Reactive Traffic Control Mechanisms for Communication Networks with Self-Similar Bandwidth Demands

Sven A. M. Östring, BE (Hons.)

A thesis presented for the degree of
Doctor of Philosophy
in
Electrical and Electronic Engineering
at the
University of Canterbury,
Christchurch, New Zealand.

January 2001

ABSTRACT

Communication network architectures are in the process of being redesigned so that many different services are integrated within the same network. Due to this integration, traffic management algorithms need to balance the requirements of the traffic which the algorithms are directly controlling with Quality of Service (QoS) requirements of other classes of traffic which will be encountered in the network. Of particular interest is one class of traffic, termed *elastic traffic*, that responds to dynamic feedback from the network regarding the amount of available resources within the network. Examples of this type of traffic include the Available Bit Rate (ABR) service in Asynchronous Transfer Mode (ATM) networks and connections using Transmission Control Protocol (TCP) in the Internet. Both examples aim to utilise available bandwidth within a network.

Reactive traffic management, like that which occurs in the ABR service and TCP, depends explicitly on the dynamic bandwidth requirements of other traffic which is currently using the network. In particular, there is significant evidence that a wide range of network traffic, including Ethernet, World Wide Web, Varible Bit Rate video and signalling traffic, is self-similar. The term self-similar refers to the particular characteristic of network traffic to remain bursty over a wide range of time scales. A closely associated characteristic of self-similar traffic is its long-range dependence (LRD), which refers to the significant correlations that occur with the traffic. By utilising these correlations, greater predictability of network traffic can be achieved, and hence the performance of reactive traffic management algorithms can be enhanced.

A predictive rate control algorithm, called PERC (Predictive Explicit Rate Control), is proposed in this thesis which is targetted to the ABR service in ATM networks. By incorporating the LRD stochastic structure of background traffic, measurements of the bandwidth requirements of background traffic, and the delay associated with a particular ABR connection, a predictive algorithm is defined which provides explicit rate information that is conveyed to ABR sources. An enhancement to PERC is also described. This algorithm, called PERC+, uses previous control information to correct prediction errors that occur for connections with larger round-trip delay. These algorithms have been extensively analysed with regards to their network performance, and simulation results show that queue lengths and cell loss rates are significantly reduced when these algorithms are deployed. An adaptive version of PERC has also been developed using real-time parameter estimates of self-similar traffic. This has excellent performance compared with standard ABR rate control algorithms such as ERICA.

Since PERC and its enhancement PERC+ have explicitly utilised the index of self-similarity, known as the Hurst parameter, the sensitivity of these algorithms to this parameter can be determined analytically. Research work described in this thesis shows that the algorithms have an asymmetric sensitivity to the Hurst parameter, with significant sensitivity in the region where the parameter is underestimated as being close to 0.5. Simulation results reveal the same bias in the performance of the algorithm with regards to the Hurst parameter. In contrast, PERC is insensitive to estimates of the mean, using the sample mean estimator, and estimates of the traffic variance, which is due to the algorithm primarily utilising the correlation structure of the traffic to predict future bandwidth requirements.

Sensitivity analysis falls into the area of investigative research, but it naturally leads to the area of robust control, where algorithms are designed so that uncertainity in traffic parameter estimation or modelling can be accommodated. An alternative robust design approach, to the standard maximum entropy approach, is proposed in this thesis that uses the maximum likelihood function to develop the predictive rate controller. The likelihood function defines the proximity of a specific traffic model to the traffic data, and hence gives a measure of the performance of a chosen model. Maximising the likelihood function leads to optimising robust performance, and it is shown, through simulations, that the system performance is close to the optimal performance as compared with maximising the spectral entropy.

There is still debate regarding the influence of LRD on network performance. This thesis also considers the question of the influence of LRD on traffic predictability, and demonstrates that predictive rate control algorithms that only use short-term correlations have close performance to algorithms that utilise long-term correlations. It is noted that predictors based on LRD still out-perform ones which use short-term correlations, but that there is potential simplification in the design of predictors, since traffic predictability can be achieved using short-term correlations.

This thesis forms a substantial contribution to the understanding of control in the case where self-similar processes form part of the overall system. Rather than doggedly pursuing self-similar control, a broader view has been taken where the performance of algorithms have been considered from a number of perspectives. A number of different research avenues lead on from this work, and these are outlined.

ACKNOWLEDGEMENTS

I would like to thank my research supervisors, Associate Professor Harsha Sirisena and Dr. Irene Hudson, for their guidance of my research over the last few years. I am particularly thankful to Harsha for introducing me to a fascinating research area and for his research ideas, which have aided the progression of my research. I also appreciate Irene's input into my work, especially in pointing out the necessary milestones of the degree itself. Krys Pawlikowski's constructive comments on my research publications have been very valuable. I would like to acknowledge the financial support of the University of Canterbury Doctoral Scholarship, the Royal Society of New Zealand's R. H. T. Bates Postgraduate Scholarship and research funding from two Telecom New Zealand research grants.

I am also very grateful for the hospitality of Professor Fergus O'Brien during my visit to the Software Engineering Research Centre (SERC) at Royal Melbourne Institute of Technology. During this visit I was able to have interesting discussions with Matthew Roughan and Darryl Veitch. While I was in Melbourne, I enjoyed spending time with Richard Nelson, my former fellow PhD student at Canterbury, and his family—Sarah, Daniel and Jacob. Also, my research was enhanced by meeting Eitan Altman, Chadi Barakat and Konstantin Avratchenkov during my stay at the Institut National de Recherche en Informatique et en Automatique (INRIA) in Sophia-Antipolis, France. Philippe Nain kindly hosted this visit, and Konstantin introduced me to some of the quaint *villages perchés* in Provence.

My friends have been vital in giving meaning to areas of my life other than my research. An exception, though, would be Stephen Simpson who provided practical tips along the way, particularly in the area of programming. He also proofread my thesis in the early stages of writing. However, our friendship extends beyond the engineering environment, and we have enjoyed many other things together such as jogging and discussing theology. I also really appreciate my friendship with Stephen Larsen, who has shifted from being my string tutor at the youth orchestra to being a very dependable friend. His encouragement of my going to Cambridge was a significant factor in my decision. I have had some great discussions with Paul Rankin during our weekly lunches, and Jill and Ross Saunders have always been interested in the travel that was necessary in my research. Andrew Rudge and I have enjoyed using our engineering background in developing the technical aspects of 3D Worship, together with the spiritual components with Lance Boulton. Finally, both the discussions and the social fun that I have had with my friends in my Bible study group—Jared and Beatriz French, Arron Galbraith and Naomi Davis, Vicky and Adrian Bell, Tony and Andrea Williams, Donald Relihan and Amy Leach—have made me a lot more human.

Lastly, I would like to thank my family, Elizabeth, Roland and Genevieve, who have been there for me at all the different steps along the way. While I have been grappling with the challenges of research and enjoying opportunities to travel to conferences, they have been very encouraging and have always pointed me to the true purpose of my work.

I would like to dedicate this thesis to my twin sister Genevieve who, despite being younger than me, beat me to the title of Doctor—the rascal!

PREFACE

The following papers were prepared during the course of this work:

[1] Östring, S., Sirisena, H. and Hassan, M., 'Optimization of a rate feedback control with feedforward for traffic management in ATM networks', *Proceedings of APCC '97*, Vol. 3, December, 1997, pp. 1551–1555.

[2] Östring, S., Sirisena, H. and Hudson, I., 'Dual dimensional ABR control scheme using predictive filtering of self-similar traffic', *Proceedings of ICC '99*, Vol. 3, June, 1999, pp. 129–134.

[3] Östring, S., Sirisena, H. and Hudson, I., 'Robustness of ABR congestion control algorithms with regards to self-similar traffic models', *IEEE LAN/MAN'99 Workshop*, November, 1999, pp. 112–115.

[4] Östring, S., Sirisena, H. and Hudson, I., 'Rate control of elastic connections competing with long-range dependent network traffic', *Accepted for IEEE Transactions on Communications*, 2000.

[5] Östring, S., Sirisena, H. and Hudson, I., 'Robust ABR control for uncertainties in long-range dependent traffic', *Accepted for post-IEEE LAN/MAN'99 Publication*, 2000.

[6] Östring, S., Sirisena, H. and Hudson, I., 'Sensitivity of ABR congestion control algorithms to Hurst parameter estimates', *Proceedings of IFIP-TC6 Networking 2000*, May, 2000, pp. 36–48.

[7] Östring, S., Sirisena, H. and Hudson, I., 'Designing and managing networks for self-similarity', *Proceedings of EMACS 2000*, September, 2000.

[8] Östring, S. and Sirisena, H., 'The Influence of Long-range Dependence on Traffic Prediction', *Accepted for ICC'01*, June, 2001.

CONTENTS

LIST OF FIGURES

LIST OF TABLES

GLOSSARY

ABR	Available Bit Rate
ACR	Allowable Cell Rate
AF	Assured Forwarding
AIR	Additive Increase Rate
ATM	Asynchronous Transfer Mode
BE	Best Effort
B-ISDN	Broadband Integrated Services Digital Network
BT	Block Transfer
CA	Congestion Avoidance
CAC	Connection Admission Control
CBR	Constant Bit Rate
CCR	Current Cell Rate
CBQ	Class-based Queueing
CDT	Cell Delay Tolerance
CDV	Cell Delay Variance
CDVT	Cell Delay Variation Tolerance
CLR	Cell Loss Ratio
CI	Congestion Indication
cwnd	Congestion Window
DiffServ	Differentiated Services
ECN	Explicit Congestion Notification
EF	Expedited Forwarding
EFCI	Explicit Forward Congestion Indication
ER	Explicit Rate
f-ARIMA	fractional Autoregressive Integrated Moving Average
FRCV	Fast Recovery algorithm
FRXT	Fast Retransmit algorithm
GCRA	Generic Cell Rate Algorithm
ICR	Initial Cell Rate
IETF	Internet Engineering Task Force
IntServ	Integrated Services
IP	Internet Protocol
ISDN	Integrated Services Digital Network

LAN	Local Area Network
LRD	Long-range Dependence
MCR	Minimum Cell Rate
MBAC	Measurement-based Admission Control
MPEG	Motion Picture Experts Group
nrt-VBR	Non-real time Variable Bit Rate
PCR	Peak Cell Rate
PERC	Predictive Explicit Rate Control
PERC+	PERC with prediction compensation
QoS	Quality of Service
RDF	Rate Decrease Factor
RED	Random Early Detection
RM	Resource Management
R/S	Rescaled Adjusted Statistic Analysis
RSVP	Reservation Protocol
RTT	Round-Trip Time
rt-VBR	Real time Variable Bit Rate
SCR	Sustainable Cell Rate
SONET	Synchronous Optical Network
SLA	Service Level Agreement
SRD	Short-range Dependence
SS	Slow Start
ssthresh	Slow Start Threshold
TCP	Transmission Control Protocol
UBR	Unspecified Bit Rate
UPC	Usage Parameter Control
VBR	Variable Bit Rate
WRR	Weighted Round Robin

Chapter 1

INTRODUCTION

There is a tremendous drive within the communication networks industry to provide a universal network that seamlessly integrates different services using a standard technology. This goal is close to being achieved [Schwartz 1996, Walrand and Varaiya 1996]. The process began with conceptualisation of the *Integrated Services Digital Network* (ISDN) and progressed to the *Broadband Integrated Services Digital Network* (B-ISDN). In parallel with these architectural concepts, technologies such as *Synchronous Optical Network* (SONET), *Asynchronous Transfer Mode* (ATM) and *Internet Protocol Differentiated Services* (IP DiffServ) have been developed and refined to support the concept of an integrated network. This thesis contributes to the development and understanding of traffic management algorithms within the context of a communication network servicing a heterogeneous mixture of traffic derived from a variety of application classes.

1.1 THE CONCEPT OF QUALITY OF SERVICE

The convergence of traffic being transported through an integrated service network requires a paradigm shift in the way networking is approached. To achieve an integrated networking environment, traffic management is required to ensure that the differing requirements of the various classes of users are met. These user requirements are termed Quality of Service (QoS) and depend on the application that is being used.

Traditional networks simply provide a single service for a very specific type of traffic or application where the quality requirements are uniform for all connections. For example, telephone companies are involved with the bundling of telephone calls through cables, so these companies can provide the quality of service required by customers by having a good understanding of characteristics of the human voice and the implicit technical demands of two people communicating using the telephone. Another example would be packet networks, like ARPANET, which developed out of the need for people, working on joint projects using computers, to transfer data between different locations. Since data loss could not be accommodated, reliable transport protocols have been devised that check packet sequence numbers and re-transmit lost packets. Finally, cable television companies need to ensure that quality TV signals are received at the residential edges of the network, and this is usually achieved by over provisioning the capacity of cable networks.

An integrated network requires a totally different approach. A request for access to the network has the potential to involve a wide range of quality requirements. Integrating these different requirements makes network design much more complex, and it would certainly be simpler to have dedicated networks for each type of traffic. There are, however, two significant reasons for this move to integration. The first drive is the demands of users themselves. People are now well-accustomed to using a wide variety of communication means—email, fax, cellular phone—and are interested in having these services available from the same communications device. Furthermore, there is a public interest in additional communication applications, like network gaming, video-on-demand, and wireless Internet access, to name a few. In general these network applications must be accessible from a small number of devices and using a reasonably standard technical interface.

The second reason driving a move to an integrated network involves economic considerations. Building separate networks for telephone, data and video applications involves a huge capital expense and the utilisation of the network resources within each dedicated networks is much lower. By building an integrated network, capital expense can be minimised, and multiplexing larger numbers of connections onto the same network increases utilisation of network resources. Furthermore, network access providers can gain more revenue through a graded suite of services. Customers could then choose to pay more for a higher quality service because they want a greater proportion of network resources guaranteed to their session, or because the transfer of information is more critical and requires higher priority throughout the network.

1.2 ELEMENTS OF AN INTEGRATED NETWORK

There are three broad elements that constitute an integrated network [Schwartz 1996, Walrand and Varaiya 1996]. Firstly, a set of services needs to be defined that is targetted for the primary classes of applications which will utilise the network. Each service is based on the type of traffic which the application class will generate, the generic quality of the service demands of the traffic, and the flexibility of a session with regards to being multiplexed with other connections or responding to feedback regarding network state. A network service can also include other network management functionalities, such as traffic shaping and selective packet discard or marking. The definitions of these services requires flexibility so that the network can accommodate new applications that arise in the future.

Secondly, the network needs to have a procedure for defining individual agreements or contracts with each connection regarding the specific resources and quality which the user expects. This process can include negotiation during the admission control phase, but it must be clear for both the network and the user what the final technical arrangement includes. This agreement is important both from an economic point of view, where the user is paying for a specific type of service, and also from a technical perspective, where the network can execute more drastic congestion control actions on the arrival of traffic which is outside the agreement.

Finally, the network requires a comprehensive set of traffic management protocols which control the entry and flow of traffic in the network. Protocols include connection admission control which determines, when a connection request arrives, whether the network has the resources to handle the connection. Traffic management also includes policing, where the network monitors the traffic to determine whether it conforms with the service agreement. A best effort service, designed for traffic that can respond to feedback from the network, also requires flow control mechanisms, which allow the network to notify sources of the resources that are currently available. Further management actions involve congestion control measures, such as selective packet discarding, which are required at the onset of overload conditions in the network.

Two networking technologies that provide integrated networking are introduced in Sections 1.3 and 1.4, namely Asynchronous Transfer Mode (ATM), and the Internet Engineering Task Force (IETF) approaches, which include the Integrated Services (IntServ) and Differentiated Services (DiffServ) architectures. These are not mutually exclusive technologies, but have developed from two different networking groups. By outlining the ATM and IETF DiffServ approaches, similar trends and issues can be observed which cultivate more creative research within this area. In particular, this thesis is concerned with the development of reactive traffic control mechanisms within a network servicing a heterogeneous traffic mixture. The ABR service in the ATM protocol suite provides the context from which a rate-control mechanism for self-similar traffic is developed, while the IP DiffServ model highlights some of the implementation issues. Thus, a unified theme of designing appropriate traffic control mechanisms underlies this thesis. The following is an introduction to the fundamentals of integrated networking technology where these control mechanisms find their application.

1.3 FUNDAMENTALS OF ASYNCHRONOUS TRANSFER MODE

The transport technology which has been developed specifically for the B-ISDN concept over the last ten years is Asynchronous Transfer Mode (ATM). In 1991, the ATM Forum was established to standardise the services provided by ATM networks and protocols used to manage traffic [Black 1995, Onvural 1995, Ritter 1998]. The uniqueness of ATM is that it provides QoS guarantees using a connection-oriented approach, where traffic is managed according to service classes and individual contracts. However, ATM is not purely a connection-switched networking technology, as cells from different ATM services are multiplexed together to increase network utilisation, as signified by the term "asychronous".

ATM technology has been primarily influenced by the telecommunications industry, as can be seen from the small, fixed-size ATM cell and the connection-oriented design of the ATM services. The 53-byte length of the cell has been adopted to minimise the delay experienced by real-time voice connections in accumulating data into each cell payload [Walrand and Varaiya 1996]. If the cell length was any longer, the time required to fill each individual cell would significantly increase the delay in cell transfer, and the quality of a voice connection would be degraded. Also, there is a range for the amount of data that can be lost without having a noticeable or disturbing effect on a listener in a audio connection, and a 48-byte payload lies in the middle of this range [Black 1995]. The cell has a fixed length because switches can process cells faster—the switch automatically knows where the end of the cell is going to occur after it has read the 5 byte header from the incoming link.

Due to the connection-oriented approach of ATM networks, cells travel through the network along fixed paths called virtual circuits (VCs). This is a more appropriate architecture for achieving the desired QoS, since admission control decisions can determine the ability of each switch to handle an additional connection along the path of the connection. A switch can also isolate misbehaving connections and police them in an appropriate manner. It should be noted that ATM virtual circuits do not allocate fixed time slots to individual connections (hence the asynchronous nature of the technology), which allows the network to statistically multiplex more connections onto the same link, leading to increased utilisation of the network.

1.3.1 ATM Services

There are five categories of services specified by the ATM Forum:

1. Constant Bit Rate (CBR);
2. Real time Variable Bit Rate (rt-VBR);
3. Non-real time Variable Bit Rate (nrt-VBR);
4. Unspecified Bit Rate (UBR);
5. Available Bit Rate (ABR),

and the attributes for these services are summarised in Table 1.1 [Walrand and Varaiya 1996, Ritter 1998]. Each of these services will be introduced in sequence in the remainder of this section.

As the name implies, Constant Bit Rate (CBR) is a service which emulates a fixed-bandwidth telephone connection, though telephony is not the only application that this

Table 1.1 ATM Service Parameters Specifications.

Attribute	**CBR**	**rt-VBR**	**nrt-VBR**	**UBR**	**ABR**
Traffic Parameters					
PCR, CDVT	Specified	Specified	Specified	Specified	Specified
SCR, BT	N/A	Specified	Specified	N/A	N/A
MCR	N/A	N/A	N/A	N/A	Specified
QoS Parameters					
CLR	Specified	Specified	Specified	UnSpecified	Specified
CDT, CDV	CDV and max CTD	CV and max CTD	Mean CTD only	Unspecified	Unspecified
Congestion Feedback	No	No	No	No	Yes

service is intended for. In terms of bandwidth requirements, a CBR connection is specified by its Peak Cell Rate (PCR). Since it is intended for real-time voice and video traffic, the service also has delay constraints built into its service specifications. Thus, CBR provides a high quality connection to applications desiring guaranteed access to a fixed amount of network resources. However, it is not necessary that the connection always maintain a cell rate at the specified PCR, and the ATM network can statistically multiplex cell streams from other connections into the unused CBR bandwidth as long as the QoS, such as delay constraints, for the CBR connection is not degraded. In spite of this, a user pays for all of the bandwidth guaranteed to its CBR connection, since the network has reserved bandwidth resources for the connection at all switches along the path of the connection, and will accept the Peak Cell Rate at any time during the lifetime of the CBR connection.

It is possible that a user has more knowledge of the characteristics of his/her traffic than just the peak cell rate, which means that statistical multiplexing can be explicitly used to increase network utilisation. This traffic is serviced by the VBR service category. More specifically, a user can supply a variable called Sustainable Cell Rate (SCR), which is related to the mean traffic level of the connection. With this information, network resources can be used more efficiently, since admission control mechanisms do not need to reserve the peak bandwidth for the connection. Instead, the network agrees to serve the SCR, with possible bursts up to PCR for finite periods of time. The PCR parameter is still required by the VBR service, but a user of the VBR service simply pays for the amount of bandwidth which is dynamically consumed. Tight delay constraints can be incorporated within this framework, as offered by the real time VBR service. This is specifically envisaged for video streaming applications. However, data applications can also use the VBR service, and since these applications do not have strict delay requirements, the user can opt to use the nrt-VBR service. This gives them guaranteed access to peak cell rates below the PCR and average cell rates below the SCR, but without any delay bounds.

The basic service provided by traditional packet switched networks is simply connectivity to the network, with no guarantees on the bandwidth. In ATM networks, this type of service is supplied by UBR. No attempt is made by the source to characterise its traffic, except a provisional PCR, and cells can be delivered at any rate to the network. The agreement is that the network can discard cells from a UBR stream whenever congestion conditions require it. This means that ATM networks provide no upper bounds on cell loss rates or delays for the UBR service. However, it should be noted that this does not necessarily mean that the source continues transmitting cells oblivious to loss—any rate adaptation or loss recovery is performed at higher protocol layers, for example by TCP, which provides a reliable service over the unreliable datagram service provided by IP.

Finally, a unique service is provided by ABR in the ATM protocol suite. Many data applications are capable of adapting to the state of the network by adjusting their transmission rates based on feedback information returned to the source. This concept is explicitly incorporated into the ABR service, where an ABR source is notified of available resources within the network by special feedback cells. The service provides dynamic bandwidth allocation with guarantees on the maximum loss rates experienced by the connection. A Minimum Cell Rate (MCR) can be specified by an ABR connection, since the applications using this type of service often require a nonzero flow of management information to ensure the integrity of the connection is maintained.

1.3.2 ATM Traffic Contracts

The ATM service categories themselves do not completely specify the details of the agreement between a connection and the network. The details are established within the traffic contract, which the source negotiates with the network during the connection admission phase. There are three parts to any traffic contract—the service which the connection is subscribed to, the traffic descriptor, and the QoS parameters. The traffic descriptor includes not only the cell rates like the PCR, SCR and MCR, but also non-ambigious specifications of the mechanisms which will be used to monitor the conformance of the cell stream to the traffic contract [Ritter 1998]. It is the function of the connection admission control protocol to determine whether the network has sufficient resources to support the connection as specified by the traffic descriptor. If it cannot, then the source has the option of re-negotiating another traffic contract. Once the admission control mechanisms has agreed to the contract, it must create the virtual circuit through the network and reserve the appropriate resources. During the life of the connection, the network can monitor the stream of cells to ensure that the connection is meeting its terms of the contract.

The QoS required by a particular connection depends on the application which is using the connection. QoS is concerned primarily with cell losses and delays, and the ATM protocols include the following parameters:

- cell loss ratio (CLR);
- cell delay variation (CDV);
- maximum cell transfer delay (maxCTD);
- mean cell transfer delay (meanCTD).

Cell losses are particularly critical for data transfers—if losses do occur, larger frames from protocols in the higher layers may have to be re-transmitted. Cell losses are caused by congestion conditions within the network due to input work loads that are greater than the capacity of the network, so that switches are forced to discard cells from queues. These cell losses can reduce the effective throughput of original data (also termed *goodput*), and result in the network being overloaded with re-transmitted cells.

Other QoS parameters are concerned with network delays. While delays are not as critical for data applications, the quality of real-time voice and video applications deteriorates rapidly if cells experience significant delays within the network, and QoS for these applications is defined in terms of upper bounds on the delays. Users experience unsatisfactory service if either the transfer delay or the variation in the delays become excessively large. Components that contribute to the delay in the network include the packetisation delay, transmission and propagation delays, and delays associated with buffering and scheduling cells at switches.

When a connection is established, the network can achieve the specified QoS requirements by appropriate routing choices and by allocating the correct scheduling algorithms at the intermediate switch along the virtual circuit. It is also important to note that the QoS specifications are values averaged over time. Due to local congestion, the instantaneous QoS experienced by the connection may deviate from the traffic contract for a brief period of time.

1.3.3 Traffic Management in ATM Networks

An integrated network that intends to meet the QoS requirements of a variety of applications must take an active role in managing traffic that utilise resources within the network. Mechanisms for traffic management need to be designed and implemented at a number of different locations within a network—at interfaces to the network and in control modules incorporated within switches. These management functions include [Ritter 1998]:

- Connection Admission Control (CAC) and Resource Allocation;

- Usage Parameter Control;
- Traffic Shaping;
- Traffic Flow Control.

These functions can be loosely divided into two categories, depending on whether they perform preventive or reactive control actions. Preventive traffic management, which includes admission control, usage parameter control and traffic shaping, occurs primarily at the edge of the network, and involves blocking or delaying the arrival of traffic that would overload the network [Cruz 1991, Parekh and Gallagher 1993, Parekh and Gallagher 1994]. However, preventative control cannot avoid all congestion situations within the network, and reactive control management, such as traffic flow control, uses feedback from the network to adapt the flow of traffic across a network so that high network utilisation is achieved while congestion is minimised. Both types of traffic management operate in order that QoS is maintained for connections within the network.

1.3.3.1 Preventive Traffic Management

The most important preventive control action which the ATM network can take is admission control. At this stage in the management process, the connection is still simply a request and no significant transmission of cells has occurred which could impact the QoS requirements of any existing connections. The CAC mechanism is supplied the resource demands of the connection and its associated QoS requirements, and the admission controller proceeds to determine whether the network can support the connection. The simplest decision is non-statistical, where the peak bandwidth requirements are reserved throughout the network. However, this wastes network resources and higher utilisation can be achieved using the statistical multiplexing [Qiu and Knightly 1999, Boorstyn *et al.* 2000]. This requires more detailed traffic characterisation, along the lines of the concept of effective bandwidth [Kelly *et al.* 1996], but this may not be available or may be too difficult to calculate. Hence, some risk is involved on the part of the network if statistical multiplexing is used, since its ability to meet the QoS requirements of the individual traffic is now dependent on the mixture of instantaneous demands of the traffic. There is a tradeoff between providing absolute guarantees on QoS and increasing the utilisation of the network.

If the decision of the admission control is to accept the connection, then the network signals the reservation of resources along the path of the virtual circuit [Cruz 1995]. ATM also uses the concept of bundling multiple virtual circuits together into virtual paths, which allows a network to utilise resources that have already been reserved along the virtual path. This simplifies the admission control process, since virtual paths can be established prior to bandwidth resources actually being needed by users, which reduces delays in connection setup delays. Moreover, it allows the network to logically collect similar types of connections into a single statistical pipe.

Having accepted the connection, the network needs to monitor and police the connection based on its traffic contract. In ATM network terminology, this is known as Usage Parameter Control. The key management block used for this function is the *Generic Cell Rate Algorithm* (GCRA) [Onvural 1995], which is an abstraction of more familiar monitoring mechanisms, such as leaky buckets. The specific implementation of the GCRA is open to the vendor, as approaches other than leaky buckets, such as virtual scheduling algorithms, can be used to monitor traffic parameters and QoS constraints. The purpose of the GCRA is to determine whether an arriving cell conforms to its traffic contract. No action is taken against conforming cells, while nonconforming cells may be tagged as low-priority or selectively discarded. Any cells that are discarded will be detected by the source, causing transmission protocols at higher layers to slow down. In particular, discarding cells, as opposed to marking them, avoids congestion further down the path of the connection.

Finally, traffic shaping is a preventive management technique that trades delaying cells for enforcing conformance to the traffic profile. This is done by intelligently delaying certain cells in the traffic stream as they enter the network. In general, it is implemented at the interface to the network. Traffic shaping ensures that the cell sequence is maintained and the desired QoS is achieved. It also allows modification of such traffic parameters as the peak rate, the burst lengths and the cell delay variation.

