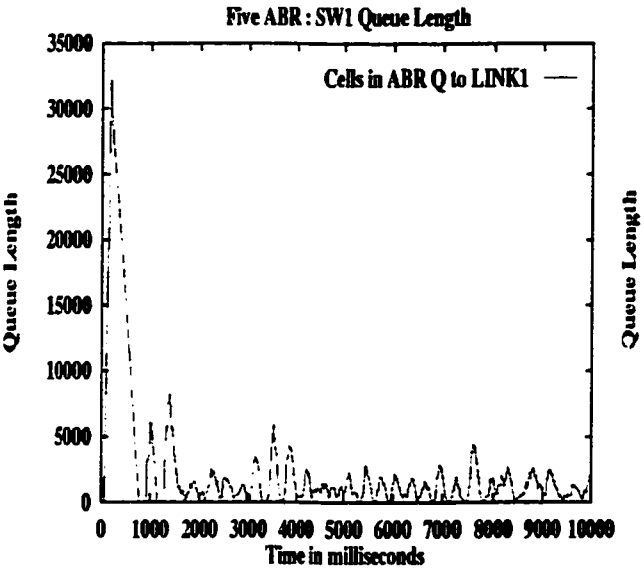
This appendix contains the performance results for the satellite configuration with and without VSVD using ERICA+ control algorithm. The results presented in this section are extracted from [GoCa98].