1.3.3.2 Reactive Traffic Management

Reactive control management includes traffic flow control which responds dynamically to congestion situations within a network [Yang and Reddy 1995]. The network returns feedback to sources regarding the current state of the network, with feedback information involving implicit or explicit requests for changes in the cell rate. Within the ATM service classes, the ABR service has been designed with this type of feedback mechanism at the core of its traffic management. In this case, switches within the interior of the network monitor local congestion levels using parameters such as average queue lengths and link utilisation, and then return feedback information to sources via specialised Resource Management (RM) cells. Feedback information can take a number of forms, including binary information such as the Explicit Forward Congestion Indication (EFCI) bit, the Relative Rate (RR) field (which requests a relative change to the current cell rate), or explicit information using the Explicit Rate (ER) field. This forms an end-to-end, closed-loop rate control mechanism. Traffic management of this type dynamically allocates available resources within the network, thus increasing network utilisation and being responsive to varying bandwidth requirements of higher priority traffic.

1.4 FUNDAMENTALS OF IP DIFFSERV

There has been tremendous growth in the use of the Internet during the last five years, which can be observed by the exponential increase in the number of registered domain names and the amount of Internet traffic that is being handled by routers [Govindan and Reddy 1997]. Of this traffic, 95% is being transported by the transmission control protocol (TCP) and approximately 70% of the messages are HTTP requests or files [Thompson *et al.* 1997]. It is therefore apparent that the philosophy that lead to the development of the Internet and its associated protocol TCP has been successful in achieving a communication medium that is both flexible and accessible. In particular, the Internet has maintained scalability and flexibility by employing distributed control rather than centralised control.

TCP is an end-to-end transport protocol that determines the changes it should make to its transmission rate by inferring information regarding congestion within the network based on the packet loss rate that it observes. TCP assumes that packet loss indicates that its current transmission rate is too high, and responds by decreasing its rate to match the available network capacity more closely. Thus, TCP provides a best effort service to network applications, where bandwidth is utilised as it becomes available within the Internet. However, the service offered by TCP in the current Internet cannot provide any assurances regarding QoS.

1.4.1 Providing QoS in the Internet

It has already been noted that there is an enormous drive to develop a networking architecture which provides a range of services, to satisfy the development of an increasing variety of applications which the public want to use across the Internet. Internet Service Providers (ISPs) are also interested in developing an architecture which supports a commercial infrastructure that allows ISPs to provide higher quality of service to users who are willing to pay more. Unfortunately, the current Internet is not designed to achieve this, as it simply offers a default best-effort service, with packets being forwarded through the network if resources are available.

To date, there have been a number of proposals that have suggested ways of enhancing IP to include some form of QoS asssurance [White and Crowcroft 1997, Metz 1999, Xiao and Lionel 1999]. The model which the Internet Engineering Task Force (IETF) currently favours is the Differentiated Services (DiffServ) framework [Blake *et al.* 1998, Dovrolis *et al.* 1999, Ren and Park 2000]. The reason for its adoption, in the face of other proposals, is its simplicity and deployability, since mechanisms within the DiffServ model are scalable. In contrast, Integrated Services (IntServ), the other significant proposal for providing QoS within the Internet, follows a connection-oriented approach and uses a signalling protocol called ReSerVation Protocol (RSVP). RSVP

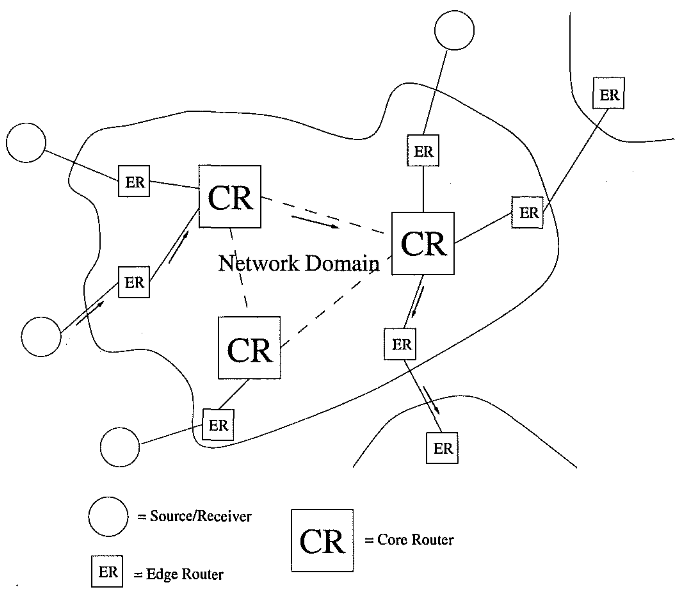


Figure 1.1 Model of the Differentiated Services Architecture.

reserves network resources for a particular connection throughout the Internet. IntServ has the significant problem of scalability, since the size of the state information tables at core nodes increases rapidly as the number of connections grows, and hence the tables soon become unmanageable.

In DiffServ, traffic from an individual source-destination pair is classified according to a set of service classes that define traffic using coarse granularity. Any sophisticated packet processing is performed by edge routers, while core routers in the interior of the network only handle packets based on the class each packet belongs to (Figure 1.1). Such a design is important, as it allows the network to scale. Otherwise, each packet arriving at an interior node would need to be mapped to a specific microflow, requiring core routers to store significant amounts of flow information.

1.4.2 Service Level Agreements

Providing quality of service requires a user to negotiate a Service Level Agreement (SLA) with its ISP. This includes the service class that is desired and any bandwidth requirements. At present, there are three groups of service classes in DiffServ—Expediated Forwarding (which is the premium service), the Assured Forwarding group (which offers graded levels of drop precedence depending on how much the customer wants to pay for bandwidth assurances) and finally the Best Effort service, which has been carried over from the current Internet. Any QoS agreement that the user has with its ISP are only assurances rather than hard guarantees, since DiffServ does not guarantee bandwidth allocations or upper bounds on delays through the network. Any deviation from these QoS assurances can be reflected in the accounting process, though this is not absolutely

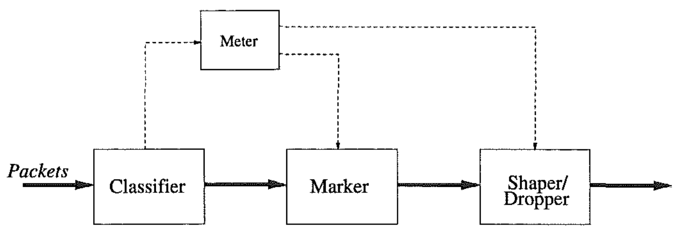


Figure 1.2 Traffic conditioning at the edge node.

necessary. Thus, the QoS that a class of traffic receives is defined relative to other classes, rather than in absolute terms that the network must provide.

1.4.3 Traffic Conditioning at Edge Routers

Edge routers perform any sophisticated packet processing, known as traffic conditioning. The different functions of a traffic conditioner are shown in Figure 1.2, and include packet classification, metering and marking. The conditioner can also shape the traffic in order to either optimise its flow through the network or limit its impact on other traffic. Finally, it can drop packets to enforce conformance to the profile.

Classification at the traffic conditioner requires flows to be mapped to a particular service class based on the SLA that the particular flow has with the edge router. These correspond to DiffServ (DS) codepoints that are written into the DS field in the IP header. In IPv4, the type-of-service (TOS) field has been redefined as the DS field, while in IPv6 the traffic class field is used. At present the principal codepoints are $\{EF, AF1x, AF2x, AF3x, AF4x, BE\}$ corresponding to the Expediated Forwarding class, the four classes in the Assured Forwarding group and the Best Effort class. The Assured Forwarding classes have three levels of drop precendence which are also included in the DS codepoint (denoted by x). The DS codepoint requires 6 bits of the 8-bit traffic class field, thus leaving the final 2 bits unused.

Expediated Forwarding (EF) is intended for real-time traffic or high-priority traffic that requires strict bandwidth or delay guarantees [Jacobsen *et al.* 1999]. Connections use a reservation protocol, such as RSVP, which checks whether the required resources are available throughout the network. If admitted to the network, EF packets receive priority treatment throughout the network. To ensure that the delays are maintained within the desired constraints, it is recommended that EF traffic be limited to about ten percent of the overall capacity at each link within the network [Xiao and Lionel 1999]. In addition, there are priority queueing disciplines that would allow EF traffic unlimited access to the outgoing link. It is therefore important that EF traffic is rate-limited both at the conditioner and within the network. Hence, conditioning EF involves metering and discarding packets that are out-of-profile.

The Assured Forwarding group provides graded levels of bandwidth assurances,

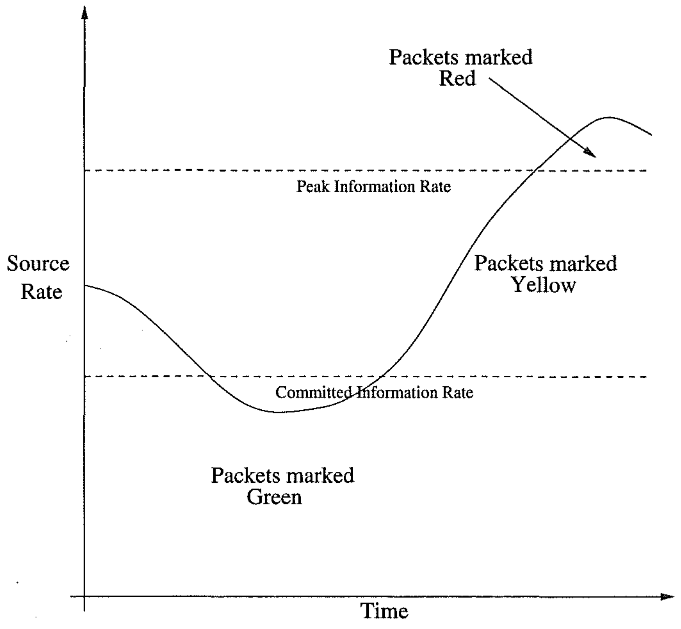


Figure 1.3 Packet colouring scheme based on the arrival rate of incoming packets.

where packets are differentially dropped based on their drop precedence level [Heinanen *et al.* 1999]. This is intended for commerical customers who desire greater assurances of Internet access to their Web sites and higher costs are associated with higher tiered AF classes. Within an SLA, a profile is constructed that specifies the bandwidth assurances to an AF user and how AF packets will be marked regarding their drop precedence. The current approach is to use three drop precedence levels, where the drop precedence to which a packet is assigned depends on the current transmission rate and two rate settings, the Committed Information Rate and the Peak Information Rate as shown in Figure 1.3 [Heinanen and Guerin 1999]. The traffic conditioner does not necessarily drop AF packets that are out-of-profile. Instead it simply marks them with their appropriate codepoint, and the resulting per-hop behaviour of each packet within a DS domain depends on the packet's drop precedence level.

At this point, it is useful to consider the meter mechanisms that have been proposed. Traditional token buckets can be used, since they are simple to implement. A token bucket is filled at the specified token rate, and makes tokens available to arriving packets. The depth of the token bucket defines the burst size that is allowed within the packet flow. If all of the tokens in the bucket have been depleted when a packet arrives, the packet is considered out-of-profile and is either marked or dropped. While the token bucket is conceptually simple, it can sometimes be difficult to configure and is not well-matched to the bursty nature of TCP traffic. Hence, Clark and Fang proposed using the time-sliding window approach for rate estimation [Clark and Fang 1998]. This mechanism estimates the average packet arrival rate that corresponds to a window of length Win_len of the arrival rate parameter, which is updated whenever a packet arrives. This rate estimation

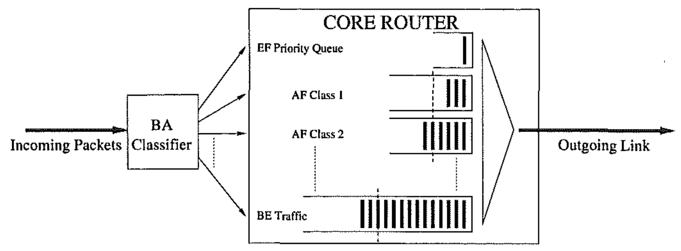


Figure 1.4 Priority queueing implement of per-hop behaviour in DiffServ.

approach has even less parameters to configure than a token buffer, and is more well-suited to TCP connections. Furthermore, it has also been adapted for use with AF traffic [Fang *et al.* 2000].

Finally, there is no traffic conditioning required for Best Effort (BE) traffic, other than to label the packets with the BE codepoint. As in the current Internet, this traffic simply makes use of any available bandwidth within the network, and users accept whatever loss rate occurs. Of course, EF and AF traffic receive priority service before BE traffic, so the BE service in DiffServ is not entirely equivalent to the current best effort service offered in the Internet.

1.4.4 Packet Forwarding at Core Routers

Within a DiffServ domain, packets are no longer processed in relation to an individual microflow, but are classified according to the Behaviour Aggregate of the class of service which is indicated by the DS codepoint. Each packet receives a particular forwarding treatment at a node, hence experiencing a per-hop behaviour that is standard for any packet within its particular service class. This is achieved without mapping the packet to a specific connection. There are a number of ways of implementing the per-hop behaviour, including priority queueing, weighted round robin (WRR) scheduling and class-based queueing (CBQ) [Kleinrock 1975, Floyd and Jacobson 1995, Jacobsen *et al.* 1999]. Class-based forwarding is illustrated in Figure 1.4 using priority queueing. It should be emphasized again that absolute QoS guarantees are not the intention of DiffServ, and in fact are not possible with this architecture. DiffServ sacrifices hard QoS limits for functionality and scalability. With the development of Dense Wavelength Division Multiplexing (DWDM) and other optical technologies, link capacity will not be the issue. Rather it will be the speed with which nodes can process packets—hence the need to maintain simplicity at core routers.

Expedited Forwarding is provided for high priority traffic which requires low loss, latency and jitter [Jacobsen *et al.* 1999]. Packets that are classified as requiring this forwarding treatment are directed to the high priority queue. This queue is always guaranteed to have minimal delay by scheduling any packet within this queue to be immediately transmitted on the outgoing link. Hence, this Premium Service is guaranteed

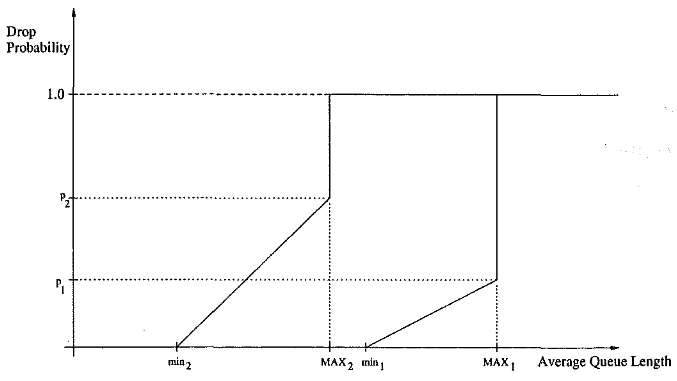


Figure 1.5 The multi-RED algorithm.

full access to the bandwidth resources at a node to ensure low latency and jitter. However, it should be noted that admission control is required to ensure that overall relative load of the EF traffic is low, with figures around 10% typically being mentioned in the literature [Xiao and Lionel 1999]. Otherwise the objectives of the end-to-end service like low latency cannot be achieved. Also, the amount of damage which EF traffic can inflict on other services needs to be bounded, for example using token bucket mechanisms at the boundary nodes [Jacobsen *et al.* 1999], thus enforcing a peak rate on the EF traffic.

The treatment which a packet from the Assured Service class receives is defined within a set called the Assured Forwarding Group [Heinanen *et al.* 1999], which defines a set of tiered classes of traffic, where queues for the higher classes receive priority scheduling. Within each class, the per-hop behaviour experienced by a packet involves three levels of drop precedence. The probability of dropping a particular packet is determined by a multi-RED algorithm which estimates the level of local congestion by an exponentially weighted average queue length and a set of thresholds [Sahu *et al.* 2000], as shown in Figure 1.5. RED, an acronym for Random Early Detection [Floyd and Jacobsen 1993], is an active queue management technique which allows a node to gracefully avoid congestion situations by probablisitically marking or dropping packets, rather than using drop tail queueing techniques which can have a drastic effect on transport control protocols.

Finally, best effort traffic receives low priority forwarding treatment similar to the forwarding which it receives in the current Internet, with the only exception being that packets from other services will be forwarded first. Users of this service pay a low price for this per-hop behaviour and receive bandwidth resources only as they become available. Thus, connectivity is maintained with no assurances on the forwarding behaviour of the packets through the network.

1.5 DISCUSSION

In this introduction, the concept and motivation for integrated services network architectures have been presented, using ATM and IP DiffServ as illustrative examples. While these networking technologies differ in many details, a common theme is that both types of networks require traffic management protocols that have been designed to support the QoS requirements of a variety of different applications. In particular, a set of services with graded QoS guarantees or assurances are included in the each network architecture, and much research work has been involved in developing the management functions that can ensure that these services meet their objectives.

Best effort services form an important class within an integrated services network, as they supply users with low-cost connectivity to the network and increase the utilisation of the network. The potential of these services can be seen in the widespread use of TCP. However, it is important that management protocols which control the flow of best effort traffic are aware of the bandwidth and QoS requirements of higher priority traffic. This requires the cooperation of interior nodes that provide feedback to sources regarding current traffic loads, and protocols located at sources which determine the transmission rate. By accurately modelling properties of high priority traffic, such as self-similarity, reactive traffic management can be designed to support the QoS objectives of a network more closely. The following two chapters (Chapters 2 and 3) discuss in detail the concepts of reactive traffic management and self-similarity of network traffic.

Chapter 2

REACTIVE TRAFFIC CONTROL MECHANISMS

Traffic management is an important aspect of network engineering. In an idealistic world, the network would be a nebulous entity which would be able to accomodate any demand which was placed on it, including being able to allocate large blocks of bandwidth with users experiencing unnoticable delays and zero losses during their communication sessions. Optical fibre technology has extended the physical boundaries much closer to this ideal, but other economic and physical constraints (such as spectrum in wireless communications and the speed of opto-electronic interfaces) still limit the resources which a network can provide its users. Resources need to be shared efficiently and fairly between network users running a variety of applications, many of which differ in their networking requirements.

There are a number of types of control mechanisms which a network uses to manage the traffic offered to it, and these operate at different time scales within the traffic managment framework and have different aims [Black 1995]. Connection admission control (CAC) determines whether the network has the resources to service the requirements of a connection, and only admits the connection if it can be supported. This is an important initial preventative step which ensures that the connection will not adversely affect other existing connections before the source begins transmission. Once the connection has been accepted by the network, the connection needs to be monitored and policed by the network (Usage Parameter Control in ATM terminology) so that it does not either inadvertently or intentionally monopolize network resources. Fundamentally CAC and UPC are open-loop control approaches, where the network accepts the traffic offered to it within the given framework of the connection contract and does not shape *compliant* cell streams regardless of conditions within the network.

Situations can arise, however, where the network cannot provide the necessary resources for an application to achieve its natural bit rate even if it is compliant to its connection contract [Black 1995]. At this point, reactive control mechanisms such as traffic and congestion control algorithms must be utilised [Yang and Reddy 1995]. The terms traffic control and congestion control are used fairly interchangeably, but in fact they refer to different management approaches to congestion. Congestion occurs when the aggregated input load to the network exceeds the ability of the network resources to service the input, resulting in much higher queue lengths (or even buffer overflows) and dead-locked links within the network. In this scenario, connections experience significantly higher delays through the network, lower throughput and increased losses. Congestion control is the process by which the network returns to an appropriate operating condition from a congestion event, and is typically achieved by discarding excess packets from the existing queue. In contrast, traffic control is a preventative approach to congestion, where the onset of congestion is detected early and control mechanisms react to avoid congestion. Queue length is often used to estimate how close a particular link is to congestion—primarily by signaling when the queue length, or a filtered average of the queue length, crosses a specified threshold.

2.1 PRINCIPLES OF REACTIVE TRAFFIC CONTROL

Reactive traffic control mechanisms are an important part of traffic management in modern broadband networks, primarily because there are data applications that can utilise a best effort service. In addition, the increased volume of traffic in the network and the variety of quality of service requirements of applications using the network mean that reactive control is necessary both to avoid overloading the network and for increasing the network utilisation. A fairly unsophisticated approach to traffic control is tagging packets for possible discard elsewhere in the network. More advanced approaches signal sources regarding the state of the network, providing information which the sources respond to. Signals can be in the form of binary information, such as using either congestion notification bits in packets or returning acknowledgements to the sources for received packets, or explicit information which informs a source of the bit rate or block size which the network can currently accommodate.

2.1.1 Stability

Stability is an important issue in the design of traffic control mechanisms in broadband networks, where the delay-bandwidth product is large [Schwartz 1996]. The delay-bandwidth product is related to the number of cells in flight (currently traversing) the network, and is the load on the network which will be unaffected by any control information. As the delay-bandwidth product increases, the network can become unstable if inappropriate control mechanisms are employed. Instability occurs when the source rates oscillate due to the control mechanisms fluctuating between signalling the situations of congestion and low network load, with the oscillations being intensified by synchronisation between neighbouring sources.

2.1.2 Fairness

Another issue in the design of traffic control mechanisms is that of fairness. While standard control theory addresses issues like stability and convergence, fairness is an issue which is rather unique to traffic control in a communication network, being a multiuser system. This is due to the potentially large number of controlled connections (in other words, individual control loops) and switches, with locations of congestion being variable at different points in time. Intuitively, fairness means that no users would be denied access to a network and none of the connections using this network would be penalized because of their path or because of the time at which they requested a share of the bandwidth. Just as importantly, fairness implies that a connection should be allocated any available bandwidth that cannot be utilised by the other connections.

Fairness is an elusive concept, and the difficulty in specifying it is reflected in the variety of definitions that have been promoted [Jaffe 1981, Mazumdar *et al.* 1991, Arulambalam *et al.* 1996, Kelly *et al.* 1998, Mo and Walrand 2000]. In a single node model, fairness simply refers to each connection receiving an equal share of the available bandwidth at a given link irrespective of the round-trip delay of the connection—in other words, the bandwidth is divided *fairly* among the users. In a network with multiple nodes, this definition is no longer adequate because a subset of the connections at a particular node can be bottlenecked elsewhere in the network. As a result, they cannot utilise their fair share of the available bandwidth at the node being considered, even if it was offered to them. In this case, the principle of *max-min fairness* is the most appropriate fairness definition. Another definition of fairness, which is termed *proportional fairness* [Kelly *et al.* 1998], evaluates the overall impact of a change in the rate allocations. A solution that achieves proportional fairness is one where any other rate allocation would result in the summation of individual proportional changes being non-positive. These fairness definitions are outlined in Sections 2.2.5 and 2.3.6.

2.1.3 Examples of Reactive Traffic Control Protocols

Traffic control is used in both ATM and IP network protocols to avoid congestion and to increase the utilisation of the network. The purpose of the ABR service in ATM networks is to provide a best-effort service for non-real time data communications that would utilise any available bandwidth within the network, and yet still provide specified cell loss rate guarantees. As such, a feedback mechanism is required that allows network nodes to inform the ABR sources of the allowable transmission rate—an example of rate-based control. A closely related concept is window-based control, which is used by the Transmission Control Protocols (TCP) in IP networks. Instead of specifying a limit on the packet transmission rate, a window limits the number of unacknowledged packets that can be in flight within the network. The TCP/IP source adapts its window size based on the arrivals of acknowledgements it receives regarding packets successfully received by the receiver. Such control mechanisms are described in more detail in Sections 2.2 and 2.3. It may be noted that reactive traffic control is also used for streaming video/audio over UDP, for example using TCP-friendly mechanisms in the higher layers [Floyd *et al.* 2000]. However, this thesis concentrates on strictly elastic flows which respond rapidly to explicit rate control information or a single packet loss.

2.2 ABR TRAFFIC CONTROL IN ATM NETWORKS

In late 1994, the ATM Forum introduced a new service class within its traffic management specification called the Available Bit Rate (ABR) service [Black 1995, Onvural 1995]. The ABR service is intended for applications that do not have stringent end-to-end delay requirements, so the source rate of a connection can be reduced without seriously affecting the quality of the connection. At the same time these applications are still sensitive to data loss. This means that the sources cannot transmit cells into the network irrespective of the current state of the network. As a result, the ABR service includes QoS guarantees for cell loss rates experienced by an ABR connection, provided that the transmission rate of a source is not above the allowed cell rate specified by the network. ABR is thus an essentially best-effort service which the ATM Forum designed to improve the utilisation of a network, and the service functions by distributing available bandwidth within the network to ABR connections. At the same time, the QoS for other services classes provided by the ATM network is maintained, since ABR connections respond to the current load within the network and reduce their rates if bandwidth is required by higher priority classes such as CBR or VBR.

The object of the ABR service is represented graphically in Figure 2.1. The network load generated by connections using ATM service classes other than ABR is a dynamically varying quantity dependent on the number of connections, the time of day at which the load is measured, and the types of applications being used. At any particular instant in time, this load will be using some proportion of the overall capacity of the network, with the remaining unused resources being referred to as the *available bandwidth.* The purpose of the ABR service is to distribute this available bandwidth fairly between ABR sources, thus increasing the overall utilisation of the network. ABR traffic can be described as opportune traffic that is willing to modulate its throughput depending on the current demand on network resources. Implementing ABR traffic management involves an ATM network returning appropriate information to ABR sources regarding the available resources within the network, with the sources responding promptly to this information. After considering two approaches, the ATM Forum chose rate-based control as the method of managing ABR traffic, which is described in detail in Sections 2.2.1 and 2.2.2. The other proposal that was considered was credit-based control, which is introduced briefly in Section 2.2.4 for the sake of completeness.

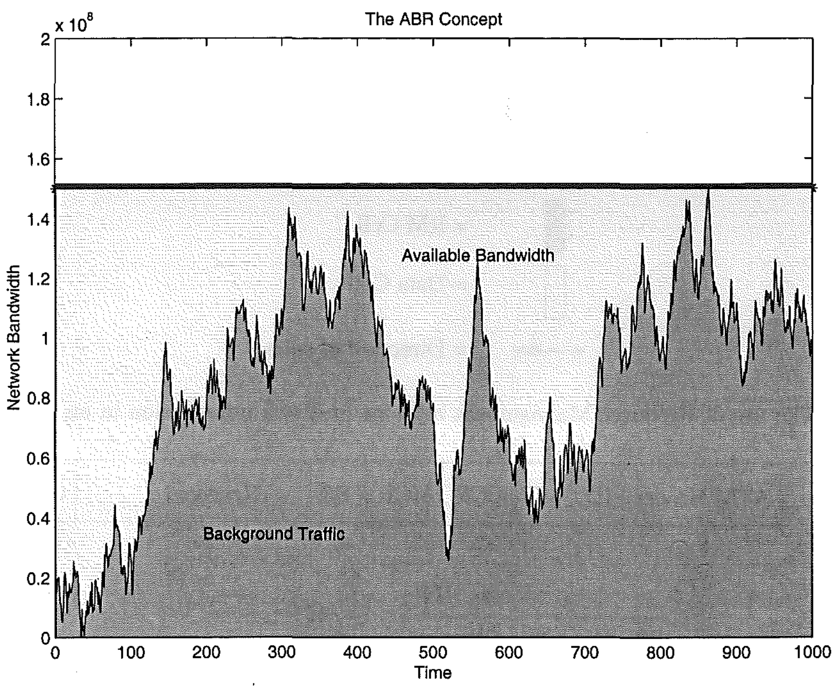


Figure 2.1 A bandwidth perspective of the ABR concept.

2.2.1 Defining the ABR Service

An ABR connection can be considered as a complete control system involving the source end system, the intermediate switches within the network and the destination end system [Bonomi and Fendick 1995]. Each component plays a part in completing the control loop so that a connection can utilise the available bandwidth that is allocated to it. A source regularly transmits resource management (RM) cells for acquiring control information from the network. Within the network, switches determine the transmission rate that they can support and compare this with rate information that is stored in the RM cells, before forwarding the cells on through the network. Finally, the destination loops RM cells back towards the source, which in turn responds with the action specified by the rate-control protocol.

2.2.1.1 Phases of an ABR Connection

An ABR source initiates a connection by signalling a request through the network specifying the desired values for the parameters given in Table 1.1 in Chapter 1. Once the connection has been granted admission to the network, the ABR source begins transmitting cells at the Initial Cell Rate (ICR), with special Resource Management (RM) cells interspersed between data cells, as shown in Figure 2.2. These RM cells have designated fields (Figure 2.3) which are used to convey the control information through the network [Chen *et al.* 1996, Ritter 1998]. The intermediate switches along the ABR connection can notify the source of the available bandwidth or the onset of congestion either by

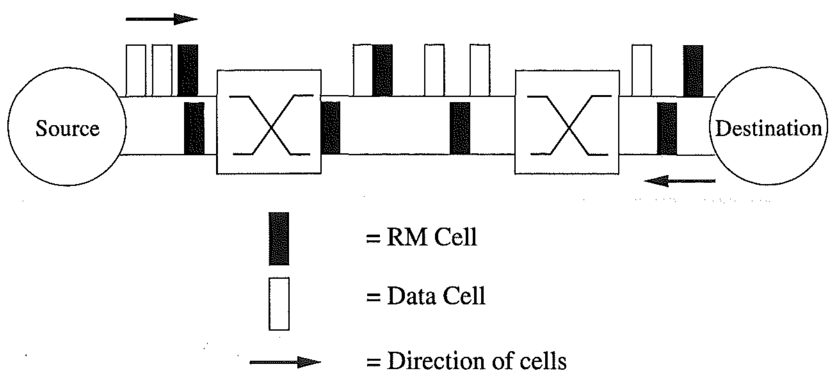


Figure 2.2 The use of Resource Management cells for feedback information in an ABR connection.

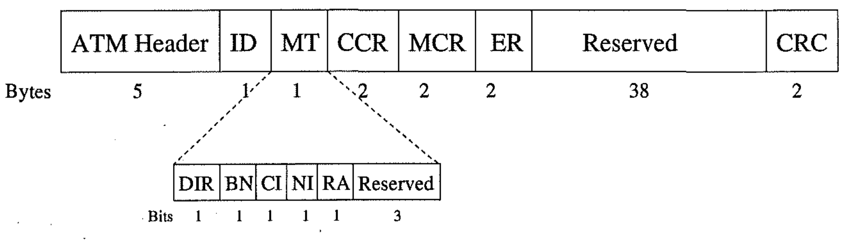


Figure 2.3 The format of the Resource Management cell, outlining the fields and their position.

reducing the value in the Explicit Rate (ER) field in the RM cells or by setting the Congestion Indication (CI) bit in the data cells. The switches cannot reverse either of these two actions (resetting the CI bit or increasing the ER value), as this would lead to a loss of control information from previous switches. Loss of this information could result in other parts of the network becoming congested. The destination then re-routes the RM cells back to the source. It is allowable for the switches to make alterations to the contents of the RM cells travelling in the backward direction. Upon reception of the feedback from the network, the ABR source calculates its Allowable Cell Rate (ACR) according to its specific rate-control protocol, and adjusts its transmission rate so that the rate is no higher than the ACR value.

From the description of the ABR service above, three technical issues emerge which must be addressed in the design of an ABR traffic management scheme. Firstly, a method of measuring the local network state must be chosen. Secondly, an algorithm must be implemented at switches within the network that converts this state information into feedback information. Finally, the way the source responds to feedback information needs to be coded into a control algorithm that governs the transmission rate.

2.2.1.2 Measuring Congestion

The measure that is used for the local congestion at a switch can have significant impact on the performance of a traffic management algorithm, in terms of the stability of the control loop and the implementation complexity of an algorithm [Bonomi and Fen-

Table 2.1 Definitions of the fields in the RM cell for the ABR service.

Field Name	Definition
ID	Protocol Identifier
MT	Message Type
DIR	Direction
BN	Backward explicit congestion Notification
CI	Congestion Indication
NI	No Increase
RA	Request/Acknowledgement
CCR	Current Cell Rate
MCR	Minimum Cell Rate
ER	Explicit Rate
CRC	Cyclic Redundancy Check (10 bits)

dick 1995]. Other considerations include the specific architecture of the switch, which may limit the way an algorithm can be constructed, the resource that needs to be managed more strictly, either buffer memory or link capacity, and finally, the time scale within which the control algorithm needs to be effective.

The simplest approach for measuring the local network state at a node is observing the instantaneous queue length at the output port of a switch. The algorithm compares this length with a specified threshold, and the outgoing link is considered congested if the queue length is greater than the threshold. While this only provides binary information regarding the state of the network, it is does indicate that the resources at the node are being heavily utilised at that point in time. More complex approaches include using multiple thresholds to differentiate between different levels of congestion at the node. With multiple thresholds, hysteresis can be introduced to stabilise the response of the system to instantaneous variations in the queue length. Finally, a differential queue length can also be calculated as a measure of the mismatch between the input traffic load and the link capacity.

However, measuring the queue length is not the only approach to congestion detection. In fact, the mechanism of queueing cells introduces an integral term into the control loop which can be difficult to handle. There are two other primary sources of congestion information—the explicit bandwidth demands at the input port and the queueing delays experienced by connections at a particular node. In some ways, these quantities are more direct measures of traffic load, but there is a tradeoff, since the implementation complexity is higher for both of these approaches. Estimating the bandwidth demands of the aggregate input traffic is complicated by the need to choose a time constant for the estimator, and there is no natural choice for this time constant since the aggregate traffic comprises multiple connections with heterogeneous round-trip delays. With regards to measuring the queueing delay of individual connections, scalability becomes a problem. However, these are valid approaches that can be explored.

2.2.1.3 Generating Feedback Information

Once congestion information has been obtained from the switch, a traffic management algorithm must convert this into feedback information that will control the ABR sources utilising the available bandwidth at the node. This can be simply binary information, based on marking the CI bit in RM cells if the queue is over a threshold or the input traffic load is greater than specified utilisation level of the outgoing link. However, this can lead to oscillations in the transmission rates of the ABR sources. More elaborate approaches include calculating explicit rate allocations for the ABR connections, which allow a switch to distribute available bandwidth in a fair manner, and to ensure that it can support the aggregate rate that arrives at its input port. This is described in more detail in Section 2.2.2 and the major subject of this thesis.

2.2.1.4 Source Rate Control

Finally, an algorithm must be implemented at an ABR source that determines the appropriate transmission rate, based on the information that is received from the switches. In this instance, explicit rate feedback provides the simplest information, as the source can directly adjust its rate according to the ER information it receives. However, the ATM Forum has considered the situation where switches may not be capable of returning explicit rate information, and instead mark CI bits in RM cells. Alternatively, a switch may choose to use this feedback mechanism. In this case, the source algorithm needs to calculate the allowable cell rate, and adjust its rate to match this value. When using binary information, the algorithm requires parameters that specify how much the transmission rate is increased or decreased, and this is typically done in a linear increase, multiplicative increase fashion as in TCP. Since the ATM Forum has chosen rate-based control with an implied emphasis on explicit rate information, most research has approached ABR rate control simply using the explicit rate information without any further processing required at the source.

2.2.2 ABR Feedback Control Mechanisms

The ABR rate control mechanisms have evolved over time from simple binary rate feedback mechanisms to advanced explicit rate control where the ATM network has more direct influence on the transmission rates of the ABR sources [Bonomi and Fendick 1995, Fendick 1996]. Binary feedback was inherited from previous network protocols, such as DECnet and TCP/IP, which set single congestion indication bits in data packets or use the absence of acknowledgements for transmitted packets as indication of congestion within the network. The advantage of binary feedback is its simplicity and minimal computational expense. However, binary feedback results in an oscillatory response from the sources that is due to non-linearity within the control system [Schwartz 1996, Rohrs *et al.* 1996]. Later proposals for the ABR service rate-control mechanisms have included feedback of explicit rate information, which allows the system to adapt to network conditions more quickly than binary feedback, and subsequently make better use of network resources. It also does not suffer from the problem of oscillations when designed correctly. The two types of feedback are compared in detail in the following two sections.

2.2.2.1 Binary Feedback

Binary rate feedback initially considered for the ABR service used the concept of Explicit Forward Congestion Indication (EFCI) [Bonomi and Fendick 1995]. If switches in the path of an ABR connection detected congestion, then the Congestion Indication (CI) bit was set in the headers of all ATM data cells within that connection until the congestion condition no longer existed. At regular intervals, the destination would determine whether or not the CI bits were set in the data cells being received. If the CI bit was *not* set, then the destination would send an RM cell to the source as a positive feedback signal that the source could increase its rate. The source would then increase its rate by a fixed amount, known as the Additive Increase Rate (AIR).

If however, the CI bit *was* set in the data cells arriving at the destination, the destination would stop sending RM cells. Since only the arrival of RM cells allowed the source to increase its rate, the absence of RM cells indicated that the network was experiencing congestion. If an RM cell was not received for a certain period, the source would then decrease its rate by factor proportional to its current rate—using the Rate Decrease Factor (RDF). This resulted in a source response with a linear increase, multiplicative decrease behaviour reminiscent of congestion avoidance and exponential back-off phases in TCP source behaviour. By employing positive feedback, where RM cells indicated permission to increase the source rate, the traffic management protocol ensured that the feedback system was robust, since source rates would be automatically decreased if RM cells were lost due to buffer overflows in the network. It should also be noted that the source was not required to transmit RM cells within its stream in this approach—RM cells are generated by the destination.

2.2.2.2 Explicit Rate Feedback

Explicit rate feedback is conceptually simpler, in the abstract, with network nodes calculating a desired rate based on the measurements of the available bandwidth or by inferring this quantity from queue lengths. An ABR source thus responds directly to this information by limiting its rate to the explicit rate. In the ABR service, the explicit rate information is conveyed by means of the ER field in RM cells, which means that a source is required to transmit a stream of these cells within the stream of data cells. Feedback information can be returned to the source by reducing the ER field in the RM cells, then forwarding the RM cells through the network, with the destination re-directing them back to the source. As can be imagined when considering the potential size and complexity of an ATM network, designing the mechanism that allocates explicit bandwidth to ABR connections is non-trivial, and the explicit rate framework for the ABR service has resulted in a large number of proposals within the literature (refer to the reviews in [Bonomi and Fendick 1995, Arulambalam *et al.* 1996, Fendick 1996] for some of the most significant proposals).

The ATM Forum finally adopted a framework called the Enhanced Proportional Rate Control Algorithm (EPRCA), which combined the approaches of binary feedback and explicit rate control. EPRCA allows the switch manufacturer the flexibility of using either the single CI bit or the ER field as feedback information or even both fields. The source then calculates its Allowable Cell Rate (ACR) based on the AIR and RDF values, compares this with the ER value and chooses the minimum value. Thus, the ABR service could be provided over networks which used switches manufactured by a variety of vendors, instead of limiting it to networks which were only capable of explicit rate feedback.

An explicit rate algorithm that a number of ATM switch manufacturers are currently considering is the algorithm called ERICA (Explicit Rate Indication for Congestion Avoidance) which has been proposed by Kalyanaraman, Jain and their colleagues [Kalyanaraman *et al.* 2000]. A brief outline of ERICA is included as a representative heuristic approach to ABR rate control which highlights some of the key technical issues in designing ABR feedback mechanisms. The interested reader is referred to the journal paper for further details and the pseudocode.

With the standard goals of the ABR service in mind—including maximising utilisation, minimising queueing delay, achieving fairness, stability, good transient performance and robustness—Kalyanaraman *et al.* specify that ERICA monitors a number of parameters at a switch. These are (i) the traffic load on the outgoing link, (ii) the available bandwidth, (iii) a quantity which they define as the load factor (related to the proportion of the available bandwidth which the aggregate ABR traffic is actually utilising), and finally, (iv) the number of active ABR connections. After obtaining estimates of these quantities from the aggregate traffic, ERICA determines the fair share of the available bandwidth for an individual ABR connection. The algorithm also stores the maximum bandwidth allocation in the previous control interval.

ERICA achieves max-min fairness (a concept described more fully in Section 2.2.5) by making an appropriate decision between the fair share, the previous maximum bandwidth allocation and a scaled value of the CCR field in a recently arrived RM cell as

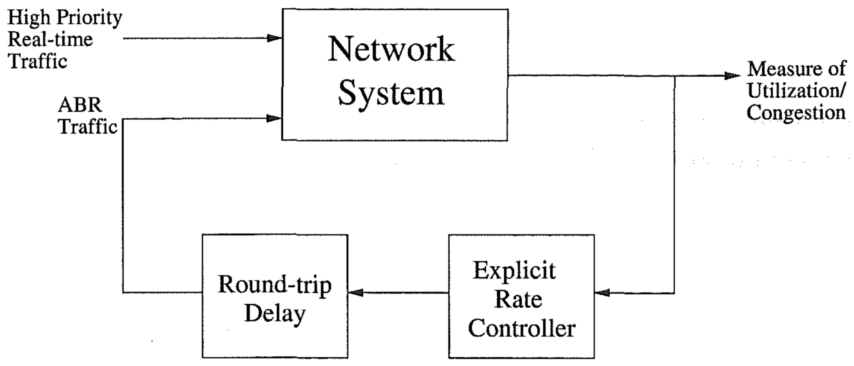

High Priority Real-time Traffic
ABR Traffic
Network System
Measure of Utilization/ Congestion
Round-trip Delay
Explicit Rate Controller

Figure 2.4 Block diagram of the ABR explicit rate control system.

to the value which should be entered in the ER field of the RM cell. This fairness is demonstrated through simulations and a limited analytical proof in [Kalyanaraman *et al.* 2000]. As it stands, ERICA provides a benchmark explicit rate algorithm which can be used to evaluate the performance of algorithms that are proposed in this thesis.

2.2.3 Theoretical Approaches to Explicit ABR Control

Explicit rate control immediately presents itself as an application for the vast amount of control theory which has been developed over the years, since it can be defined as a closed-loop control system, and researchers have lost no time in proposing different control schemes, based on a variety of theoretical approaches from the control domain, the most significant of which are reviewed in this section. This review provides theoretical background for the research work in this thesis, and exposes areas of research that need further study.

The basic control concept is shown in the block diagram in Figure 2.4. Non-adaptive, real-time traffic requires high priority treatment through the network, and in general enters the system without responding to any feedback. In contrast, the ABR system incorporates feedback control, as shown in the diagram. Designing the traffic control mechanism for the ABR service includes using appropriate information regarding the state of the network, and converting this information to rate information to be returned to the ABR sources. The round-trip delay component in the control system has been highlighted in the block diagram, since it is the main cause of instability. To proceed with the control design, models for the network system are required, together with accurate models of the high priority traffic.

2.2.3.1 Application of Classical Control Theory to Rate-based Control

An initial simplistic choice for the explicit rate controller would be a proportional gain K, with the standard gain and phase margins being used to ensure the stability of the system [Hassan *et al.* 1996, Rohrs and Berry 1997]. However, due to the significant round-trip delays that occur in ABR connections, the value of the proportional gain which results from a classical control systems design is so low that the throughput of an ABR connection suffers. Thus, more sophisticated control approaches are required which take into account the unique characteristics of the ABR system.

Benmohamed and Meerkov wrote a notable journal paper on the application of control theory to rate-based congestion control [Benmohamed and Meerkov 1993], in which they appear to anticipate the ABR service even though the paper was not directly written in the context of ABR traffic control. They consider a single congested node with a feedback mechanism for controlling the transmission rates of responsive connections which have paths that traverse the particular node. The rate information is returned back to the sources, with intermediate nodes within the network being able to reduce the value of allowable transmission rate based on the local traffic level at that particular node. As expected, some details of the control protocol do not match the ATM Forum's definition of the ABR service, like Benmohamed and Meerkov return rate information in a control field that exists in every packet generated within their network. In comparison, ABR connections use specialised RM cells for this purpose. However, since the paper was published, the structure of the controller proposed by Benmohamed and Meerkov has been applied to the ABR service by the same authors and other researchers [Zhang and Yang 1997, Benmohamed and Wang 1998, Kolarov and Ramamurthy 1999]. Also, it falls in a class of controllers that use classical control theory for the design of rate-based feedback mechanisms, so the approach is outlined here and its limitations are discussed.

In Benmohamed and Meerkov's original paper [Benmohamed and Meerkov 1993], the control algorithm is based on using queue length measurements and previous control information to generate the current rate-control value that is returned to the sources. The rate information is generated using the following algorithm:

$$u_i(n+1) = \mathrm{Sat}_{u_{\max}} \left\{ u_i(n) - \sum_{j=0}^{J} \alpha_j \left[x_i(n-j) - x^* \right] - \sum_{k=0}^{K} \beta_k u_i(n-k) \right\}, \tag{2.1}$$

where the u variables are the calculated rates and the x variables are the measured queue lengths. The queue threshold is defined as x^*, and the set of variables, α_j and β_k, are the control parameters. The function $\mathrm{Sat}_{u_{\max}}$ enforces the maximum value $u_{\max}$ and the minimum value of 0 for the explicit rates u_i.

The resultant task of designing the controller is to determine the values of the control parameters α_j and β_k such that the desired transient response and stability objectives are met. This involves the appropriate placement of the poles of the closed loop system—a fairly standard process in classical control design. Stability requires that closed loop poles lie within the unit disk, while achieving good transient performance involves placing the poles so that the system has fast time constants without oscillating too wildly. By investigating the resulting system in steady state, Benmohamed and Meerkov show that the constraints on the control parameters are [Benmohamed and Meerkov 1993]:

$$\begin{aligned} \sum_{j=0}^{J} \alpha_j &> 0, \\ \sum_{k=0}^{K} \beta_k &= 0. \end{aligned} \tag{2.2}$$

Stability analysis further shows that the number of control parameters could be chosen with $J = 1$ and $K = D$, where D is the maximum round-trip delay. This number of control parameters results in a proportional-derivative (PD) controller with respect to the queue length—a familiar controller from classical control theory and one which has well-known characteristics. Benmohamed and Meerkov investigate both adaptive and robust controllers in their paper, and show, using simulations, that the adaptive controller achieved higher performance because of its better transient response while incurring a higher computational cost due to the expense involved with updating the control parameters parameters. They also investigate design issues like the choice for the control update period and the queue threshold, and operational issues such as the interaction of the control protocol with dynamic routing and methods of monitoring connections to ensure that they are cooperating with the control information returned by the nodes. Thus, the paper covers many of the areas that are of interest for the ABR service.

The significance of Benmohamed and Meerkov's approach is that, in its general form, the control is in the class of classical controllers, where the network system is represented by a linear transfer function and the controller consists of a rational function that places the closed-loop poles correctly. Thus, their approach is well-suited to further development using classical control theory. One limitation of their original paper is that they only analysed the case of a single congested node—since then, they have analysed a more general network scenario with multiple congested nodes [Benmohamed and Meerkov 1997]. However, a deeper issue that they have not adequately addressed in their research is the stochastic structure of non-cooperative, high priority traffic that may also be entering the node. They do introduce randomness into their traffic towards the end of the paper, but this is only done by modelling the rate of the connections as independent random variables using a uniform distribution and the subject is only given a brief treatment just prior to the conclusions. Considering that modern networks carry a heterogeneous mixture of traffic types, and that connections must be serviced based on their QoS requirements, it is important that rate control design includes the characteristics of any non-adaptive traffic that will be demanding bandwidth from the outgoing link. A broader view of the network traffic needs to be taken, as opposed to simply applying classical control theory to the problem.

2.2.3.2 Global Optimisation of the ABR Service

The focus of the ABR rate control problem can be considered in a broader sense than just optimising the performance of a single congested node within a network, since a complete system almost invariably involves multiple nodes and multiple ABR users competing for resources within the set of network nodes. The theoretical framework for approaching this view of the network is global optimisation, based either on dynamic game theory [Altman and Başar 1998] or operational research techniques using pricing mechanisms [Kelly *et al.* 1998, Low and Lapsley 1999]—approaches that bear some resemblence to each other. Global optimisation requires a measure of the global performance of the network as a system, such as a global utility function or a cost function, and optimisation of the system performance implies either maximising the global utility function or, equivalently, minimising the cost function.

Due to the size and complexity of the networks in question, solving the primal problem, which involves coordination between all the network users, is a difficult task and it is more feasible to solve the dual of the optimisation problem, where each user optimises their own individual performance objective. This approach results in decentralized control, such that individual components in the network measure local performance and convey this information in an uncomplicated and unambiguous manner to users. Users then choose their transmission rates to maximise their own individual utility functions. The goal of the dual problem is to converge to the Nash equilibrium, which is the network state where there is no incentive for any users to move away from that particular point of operation. Clearly, the purpose of the optimal control is also to design multiuser policies that are stable. The key results proposed by [Altman and Başar 1998, Low and Lapsley 1999] are summarised below to illustrate the theoretical concepts involved with the global optimisation approach. Kelly's work [Kelly *et al.* 1998] is described in the context of the dynamics of TCP in Section 2.3.5, as his rate control model is closely related to TCP.

The approach of Altman and Başar [Altman and Başar 1998] is similar to optimal control, since they define an individual cost function of the form (an example function among others defined in the paper):

$$J_m(u) = \int_0^\infty \left(|x(t)|^2 + \frac{1}{c_m}|u_m(t)^2| \right) dt, \tag{2.3}$$

for user m, where $x(t)$ is the deviation of the queue around a threshold, c_m is a positive constant and $u_m(t)$ is the difference between the transmission rate of controlled user and its allocated proportion of the available bandwidth. The objective is to determine the function $u_m(t)$ that minimises the cost function $J_m(u)$. This does not necessarily achieve the global optimum, since it is only the individual user's performance that is being maximised—hence the term noncooperative differential game. Global optimisation requires minimising a team cost function of similar form to Equation (2.3) as defined in the paper, which leads to a cooperative game scenario.

The focus of the Altman and Başar's paper is to choose controllers that solve these differential games and then to demonstrate asymptotic convergence of the solutions to the Nash equilibrium. Altman and Başar considered candidate solutions of the form

$$u_m(t) = -\beta_m x(t) \tag{2.4}$$

with $\beta_m = \bar{\beta} - \sqrt{\bar{\beta} - c_m}$, where $\bar{\beta} = \sum_{m=1}^{M} \beta_m$. This value for the parameter β_m achieves the Nash equilibria with the value of the cost function given by

$$J_m(u) = \frac{\beta_m}{c_m} x_0^2, \tag{2.5}$$

for an initial state x_0. Altman and Başar consider implementing the above differential game using a general greedy decentralized algorithm where the choice of subset of users that update their transmission rates is given by one of the following algorithms: Parallel Update, Round Robin, Random Polling or Stochastic Asynchronous Algorithms. It is important that any decentralised algorithm converges to the unique equilibrium of the system, and Altman and Başar proved the convergence of the Parallel Update and Round Robin Algorithms. Essentially, convergence to the unique equilibrium depends on the eigenvalues of the characteristic equation of the system being within the unit disk—a consideration that parallels placing the poles within the unit disk for Benmohamed and Meerkov's controller which has been outlined in Section 2.2.3.1.

While Altman and Başar's work uses some powerful analytical results from dynamic game theory, it is limited in a number of ways in being applied directly to the ABR service. Firstly, they do not consider any delay in returning the control information to the users, which is unrealistic since any ABR connection will have non-neglible latency. This limitation is noted in the paper and is referred to as further work. However the significance of this limitation can be observed by recognizing that the controller they are proposing in Equation (2.4) is actually a proportional controller. To ensure the stability of a system with large feedback delays, the gain of a proportional controller must assume a very low value which means that the throughput of the connection will suffer. A further limitation of the work is that it did not consider the impact of non-adaptive, real time traffic flowing through the system. While multiple types of traffic are considered, these are fundamentally rate-controlled and do not introduce the complex stochastic structure that traffic like video and multimedia traffic is known to have.

Low and Lapsley's [Low and Lapsley 1999] approach to global optimisation differs from Altman and Başar's work in that they develop a pricing mechanism where the network links advertise bandwidth "prices" which a source s can use to determine its transmission rates based on its individual utility function U_s. These prices can be associated with a charging system, but this is not necessarily the case and prices can simply be viewed as control information conveyed to users. Optimisation of the system requires steering the network towards the point where the total utility over all sources is maximized. By considering the dual problem, the problem can be solved in a decentralized manner, where nodes calculate local prices based on link utilisation and sources use prices aggregated along their connection path to determine their rates. The specific algorithm for adjusting the link prices is based on the gradient projection method and is defined as:

$$p_l(t+1) = \left[p_l(t) + \gamma\left(x^l(p(t)) - c_l\right)\right]^+, \tag{2.6}$$

where p_l is the link price, γ is the step-size, x^l is the total demand on the link and c_l is the link capacity. Prices are non-negative, which is achieved by the operator $[\cdot]^+ = \max(\cdot, 0)$. Each source determines its transmission rate based on:

$$x_s(p) = \left[U_s'(p)\right]_{m_s}^{M_s}, \tag{2.7}$$

where m_s and M_s are the minimum and maximum transmission rates respectively of source s. Provided that the individual utility functions U_s of the sources in the network are increasing, strictly concave and twice continuously differentiable, and the curvatures of U_s are bounded away from zero, Low and Lapsley prove that the synchronous distributed algorithm will reach an equilibrium point that is optimal.

Having developed the basic structure of their approach, Low and Lapsley [Low and Lapsley 1999] extend this work to an asynchronous distributed control algorithm for the system where link prices are updated at times other than when the sources calculate their transmission rates. This network model allows the introduction of non-neglible round-trip delays which connections experience through a network. The adjustment that Low and Lapsley make to the basic control algorithm to handle these network delays is to use estimates of control variables based on weighted averages of the past values of the variable. Thus, a network link uses a weighted average of the input traffic rate arriving at the link to update its price, while sources average over recent price information to determine its current transmission rate. Low and Lapsley show that this asynchronous algorithm also converges to the dual optimal solution. They complete the paper with a discussion of other issues concerning their algorithms, such as fairness and performance in a time-varying environment, and then include experimental results that compare a testbed implementation of the system with the theoretical performance of the system.

The work of Low and Lapsley stands as a very thorough and enlightening study into the mechanisms that can be implemented for rate-control. They have developed the algorithm with reasonable generality, in terms of the utility function and estimation methods, so that proposed control protocols can be modelled and evaluated—as Low has shown with TCP Vegas [Low *et al.* 2000]. However, the approach still does not integrate other types of traffic that do not respond to feedback information from the network. Certainly, they have included a filtering mechanism for estimating system state variables (specifically, a weighted moving average), and this can be exploited to utilise the stochastic structure of high priority traffic. The filtering mechanism can, however, be constructed based on the self-similar nature of network traffic, and it is this area which is the focus of this thesis.

2.2.3.3 Explicit Integration of Non-adaptive Traffic

The argument so far in the appraisal of previous theoretical work in ABR traffic control has been that explicit integration of the characteristics of high priority traffic is an important consideration that has not been studied to any particular depth. Considering that ATM is an integrated services technology that aims to satisfy the quality requirements of both adaptive traffic and non-adaptive traffic, it seems necessary that the interaction between these traffic types is considered, particularly for adaptive traffic. This is because adaptive traffic aims to maximise its utilisation of any bandwidth remaining after the bandwidth demands of the high priority traffic have been satisfied. To do this, the nature—both the instantaneous distribution and the correlation structure across time—of the high priority traffic needs to be recognized and incorporated into the design of the ABR control mechanisms. This is the fundamental area of research on which this thesis is based. In this section, some previous research is introduced that has considered this issue in a limited sense.

In our early work, this integration is accomplished by in a simplisitic manner by augmenting proportional control with a feedforward term [Östring *et al.* 1997]. Feedforward control involves measuring the available bandwidth at a link and using this information in the control decision, which explicitly incorporates the instantanteous bandwidth requirements of any background traffic into the control algorithm. Our paper considered the stability of the system and its performance, in terms of the variance of buffer level and its response time. A figure of merit was introduced—the ratio of the buffer variance to the $3dB$ bandwidth—to demonstrate the robustness of the system with regards to parameters of an AR(1) model of the available bandwidth. Extending this work involves using more accurate models of the utilisation of bandwidth at a link, and this thesis considers self-similar models for the background traffic bandwidth requirements.

The other logical approach to incorporating bandwidth utilisation into the control algorithm is to filter previous utilisation measurements to estimate future bandwidth requirements of the high priority traffic. This is the approach taken by Zhao *et al* [Zhao *et al.* 1997], who develop an explicit-rate ABR control scheme using a low-pass filter for VBR traffic. Their motivation is the same as ours—to provide the QoS requirements for the VBR traffic and at the same time make optimum use of the available bandwidth, while their choice of filter is based on previous research results of San-qi Li *et al* [Li *et al.* 1995]. In this paper, the authors show that low-frequency components of traffic dominate its behaviour through a network, hence Zhao *et al.*'s choice of the low-pass filter.