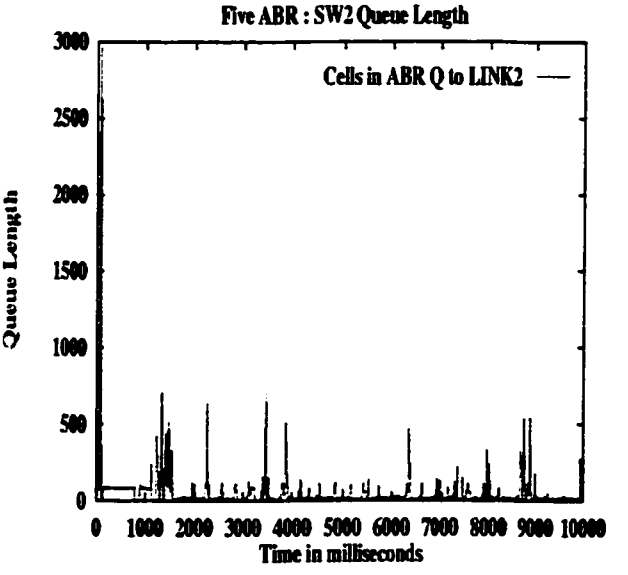
(a) Switch 1 Queue Size – No VSVD



(b) Switch 2 Queue Size – No VSVD

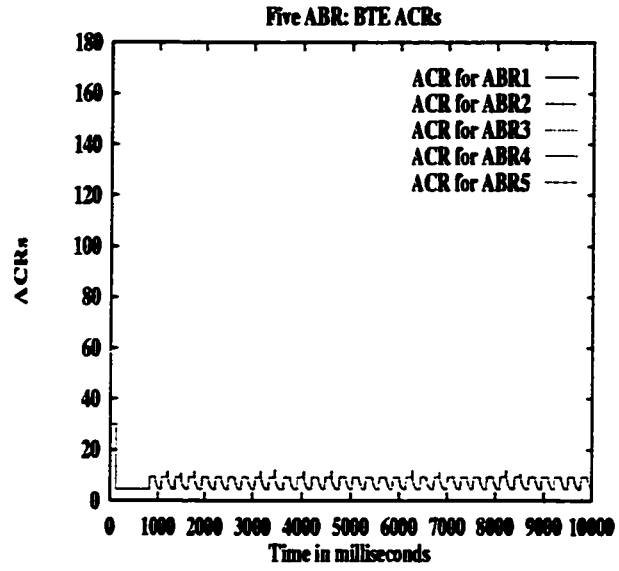


(c) Switch 1 Queue Size – VSVD

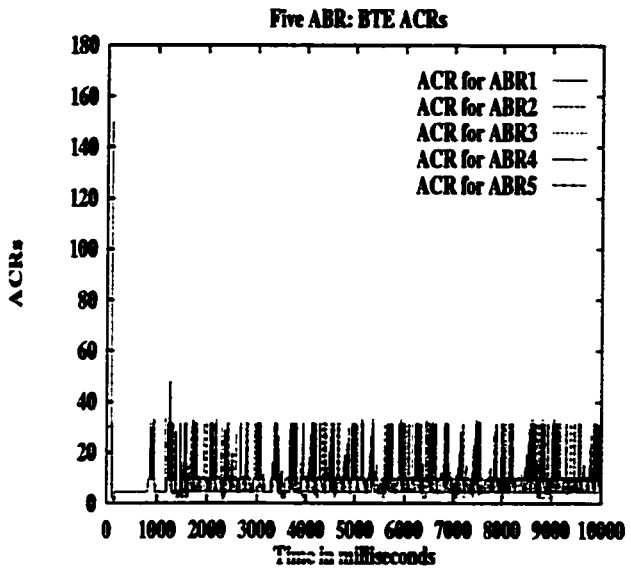


(d) Switch 2 Queue Size – VSVD

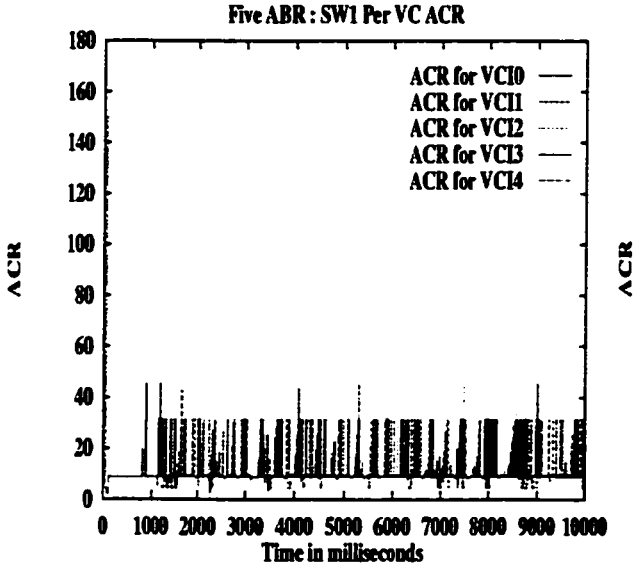
Figure C- 1: Switch Queue Size for VSVD and No VSVD when using ERICA+ in LEO Configuration