By decoupling the feedback loops for different ABR connections and modelling the low-pass filtered VBR rate as a level process, Zhao *et al.* [Zhao *et al.* 1997] develop the following ABR control mechanism based on achieving a target link utilisation:

$$u(n) = \frac{\rho C - r_L(n)}{N} \tag{2.8}$$

where $u(n)$ is the explicit rate information returned to the ABR source, ρ is the link utilisation, C is the link capacity, $r_L(n)$ is the filtered, low-freqency VBR rate and N is the total number of ABR connections using the link. This design is based on using H_2 optimal control. Through simulation studies, Zhao *et al* compare the performance of their algorithm with the OSU scheme proposed by Raj Jain and other researchers at Ohio State University, and demonstrate the excellent transient behaviour of the H_2 explicit-rate control mechanism. Zhao *et al.* also show that their controller is robust to the highly bursty nature of VBR by varying the number of MPEG video traces that are aggregated to simulate the background VBR traffic.

A significant amount of San-qi Li's research is based on the premise that the most meaningful information regarding network traffic lies in its low frequency components. While it is true that there is significant power in the low-frequency region of the spectrum of long-range dependent processes (due to the divergence at the origin), this does not necessarily mean that adaptive traffic, that seeks to make use of available bandwidth, is best controlled by low frequency information. In any case, it is worthwhile studying this area to determine the optimal filtering process for controlling elastic traffic that is competing with long-range dependent background traffic. Later chapters in this thesis address this very issue.

2.2.4 Credit-based control

The ATM Forum also considered a credit-based flow control mechanism for the ABR service [Ozveren *et al.* 1994, Kung and Morris 1995, Ramakrishnan and Newman 1995], but the rate-based control approach was overwhelmingly accepted after considering extensive simulations of both the rate-based and the credit-based schemes. For the sake of completeness, the credit-based approach is introduced in this section, together with some of the reasons for the Forum's choice of rate-based control.

The credit-based control scheme considered by the ATM Forum is a link-by-link flow control approach where downstream nodes return credits to upstream nodes regarding the available buffer resources at the downstream node. Each credit received by the upstream node allows it to transmit one cell, and the node can continue sending cells until it has zero credits. At this point, it has to cease sending cells and it can only resume transmission when it has received further credits from the downstream node. Flow control still occurs from an end-to-end perspective as cells will migrate through the network when credits are arrive at each node. However, each node still retains greater control of the amount of traffic arriving at the node.

The advantages of the credit-based approach are twofold. The first is that there are shorter control loops in the scheme, since it operates in a link-by-link fashion, and this allows it to achieve higher performance in terms of utilisation and fairness. This is because each node can determine its link utilisation by monitoring its queue size, and promptly increase throughput by sending credits to the upstream node. Conversely, if the node is approaching congestion, then it can suppress the generation of credits. Credit-based control is also efficient with regards to fairness, as it maintains per-VC queues. The second advantage is the control mechanism can handle transient "cross" traffic more effectively. This is important as traffic is characteristically bursty, with the majority of file transfers being of relatively short duration. When a bursty "cross" connection reduces the amount of available bandwidth at a node, the node will send less credits to its upstream node, and this effect will ripple up the path of controlled connection. Thus excess cells transmitted by the source will be buffered at intermediate nodes in the path, rather than all of the excess cells arriving at the queue of the node that is bottlenecked by the bursty connection.

The disadvantages of credit-based flow control are that it requires much more complex and expensive hardware at the switch, as credits accounting is computationally expensive and must be implemented in the hardware for the switch to be able to handle large numbers of cells. Also, significant buffering is required when this control mechanism is applied in wide area networks that support large numbers of connections. Large delays and high link speeds exist within these network configurations, and the network must be able to store large quantities of cells until further credits arrive from down-stream nodes. Credit-based control also uses per-VC queueing, which further increases the complexity of managing the buffer. After considering the advantages and disadvantages of both approaches and reviewing extensive simulation studies, the ATM Forum chose the rate-based approach because of the simplicity and flexibility of its implementation within an ATM switch [Kung and Morris 1995].

2.2.5 Max-min Fairness

Another important consideration in the design of ABR rate control is the issue of fairness. While some flexibility for the fairness definition is required for ABR service providers, who may place a heavier emphasis on one particular performance criteria, the principle of *max-min fairness* is the definition used by the ATM networking community [Arulambalam *et al.* 1996]. In max-min fairness, connections are categorized at a specific node into constrained connections, which are connections that are bottlenecked elsewhere in the network, and unconstrained connections, which are only bottlenecked at that particular link and can utilise any bandwidth which is offered by the node. Fairness is achieved when the unconstrained connections are offered an equal share of the bandwidth not utilised by the constrained connections [Bonomi and Fendick 1995]:

$$\Lambda = \frac{\mu - \sum_{ConstrainedSet} \lambda_i}{N - M}, \tag{2.9}$$

where Λ is the bandwidth offered to the unconstrained connections, μ is the link capacity, λ are the rates of the M constrained connections and N is the total number of connections using the link. The goal of max-min fairness is to achieve full utilisation of the link bandwidth by distributing any unused bandwidth from individual connections to other connections which can use it. Max-min fairness leads to maximum total throughput, with the only aspect limiting the bandwidth achieved by a particular connection being its own bottleneck link.

In terms of implementing max-min fairness, if switches within an ATM network had global information regarding (i) available bandwidth at each node and (ii) the resources allocated to ABR connections elsewhere in the network, then achieving max-min fairness is reasonably straightforward [Arulambalam *et al.* 1996]. The solution can be found by starting at the most heavily loaded link, and bandwidth is iteratively allocated to connections throughout the network until all the available bandwidth is distributed to the ABR sources. In reality, the size of networks and complexity of such an approach is not feasible, and fairness has to be achieved by distributed algorithms. However, optimally these algorithms need to (i) converge rapidly to the optimal solution, (ii) should be scalable as the number of connections increase and (iii) must be interoperable with switches in other domains of the network that may using other types of rate algorithms.

A number of ABR rate algorithms have been proposed that achieve fairness in a way that is computational feasible. These include the enhanced proportional rate control algorithm (EPRCA), ERICA and the fast max-min rate allocation algorithm (FMMRA) [Arulambalam *et al.* 1996]. Algorithms that endeavour to achieve fairness, however, do not need to be explicit rate algorithms. Nevertheless, when considering this type of algorithm, the basic procedure includes checking RM cells that arrive at a node to determine whether the connection is able to utilise its fair share of the link capacity, or whether it is bottlenecked elsewhere. The switch also must keep track of how much of the available bandwidth is being used and the maximum ER value being written into outgoing RM cells. The specific way that the switch uses this information to achieve fairness depends on the particular rate algorithm.

2.3 WINDOW CONTROL DYNAMICS IN IP NETWORKS

Another important example of reactive traffic control is the Transmission Control Protocol (TCP) used in IP networks. While TCP uses a window-control protocol to control its transmission rate, rather than an explicit rate control similar to that incorporated by the ABR service, the concept involved is not totally unrelated. TCP sources still adapt their transmission of packets into the network based on feedback, which in this case is information regarding packets that have been received successfully. Indeed, modelling window dynamics can be abstracted so that a rate-based approach can be applied [Schwartz 1987, Bolot and Shankar 1990].

2.3.1 The Purpose of TCP and its Window Control

IP functions at the network layer and provides mechanisms for routing packets from source to destination, but does not ensure reliable end-to-end delivery of packets. This functionality must be implemented in the higher layers, and TCP can be layered over IP to provide a connection-oriented, reliable, byte-stream service [Stevens 1994]. This service establishes a connection between the source and destination, with a reliable byte flow being ensured for this connection by TCP keeping track of the number of bytes successfully transferred. TCP's reliability involves the receipt of acknowledgements (ACKs) from the destination that specify which byte it is waiting to receive. TCP detects and retransmits packets that have been lost or corrupted, since no ACK is received for these packets.

The information that TCP requires is contained in the 20-byte TCP header that is added to the head of an application packet, which is shown in Figure 2.5 [Stevens 1994]. The names of the fields are self-explanatory, while the functions of the header flags are given in Table 2.2. Of particular interest are the fields that store the sequence number (for a packet with a payload containing data) and the acknowledgement number (employed in ACKs). It is these fields which allows the destination to order the packets it receives, and acknowledge the byte that it is waiting to receive, so that the source can retransmit any packets that have been lost in the network.

The use of individual acknowledgements to ensure the reliable delivery of every packet can be relatively inefficient in terms of data transfer, since the transmission rate is limited to one packet per round-trip time. This can be improved by using a sliding

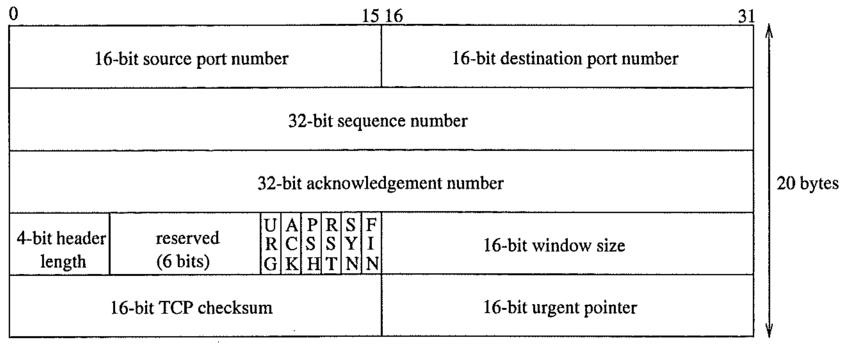


Figure 2.5 The TCP header fields.

Table 2.2 Definitions of the flags in the TCP header.

Flag	Definition
URG	Urgent pointer is valid.
ACK	Acknowledgement number is valid.
PSH	Data should be passed to application immediately.
RST	Reset TCP connection.
SYN	Synchronize sequence numbers.
FIN	Sender finished transmitting data.

window, where the source can transmit multiple packets into the network without receiving acknowledgement for the first packet transmitted within the group of packets. This concept is shown in Figure 2.6. The window size limits the number of unacknowledged packets that can be in flight in the network, which ensures that neither the network nor the destination is overwhelmed with packets.

A static window size can be used, but it is highly unlikely that the window will be well matched to the capacity of the network. This means that either the network will be overwhelmed with packets, if the window is too large, or the transmission rate of the source will be unnecessarily restricted, if the window size is too small. Thus, a more efficient approach is adapt the window size, and window dynamics form the basis of the traffic control algorithm within TCP. ACKs are used by TCP as positive feedback to increase the window size, which increases the transmission rate of the source. Duplicate ACKs and time-outs of the retransmission timer indicate that packets have been lost within the network, and TCP responds by reducing the size of the window to avoid further congestion. The window-control protocol used in TCP is described in detail in this section and some analytical methods of modelling the window dynamics are presented. More recently there have been a number of proposals that suggest using explicit congestion notification, where routers set an ECN bit in the IP header if they experience heavy traffic loads, rather than simply using packet losses as feedback [Floyd 1994, Kelly *et al.* 1998]. ECN proposals are described in Section 2.3.4.2.

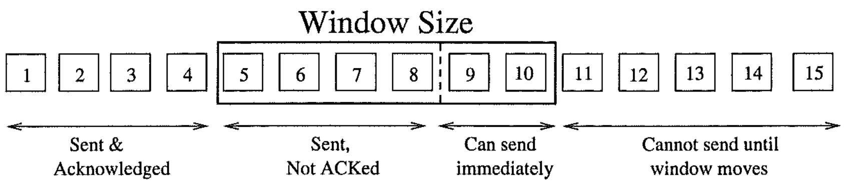


Figure 2.6 A graphical representation of the concept of a window with a sequence of packets.

2.3.2 Defining the Window Control Protocol In TCP

There are three main versions of TCP—Tahoe, Reno and Vegas. Tahoe is the TCP protocol that was designed by Jacobson to avoid congestion collapses that were occurring in the Internet with earlier versions of TCP [Jacobson 1988]. It would now be considered the base TCP version from which other versions have their roots. Later versions of TCP (Reno and Vegas) have been introduced to increase bandwidth utilisation. The basic difference between the TCP versions is the way the protocols infer information regarding the network state and packet losses from the ACKs they receive, and how they adjust the window size in response to this information. The Tahoe, Reno and Vegas versions are introduced progressively in Sections 2.3.2.1, 2.3.2.2 and 2.3.2.3.

It should also be noted that there are many variants of these three basic versions of TCP, including NewReno, Selective Acknowledgement (SACK) and Forward Acknowledgement (FACK). Since our purpose is to introduce the concept of window-based congestion control, only Tahoe, Reno and Vegas are described, and the interested reader can find the details of their variants elsewhere [Fall and Floyd 1996, Mathis and Mahdavi 1996, Floyd and Henderson 1999, Barakat *et al.* 2000].

2.3.2.1 TCP Tahoe

There are two phases of the dynamics of the congestion window (*cwnd*) in the Tahoe version, the slow start (SS) phase and the congestion avoidance (CA) phase [Jacobson 1988, Stevens 1994]. The congestion window starts at an initial value of 1 packet and during the slow start phase the value of *cwnd* increases by 1 for each acknowledgement received from the destination. This results in an exponential increase in the window size when the window size is plotted using the TCP connection's round-trip delay as the time unit. A typical trace of the window dynamics of TCP Tahoe is shown in Figure 2.7, where the rapid increase in window size can be observed as the source begins transmission. Figure 2.7 also shows that the SS phases is re-entered whenever a packet loss is detected. Thus, during SS the algorithm can be summarized as follows:

$$\begin{aligned} &cwnd \leftarrow 1, && \text{Initial value} \\ &cwnd \leftarrow cwnd + 1, && \text{while } cwnd < ssthresh. \end{aligned} \tag{2.10}$$

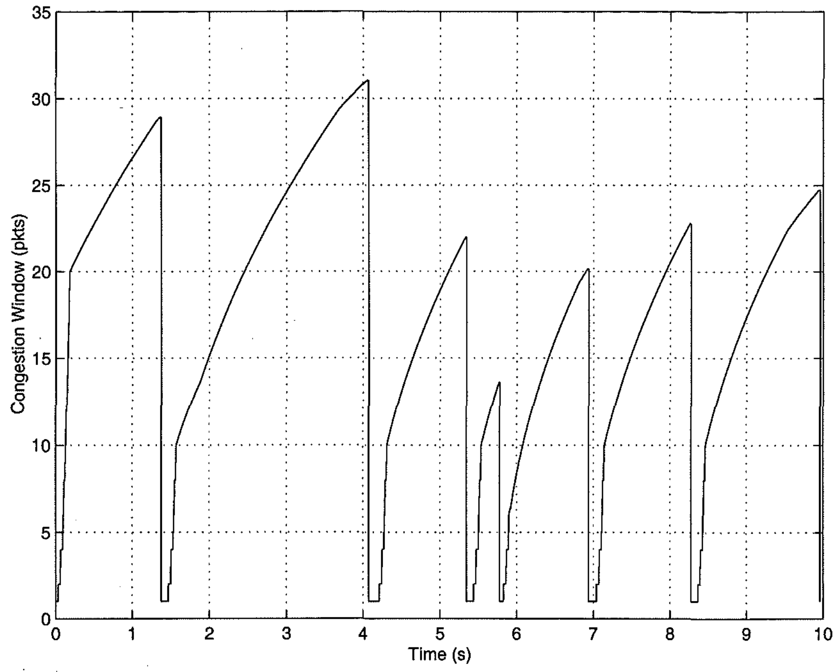


Figure 2.7 Trace of the window dynamics of TCP Tahoe.

The second phase of the window dynamics begins when the window reaches a threshold called the slow start threshold (*ssthresh*). This threshold is a state variable within the TCP algorithm which is an estimate of the capacity of the network. Upon reaching this threshold, the congestion avoidance phase is entered—the increase in window size is approximately linear (refer to Figure 2.7), and the source increases its transmission rate in a more cautious manner for each ACK received:

$$cwnd \leftarrow cwnd + \frac{1}{cwnd}, \text{ when } cwnd > ssthresh. \tag{2.11}$$

In this phase the TCP is slowly probing the network for more bandwidth until a packet loss is detected.

Since the IP layer does not signal the source if a packet is lost on route, the TCP source must infer this lost from the absence of an ACK for the packet. One method of doing this is by employing a retransmission timer. If this timer expires for a particular packet, TCP considers the packet lost and retransmits the packet. However, retransmission timers used in TCP protocols have a very coarse granularity, typically in the order of 500ms, to ensure that packets are not retransmitted unnecessarily. This means that a TCP source will be idle for a significant period of time if it waits for a retransmission timeout, and it is possible for TCP to deduce that a packet has been lost by other means. In particular, TCP Tahoe uses duplicate ACKs as early indication of a packet loss. Whenever a TCP receiver receives a packet that is out of sequence, it returns a duplicate acknowledgement for the byte that it is still waiting to receive, thus signalling a missing packet in the byte-stream. This could simply be due to the missing packet experiencing a longer delay through an alternative route in the Internet, so every duplicate ACK received does not automatically mean a packet has been lost. However, the Fast Retransmit (FRXT) Algorithm interprets the third duplicate ACK to mean that a packet has actually been lost in the network, and retransmits the packet. In this way, the source does not need to wait for a lengthy timeout to occur before recovering the loss.

After detecting that a packet has been lost using either method, TCP Tahoe resets *ssthresh* to half of the current window size, effectively reducing its estimate of the available capacity in the network. Also, the congestion window is reset to 1:

$$\begin{aligned} &ssthresh \leftarrow \frac{cwnd}{2}, \text{ when a Packet loss is detected,} \\ &cwnd \leftarrow 1. \end{aligned} \tag{2.12}$$

The TCP source once again returns to the slow start phase (see Figure 2.7).

2.3.2.2 TCP Reno

The response of TCP Tahoe to packet loss is a drastic reduction in the transmission rate of the source, since the source has to re-enter the slow start phase, and this leads to significant under-utilisation of network bandwidth. Also, the transmission rate of the source is very bursty. This problem has been addressed in TCP Reno, which executes the Fast Recovery (FRCV) algorithm in addition to the Fast Retransmit algorithm when the source receives multiple duplicate ACKs [Fall and Floyd 1996, Barakat *et al.* 2000]. The problem with using only Fast Retransmit is that the source must wait for an ACK for a retransmitted packet before it can continue transmitting other packets, which means that the "transmission pipe" quickly empties. If further duplicate ACKs arrive back at the source, this implies that packets are still being delivered to the receiver sucessfully, so the source can continue transmitting packets while waiting for the retransmitted packet to be ACKed. Hence, the aim of Fast Recovery is to keep the transmission pipe full. It uses duplicate ACKs as signals that it can transmit another packet into the network, thus maintaining the current flow rate of packets, as clocked by arrivals of duplicate ACKs.

The Fast Recovery algorithm is implemented as follows. Firstly, when the number of duplicate ACKs reaches a particular threshold (typically 3), the algorithm makes *ssthreshold* half the current size of the *cwnd*, as TCP Tahoe does. However, instead of setting *cwnd* to 1, the algorithm sets the window equal to the slow start threshold plus additional segment sizes for the number of duplicates ACKs received:

$$\begin{aligned} ssthresh &\leftarrow \frac{cwnd}{2}, \\ cwnd &\leftarrow ssthresh + ndup \times SS. \end{aligned} \tag{2.13}$$

Each arriving duplicate ACK increments the variable *ndup* and hence the congestion window, allowing the source to transmit another packet. As soon as the retransmitted packet has been acknowledged, TCP Reno exits Fast Recovery, and returns to the regular congestion avoidance phase. The congestion window is deflated back to the value of *ssthresh*

$$cwnd \leftarrow ssthresh, \tag{2.14}$$

as this is the estimate of the capacity of the "transmission pipe" when the packet loss occurred.

Since the TCP source transmission rate is not throttled as severely as in the Tahoe version, the utilisation of link bandwidths is higher. This can be seen in Figure 2.8, where a packet loss that is detected by duplicate ACKs only reduces the window size by half rather than setting it to one again. This improves the overall throughput, especially for connections with long RTT, since the *SS* phase lengthens considerably as the delay increases. Furthermore, the transmission rate from the TCP source is smoother, which improves the performance of the TCP connection. The reason for preferring smoother source behaviour over bursty behaviour is that a smooth source is less likely to experience packet drops due to overwhelming a tail drop queue within the network, and it has less impact on the performance of other connections.

2.3.2.3 TCP Vegas

The final flavour of TCP that is introduced in this thesis is TCP Vegas, which was proposed by Brakmo and Peterson [Brakmo and Peterson 1995]. Their motivation was to make more efficient use of the available bandwidth by adapting the window size in TCP using explicit throughput information deduced from variations in the round-trip time (RTT). TCP Reno attempts to find an optimal window size by linearly increasing the window until a packet loss occurs. Thus, Reno needs to create losses to determine the capacity of the network, and backs off significantly when it overestimates this capacity. TCP Vegas, on the other hand, compares the actual throughput that is currently being achieved with the expected throughput, and adjusts the window size in a manner that matches the true capacity more closely.

The Vegas algorithm maintains an estimate of the minimum RTT (the variable *baseRTT*), which corresponds to propagation delay. Thus, the expected throughput for

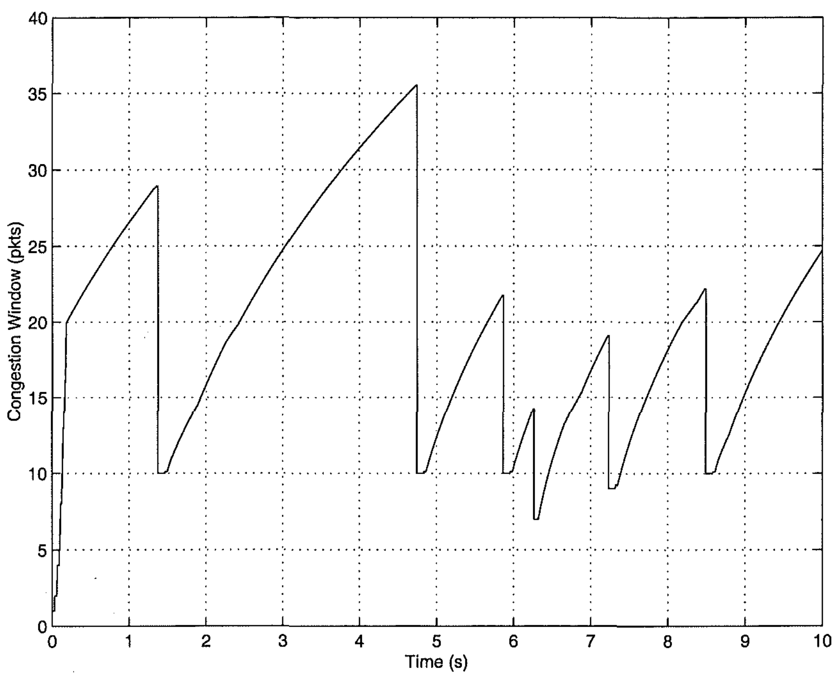


Figure 2.8 Trace of the window dynamics of TCP Reno.

the current window size, when packets do not experience any queueing delays, is:

$$Expected = \frac{WindowSize}{BaseRTT}. \tag{2.15}$$

Whenever an ACK arrives at the source, Vegas updates *baseRTT* if the timestamp on the ACK implies a lower RTT value than *baseRTT*. Every round trip, the actual throughput is also calculated, and is given by:

$$Actual = \frac{BytesTransferred}{RTT}. \tag{2.16}$$

The actual throughput is generally less than the expected throughput, since packets will be delayed at queues within the network when the network load increases. The difference between these two throughputs $Diff = Expected - Actual$ indicates the level of congestion within the network and the error that the TCP source has made in estimating the available bandwidth.

There are two thresholds for the *Diff* variable that are defined in TCP Vegas, namely α and β where $0 < \alpha < \beta$, which determine how the congestion window is adjusted. If the actual throughput is close to the expected throughput ($Diff \approx 0$), then there are negligible queueing delays and network utilisation may be low. Hence, the window size should be increased. Conversely, if the variable *Diff* is too large (greater than β), then the actual throughput is too low due to significant traffic load in the network, and the window should be reduced. The specific window control algorithm is given by [Brakmo and Peterson 1995]:

$$cwnd \leftarrow \begin{cases} cwnd+1, & \mathit{Diff} < \alpha; \\ cwnd, & \alpha < \mathit{Diff} < \beta; \\ cwnd-1, & \mathit{Diff} > \beta. \end{cases} \tag{2.17}$$

Brakmo and Peterson chose the conservative values 1 and 3 for α and β respectively, as there is a tradeoff between increasing throughput and reducing the number retransmissions. While the units for these thresholds are bytes per second, they indicate the number of excess packets that need to be buffered within the network for the current window size. Thus, the aim of TCP Vegas is to control the flow of packets entering the network so that large numbers of packets do not need to be buffered and the network load is closely matched to its capacity. At the same time, the algorithm still aims to have a small number of packets buffered, as indicated by the α threshold, to ensure that the network is not under-utilised.

The above description of TCP Vegas concerns the congestion avoidance phase of the algorithm. Brakmo and Peterson also addressed the slow start phase and the response of Vegas to packet losses. When a Vegas source starts transmitting packets, it uses the same exponential increase in window size as TCP Tahoe or Reno. To ensure that the window size does not overshoot the transmission capacity significantly, Vegas only allows exponential growth in the window on alternative ACKs during slow start, thus allowing the algorithm to settle at the optimal window size more quickly. The other modification that Brakmo and Peterson introduced was the retransmission of packets when the source receives duplicate ACKs. Instead of waiting for three duplicate ACKs, Vegas uses a single duplicate ACK as a signal to check whether it should have received an ACK for the packet in question. It bases the decision to retransmit the packet on whether the delay in receiving the ACK is greater than the estimated RTT for the connection. Thus, Vegas can retransmit lost packets faster than Reno, which may have to wait for a retransmission timeout if many packets in a single window have been lost. These minor modifications add further improvements to the performance of TCP Vegas. Indeed, it has been shown that slow start and congestion recovery mechanisms incorporated in Vegas contribute more to its higher performance than its better-known congestion avoidance mechanism [Hengartner *et al.* 2000].

A typical trace of the congestion window in TCP Vegas is shown in Figure 2.9, where the link capacity is modulated by background traffic. From the figure, it can be seen that Vegas rapidly stabilises at a smooth transmission rate between changes in the background traffic, which means that more packets will be delivered to the destination without loss. In their paper, Brakmo and Peterson demonstrate that TCP Vegas achieves higher throughput as compared with TCP Reno [Brakmo and Peterson 1995], and analytical studies by other researchers have proven other properties of the algorithm, such as

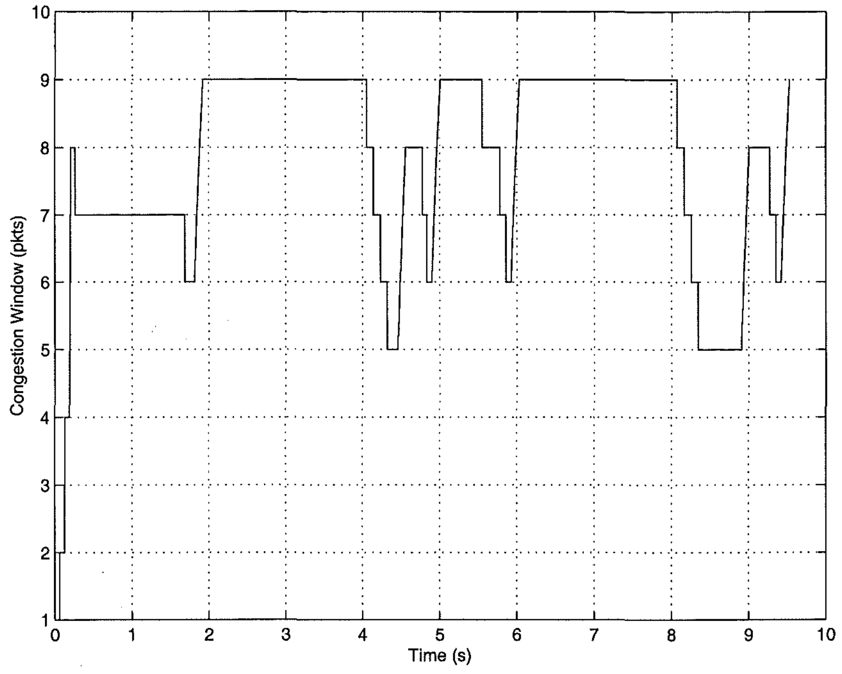


Figure 2.9 Trace of the window dynamics of TCP Vegas.

fairness and stability [Bonald 1999, Low *et al.* 2000].

2.3.3 Theoretical Modelling of TCP Window Dynamics

Modelling the window dynamics of TCP allows important features, such as throughput, fairness and stability, to be investigated. As such the performance of different versions of TCP can be compared. However, developing detailed models for these algorithms is, in some ways, more complex than modelling explicit rate control for the ABR service, since the congestion window controls the number of packets in flight, rather than specifying the packet transmission rate. In effect, the window integrates the transmission rate over the round-trip time, but further details such as the window halting until ACKs arrive and retransmissions of lost packets can complicate the model. Thus, some simplifications need to be made so that tractable models can be developed. Researchers have approached the task of modelling TCP in a number of different ways, including developing linear differential models based on the evolution of the congestion window [Altman *et al.* 1995, Schwartz 1996, Lakshman *et al.* 1997], stochastic models that incorporate packet losses as random events modulating the size of the window [Altman *et al.* 2000], and optimisation approaches using utility functions [Kelly *et al.* 1998, La and Anantharam 2000]. In this section, models for directly representing the window control dynamics are reviewed. Optimisation approaches are reviewed in Section 2.3.5.

2.3.3.1 Models for TCP Tahoe and Reno

From a modelling perspective, Tahoe and Reno are not totally dissimilar, even though TCP Reno includes the fast recovery algorithm. This is because the algorithms have identical window dynamics during the slow start and congestion avoidance phases, and only differ when lost packets have been detected. This means that modelling either version can be approached using a similar procedure, with the differences between the algorithms being included when they become necessary. The key performance parameter for TCP is the throughput that it can achieve, and throughput can be determined in the following manner. The first stage of modelling TCP is to capture the evolution of the window during slow start and congestion avoidance. Having modelled this part of the algorithm, calculating throughput involves determining the total number of packets transferred in each phase and the length of the phases, in units of time. This is demonstrated by reviewing a number of significant papers on TCP modelling.

Representing the window by the variable W, Lakshman *et al.* modelled the window dynamics as follows [Lakshman *et al.* 1997]:

$$W = \begin{cases} W+1, & \text{if } W < W_{th}; \\ W + 1/W, & \text{otherwise.} \end{cases} \tag{2.18}$$

where W_{th} represents the slow start threshold. The motivation behind these equations is obvious, after referring to Equations (2.10) and (2.11). These window adjustments occur whenever the source receives an ACK, as described previously in Section 2.3.2. Transforming Equation (2.18) into a differential equation model allows the integration the window size over time, so that the number of packets transferred in each phase can be calculated. During slow start, the window can be well-represented as having exponential growth of the form $W(t) = 2^{t/T}$, where T is the round-trip time. Also, after noting that the arrival rate of acnowlegements is equivalent to the throughput, the window dynamics during congestion avoidance can be represented by the differential equation [Lakshman *et al.* 1997]:

$$\frac{dW}{dt} = \begin{cases} \frac{1}{T}, & W \le \mu T; \\ \frac{\mu}{W}, & W \ge \mu T. \end{cases} \tag{2.19}$$

where μ is the service rate of the bottleneck queue. Integrating these window dynamics, Lakshman *et al.* obtain analytical quantities for the number n of packet transmissions in each phase and the durations t of the phases, with the average throughputs given by:

$$\begin{aligned} \bar{\lambda} &= \tfrac{n_{ss}+n_A+n_B}{t_{ss}+t_A+t_B}, && \text{TCP Tahoe,} \\ \bar{\lambda} &= \tfrac{n_A+n_B}{t_A+t_B}, && \text{TCP Reno.} \end{aligned} \tag{2.20}$$

The subscript ss refers to the slow start phase and the subscripts A and B refer to the intervals that divide the congestion avoidance phase, depending on whether the window size is smaller or greater than the transmission capacity. Lakshman *et al.* use this model to show that TCP Reno achieves higher throughput than TCP Tahoe, which is not surprising considering the modifications that have been included in the Reno algorithm.

It should be noted that the model for TCP just described models packet losses as occurring when the bottleneck queue overflows, and if multiple TCP connections need to be modelled, packets must be dropped from all connections simultaneously. In general, packet losses occur at random locations throughout a network, and at different times for different connections. This can be included in a TCP model by defining an average packet loss probability p. By assuming fixed point behaviour for the congestion window and a total number $1/p$ of packets lost before the window is halved, Lakshman *et al.* show that the initial window of the congestion avoidance phase for TCP Reno in steady-state is [Lakshman *et al.* 1997]:

$$w_0 = \sqrt{2/3p}. \tag{2.21}$$

This assumes that the window dynamics is a perfectly periodic sawtooth wave. Taking this result one step further, Mathis *et al.* show that the bandwidth thus achieved by a TCP connection is [Mathis *et al.* 1997]:

$$BW = \frac{MSS}{RTT}\sqrt{\frac{3}{2p}}, \tag{2.22}$$

where MSS is the maximum segment size generated by TCP, and RTT is the usual symbol for the round-trip time. While many assumptions have been made in reaching this result, including steady-state behaviour of the TCP source (which has an infinite amount of data to send) and a constant packet loss probability (where losses are uncorrelated), it is a very useful result which has been labelled the *inverse square-root p* law [Roughan and Erramilli 2001]. It captures two properties of the performance of TCP, specifically the inverse relationship between the throughput and the round-trip time and the inverse square root relationship with packet losses. An application of this result has been the design of TCP-friendly transmission control algorithms for real-time applications [Floyd *et al.* 2000]. These control algorithms provide smoother transmission rates for real-time applications, while still being responsive to losses so that the algorithms do not starve other TCP connections of bandwidth.

While the simplicity of Equations (2.21) and (2.22) highlight the influence of the parameters p and RTT on the performance of TCP Reno, these equations can significantly overestimate the throughput when the packet loss probability is high. This is because the model ignores relevant features of TCP Reno, including the fast retransmit mechanism and timeouts. When the fast retransmit mechanism is included, Padhye *et al.* show that the packet send rate is given by [Padhye *et al.* 2000]:

$$\begin{aligned} B(p) &= \frac{\frac{1-p}{p} + \frac{2+b}{3b} + \sqrt{\frac{8(1-p}{3bp} + \left(\frac{2+b}{3b}\right)^2}}{RTT\left(\frac{2+b}{6} + \sqrt{\frac{2b(1-p)}{3p} + \left(\frac{2+b}{6}\right)^2} + 1\right)} \\ &= \frac{1}{RTT}\sqrt{\frac{3}{2bp}} + o(1/\sqrt{p}) \end{aligned} \tag{2.23}$$

where b is the number of packets acknowledged by each ACK received by the source. If timeouts are included, the send rate is approximated by:

$$B(p) \approx \frac{1}{RTT\sqrt{\frac{2bp}{3}} + T_0 \min\left(1, 3\sqrt{\frac{3bp}{8}}p(1+32p^2\right)} \tag{2.24}$$

where T_0 is the value for the first timeout period. These results reduce to the simple inverse square-root p law if the loss probability is small and no timeouts occur.

2.3.3.2 Stochastic Model for TCP

In general, there are significant correlations between packet losses, due to the burstiness of both TCP and other background traffic using the Internet, and a truly accurate model of TCP needs to include a general loss distribution. With this aim, Altman *et al.* have considered a model of TCP in which loss events occur at instances in time T_n described by a stationary ergodic point process [Altman *et al.* 2000]. By a *loss event*, they refer to an occasion where the TCP source is forced to reduce its sending rate, which can be caused by a single packet loss or multiple losses in a particular round-trip time. Also, they simplify the analysis by modelling the transmission rate $X(t)$ of the TCP source, rather than its window size W. The transmission rate can be approximated by dividing the window size by the RTT of the connection. This leads to the relatively simple model:

$$X_{n+1} = \nu X_n + \alpha S_n \tag{2.25}$$

where ν is the multiplicative decrease factor, α is the linear increase in transmission rate between loss events, and $S_n = T_{n+1} - T_n$ is the point process modelling intervals between loss events. If the loss events have intensity λ and a correlation structure $R(k)$, Altman *et al.* show that the throughput of the TCP source is given by:

$$\begin{aligned}\bar{X} &= \lambda\alpha\left[\frac{1}{2}R(0)+\sum_{k=1}^{\infty}\nu^k R(k)\right]\\ &= \frac{1}{RTT\sqrt{bp}}\sqrt{\frac{1}{2}\hat{R}(0)+\sum_{k=1}^{\infty}\nu^k\hat{R}(k)}\end{aligned} \tag{2.26}$$

where b is the number of packets acknowledged by each ACK as before, and $\hat{R}(k) = \lambda^2 R(k)$. Equation (2.26) confirms the relationship between TCP performance and the parameters RTT and p. However, the model considers a general correlation structure for the packet losses, which means that the performance of TCP can be analysed for more general loss processes and TCP-friendly algorithms can be designed on a more precise model of TCP.

2.3.3.3 Models for TCP Vegas

As described in Section 2.3.2.3, TCP Vegas adjusts the window size based on the measured throughput. Vegas uses RTT estimates to calculate the measured throughput, and the window size is incremented or decremented, depending on the difference between the estimated throughput and the actual throughput and two thresholds α and β. These window dynamics are modelled by Bonald [Bonald 1999] using the following differential equation:

$$\frac{dW}{dt} = \begin{cases}\frac{\epsilon(t)}{RTT(t)}, & \text{during congestion avoidance}\\ -\gamma W, & \text{at packet loss,}\end{cases} \tag{2.27}$$

where $\epsilon(t) = -\frac{1}{2}\left(\operatorname{sgn}(X(t)-\alpha)+\operatorname{sgn}(X(t)-\beta)\right)$ with the operand $\operatorname{sgn}(\cdot)$ defined as:

$$\operatorname{sgn}(x) = \begin{cases}1 & x>0,\\ -1 & x<0.\end{cases} \tag{2.28}$$

The variable $X(t)$ represents the number of excess packets that need to be buffered within the network, and is equivalent to:

$$\begin{aligned}X(t) &= \mathit{Diff}(t)\times\tau\\ &= \left(\frac{W(t)}{\tau}-\frac{W(t)}{RTT(t)}\right)\\ &= W(t)-\lambda(t)\tau\end{aligned} \tag{2.29}$$

where τ is the minimum RTT (which can be approximated by the propagation delay of the connection) and $\lambda(t)$ is the current throughput.

The total throughput and buffer occupancy for all K Vegas connections through a link with capacity μ is:

$$\begin{aligned} \lambda &= \mu \\ K\alpha \leq X &\leq K\beta. \end{aligned} \tag{2.30}$$

To ensure that the windows of all the K connections stabilise to values w_k, where $\alpha \leq w_k \left(1 - \frac{\mu\tau}{\sum w_k}\right) \leq \beta$, enough buffering B needs to be provided within the network, such that $K\alpha < B$. However, if this buffering is not available in the network, TCP Vegas is not unstable—it simply behaves like TCP Reno, due to similar behaviour of the two algorithms when a packet loss is detected. Refer to Equation (2.27). Using this model, Bonald [Bonald 1999] also shows that TCP Vegas is more efficient in transferring files of finite size, maintaining a higher network utilisation than TCP Reno, while at the same time keeping the buffer occupancy low. TCP Vegas also achieves fairness between connections with heterogeneous RTTs, provided that $\alpha = \beta$.

2.3.4 Active Queueing Mechanisms in IP Networks

TCP has been designed using an end-to-end traffic control approach where knowledge of the network state is minimal. The algorithm infers information about congestion within the network when it detects a loss of a packet in the network which, in general, is due to buffer overflows in current networks. This is because most routers deployed within the Internet use simple tail drop queueing disciplines, and packets are thus discarded if they arrive when the buffer is full. The other potential reason for losing a packet is packet corruption during transmission across wireless links.

However, this does not necessarily lead to the best performance when considering traffic control in the Internet in a global sense, and it is possible for a node within networks to actively manage queues, such as dropping or marking packets, in order to manage the traffic flowing through the node. This has been labelled as *active queue management* [Floyd and Jacobsen 1993]. This allows nodes to actively signal TCP sources to adjust their transmission rates, so that connections can respond gracefully to heavy traffic loads. In contrast, tail drop queues can drop significant numbers of packets from a TCP window, which means that TCP cannot detect losses using duplicate ACKs and must rely on less efficient timeouts. This reduces the throughput of the connection, as it has to re-enter the slow start phase. With tail drop queues, there is also the potential of global synchronisation, where groups of TCP sources simultaneously reduce their window sizes at the same time. This can lead to significant oscillations in aggregrate traffic arriving at the node. Hence, there is significant benefits from deploying active

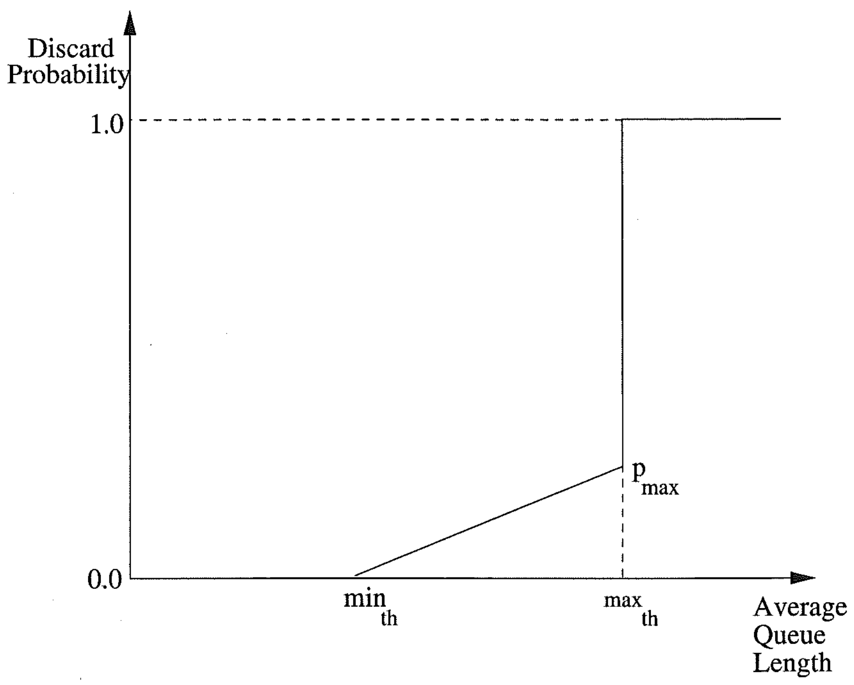


Figure 2.10 The function for determining the packet discard probability based on the average queue length.

queue management within nodes in the Internet. Random Early Detection, an active queue management mechanisms being considered by IETF, and an associated feedback scheme called Explicit Congestion Notification is discussed in this section.

2.3.4.1 Random Early Detection

Random Early Detection (RED) is one of the key mechanisms that has been proposed for active queueing in the Internet, and it was introduced by Floyd and Jacobson [Floyd and Jacobsen 1993]. It includes two components—an algorithm which calculates the average queue length, and a probabilistic queue management function which marks or discards packets based on the average queue. The average queue length is a measure of the long-term congestion at the node, and is not significantly influenced by instantaneous bursts of traffic arrivals. By using an average queue length, the stability of the system is ensured, since the node will not react unnecessarily to rapid changes which occur in its input traffic. A simple averaging algorithm is exponential weighted moving average. Once the average queue length has been calculated, the probability of marking or dropping a packet is determined using the function shown in Figure 2.10. There are two thresholds defined in RED, min_{th} and max_{th}. If the average queue is below the minimum threshold when a packet arrives, then the probability that the packet is marked or dropped is zero. When the average queue length is between the thresholds, the probability increases linearly up to a maximum probability p_{max}. Finally, if the average queue length is greater than max_{th}, arriving packets will always be marked or dropped.

There are a number of reasons for the construction of the RED mechanism. Using the average queue length as a measure of the load at the node ensures that the overall traffic load is close to the capacity of the network, since the link utilisation must be close to unity for the average queue length to be significantly greater than zero. At the same time, the probabilisitic algorithm reduces the likelihood that a batch of consecutive packets will be discarded from a single TCP flow. Random discarding means that the throughput of individual connections will be higher since a particular source is less likely to enter *Slow Start.* RED also probabilisitically distributes the congestion signals among all the TCP connections traversing the node, so that it is less likely that packets will be dropped from a significant number of connections simultaneously. This feature minimises the problem of global synchronization. Finally, RED is not biased towards bursty traffic since packets are marked/dropped probabilisitically, so that the overall probability of a packet being marked or dropped is proportional to the average packet arrival rate of a particular flow rather than the size of a burst of packets arriving simultaneously. This means that RED provides a fairer service to connections.

2.3.4.2 Explicit Congestion Notification

The issue of whether a packet is marked or dropped as a signal of congestion depends on whether the source is enabled to cooperate with the congestion signal conveyed by the congestion notification mark. When the transport protocols are unresponsive to congestion notification, dropping packets from the queue is the safe approach to controlling congestion at the node. It is also the appropriate choice for TCP versions such as Tahoe and Reno which use packet loss as signals to reduce their window size. However, it is possible to utilise marks on packets to signal the sources to adjust their transmission rates. This feedback mechanism is the basis of Explicit Congestion Notification (ECN) [Floyd 1994] being considered by the IETF.

In the Explicit Congestion Notification approach, nodes set the ECN flag in the IP header of arriving packets whenever the nodes are experiencing heavy traffic loads. The receiver copies the ECN bit from a data packet that it receives to the equivalent bit in the next ACK which is being returned to the source. The TCP source then responds to the feedback signal, if it receives an ACK with the ECN bit set. This response can be identical to TCP's normal response to a packet loss when it receives duplicate ACKs, or a more sophisticated approach can be used which filters ECN marks to determine an appropriate transmission rate.

One advantage of the ECN approach is that, because packets are marked and not dropped in the network, packets are not lost from connections which have very low throughput, like telnet applications. Losses in these instances would generally need to be detected by retransmission timeouts, which would increase the delay experienced by the user, which in turn can be very significant. In contrast, ECN schemes provide rapid and unambiguous information regarding congestion inside the network. The other advantage is that the performance of the system can be optimised, as will be shown in Section 2.3.5. The only drawback with ECN is that congestion messages can be lost if packets or ACKs with the ECN bit set are forced to be dropped inside the network. However, this issue occurs in any traffic management scheme which uses congestion messages rather than packet losses as signals to notify the sources of the transmission rate adjustments desired by the network.

2.3.5 Optimising Global TCP Performance

The initial objective in designing TCP algorithms has not been to achieve an operating point which is globally optimal, as has been noted in our introduction to active queue management. Instead, the objective of TCP is to actively utilise available bandwidth, but remain responsive to network conditions so that reliable transfer of packets can be achieved. Superficially, reaching a globally optimum operating point would require a global controller, which is not feasible due to the size of the Internet and the prohibitive amount of state information needed. Instead, individual TCP sources attempt to maximise their own performance without direct reference to the window sizes of other sources in the network. A source only modifies its transmission rate when its performance suffers. While this may appear to be a non-cooperative environment which does not achieve a global optimum, it can be shown that the system maximises an overall objective—in this case, the overall objective is the combination of individual source performances. When the individual objective functions are designed correctly, the global objective is simultaneously maximised, so traffic control within the Internet becomes a distributed optimisation problem and can be modelled as such.

This perspective of modelling the Internet has been approached using pricing mechanisms by a number of researchers, where connections respond to pricing information generated by nodes within the network [Kelly *et al.* 1998, Gibbens and Kelly 1999, La and Anantharam 2000, Athuraliya *et al.* 2000]. In this situation, TCP sources attempt to maximise their individual utility functions $U_r(x_r)$, which are functions of their current transmission rate x_r. Inside the network, each node determines the cost of handling its current traffic load and returns price information $\mu_j(t)$ back to the sources. While packet losses can be viewed as conveying this price information, active queueing mechanisms can also notify sources of the current prices using schemes like ECN, where the rate of arrival of ECN marks is proportional to the path price. Regardless of which type of congestion signal is adopted, it is clear that active queue management must be deployed at interior nodes for these approaches to operate.

Kelly *et al.* models the response of a TCP source to this price information as [Kelly *et al.* 1998]:

$$\frac{d}{dt}x_r(t) = \kappa \left(w_r - x_r(t) \sum_{j \in r} \mu_j(t) \right). \tag{2.31}$$

This captures the linear increase, multiplicative decrease behaviour of TCP connections, where the parameter w_r reflects the willingness of a TCP source to pay for resources in the network and is associated with the phase where the transmission rate is increasing linearly. The rate is decreased in proportion to the stream of congestion signals it receives, in this case the path prices μ_j. Provided that the individual utilities $U_r(x_r)$ are increasing, strictly concave functions, Kelly *et al.* has shown that the system converged to a stable point which maximises the aggregate utility function given by the Lyapunov function:

$$\mathcal{U}(x) = \sum_{r \in R} U_r(x_r) - \sum_{j \in J} C_j \left(\sum_{s:j \in S} x_s \right) \tag{2.32}$$

Furthermore, Kelly has shown that individual rates in the system are Pareto Efficient, ensuring that the system achieves fairness [Kelly 1999], in particular achieving proportional fairness. Thus, the distributed nature of TCP congestion control can optimise the overall performance of the Internet by maximising the utilities of individual sources, while providing fair distribution of network resources.

These pricing models are not limited to the linear increase, multiplicative decrease behaviour of TCP Reno, as Lo *et al.* have shown with their model of TCP Vegas [Low *et al.* 2000]. They demonstrate that TCP Vegas uses a utility function given by $U_r = \alpha d_r \log x_r$, where α is one of the parameters of TCP Vegas and d_r is the propagation delay of the TCP connection. Pricing information for Vegas can be identified as queue length at a node, normalised by the capacity of the outgoing link. Thus, Vegas can be interpreted as implementing an approximated gradient projection algorithm, and converging towards the dual optimal solution.

This interpretation of TCP Vegas allows Low *et al.* to explain the purpose of Vegas from another perspective, namely that the algorithm attempts to set its transmission rate to be proportional to the ratio of the propagation delay and the queueing delay, with the ratio lying between the thresholds α and β [Low *et al.* 2000]. Previously, Vegas has been explained as attempting to keep the number of packets buffered in the network between αd_r and βd_r. Also, the machinery of system optimisation shows that Vegas achieves proportional fairness, or even weighted proportional fairness if individual α parameters are set to be proportional to the propagation delay of the connection. Finally, Low *et al.* show that a problem with TCP Vegas, which is that the buffering required within the network increases proportionally with the number of sources, can be effectively solved by adapting the algorithm to respond specifically to ECN marks rather than RTT estimates.

2.3.6 Proportional Fairness

In conjunction with their model for rate control, Kelly *et al* also propose another definition of fairness, called proportional fairness [Kelly *et al.* 1998], which can be described as follows. Firstly, a set of elastic connections using a network can be characterised by their rate vector $x = (x_r, r \in R)$ where R is the set of possible routes in the network. A proportional fair rate allocation is then defined to be x^*, which satisfies:

$$\sum_{r \in R} \frac{x_r - x_r^*}{x_r^*} \leq 0, \tag{2.33}$$

where $x = (x_r, r \in R)$ is any other possible rate vector. This definition of fairness can be interpreted as the solution in which any change would have a non-positive overall impact on all the rates of the users, and hence is closely connected to the Nash bargaining solution in game theory. Thus, it is a definition that naturally suggests itself in the framework of optimising global performance. A less obvious feature of this definition of fairness, but a rather intuitive one none-the-less, is that a proportionally fair solution allocates rates to connections in proportion to the network resources that the connections require. If the rate of a connection, that has a path which includes many links, is increased, this would adversely affect a significant number of other connections, and it is likely that the resulting rate allocation would not be proportionally fair. Compare this situation with a connection that uses one link—increasing its rate would only affect other connections crossing that link, and hence the overall affect on all users within the network is most likely to be less significant.

While proportional fairness has been proposed in the context of maximising the utility functions of a set of connections, it is the natural definition for an linear increase, multiplicative decrease protocol like TCP. TCP seeks to find the optimum transmission rate, and determines this by inferring the affect of increasing its rate on the state of the network using either losses or ECN marks as signals of congestion. This process of quantifying the effect of increasing or decreasing a particular user's rate is also the way proportional fairness has been formulated. One can add further definitions of fairness to the collection by defining (p, α)-proportional fairness, which generalise proportional fairness and max-min fairness [Mo and Walrand 2000]. The above description of proportional fairness, however, illustrates the principles of fairness as they relate to an algorithm like TCP.

2.4 CONTRIBUTIONS OF THIS THESIS

This chapter has thoroughly reviewed the concepts of reactive congestion control for the management of network traffic. Review has included examples of the use of reactive control in modern networks, with outlines of the algorithms used by the ABR service in ATM networks and TCP in IP networks. Analytical models used to investigate the performance of these algorithms have been introduced, and the associated control issues been discussed, laying down the foundation research work for results provided in the rest of this thesis.

A concept which has not been addressed within the design of rate control algorithms is the incorporation of the self-similar nature of network traffic in calculating explicit rate information that can be fed back to controlled sources. Whether in the context of the ABR service in ATM networks or TCP in IP networks, traffic originating from these sources are competing for network resources with higher priority traffic which is uncontrolled. There is significant evidence that network traffic is self-similar [Beran *et al.* 1995, Crovella and Bestavros 1997, Leland *et al.* 1994], and to ensure that the service to this high-priority, background traffic is not degraded, the performance of congestion control in the context of self-similarity needs to be investigated. While previous studies have included the stochastic nature of the background traffic within studies of congestion control algorithms in some capacity [Zhao *et al.* 1997, Lakshman *et al.* 1997, Altman *et al.* 2000], the only significant study of self-similarity in relation to congestion control is the work by Tuan and Park [Tuan and Park 1999], which is reviewed at the end of Chapter 3. This thesis describes, in detail, a rate-based congestion control algorithm which addresses the issue of self-similarity. The traffic control algorithm is presented in the context of the ABR service, and fundamental aspects of this control approach are analysed, including the sensitivity of the algorithm to estimates of traffic parameters and robust control solutions are developed.

Chapter 3

SELF-SIMILARITY OF NETWORK TRAFFIC

The use of mathematical models to describe network traffic allows engineers to understand some of the fundamental properties underlying the traffic and also to analyse the performance of proposed methods of supporting traffic within the network. These analytical results allow engineers to develop solutions which specifically take into account the unique characteristics of the traffic, and so optimise the operation of the network. Such models form the analytical branch of network modelling, which is balanced by the more practical contributions that simulated models provide.

Modelling network environments using either analytical or simulation techniques forms an important initial step in network development. The network layout and management algorithms can be explored in an economic way, before capital is invested in developing testbeds, or the actual system is constructed. The accuracy of mathematical models in capturing important aspects of network traffic is critical to ensure that the subsequent network performance is neither overestimated, resulting in significantly greater network delays and losses; nor underestimated, in which case conservative management policies will be implemented and the network will be poorly utilised.

In the first half of the twentieth century, the type of network which teletraffic engineers were creating and managing was the telephone system, where arrivals of telephone calls could be modelled as a Poisson process and the duration of calls were assumed to be exponentially distributed. Hence the standard M/M/1 queue which is used to introduce networking concepts [Schwartz 1987]. In packet networks, the basic model example is the ON/OFF source, which spends exponentially distributed periods in both ON and OFF states. From these fundamental mathematical models, more complex models can be developed, such as the interrupted Poisson process and the Markov modulated Poisson process [Adas 1997].

Models based on the Poisson process can be used to represent particular events within the network, such as telephone calls or arrivals of individual packets. However, as the speed of networks and the potential number of events occuring increases drastically, fluid models are used to capture the key features of the traffic without modelling specific events [Adas 1997]. Fluid models consider network traffic as a dynamically varying stream of data, measured in bits/sec for example, where key events are changes in the rate of traffic. Fluid models include the single ON/OFF source model, where the source generates a constant flow during its ON period and the Markov modulated fluid models, where states within the Markov chain represent different flow rates.

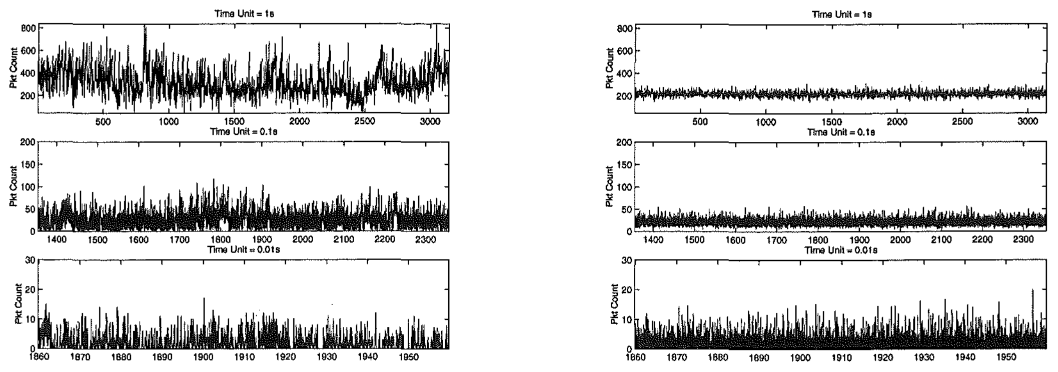
An important characteristic of the models which have been introduced so far in this chapter is that they are Markovian, in that the future state of the traffic is only dependent on the current state and that they are short-range dependent (SRD), where the correlations within the traffic decay rapidly in time (at least exponentially fast). The structure of the short-range dependence can be modified using regression models, which use finite linear combinations of previous states and an uncorrelated iid noise process producing a fluid-type model. However, these regression models are still short-range dependent.