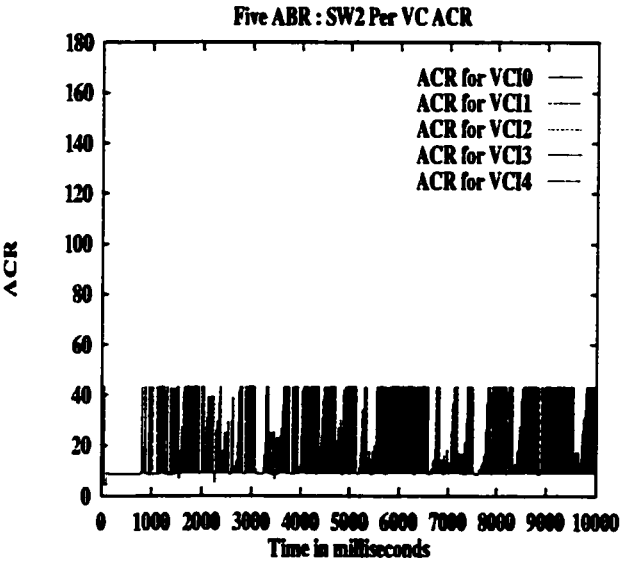
(a) ACR at Source - No VSVD



(b) ACR at source –VSVD

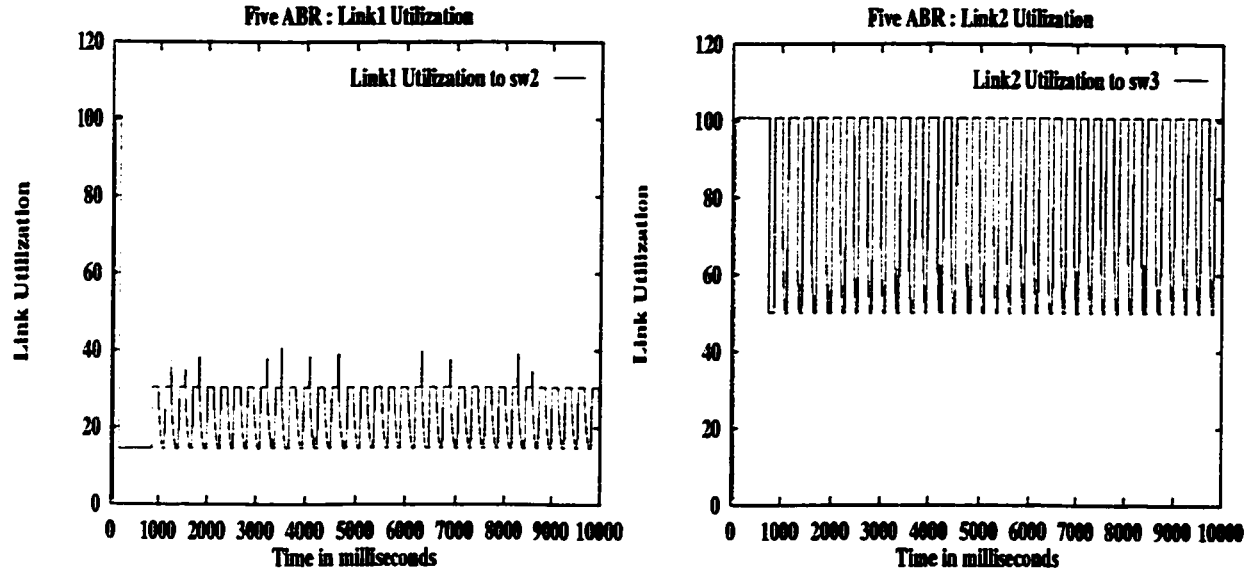


(c) ACR at Switch 1 – VSVD

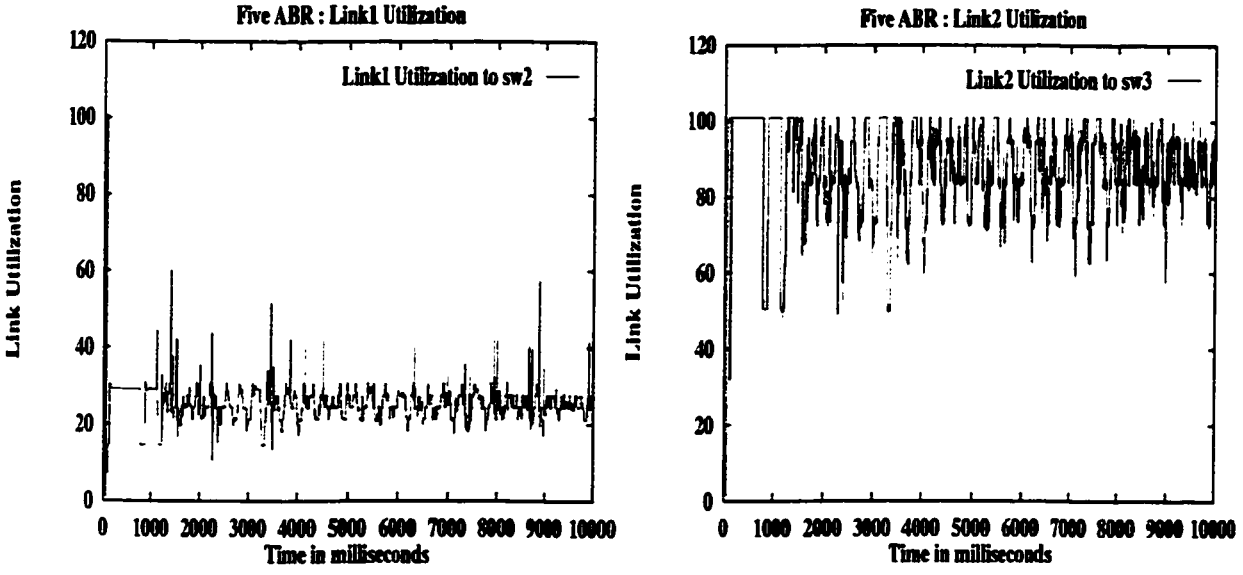


(d) ACR at Switch 2 – VSVD

Figure C- 2: ACR for VSVD and No VSVD when using ERICA+ in LEO Configuration

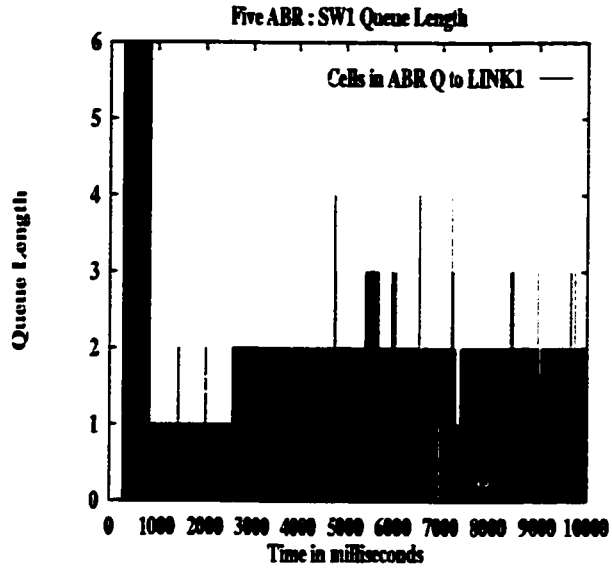