There is significant statistical evidence that a wide class of network traffic is self-similar and has an associated property of long-range dependence. Self-similarity refers to the invariance of the distribution of the traffic as the traffic is scaled in time. Examples of self-similar behaviour has been observed in a number of different fields, including geophysics, hydrology, turbulence, economics, communications, and $1/f$ type noises. Self-similar models are also applicable in such areas as renormalization group theory and critical phenomena in theoretical physics [Samorodnitsky and Taqqu 1994]. Long-range dependence refers to a non-degenerate correlation structure at large lag within self-similar traffic.

In this chapter, the concepts of self-similarity and long-range dependence are introduced, together with models which are used for processes which exhibit these characteristics. These models will be used later in this thesis for developing congestion control mechanisms that take the self-similarity of network traffic into account. Methods of estimating key parameters within self-similar traffic are discussed, and the theoretical basis for predicting long-range dependent processes is described. These predictors will form the core of the congestion control algorithms which are developed in this research. The studies analysing the self-similarity of network traffic are reviewed, and then reasons are outlined for including self-similarity within network management algorithms. The rationale being that the flow of self-similar traffic through a network is fundamentally different from Poissonian traffic.

3.1 INTRODUCTION TO SELF-SIMILARITY

The term "self-similar" is used to describe a class of processes which appear "similar" as the processes are scaled in time. This is in contrast to Poisson processes which lose their burstiness and flatten out when scaled appropriately. The burstiness of self-similar traffic, compared with a Poisson process, is shown graphically in Figure 3.1, where the graphs zoom in on the number of packet counts from Ethernet traffic at smaller time



(a) Ethernet Traffic (b) Poisson Data

Figure 3.1 Comparing the scaling behaviour of (a) self-similar Ethernet and (b) Poisson data.

scales, moving from time units of $1s$ to $0.1s$ and finally $0.01s$. Moving down the plots, a particular interval is chosen from the previous plot, revealing a more detailed packet count process within the smaller time unit. The burstiness of the self-similar data remains evident for several orders of magnitude. In contrast, Poisson traffic flattens out at larger time scales.

3.1.1 Defining Self-similarity

Mathematically, invariance in the distribution, characteristic of self-similar processes, is defined as follows [Samorodnitsky and Taqqu 1994]. The process $X(t)$ is self-similar with self-similarity parameter H if

$$X(\alpha t) \overset{\mathrm{d}}{=} \alpha^{H} X(t), \tag{3.1}$$

where $\alpha > 0$ and the equivalence in distribution is symbolized by $\overset{\mathrm{d}}{=}$. Self-similarity manifests itself within data in three fundamental ways—slowly decaying variances as the data is aggregated, long-range dependence within the covariance structure, and a spectral density which obeys a power-law with divergence near the origin.

3.1.2 Hurst Parameter

The Hurst parameter captures the self-similar intensity of a process. For a process to be realisable, the Hurst parameter of the process must lie in the interval $H \in [0, 1.0)$, since the absolute autocovariance function must be strictly decreasing. Processes can be distinguished as long-range dependent (LRD) or short-range dependent (SRD) depending on whether the Hurst paramater is greater than 0.5 or not. The difference between these types of processes is the asymptotic behaviour of the autocovariance function, as

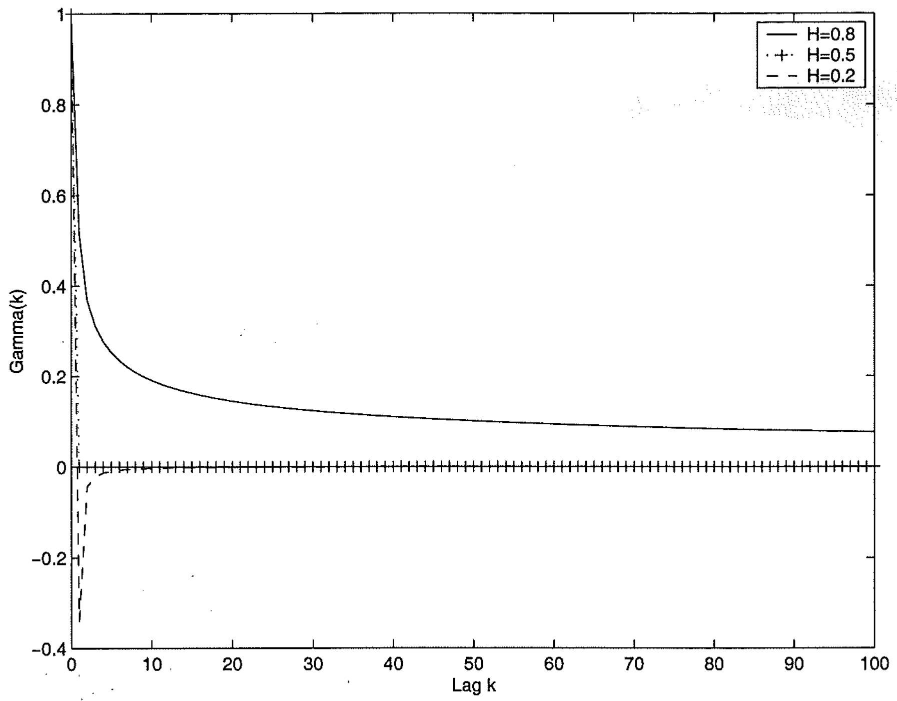


Figure 3.2 Autocovariance function for various values of the Hurst parameter.

shown in Figure 3.2. It should be noted that the autocovariance functions shown in Figure 3.2 are for *exactly second-order self-similar* processes [Park and Willinger 2000]. More general asymptotically second-order self-similar processes can be defined, which is a weaker condition, where a process must be time-aggregated with a sufficiently large block size before the asymptotic form of the autocovariance function commences.

3.1.3 Slowly Decaying Variances

In many engineering problems, information regarding a particular process is obtained as a set of discrete-time measurements. Self-similarity can be defined in this context by considering the behaviour of the autocorrelation function of the process as it is aggregated in time [Leland *et al.* 1994]. If a covariance stationary stochastic process $X = (X_t : t = 0, 1, 2, \dots)$, with mean μ, variance σ^2 and autocorrelation $\gamma(k)$ at lag k, is exactly self-similar with self-similarity parameter H, then the aggregated process $X_t^{(m)}$, which is defined as

$$X_t^{(m)} = \frac{1}{m}(X_{tm-m+1} + \cdots + X_{tm}),\ t \geq 1, \tag{3.2}$$

using block size m, satisfies

$$\operatorname{var}\left(X^{(m)}\right) = \sigma^2 m^{2H-2}, \tag{3.3}$$

where $m > 1, m \in I$. In contrast, the variance of a short range dependent (SRD) process would decrease as m^{-1} with increasing block size. For an exactly self-similar process, the autocorrelation function does not change with block size:

$$\gamma^{(m)}(k) = \gamma(k) \tag{3.4}$$

Alternatively, a process can be asymptotically self-similar, in which case the autocorrelation function behaves such that $\gamma^{(m)}(k) \to \gamma(k)$ as $m \to \infty$. Equation (3.4) captures the essence of the invariance of the stochastic structure as the process is scaled, in the same manner as the definition in (3.1).

3.1.4 Long-range Dependence and Divergence of Spectral Density

When the Hurst parameter, the index of self-similarity, lies in the interval $1/2 < H < 1$, the process displays long-range dependence. The correlations within LRD traffic decay hyperbolically, namely:

$$\gamma(k) \sim C_\gamma k^{2H-2},\ k \to \infty, \tag{3.5}$$

as compared with SRD traffic, which have correlations that are either zero for lag $k > 1$, or rapidly approach zero, as for an exponentially decaying autocovariance function. Due to the significant correlations at asymptotically large lag, the autocovariance function is non-summable:

$$\sum_k \gamma(k) \to \infty. \tag{3.6}$$

In the frequency domain, the autocorrelation function transforms into the spectral density, and long-range dependence manifests itself as divergence of the spectral density at the origin, with a power-law of the form:

$$f(\lambda) \sim C_f \lambda^{1-2H},\ \lambda \to 0, \tag{3.7}$$

where $f(\lambda)$ is the spectral density component at frequency λ, and C_f is some positive constant that depends on the specific process.

The above definition of self-similarity introduces the nature of the traffic that is being considered by this research work. It also identifies key properties by which self-similarity can be recognized, and trends within self-similar processes which can be used to estimate important parameters, such as the Hurst parameter. Properties, like the significant correlations that occur in long-range dependence, can be utilised within network management algorithms, that incorporate more accurate predictions of long-range dependent traffic. Finally, the scaling behaviour of self-similar processes can be coupled with the scaling which occurs in wavelet theory which means that wavelets can be used to analyse self-similar network traffic. In addition, wavelets can be used to generate synthetic self-similar traces.

3.2 SELF-SIMILAR TRAFFIC MODELS

Fractional Brownian motion and fractional Gaussian noise are standard models for self-similar phenomena, in that they parsimonously capture the long-range dependence that occurs in the process using the Hurst parameter, the mean and variance of the data. As a result, a number of properties of self-similar traffic have been demonstrated using these models [Norros 1994, Gripenberg and Norros 1996, Willinger *et al.* 1997]. Other models include fractional ARIMA processes, which can adjust the short-range dependence in traffic models. Finally, self-similarity can be derived using ON/OFF sources with heavy-tailed distributions for ON (and possibly OFF) periods.

3.2.1 Fractional Brownian Motion

The canonical example of a self-similar process is fractional Brownian motion (fBm), which has a Hurst parameter in the range $0 < H < 1$. The fBm process $B_H(t)$ was first discovered by Kolmogorov. It was given much wider exposure in Benoit Mandelbrot and John van Ness's pioneering paper [Mandelbrot and Van Ness 1968], where it was proposed as a model for strong dependence observed in data from economics and hydrology. The well-known Brownian motion is a special case of fractional Brownian motion, with $H = \frac{1}{2}$ and increments that are independent. In contrast, fractional Brownian motion $B_H(t)$ has a covariance structure given by [Samorodnitsky and Taqqu 1994]:

$$\mathrm{Cov}(B_H(t_1), B_H(t_2)) = \frac{1}{2}\left\{|t_1|^{2H} + |t_2|^{2H} - |t_1 - t_2|^{2H}\right\}\mathrm{Var}(B_H(1)) \tag{3.8}$$

When $\frac{1}{2} < H < 1$, fBm becomes long-range dependent.

Fractional Brownian motion $B_H(t)$ can also be defined by a moving average representation of the increments of Brownian motion $B(t)$, using the weighting kernel $(t-s)^{H-\frac{1}{2}}$. By representing these increments as dB, fractional Brownian motion can be defined as

$$\begin{aligned} B_H(0) &= B_0, \\ B_H(t) - B_H(0) &= \frac{1}{\Gamma(H+\frac{1}{2})}\left\{\int_0^t (t-s)^{H-\frac{1}{2}} dB(s) + \int_{-\infty}^0 \left[(t-s)^{H-\frac{1}{2}} - (-s)^{H-\frac{1}{2}}\right] dB(s)\right\}. \end{aligned} \tag{3.9}$$

The weighting kernel demonstrates the dependence of an increment in fBm on all previous increments. As such, fBm in Equation 3.9 can be shown to have the covariance structure of Equation (3.8), and hence the property of self-similarity.

A useful process is normalized fBM, which can be used to construct other self-similar processes with different mean and variance parameters. The properties of normalized fBm, symbolised by Z_t, can be delineated as follows [Norros 1994]:

1. Z_t has stationary increments;
2. $Z_0 = 0$, and $\mathbf{E}[Z_t] = 0$ for all t;
3. $\mathbf{E}\left[Z_t^2\right] = |t|^{2H}$ for all t;
4. Z_t has continuous paths;
5. Z_t is Gaussian.

Norros then proceeded to develop a fractional Brownian traffic model A_t, based on normalized fBm, which represents the cumulative amount of work arriving at a point in the network, ie.

$$A_t = mt + \sqrt{am}Z_t, \tag{3.10}$$

where the parameter m is the mean input traffic rate and a is the variance coefficient. This traffic model has been used to analyse queueing performance with self-similar traffic by Norros [Norros 1994], who found that the buffer overflow probabilities are heavy-tailed.

3.2.2 Fractional Gaussian Noise

The incremental process of fractional Brownian motion is a stationary sequence, which is termed fractional Gaussian noise (fGn)—analogous to white Gaussian noise which is the incremental process derived from Brownian motion [Samorodnitsky and Taqqu 1994]. Fractional Gaussian noise is defined as

$$Y_k = B_H(k+1) - B_H(k),\ k \in I, \tag{3.11}$$

and its properties of stationarity, zero mean and variance $\mathbf{E}\left[Y_j^2\right] = \mathbf{E}\left[B_H^2(1)\right] = \sigma_0^2$ are derived from fBm. The autocovariance function of fGn is

$$\gamma(k) = \frac{\sigma_0^2}{2}\left(|k+1|^{2H} - 2|k|^{2H} + |k-1|^{2H}\right), \tag{3.12}$$

using the covariance structure of fBm. Transforming this covariance structure into the frequency domain gives the spectral density

$$f(\lambda) = C_f \sigma_0^2 |e^{i\lambda} - 1|^2 \sum_{k=-\infty}^{\infty} \frac{1}{|\lambda + 2\pi k|^{2H+1}}, \tag{3.13}$$

where C_f is given by

$$C_f = \left(\frac{H\Lambda(2H) \sin H\pi}{\pi} \right)^{1/2}. \tag{3.14}$$

When the parameter $H = \frac{1}{2}$, the sequence reduces to white Gaussian noise. However, in the interval $\frac{1}{2} < H < 1$ the fractional Gaussian noise has the property of long-range dependence, which can be seen by the asymptotic behaviour of the autocovariance function as $k \to \infty$:

$$\gamma(k) \sim \sigma_0^2 H(2H-1)k^{2H-2}, \tag{3.15}$$

and the spectral density as $\lambda \to 0$:

$$f(\lambda) \sim \sigma_0^2 C_f^{-2} \lambda^{1-2H}. \tag{3.16}$$

This power-law behaviour results in a non-summable autocovariance function and a spectral density that diverges at the origin, which are the characteristics of long-range dependence as shown in Section 3.1.

3.2.3 Fractional ARIMA Processes

As such, fractional Brownian motion and fractional Gaussian noise model exactly self-similar processes, but the short-range correlations of a data set may, however, differ from the correlation structure of these models. Unfortunately, such models do not have the flexibility to incorporate these variations in short-range correlations. The need to model both LRD and SRD within the correlation structure was recognized by Hosking [Hosking 1981], resulting in his development of the family of models called fractional ARIMA processes. These processes are natural extensions of ARIMA(p, d, q) models, where the differencing parameter d for fractional ARIMA processes can assume real values rather than just being restricted to the set of positive integers.

The ARIMA(p, d, q) model [Hosking 1981, Box *et al.* 1994] is mathematically described by

$$\Phi(B)\Delta^d X_n = \Theta(B)\epsilon_n, \tag{3.17}$$

where X_n is the fractional differenced random variable with zero mean. The innovations ϵ_n are i.i.d. normal random variables with zero mean and variance σ_ϵ^2. The polynomials Φ and Θ are of order p and q respectively

$$\begin{aligned}\Phi(B) &= 1 - \phi_1 B - \cdots - \phi_p B^p,\\ \Theta(B) &= 1 + \theta_1 B + \cdots + \theta_q B^q,\end{aligned} \tag{3.18}$$

in the backward-shift operator B, which is defined as $B(X_n) = X_{n-1}$. These polynomials define the short-range dependent structure of the variable X_n. The fractional differencing component $\Delta^d = (1-B)^d$ introduces the long-range dependent structure, and can be expanded as follows:

$$\begin{aligned}\Delta^d &= (1-B)^d\\ &= \sum_{j=0}^{\infty} \frac{\Gamma(j+d)}{\Gamma(d)\Gamma(j+1)}.\end{aligned} \tag{3.19}$$

When $0 < d < \frac{1}{2}$, the process is long-range dependent with Hurst parameter $H = d + \frac{1}{2}$. This can be shown by considering the spectral density of X_n which is

$$\begin{aligned}f(\lambda) &= \frac{\sigma_\epsilon^2}{2\pi}\frac{|\Theta(e^{i\lambda})|^2}{|\Phi(e^{i\lambda})|^2}|1 - e^{i\lambda}|^{-2d},\\ &\sim \frac{\sigma_\epsilon^2}{2\pi}|\lambda|^{-2d}, \text{ as } \lambda \to 0.\end{aligned} \tag{3.20}$$

Once again in Equation (3.20), as in Equation (3.7), the power-law at the origin is observed. Comparing (3.20) with (3.7), the relationship $d = H - \frac{1}{2}$ is derived. By transforming the spectral density into the time domain, the asymptotic behaviour of the covariance of a fractional ARIMA process is found to be

$$\gamma(k) \sim \frac{\sigma_\epsilon^2}{\pi}\frac{|\Theta(1)|^2}{|\Phi(1)|^2}\Gamma(1-2d)\sin\pi d|k|^{2d-1}, \text{ as } k \to \infty. \tag{3.21}$$

Equation (3.21) shows that the correlation function is non-summable.

Explicit formulae for the covariance function of general fractional ARIMA processes are complicated [Hosking 1981], so the simplest cases are considered here. The fractional ARIMA$(0, d, 0)$ process $\Delta^d x_k = \epsilon_k$, which functions like fractional noise, has the autocovariance function

$$\gamma_x(k) = \sigma_\epsilon^2 \frac{(-1)^k \Gamma(1-2d)}{\Gamma(k-d+1)\Gamma(1-k-d)} \tag{3.22}$$

and spectral density

$$\begin{aligned} f_x(\lambda) &= \left(2\sin\frac{\lambda}{2}\right)^{-2d} \\ &\sim \lambda^{-2d}, \text{ as } \lambda \to \infty. \end{aligned} \tag{3.23}$$

The autocovariance function of the fractional ARIMA$(1,d,0)$ process $(1-\phi B)\Delta^d y_k = \epsilon_k$ with autoregressive parameter ϕ is

$$\gamma_y(k) = \gamma_x(k)\frac{F(1,d+k;1-d+k;\phi)+F(1,d-k;1-d-k;\phi)-1}{1-\phi^2}, \tag{3.24}$$

and the spectral density is

$$\begin{aligned} f_y(\lambda) &= \frac{\left(2\sin\frac{\lambda}{2}\right)^{-2d}}{1+\phi^2-2\phi\cos\lambda} \\ &\sim \frac{\lambda^{-2d}}{(1-\phi)^2}, \text{ as } \lambda \to \infty. \end{aligned} \tag{3.25}$$

Finally, the fractional ARIMA$(0,d,1)$ process $\Delta^d z_k = (1-\theta B)\epsilon_k$ with moving average parameter θ has an autocovariance function given by

$$\gamma_z(k) = \gamma_x(k)\frac{(1-\theta)^2k^2-(1-d)[(1-d)(1+\theta^2)-2\theta d]}{k^2-(1-d)^2}, \tag{3.26}$$

and a spectral density

$$\begin{aligned} f_z(\lambda) &= (1+\theta^2-2\theta\cos\lambda)\left(2\sin\frac{\lambda}{2}\right)^{-2d} \\ &\sim (1-\theta)^2\lambda^{-2d}, \text{ as } \lambda \to \infty. \end{aligned} \tag{3.27}$$

3.2.4 Aggregation of High Variability ON/OFF Sources

While fractional Brownian motion, fractional Gaussian noise and fractional ARIMA processes can capture the long-range dependence, they do not provide any physical explanation for LRD behaviour, in that they model self-similarity using a fluid-type approach and do not account for individual packet arrivals or duration of connections. Self-similarity of network traffic can, however, be arrived at by considering the aggregation of a large number of ON/OFF sources [Willinger *et al.* 1997, Taqqu *et al.* 1997], where the distribution of the either the ON- or OFF-period, or both, is heavy-tailed (has infinite variance). A distribution that is commonly used to model heavy-tailed behavour is the Pareto distribution [Park and Willinger 2000] with parameter α. When $1 < \alpha < 2$, the distribution has finite mean, but infinite variance, while it has infinite mean and variance when $\alpha < 1$.

Willinger *et al* [Willinger *et al.* 1997] consider the cumulative packet count from a large number of ON/OFF sources which have heavy tailed distributions for ON (and possibly OFF) periods. These sources are represented by the reward sequence $W^{(m)}(t)$, where a reward of 1 occurs during an ON period and an OFF period is represented by reward 0. Denoting the cumulative packet count from M sources as $W^*_M(Tt)$, Willinger *et al.* show that:

$$\begin{aligned} W^*_M(Tt) &= \int_0^{Tt} \left(\sum_{m=1}^{M} W^{(m)}(u) \right) du \\ &\to TM\frac{\mu_1}{\mu_1+\mu_2}t + T^H\sqrt{L(T)M}\sigma_{\lim}B_H(t), \text{as } M \to \infty,\ T \to \infty, \end{aligned} \tag{3.28}$$

where μ_1 and μ_2 are the mean ON and OFF period respectively. The function $L(t)$ is a slowly varying function at infinity that depends on the distributions of the ON/OFF periods. The parameter $\sigma_{\lim}$ is specified in Willinger *et al.*'s paper [Willinger *et al.* 1997], for the interested reader. The limiting process $W^*(t)$ has the form of the fractional Brownian traffic model proposed by Norros [Norros 1994]. The Hurst parameter for the fractional Brownian traffic is related to the parameter α, which defines the weight of the heavy-tail in the distribution for the ON/OFF periods, by the relationship $H = (3-\alpha)/2$.

This is an important result for relating properties of individual connections with the behaviour of the aggregate traffic entering a node in a backbone network. If the packet lengths or connection durations, which constitute the traffic, are characterized by a heavy-tailed distribution, this result shows that the traffic itself will be self-similar in nature. The model is also applicable to the aggregation of TCP connections, even though there is significant nonlinearity and dependence within this traffic. This is because TCP stretches out the transfer of files into a packet train, as clocked by arrival of ACKs at the source, so that packet losses are minimised. Hence, TCP preserves the distribution of file sizes [Park *et al.* 1996]. It also provides a theoretical basis for generating synthetic data to represent self-similar traffic using ON/OFF sources that are heavy-tailed.

3.3 ESTIMATION OF LONG-RANGE DEPENDENT TRAFFIC PARAMETERS

3.3.1 Estimation of the Hurst Parameter

As demonstrated in the previous section, the Hurst parameter H plays a key role in characterising the self-similarity of a process. Thus, the estimation of this parameter from a set of measurements is crucial in determining whether a process is self-similar, and estimation methods have attracted a significant amount of research. Methods of estimating the Hurst parameter are based on quantifying one of the three fundamental behaviours of self-similar processes—slowly decaying variances, LRD in the covariance structure and divergence of the spectal density—with estimation proceeding by applying an appropriate transformation to the data and fitting a slope to the resulting graph.

3.3.1.1 Variance-time Analysis

A conceptually simple approach is to use variance-time analysis [Beran 1994]. This approach uses a property of LRD processes, in which the variance of the arithmetic mean decreases more slowly than the reciprocal of the sample size. Thus, if the averages $X^{(m)}$ of non-overlapping blocks of data of size m are obtained from the process X, such that, $X_k^{(m)} = \frac{1}{m}(X_{km-m+1} + \ldots + X_{km})$, then the variance of $X_k^{(m)}$ follows the relationship:

$$Var(X^{(m)}) \sim cm^{2H-2}, \text{ as } m \to \infty \tag{3.29}$$

with $0.5 < H < 1.0$ for LRD processes. For SRD process, however, $H = 0.5$ and this can be used to distinguish between LRD and SRD. The sample variance $S^2(m)$ of the means of non-overlapping data blocks can be plotted against block size on a log-log graph. The asymptotic slope of the graph can be used to determine the value of H.

The variance-time plot for the data set of aggregating ten randomly shifted sequences from the *Star Wars* VBR data is shown in Figure 3.3. The *Star Wars* data has been analysed previously [Garrett and Willinger 1994], and it has been shown to be LRD. As such, it is commonly used as a representative LRD data set. Figure 3.3 clearly shows that sample variance decreases much more slowly than would be expected for SRD data, where the data would lie around the slope that gives $\hat{H} = 0.5$. From this plot, the estimate of the Hurst parameter is 0.84.

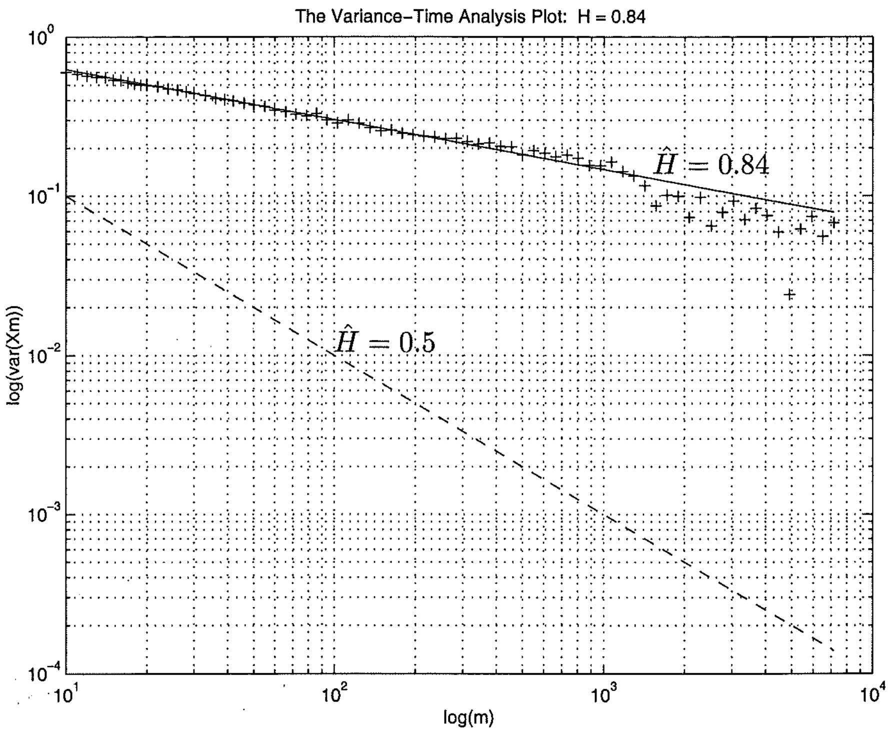


Figure 3.3 The variance-time plot for the aggregated *Star Wars* data set. The solid line represents the line fitted to the data, which gives $\hat{H} = 0.84$ as the estimate for the Hurst parameter.

3.3.1.2 R/S Statistic Analysis

The second well-known heuristic technique for estimating the Hurst parameter is R/S statistic analysis. LRD was originally observed by Hurst using the *rescaled adjusted range* statistic (or *R/S statistic*) for annual variation of the Nile river flows. The R/S statistic is calculated using the sample mean $\bar{X}(n)$ and sample variance $S^2(n)$ from observations $X_k : k = 1, 2, \ldots, n$:

$$R(n)/S(n) = \frac{1}{S(n)} \{\max(0, W_1, W_2, \ldots, W_n) - \min(0, W_1, W_2, \ldots, W_n)\} \qquad (3.30)$$

where $W(k) = (X_1 + X_2 + \cdots + X_k) - k\bar{X}(n)$, $1 \leq k \leq n$. Hurst observed in his research that the annual flow rates exhibited an effect described by the relationship:

$$\mathbf{E}[R(n)/S(n)] \sim cn^H, \text{ as } n \to \infty \qquad (3.31)$$

with the parameter H typically about 0.7. This contrasted the expected relationship $\mathbf{E}[R(n)/S(n)] \sim cn^{0.5}$ which occurs for data that has SRD. This effect has become known as the Hurst effect. R/S statistic analysis has become a standard technique for estimating the Hurst parameter.

For the aggregated *Star Wars* data set, a log-log plot of the R/S statistic versus

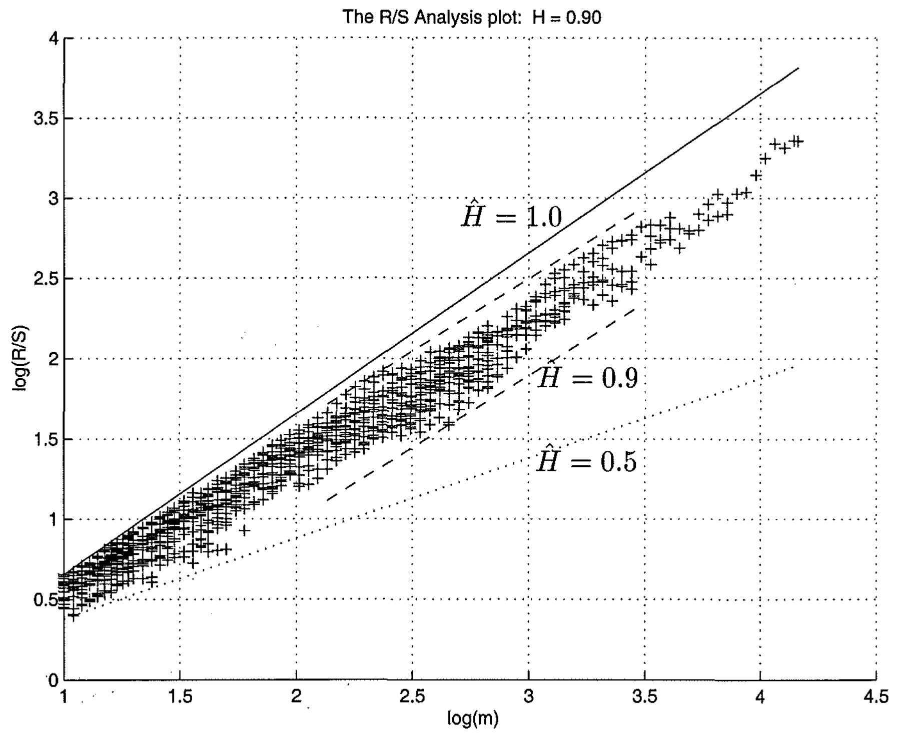


Figure 3.4 R/S statistic analysis for the aggregated *Star Wars* data set. Lines with slope 0.5 and 1.0 are shown as the dotted and solid lines, while dashed lines bound the subset of the R/S statistics used to obtain the estimate of the Hurst parameter, which is $\hat{H} = 0.90$.

the block size m is shown in Figure 3.4. Multiple R/S statistics can be calculated for non-overlapping blocks when the block size is small. The number of statistics that can be estimated from a finite length data set, however, decreases as the block size increases. Figure 3.4 also includes lines of slope 0.5 and 1.0, which are the bounds of the region for which the Hurst parameter represents LRD. It can be seen from the plot, that the slope of the data is significantly greater than 0.5, thus indicating the presence of self-similarity. The Hurst parameter is estimated using a subset of the R/S statistics. The statistics for small block sizes d are not included because the Hurst effect is an asymptotic behaviour, and the R/S statistics for small block sizes do not reflect the long-term correlations within the data. The R/S statistics for large block sizes d are also not included, because there are only a few samples of the statistic in this region, and, if they are included in estimating the slope, will result in a biased estimate for H. From the R/S statistic estimation method, the estimate of the Hurst parameter is 0.90.

Unfortunately, both the variance-time and R/S statistic analysis are heuristic methods for estimating the Hurst parameter, which do not rigorously prove self-similarity. Proof requires confidence intervals to be calculated for H. The estimated value of the Hurst parameter is also dependent on the section of the resulting graphs that are used to estimate slopes. This feature is unsatisfactory because the estimate is dependent on visually inspecting the graph for an appropriate section. It would be better if this process of selecting the section of the graph could be automated.

More refined methods of data analysis are based on maximum likelihood estimators (MLEs) and the use of periodograms which effectively transform the data into the frequency domain. The well-known Whittle's estimator [Beran 1994] is an approximate MLE which is asymptotically normal and efficient. Problems with robustness do occur because of deviations from the assumed model spectrum and also from deviations from Gaussian distribution.

3.3.1.3 Abry-Veitch Estimator

The Abry-Veitch (AV) estimator is a fast estimation technique based on the wavelet transform [Veitch and Abry 1999]. Its speed is due to the fast pyramidal algorithm that is used to implement the wavelet transform—an algorithm of order $O(n)$, as compared to the Fast Fourier Transform which is of order $O(n\log(n))$. The AV estimator is an unbiased estimator of the Hurst parameter with close to minimum variance. The wavelet transform generates a set of detail coefficients $d_{\mathbf{X}}(j,k)$ from a data set, and for an LRD process $\mathbf{X}$, the variance of the detail coefficients at each level j is given by the relationship [Veitch and Abry 1999]:

$$\mathbf{E}[d_{\mathbf{X}}(j,\cdot)^2] = C \cdot 2^{j(2H-1)} \tag{3.32}$$

with $C > 0$. The Hurst parameter can thus be estimated from the graph of the logarithm of variance, $\log \mathbf{E}[d_{\mathbf{X}}(j,\cdot)^2]$, versus the resolution level j. Clearly, the variance must be estimated from the detail coefficients generated from a finite data set, and since the expected value of the detail coefficients is zero for each level, the time average

$$\mu_j = \frac{1}{n_j}\sum_{k=1}^{n_j} d_{\mathbf{X}}^2(j,k), \tag{3.33}$$

can be used. This estimation technique, using the Daubechies wavelet D3, is shown in Figure 3.5 for the aggregated *Star Wars* data set, giving the estimated value of the Hurst parameter to be 0.837 with the 95% confidence interval for this estimate being $(0.775, 0.898)$.

An important property of the AV estimator is that it can be re-formulated to generate on-line estimates of the Hurst parameter [Roughan *et al.* 1998]. This features stems from the fact that the wavelet transform is performed using the fast pyramidal filter, which was originally intended for on-line filtering applications. The current traffic level measurement is the input to the filter, and the following sum and counter are updated for each level j when the particular filter level has an output (which varies, depending

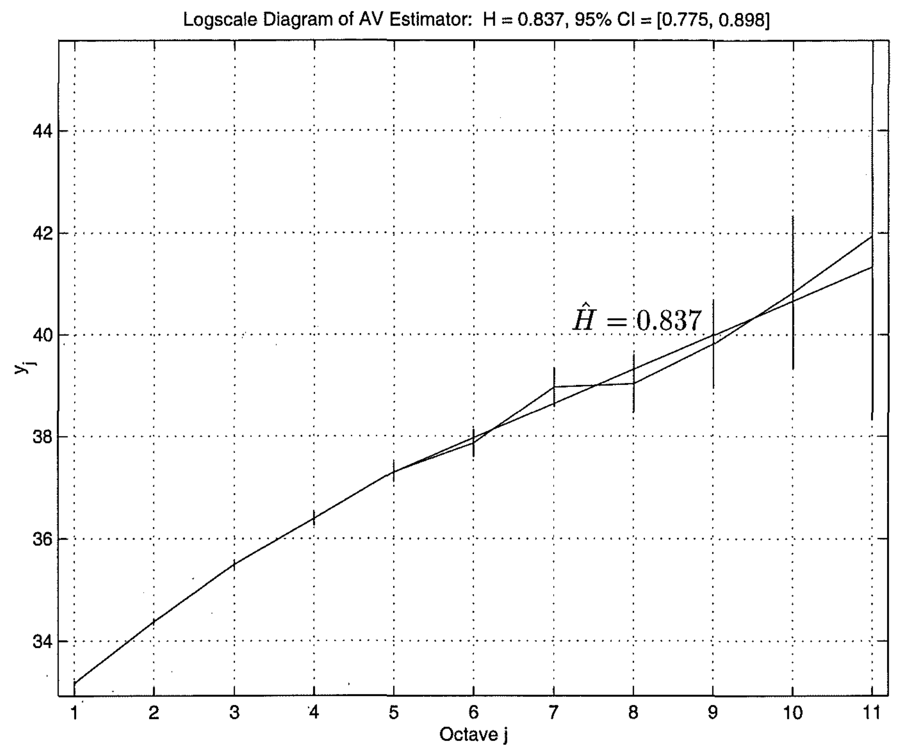


Figure 3.5 The graph of $\log \mu_j$ versus level j. The vertical bars represent the 95% confidence intervals for the estimates of the variance at each level.

on the resolution of the level):

$$\begin{cases} S_j = S_j + (d_{\mathrm{x}}(j,k))^2 \\ n_j = n_j + 1. \end{cases} \tag{3.34}$$

When an estimate of the Hurst parameter is required, the time averages for the variance of the details μ_j are calculated using

$$\mu_j = \frac{S_j}{n_j} \tag{3.35}$$

and the Hurst parameter is estimated by performing a weighted linear regression of the variables μ_j versus level j. These on-line estimates can be used within adaptive rate control algorithm, such as the ABR control mechanism proposed by the author [Östring *et al.* 2000a].

3.3.2 Estimation of the Traffic Mean

Estimation of the mean from sampled data is a fairly standard statistical problem, with the sample mean being the obvious choice of estimate. However, LRD means that the usual assumption that data is independent, or that the correlations are SRD (which allows us to aggregate data to remove the correlations), is no longer valid. The sample mean can still be used as an estimate of the mean, but the effect of these correlations is to significantly widen the confidence intervals within which the true mean lies, thus increasing the uncertainty associated with the estimate. One approach is to use whitening filters that can strip fractional processes of long-range correlations which occur within the data. The output of the filters is then used to detect the mean signal [Barton and Poor 1988]. The best linear unbiased estimator (BLUE) method also uses the correlation structure to determine the mean from a data set. However, the sample mean performs well compared to the more sophisticated techniques of whitening filters and BLUE, as will be shown in this section.

Considering a linear estimation approach, the mean μ_b^* can be estimated using a weighted sum of sampled traffic data [Beran 1994],

$$\hat{\mu}_b = w'\mathbf{R}_b \tag{3.36}$$

where $w = (w_1, w_2, \dots, w_n)'$ is the vector of weights and $\mathbf{R}_b = (R_b(k), R_b(k-1), \dots, R_b(k-n+1))'$ is the vector of stored traffic measurements. The weights w must sum to unity $w'\mathbf{1} = 1$, with the vector notation $\mathbf{1} = (1, 1, \dots, 1)'$. The variance of $\hat{\mu}_b$ is given by

$$var\,(\hat{\mu}_b) = w'\Gamma_n w \tag{3.37}$$

where the autocovariance matrix of the background traffic is represented by the $n \times n$ matrix Γ_n. With this notation, the weight vector for the sample mean is $w = 1/n \cdot \mathbf{1}$.

For an LRD process with an autocovariance function that has the form $\gamma(k) \sim c_\gamma |k|^{2H-2}$, the generalized Central Limit Theorem gives the asymptotic variance of the sample mean $\bar{R}_b$ as

$$var(\bar{R}_b) \to \frac{c_\gamma n^{2H_2}}{H(2H-1)} \tag{3.38}$$

The sample mean is not the optimal linear estimator of the mean, in that if the correlation structure is known, BLUE is the optimal estimator. However, the performance of these two estimation techniques can be compared using the relative asymptotic efficiency as given by [Beran 1994]:

$$eff(\bar{R}_b, BLUE) = \frac{2H\Gamma(H+\frac{1}{2})\Gamma(3-2H)}{\Gamma(\frac{3}{2}-H)} \tag{3.39}$$

As can be observed from Figure 3.6, the performance of the sample mean is nearly optimal. The worst performance occurs when the Hurst parameter is approximately

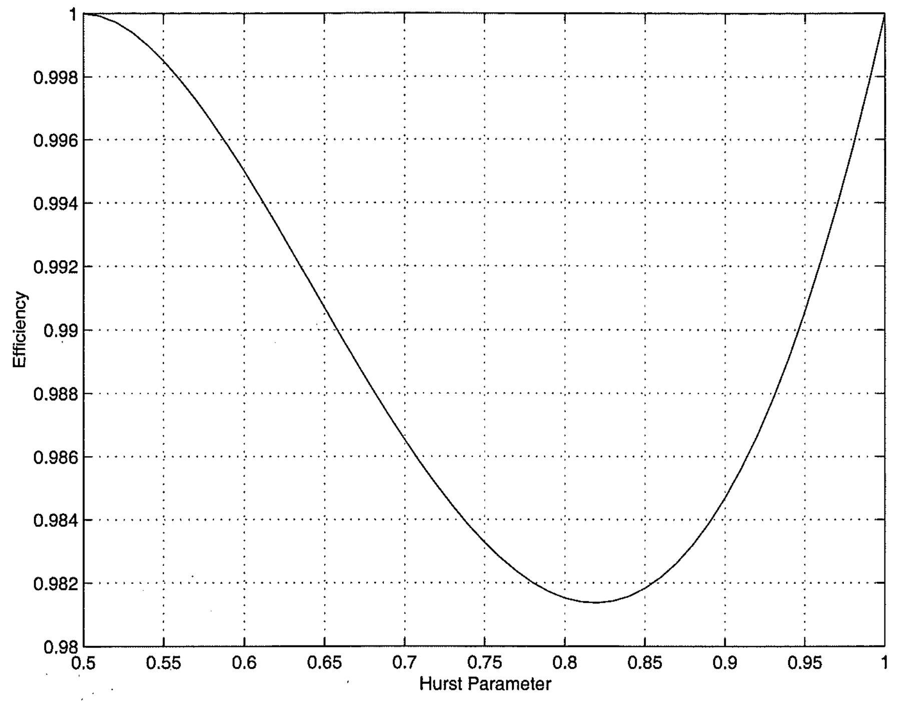


Figure 3.6 The performance of the sample mean compared with the optimal BLUE algorithm.

0.82. However, the reduction in performance is less than 2%.

3.4 PREDICTION OF LONG-RANGE DEPENDENT PROCESSES

Being able to predict future bandwidth requirements within the network is an important part of congestion control. The challenge is to predict the demand on the network resources with an adequate prediction interval which allows sufficient time for the source to react to the control information. In the context of a network providing QoS guarantees, the prediction must also have minimal error so that the performance of the system is not compromised. Thus, high utilisation of the network requires precise prediction of the traffic, yet the nature of the traffic can significantly affect its predictability [Sang and Li 2000].

Self-similar traffic has the property of long-range dependence, which means that the current level of network traffic is highly correlated with previous traffic levels. Thus, considerable information about the future of the traffic can be derived from past knowledge of the traffic, resulting in accurate predictions of the traffic. In this section, the theoretical development of predictors for LRD traffic is presented.

3.4.1 Prediction of Fractional Brownian Motion

Fractional Brownian motion is the canonical model for self-similarity. By considering prediction of this process, insight can be gained regarding predicting self-similar processes in general [Gripenberg and Norros 1996]. Aspects of prediction includes how much uncertainty in prediction can be reduced as the self-similarity of the process increases, due to the long-range dependence within the process. Also of interest is the amount of knowledge of the past required to predict a distance into the future—in other words, the window size that is necessary for the predictor to achieve a certain prediction accuracy. For instance (in the extreme), if improvement in prediction accuracy can only be achieved if all the past is known, then optimal prediction of LRD cannot be easily implemented.

Fractional Brownian motion has continuous paths, so it is natural to proceed by defining prediction as integration over the past. Since fractional Brownian motion is not a semi-martingale [Huang and Cambanis 1978, Dellacherie and Meyer 1982, Gripenberg and Norros 1996], 'reasonable' stochastic integration is not possible. However, integration can be defined if the integrand is deterministic. Gripenberg and Norros [Gripenberg and Norros 1996] represent the predictor $\hat{Z}_{a,T} = \mathbf{E}[Z_a | Z_s, s \in (-T, 0)]$ as the integral

$$\hat{Z}_{a,T} = \int_{-T}^{0} g_T(a,t) dZ_t \tag{3.40}$$

where a is the prediction interval and T is the window size. Gripenberg and Norros found the function $g_T(a,t)$ to be given by:

$$g_T(a,-t) = \frac{\sin(\pi(H-\frac{1}{2}))}{\pi} t^{-H+\frac{1}{2}} (T-t)^{-H+\frac{1}{2}} \int_0^a \frac{\sigma^{H-\frac{1}{2}}, (\sigma+T)^{H-\frac{1}{2}}}{\sigma+t} d\sigma \tag{3.41}$$

for finite T. When the past is completely known ($T = \infty$), the function becomes

$$g_\infty(a,-t) = \frac{\sin(\pi(H-\frac{1}{2}))}{\pi} \left(\frac{1}{H-\frac{1}{2}} \left(\frac{t}{a}^{-H+\frac{1}{2}} \right) - B_{a/(t+a)}(H-\frac{1}{2}, \frac{3}{2}-H) \right), \tag{3.42}$$

where $B_{\cdot}(\cdot,\cdot)$ is the incomplete beta function. The function $g_T(a,t)$ is shown in Figure 3.7, where the slow decay in $g_T(a,t)$ reveals the significant dependence of the predicted future on the past.

The performance of a predictor for a stochastic process is measured by the variance of the error between the predicted value and the actual future value. In the same paper [Gripenberg and Norros 1996], the variance of the predictor for finite T is given by

$$\mathbf{E}[\hat{Z}_{a,T}^2] = \mathbf{E}[Z_a^2] \int_0^{T/a} g_{T/a}(1,-s) \left[(1+s)^{2H-1} - s^{2H-1} \right] ds, \tag{3.43}$$

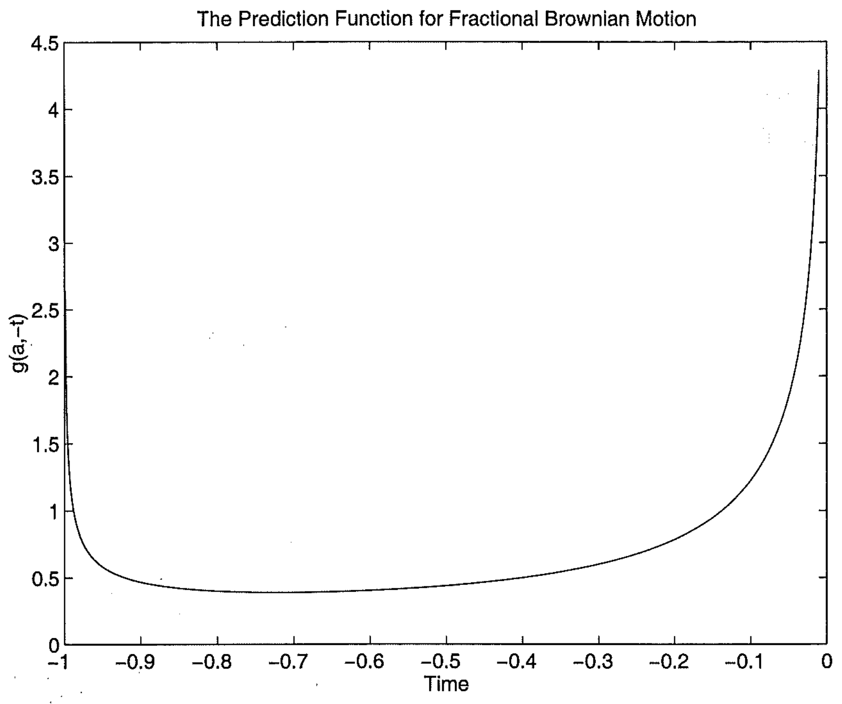


Figure 3.7 The prediction function $g_T(a, -t)$ for $H = 0.9$, $a = 1$ and $T = 1$.

and for $T = \infty$

$$\mathbf{E}[\hat{Z}^2_{a,\infty}] = \mathbf{E}[Z^2_a]\left(1 - \frac{\sin(\pi(H-\frac{1}{2}))}{\pi(H-\frac{1}{2})}\frac{\Gamma(\frac{3}{2}-H)^2}{\Gamma(2-2H)}\right). \tag{3.44}$$

Using these results, the relative variance for the prediction error can be calculated as

$$\frac{\mathbf{E}[(Z_a - \hat{Z}_{a,\infty})^2]}{\mathbf{E}[Z^2_a]} = \frac{\sin(\pi(H-\frac{1}{2}))}{\pi(H-\frac{1}{2})}\frac{\Gamma(\frac{3}{2}-H)^2}{\Gamma(2-2H)}. \tag{3.45}$$

Figure 3.8 profiles values of the relative variance versus the Hurst parameter and shows that for values of the Hurst parameter close to 1, the uncertainty in prediction of fractional Brownian motion decreases rapidly. Thus, an increase in performance gain can be achieved by using predictors developed from the correlation structure of self-similarity, as the Hurst parameter approaches 1.0.

3.4.2 Prediction of Long-Range Dependent Time Series

Define a stationary random variable $X = \{X_k : k \in I\}$ which has zero mean, variance σ^2 and autocovariance function $\gamma(k)$. A set of M measurements (where M can be varied) is a time series of the random variable, and can be stored in the vector $\mathbf{X}_M = \{X_k, X_{k-1}, \cdots, X_{k-M+1}\}'$. The optimal linear predictor $\mathbf{G}_\delta$ for a prediction interval δ

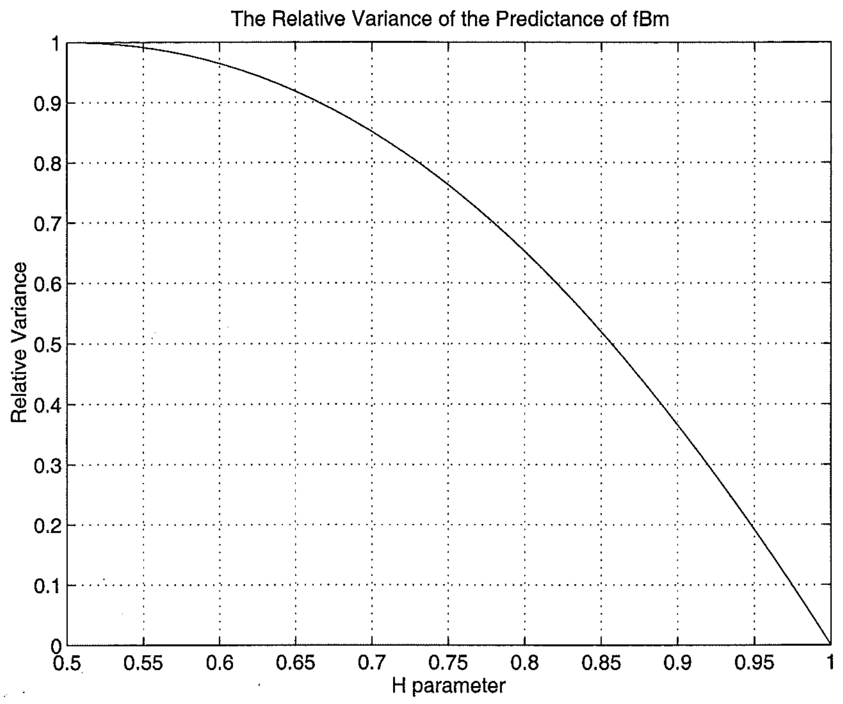


Figure 3.8 The relative variance for the prediction error versus the Hurst parameter H.

gives the predicted value of X as

$$\hat{X}_{k+\delta} = \mathbf{G}'_{\delta}\mathbf{X}_M. \tag{3.46}$$

The predictor $\mathbf{G}_\delta$ can be generated by solving a matrix equation that arises by multiplying (3.46) by $X_{k+\delta-m}$ (for $m = k - M + 1, \dots, k$) and taking the expectation. The final result is given by

$$\Gamma \mathbf{G}_\delta = \gamma_\delta, \tag{3.47}$$

where Γ is the covariance matrix:

$$\Gamma = \begin{bmatrix} \gamma(0) & \gamma(1) & \gamma(2) & \cdots & \gamma(M) \\ \gamma(-1) & \gamma(0) & \gamma(1) & \cdots & \gamma(M-1) \\ & \vdots & & \ddots & \vdots \\ \gamma(-M) & \gamma(-M+1) & \gamma(-M+2) & \cdots & \gamma(0) \end{bmatrix}, \text{ and}$$
$$\gamma_\delta = \begin{pmatrix} \gamma(\delta) \\ \gamma(\delta+1) \\ \vdots \\ \gamma(\delta+M-1) \end{pmatrix} \tag{3.48}$$

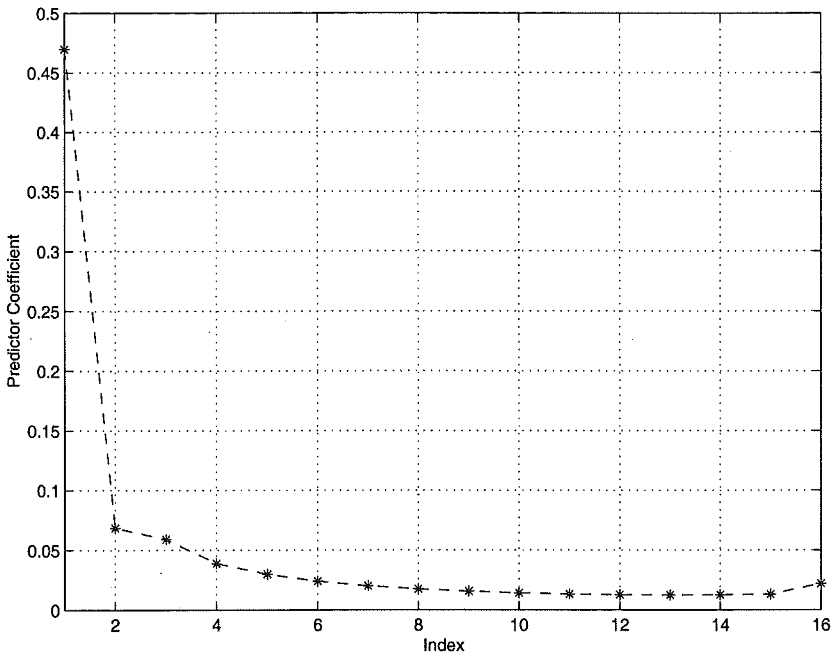


Figure 3.9 An example predictor for the discrete time series, with H = 0.85 and M = 16.

The predictor $\mathbf{G}_\delta$ can be found by directly inverting the matrix Γ, or found recursively using the Durbin-Levinson algorithm [Brockwell and Davis 1991] which uses the symmetry of the autocovariance function $\gamma(k)$. If $\gamma(0) >$ and $\gamma(k) \to 0$ as $k \to \infty$, then the components of the predictor $\mathbf{G}_\delta$ and the mean squared errors $v_n = E(X_{n+1} - \hat{X}_{n+1})^2$ satisfy

$$\begin{aligned}
G_{11} &= \frac{\gamma(\delta)}{\gamma(0)}, \\
v_0 &= \gamma(0), \\
G_{MM} &= \left[\gamma(n) - \sum_{i=1}^{M-1} G_{M-1,i}\gamma(M-i)\right] v_{M-1}^{-1}, \\
\begin{bmatrix} G_{M1} \\ \vdots \\ G_{M,M-1} \end{bmatrix} &= \begin{bmatrix} G_{M-1,1} \\ \vdots, \\ G_{M-1,M-1} \end{bmatrix} - G_{MM} \begin{bmatrix} G_{M-1,M-1} \\ \vdots \\ G_{M-1,1} \end{bmatrix} \\
v_M &= v_{M-1}(1 - G_{MM}^2).
\end{aligned} \tag{3.49}$$

The predictor $\mathbf{G}$ for prediction interval $\delta = 1$ is shown in Figure 3.9, and has been based on the autocovariance function for fractional Gaussian noise and a Hurst parameter of $H = 0.85$. It has a similar form to the continous predictor for fractional Brownian motion in Figure 3.7, both in the behaviour at the boundaries and the slow decay in the predictor coefficients.