(a) Switch 1 Link Utilization – No VSVD (b) Switch 2 Link Utilization – No VSVD

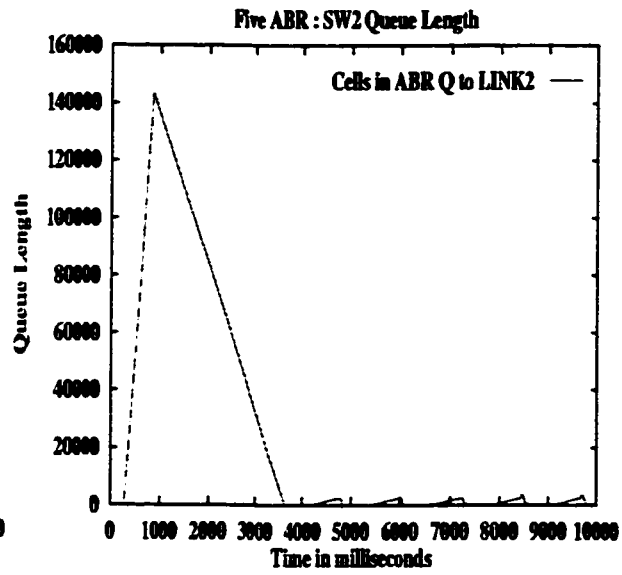


(c) Switch 1 Link Utilization –VSVD (d) Switch 2 Link Utilization – VSVD

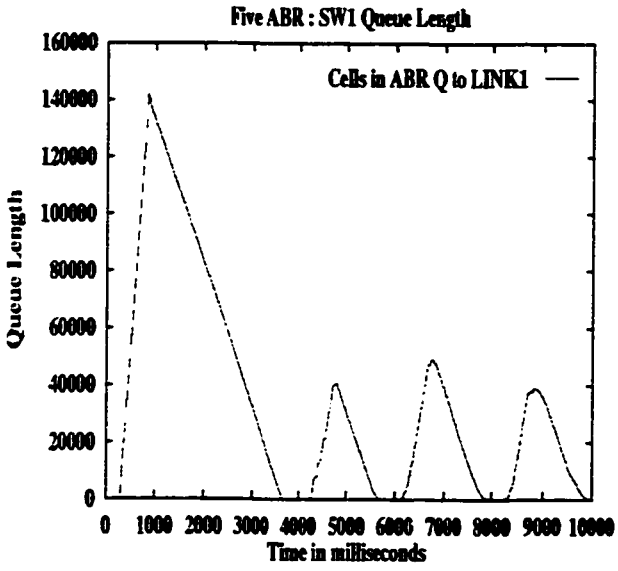
Figure C- 3: Link Utilization for VSVD and No VSVD when using ERICA+ in LEO Configuration



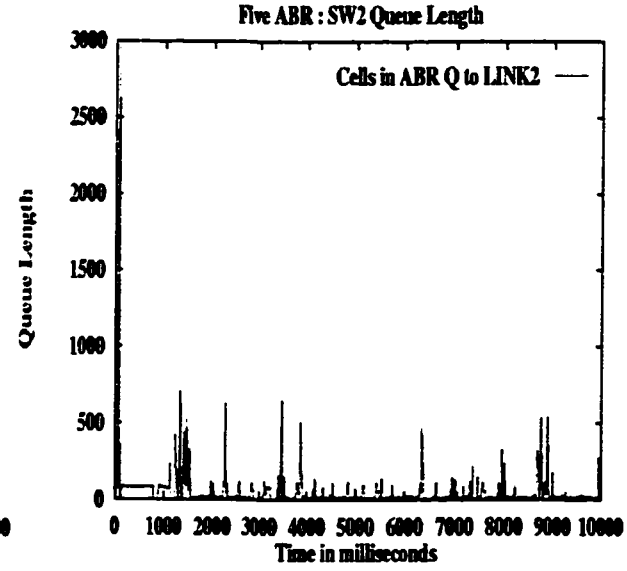
(a) Switch 1 Queue Size – No VSVD



(b) Switch 2 Queue Size – No VSVD

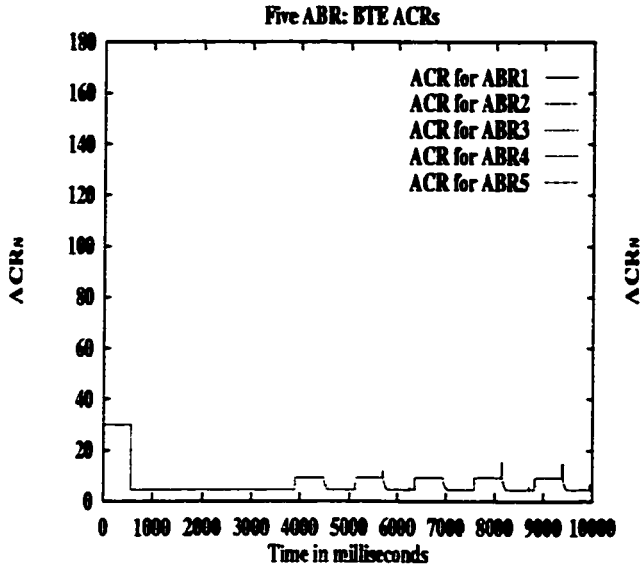


(c) Switch 1 Queue Size – VSVD

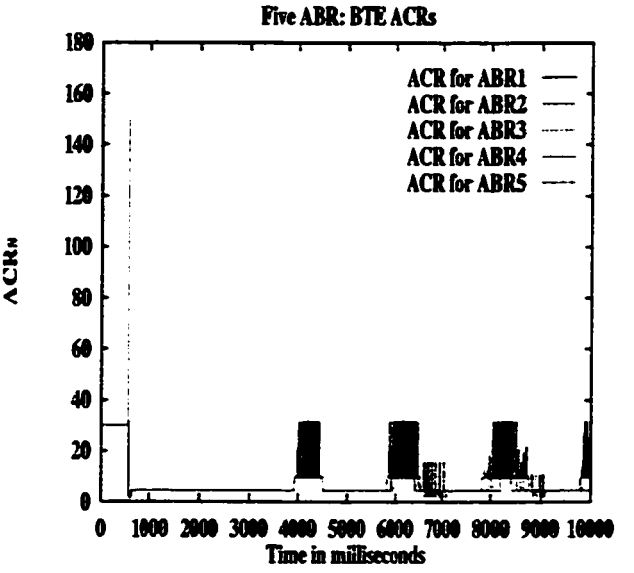


(d) Switch 2 Queue Size – VSVD

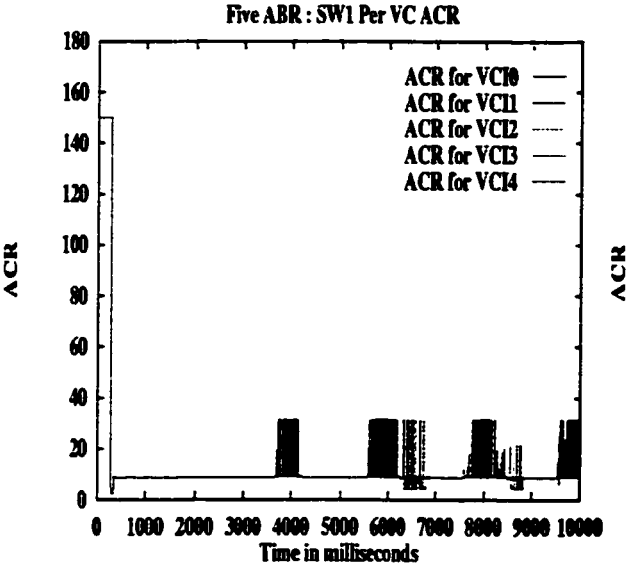
Figure C- 4: Switch Queue Size for VSVD and No VSVD when using ERICA+ in GEO Configuration



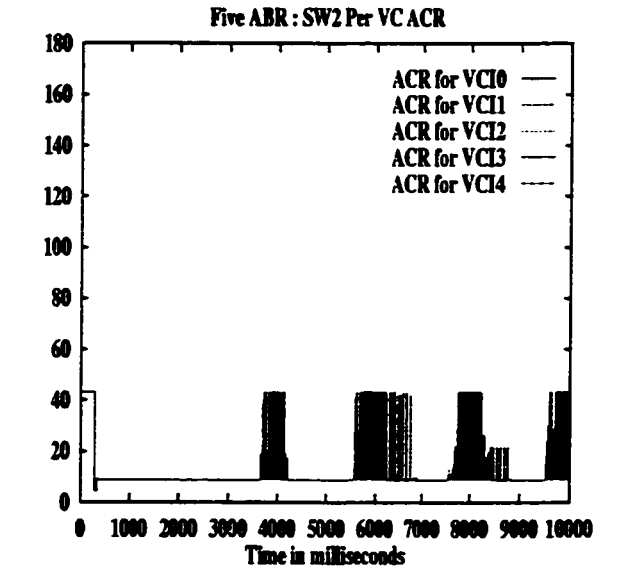
(a) ACR at Source - No VSVD



(b) ACR at source –VSVD

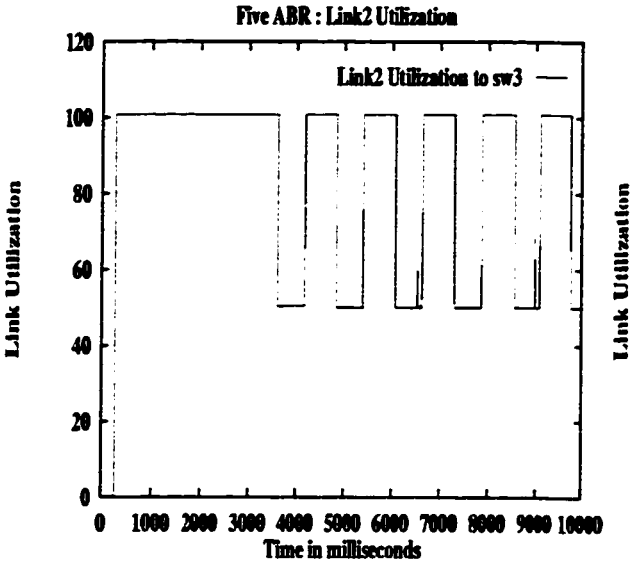


(c) ACR at Switch 1 – VSVD

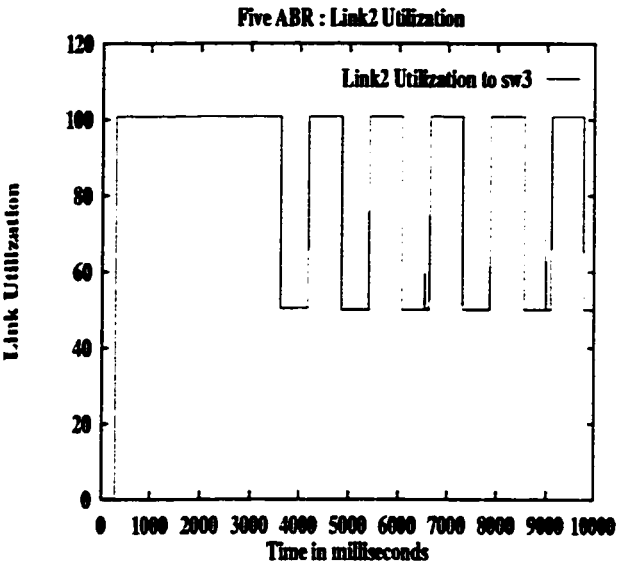


(d) ACR at Switch 2 – VSVD

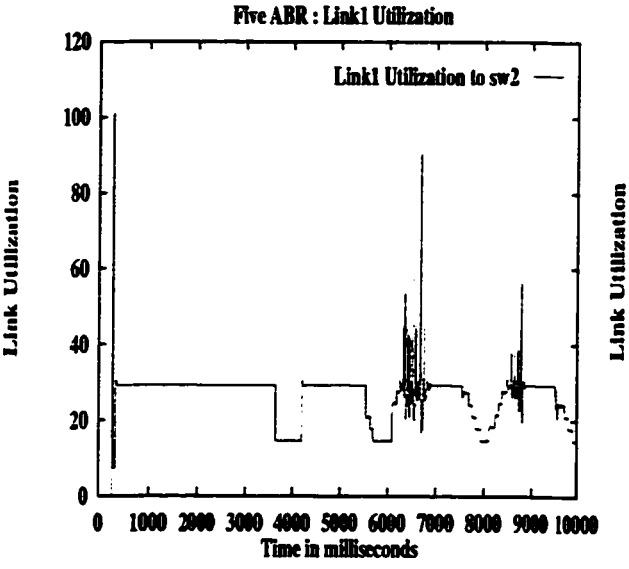
Figure C- 5: ACR for VSVD and No VSVD when using ERICA+ in GEO Configuration



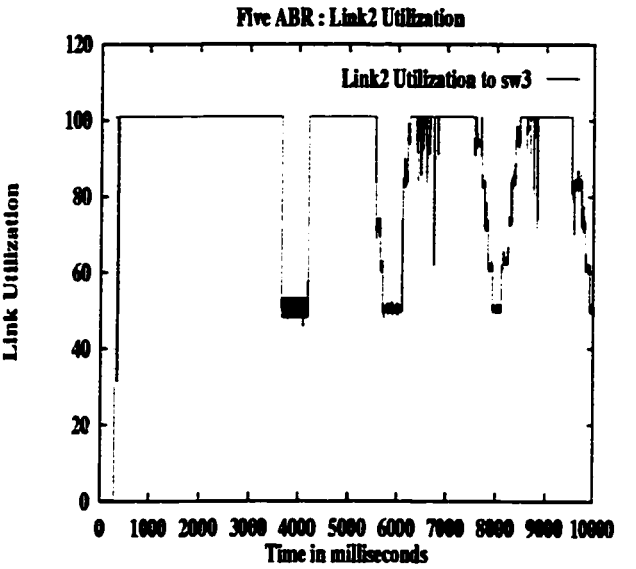
(a) Switch 1 Link Utilization – No VSVD



(b) Switch 2 Link Utilization – No VSVD



(c) Switch 1 Link Utilization –VSVD



(d) Switch 2 Link Utilization – VSVD

Figure C- 6: Link Utilization for VSVD and No VSVD when using ERICA+ in GEO Configuration