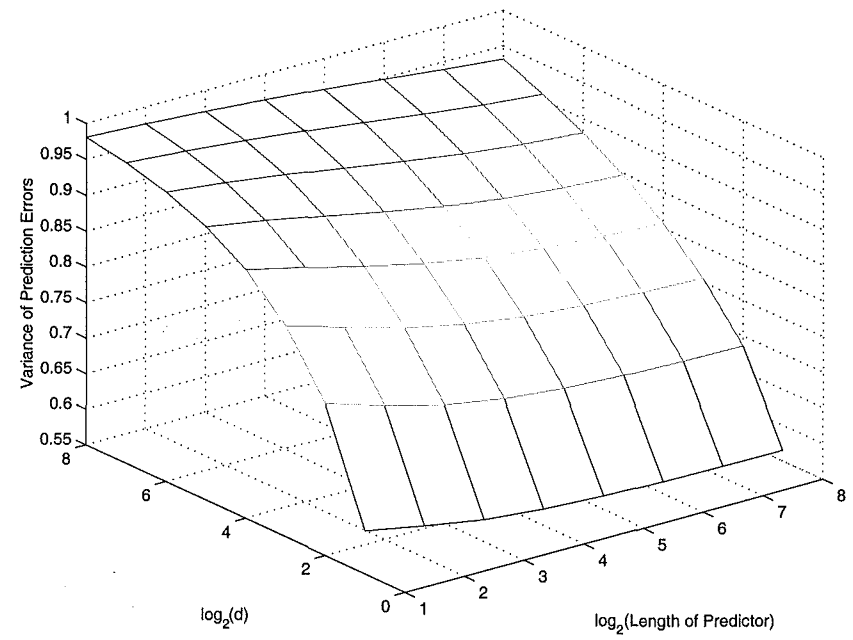


Figure 3.10 Variance surface for predictor errors as the predictor length and interval are varied.

The variance of the prediction errors is given by [Brockwell and Davis 1991]

$$v_\delta = \gamma(0) - \gamma_\delta' \Gamma^{-1} \gamma_\delta, \tag{3.50}$$

and the normalized variance is plotted as a surface function of the prediction interval δ and the predictor memory M (Figure 3.10). As is expected, the variance of the prediction errors increases as prediction further into the future is attempted. The graph shows that there is insufficient information for predicting large "distances" into the future, even though the correlations are assumed heavy-tailed due to LRD. An unexpected phenomenon from Figure 3.10 is that increasing the length of the predictors only results in minor decreases in the variance. This may be partly due to the use of a linear predictor in combination with LRD processes that still have significant correlations as the lag increases asymptotically. The practical implications of this phenomenon will be discussed in Chapter 7.

3.5 APPLICATION TO NETWORK TRAFFIC

Thus far in this chapter, the concepts of self-similarity and classes of mathematical models which can be used to represent self-similar processes have been presented. The methods for detecting self-similarity were presented. In this section, these concepts are applied directly to the situation where network traffic displays self-similarity. Firstly, types of traffic which have been shown to exhibit self-similarity are presented, followed by a discussion of the causes for this phenomenon. Secondly, the implications of self-similar traffic for network management are reviewed. Finally, the topic of this thesis is discussed: traffic control where protocols are interacting with self-similar traffic, together with review of some approaches to this problem that have already been published.

3.5.1 Types of Traffic Exhibiting Self-similarity

Many types of traffic have been shown to exhibit self-similarity. These include Ethernet traffic on Local Area Networks (LANs), Variable Bit Rate encoded video, World Wide Web and signalling traffic [Leland *et al.* 1994, Beran *et al.* 1995, Crovella and Bestavros 1997]. Even more recently, it has been shown that TCP itself can cause self-similarity and propagate LRD through a network [Veres and Boda 2000, Veres *et al.* 2000]. Since integrated networks in the future will be carrying a mix of these types of traffic, studies into the self-similar nature of traffic are reviewed.

The first significant study into self-similarity of network traffic was undertaken by Leland and his colleagues at Bellcore [Leland *et al.* 1994]. Ethernet is a widely used LAN protocol that uses a random access technique called carrier-sense multiple access with collision detection. Ethernet terminals can transmit a packet on the network at any time, but to ensure that packets are transmitted successfully, a terminal listens for packet transmissions from other terminals to ensure that packet transmissions do not collide. If a collision between packet transmission does occur, the terminal backs-off and re-transmits the packet later. From the point of view of the receiver, terminals read all packets on the network and simply copy packets which have their address recorded in the header. Leland *et al.* recorded the timestamps of hundreds of millions of Ethernet packets and analysed the packet count processes for this data. Logging of Ethernet packets occurred over a four year period. Traffic between the network at Bellcore and the rest of the Internet was also recorded in some of the data networks. Leland *et al.* then showed that, for these data sets, the Hurst parameter typically lay in the interval $[0.85, 0.95]$, hence revealing the significant self-similarity in the data. Further study into these data sets [Willinger *et al.* 1997] revealed that individual terminals could be represented by ON/OFF sources, where the distributions for ON and/or OFF periods were heavy-tailed. As has been discussed previously in Section 3.2.4, traffic that results from aggregating the packet streams of ON/OFF sources, with heavy-tailed distributions for the ON/OFF periods, is self-similar.

Due to the increased use of multimedia and video streaming applications, another traffic type that is expected to form a significant proportion of high-speed network traffic is variable bit rate encoded video traffic. The reason for using a variable encoding rate is to ensure constant quality of the video stream as the activity in scenes varies. Two studies into the nature of VBR video traces [Garrett and Willinger 1994, Beran *et al.* 1995] have shown that, independent of the particular encoding method used, LRD is an inherent feature of the data. The data sets that were used included included sequences from video conferences, sports shows, TV series and movies. All data sets showed Hurst parameters that were statistically significantly greater than 0.5. Thus, self-similarity is also an important consideration in managing network traffic when VBR video is a dominant component of traffic.

A very interesting study into self-similarity was performed by Crovella and Bestavros [Crovella and Bestavros 1997]. Their focus was World Wide Web transfers, which shifted focus away from the traffic loads at the network layer to requests and objects being transferred at the application layer. To monitor this activity, Crovella and Bestavros modified Mosaic, a Web browsing application, and installed this modified version within the computing environment of their department at Boston University. Data they recorded from the network included information regarding the session and the URL of the requested packet, and also the time stamp, size and retrieval time of objects downloaded by the browser. Analysis of this data generated Hurst parameters in the range 0.77 to 0.87, again confirming self-similarity of network traffic. However, the specific nature of their study, which involved the size of objects, request and retrieval times, allowed them to investigate the causes of self-similarity. It was shown that one of the primary reasons for the self-similarity is the heavy-tailed distribution of file sizes, a result that has also been demonstrated by Paxson and Floyd [Paxson and Floyd 1995] and Park *et al* [Park *et al.* 1996]. Thus, a large number of files involved in the Web transfers are significantly greater than the mean file size. This in turn influences the time required to transfer the object across the Internet. Crovella and Bestavros' study also observed that user interaction times were heavy-tailed, contributing further to self-similarity. The distribution of file sizes is evidence that the self-similar nature of network traffic is a result of fundamental properties of the information stored at the edge of the network and the processes which are involved with the retrieval of that information.

3.5.2 The Implications of Self-similar Traffic

Having reviewed the evidence for self-similarity in network traffic, and before introducing traffic management approaches based on self-similarity, it is worthwhile discussing the implications of self-similarity on network engineering. Self-similar traffic, by nature, is very different from Poisson models that have been traditionally assumed by network engineers. Its effect on the network is not only of theoretical interest for academia, but also motivates the development of network management schemes which take into account the unique properties of self-similarity. The noticeable difference between self-similar traffic and traditional Poisson traffic is that the burstiness of self-similar traffic remains evident for a wide range of time scales and does not decrease with aggregration (or multiplexing). Equivalently, there appears to be much longer bursts of data and also longer periods of sparse activity. Intuitively, one would expect that the impact of this type of traffic on the network would be longer queues at the nodes, greater packet losses, and that the admission control mechanisms would have to be more conservative in allowing connections into the system. These effects can be quantified more specifically by investigating the queueing performance of self-similar traffic.

In his seminal paper [Norros 1994], Norros studied a queueing model with self-similar input, where the queue was serviced at a constant rate C. The queueing process $V(t)$ itself is described by

$$V(t) = \sup_{s \leq t} [A(t) - A(s) - C(t-s)], \, t \in (-\infty, \infty), \tag{3.51}$$

where $A(t)$ is the fractional Brownian traffic model described in Equation (3.10). Using the stationarity of the increments of $A(t)$ (and hence the stationarity of $V(t)$), Norros proved that the tail behaviour of this queue was bounded by a Weibull distribution

$$P(V > x) \sim \exp(-\gamma x^{\beta}) \tag{3.52}$$

thus demonstrating the heavy-tailed nature of a queue with self-similar input. The parameters γ and β are given by:

$$\begin{aligned} \gamma &= \frac{1}{2am(1-H)^2} \left(\frac{(1-m)(1-H)}{H} \right)^{2H}, \\ \beta &= 2(1-H). \end{aligned} \tag{3.53}$$

Equation (3.52) implies that the frequency of having long queue lengths is much higher than that for Poisson models, resulting in longer delays through the network and greater packet losses due to buffer overflows in real networks (buffers being of finite size).

This fundamental queueing behaviour of self-similar traffic within a network has significant effect on network performance [Paxson and Floyd 1995]. First of all, if Poisson traffic models are used in the situation where the traffic is actually self-similar, the performance of the network will be overestimated and much greater average packet delays and maximum queue lengths will be experienced by the users. Secondly, due to the bursty nature of the traffic, periods of congestion will be much longer and losses will be heavily concentrated around these periods, resulting in large blocks of data needing to be retransmitted rather than a few packets intermittently spaced throughout the whole connection period.

An attempt to decrease losses is conventionally approached by increasing buffer sizes within a network, but unfortunately self-similar traffic results in a hyperbolic decrease in losses rather than a logarithmic decrease as buffers are increased. Therefore, a linear increase in the buffer size at a particular node will not result in a significant decrease in losses. In addition, a slight increase in the number of connections admitted to the network will result in significant increases in losses. As such, traffic admission control schemes need to be redesigned so that the performance of the system is not degraded in the face of self-similar traffic. Recently, measurement-based admission control (MBAC) has been proposed as a method of increasing network utilisation [Gibbens and Kelly 1997, Grossglauser and Tse 1999]. However, the burstiness of self-similar network traffic means that apparent silences may not indicate the true number of active sources. Thus, MBAC algorithms may make incorrect admission decisions, so admission control algorithms that specifically incorporate the characteristics of self-similarity need to be designed.

3.5.3 Self-similar Traffic Management

Due to the fundamentally self-similar nature of network traffic, and the significant impact that self-similar traffic can have on network performance, it is essential that traffic management algorithms are designed and/or evaluated using self-similar input traffic. This is even more important within integrated networks, such as ATM and IP DiffServ, which aim to meet the QoS requirements of a variety of classes of traffic. The implications of self-similarity is that during bursts within high priority traffic, lower priority traffic can experience significant packet losses, requiring numerous retransmissions if the burstiness of the traffic is ignored. Conservative traffic management schemes can be used to ensure that losses are minimised, but these algorithms will lead to under utilisation of a network when the bandwidth requirements of the high priority are low. Hence the need to consider self-similar traffic management.

This area has not gone unnoticed in the research literature, and researchers have designed admission control algorithms, packet discard policies and congestion control algorithms based on the characteristics of self-similar traffic [Adas and Mukherjee 1995, Giordano *et al.* 1996, Tuan and Park 1999, Park and Willinger 2000]. Since admission control algorithms make static decisions whether to allow connections to enter the network, admission of self-similar traffic means that control protocols must be more conservative than ones designed for Poisson traffic. When the Central Limit Theorem is applied to self-similar processes, confidence intervals for aggregate traffic levels are much wider due to higher probabilities that extreme traffic levels will be sustained for longer periods of time [Beran 1994, Giordano *et al.* 1996]. Queue distributions are also heavy-tailed. Both of these results mean that admission control needs to be more conservative when multiplexing large numbers of self-similar connections together. Network performance, with LRD traffic, can also be improved by actively discarding cells to avoid congestion [Adas and Mukherjee 1995].

The subject of this thesis concerns reactive traffic management, and particularly focuses on dynamic traffic control algorithms. Rather than requiring a conservative approach, dynamic resource allocation can utilise the correlation structure of LRD traffic to predict future bandwidth requirements. Significant related work includes the multiple time scale control (MTSC) algorithm proposed by Tuan and Park [Tuan and Park 1999].

Tuan and Park adapt linear increase/multiplicative decrease behaviour, typical of congestion control algorithms like TCP, so that the aggressiveness of such an algorithm when the algorithm is probing the network for available bandwidth, is modulated by predictions of the network load at larger time scales. If network utilisation is self-similar, the available bandwidth within the network will exhibit a non-degenerative correlation structure. Intuitively, this means that high traffic levels are more likely to be sustained, rather than returning to the mean level, and similarly with low traffic levels. Hence, the control algorithm needs to be less aggressive in the next prediction interval if the current traffic level is high, and vice versa. Tuan and Park develop an on-line prediction approach, using estimates of the conditional probabilities for shifts in traffic levels, and they show that incorporating this predictive approach into traffic control results in higher throughput when traffic is LRD. This control strategy partly solves the problem where connections with higher latency are penalised when using linear increase, multiplicative decrease control protocols. In a later paper [Park and Tuan 2000], Park and Tuan explicitly modify TCP using the same multiple time scale control approach. They have also applied this concept to redundancy control for transport of real-time traffic [Tuan and Park 2000]. As one would expect, the performance advantage in both cases is similar to that of the MTSC algorithm.

The research work in this thesis is closely related in concept, but differs in a number of regards. The most fundamental difference is that the approach used here engages active participation of a switch when estimating LRD and making control decisions. Specifically, the context of traffic control algorithm proposed here is the ABR service in ATM networks, where explicit rate information is returned to sources regarding allocated available bandwidth. Hence, it can be viewed as a per-hop approach, in contrast to Park *et al.*'s approach which is purely end-to-end. Another key difference is that structure of the predictors used in this thesis is based explicitly on the LRD structure of background traffic and incorporates round-trip delays when managing multiple ABR sources. Finally, explicit use of LRD structure has allowed the investigation of fundamental properties of the algorithm proposed in this thesis, including sensitivity to traffic parameters, the development of robust control mechanisms and the influence of LRD versus SRD statistics on traffic prediction. Thus, this thesis forms a valuable contribution to the area of control of self-similar traffic by complementing previous work.

Chapter 4

RATE CONTROL IN A SELF-SIMILAR NETWORK ENVIRONMENT

A feature of modern data networks is the ability to adaptively control the transmission rate of sources utilising a network, as opposed to traditional telephony networks which admit connections only when they can allocate the fixed bandwidth requirements to the connection. Rate control is possible because the transfer of computer-generated data is actually an elastic process which does not have the strict end-to-end delay requirements of telephone connections. Early packet networks, such as ARPANET and DECnet, were indeed designed to take advantage of this property. In essence, a connection using a service that incorporates rate control mechanisms adapts to the amount of available network resources by adjusting its transmission rate based on feedback from the network.

An important concept in developing rate-control mechanisms for elastic traffic in the framework of an integrated services network is that it must be considered from the broader perspective that includes the characteristics of the high priority traffic using the other services (or the "background" traffic). While it may seem redundant to highlight this given the definition of an integrated network, two functions of the control mechanism are important. Firstly, it ensures that the QoS for the background traffic is not affected by the controller. Secondly, by more closely tracking the actual bandwidth requirements of high priority traffic, available bandwidth can be distributed more effectively, thus resulting in higher utilisation of the network.

In this chapter, the self-similar property of network traffic is incorporated into network control mechanisms. Self-similar traffic behaves very differently to Poisson traffic, as outlined in Chapter 3, and it is necessary to include these characteristics in the design of traffic controllers to ensure high performance in network environments where self-similar traffic dominates. This type of controller has the potential not only to predict the background traffic with more accuracy, but also to compensate for the effects of self-similarity by inverting its inherent burstiness in a similar manner to noise cancellation techniques. The fundamental concept is the prediction of self-similar background traffic with prediction horizon given by the round trip delay. This information is then used to calculate the desired source rates to so utilise the available bandwidth. All the issues regarding the implementation of this type of controller, including self-similar pa-

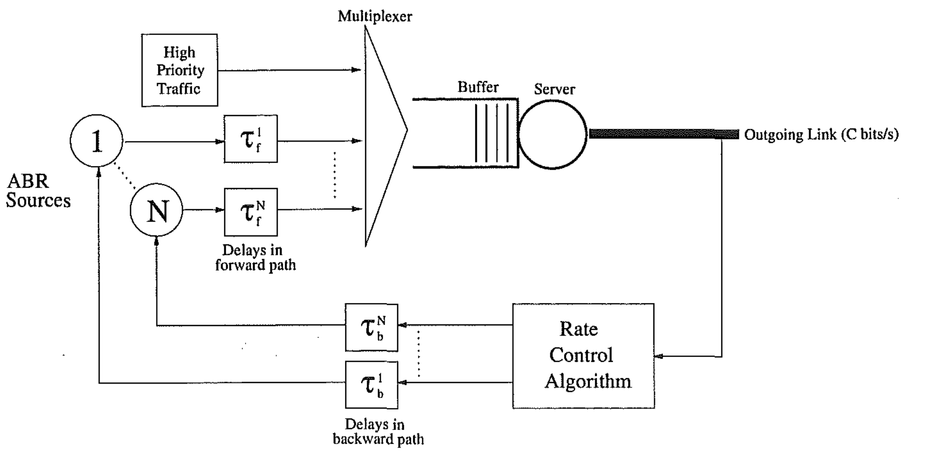


Figure 4.1 The model of the system which is used for the development of the rate control mechanism.

rameter estimation, choice of model, and the calculation of the source rate are addressed in the following sections of this chapter.

4.1 SYSTEM MODEL

The system model that is used for developing the rate control mechanism is a decentralised control system in which local congestion at a node is used as the measure of the network state (available bandwidth). The node returns explicit information to the sources regarding the desired source rates. This model is shown schematically in Figure 4.1 [Ritter 1998, Östring *et al.* 1999a]. The control information is delayed both in the forward path and the backward path before it reaches the sources. Upon receiving this information, the source then adjusts its rate so that it does not exceed the explicit rate information. Thus, the model is defined in the context of the ABR service within ATM networks, where switches calculate explicit rate information and convey this to the ABR sources using the ER field in the RM cells. The model can be viewed as a conceptual representation of a transmission control protocol for IP networks. However, the mechanism could not be directly implemented within TCP as currently there are no fields for returning explicit control information and no mechanisms for acting upon that type of information.

4.2 AVAILABLE BANDWIDTH ESTIMATION

There are two primary methods of estimating available bandwith at a node. The first measure is the length of the queue at the node (either a queue at the input port, which stores incoming cells; or an output port queue, which is situated just prior to cells being serviced by the outgoing link). Many congestion control algorithms, including ABR traffic control mechanisms, use this measure because of its ease of measurement [Bonomi and Fendick 1995]. Queue management protocols embedded in the switch already have information regarding the queue length, so it is a quantity that can be used within other traffic management protocols. Thus algorithms designed to be directly implemented in the switch generally use this measure. Simplicity in measuring queue length must be balance by the complexity the parameter adds to the controller itself—the queue length introduces an integral term into the control model. This integral is characteristic of the function of a queue in that it stores excess cells on route from the input to the output links, hence integrating bandwidth overflows.

The other primary approach is estimating bandwidth requirements of high priority traffic. The advantage of this measure is that it is directly related to the aim of rate control—the distribution of the available bandwidth. The main function of the controller is simply to determine how much utilisation of the outgoing link is currently being achieved, and then to calculate the individual ABR rates, so that remaining bandwidth is fairly allocated. In this approach, the integrating effect of the buffer is avoided in the control loop and the buffer is merely a device to handle instantaneous mismatches between the input bandwidth load and the capacity of the outgoing link. It should be noted, however, that while the buffer is not involved in the control loop, the dynamics of its queue can be used to observe the performance of the controller. The disadvantage of using bandwidth measures to develop the controller is that implementation of the controller is difficult. The actual bandwidth demand or utilisation must be estimated using packet arrivals using a specified time constant.

In this chapter, the concept of incorporating self-similarity of background traffic into rate control algorithms is investigated. Thus, the control algorithm is developed using bandwidth estimation as the control variable. This follows the approach of Zhao *et al.* [Zhao *et al.* 1997] who use a similar filtering technique based on bandwidth utilisation measurements. ERICA also uses bandwidth estimation to calculate explicit rate feedback within its control algorithm [Kalyanaraman *et al.* 2000]. The resulting control algorithm, which is implemented locally at a node, is developed as follows.

4.3 DEVELOPMENT OF THE RATE CONTROL MECHANISM

Firstly, the variables that are used in the development of the controller are defined. The total bandwidth arriving at the network node from the rate-controlled connections is defined as $U(t)$ (where U is a conventional symbol for a feedback variable in control theory), the high-priority background traffic requires the bandwidth $R_b(t)$ and the outgoing link has the bandwidth capacity C. These variables all have units bits/s. The outgoing link is considered congested if the total input load is greater than the capacity of the node,

$$\text{Total Input Load} > C. \tag{4.1}$$

In order to avoid congestion and to accommodate small instantaneous mismatches in the rate calculations, the target of the control algorithm is to maintain the utilisation of the outgoing link at a specific value ρ, where

$$U(t) + R_b(t) = \rho\, C, \tag{4.2}$$

for $0 < \rho < 1$. The network node also has a buffer associated with it to absorb any transients in the bandwidth requirements. Utilisation is then normally chosen to be close to 1.0.

The task of the rate controller is to allocate the controlled bandwidth $U(t)$ among the rate-controlled sources. The number of controlled sources, $N(t)$, is assumed to vary slowly when compared with the fundamental time constant of the control system. So $N(t)$ can be considered constant for the purposes of developing the controller. The node simply updates the number N whenever a new ABR connection is admitted to the network using that particular node. As shown in 4.1, each connection has its own round-trip delay $\tau_j(t)$ through the network. This delay $\tau_j(t)$ comprises both the forward and backward delay components. The total controlled bandwidth can now be represented by:

$$U(t) = \sum_{j=1}^{N} u_j(t - \tau_j(t)). \tag{4.3}$$

The state of the system is its deviation from the desired target, namely the utilisation value ρ. The state variable can be defined as follows:

$$x(t) = \rho\, C - \left[\sum_{j=1}^{N} u_j(t - \tau_j(t)) + R_b(t) \right]. \tag{4.4}$$

The aim of the control mechanism is to determine the explicit rate values $u_j(t)$ that should be returned to the controlled sources so that the state variable $x(t)$ is maintained at the value 0. Since this is a stochastic system with the view of recognising the self-similar characteristics of the background traffic process $R_b(t)$, a natural metric for the performance of the control system is the variance of the state variable $E[x^2(t)]$ [Åström and Wittenmark 1997].

The state variable, as defined in Equation (4.4), is a continuous variable. However, control actions can only occur at discrete time instants. Thus, it is logical to convert the control problem into discrete-time. It is assumed, for computational ease, that control information is calculated at uniformly spaced intervals, given by the time constant ΔT. This explicit rate information is entered into all the RM cells arriving during this interval ΔT. The arrivals of RM cells are, in general, not regularly spaced, but are generated every 32 data cells in the cell stream of a particular ABR connection. As the transmission rate of the controlled source varies, the number of RM cells varies in accordance. Two further simplifications are made to the model of the control system. Firstly, the round-trip delay for each controlled connection is assumed to be constant, a standard assumption used by other researchers [Benmohamed and Meerkov 1993, Zhao *et al.* 1997, Kolarov and Ramamurthy 1999]. This results in the discrete state variable:

$$x(k) = -\sum_{j=1}^{N} u_j(k-\delta_j) + A(k), \tag{4.5}$$

with the variable $A(k) = \rho C - R_b(k)$ being equivalent to available bandwidth (given the desired utilisation level). The variable δ_j is the constant round-trip delay for connection j. The other simplification is to reduce the system from multiple-input single-output to a single-input single-output system by introducing a sharing mechanism. This allows the connections to be decoupled from each other, and a simple control law to be defined. In this analysis a sharing mechanism with arbitrary weights ω_j is used, where $\sum \omega_j = 1$, which includes the case where the available bandwidth is shared fairly. By defining

$$\alpha_j(k) = \omega_j A(k),\ j = 1, \ldots, N \tag{4.6}$$

where α_j is the amount of the available bandwidth allocated to connection j, the system now becomes a collection of N subsystems, each with their own controller,

$$x_j(k) = -u_j(k-\delta_j) + \alpha_j(k),\ j = 1, \ldots, N, \tag{4.7}$$

where the round-trip delays have been ordered such that $\delta_1 \geq \delta_2 \geq \cdots \geq \delta_N$, without loss of generality. This ordering process is required for the prediction compensation algorithm, presented in the next section. The node simply needs to keep an ordered list of the connection network delays, and update the list when new connections are added to the network.

The goal of each stochastic controller is to maintain the state variables close to the desired operating point $x_j(k) = 0$. This can be achieved by minimising the quantity $E[x_j^2(k)] = 0$. Åstöm proved that this implies that the predicted value for the state variables should be zero for all k (refer to Theorem 12.2, Remark 2 in [Åström and Wittenmark 1997]). Thus, to achieve $\hat{x}_j(k) = 0$, the controlled source rate $u_j(k)$ must be calculated by:

$$u_j(k) = \hat{\alpha}_j(k + \delta_j \mid W(m) : m \leq k). \tag{4.8}$$

The desired source rate is simply the predicted value for the share of the available bandwidth designated for each connection, given that the prediction has to be calculated using the round-trip delay as the prediction horizon. Returning to the objective of this thesis, which is to incorporate the self-similarity of the background traffic into the control algorithm, the long-range dependence of this traffic can be used to predict more accurately the future bandwidth requirements of the traffic. Provided that a history of background traffic levels has been measured and stored at the network node, prediction can be achieved using the optimal predictors introduced in Section 3.4.

4.3.1 Predictive Kernel of the Rate Control Algorithm

The kernel of the control algorithm for calculating the explicit rate for connection j is a δ_j-step predictor of the background traffic level. This determines the available bandwidth at for a time slot δ_j in the future, according to the round-trip delay of connection j. The available bandwidth is then allocated using weights ω_j,

$$u_j(k) = \omega_j \left[\rho C - \hat{R}_b(k + \delta_j)\right]. \tag{4.9}$$

The control aim given in Equation (4.8) in the preceeding section is thus achieved. This basic algorithm is labelled *Predictive Explicit Rate Control* (PERC).

For the prediction of the background traffic level, $\hat{R}_b(k + \delta_j)$, the background traffic process is assumed to be a stationary process with non-zero mean μ, variance σ_b^2 and has a correlation structure given by the autocovariance function $\gamma(k)$. The long-range dependence of the background traffic is introduced by this correlation structure, in that the correlations are slowly decaying which results in a non-summable autocovariance function. Equivalently, the spectral density of the background traffic diverges near the origin due to its long-range dependence (LRD). Ideally, an entire history of the background traffic would be available so that the prediction can utilise all prior traffic levels and their correlation with a future traffic level. However, this is neither practical, as infinite memory capacity cannot be provided at the network nodes to store traffic information. Nor is it necessary. Gripenberg [Gripenberg and Norros 1996] proved that for fractional Brownian motion there is no performance gain in storing information regarding the history of the process greater than the time interval which is being predicted into the future. By appealing to their work as the basis for using finite predictors, the background traffic can be predicted as follows [Östring *et al.* 1999a, Östring *et al.* 2000a]:

$$\hat{R}_b(k+\delta_j) = \mu_b + \mathbf{G}'_{\delta_j}(\mathbf{R}_\mathbf{b}(k) - \mathbf{1}\cdot\mu_b), \tag{4.10}$$

where the predictor $\mathbf{G}_{\delta_j}$ is the optimal linear predictor detailed in Section 3.4.2. $\mathbf{G}_{\delta_j}$ is generated using the equation (3.47). Previous traffic measurements are stored in the vector $\mathbf{R}_\mathbf{b}(k)$, which has capacity for M samples of the background traffic, and $\mathbf{1} = [1, 1, \ldots, 1]'$. The performance of this predictor is measured using the variance of the prediction errors, which quantifies the mismatch between the true bandwith requirements of the high priority traffic and the estimate of these bandwidth requirements.

4.3.2 Discussion on Finite Predictors for LRD

Before continuing with the development of the control mechanism, it is worthwhile to make a slight digression into the question of whether the model which has been developed so far truly captures the long-range dependent character of the background traffic. Note the predictor only uses a finite section of the autocovariance function. Long-range dependence is an asymptotic behaviour, which defines the behaviour of the correlations as the lag tends to infinity, thus any model which does not use the autocovariance function for all lags would appear to have evaded the true nature of LRD.

Two points should be noted here. Firstly, the use of long-range dependence as a mathematical representation of network traffic is based on the goodness-of-fit of the models to the finite-lag correlations within the traffic data, and does not imply that the time span of the traffic data must be infinite for this phenomenon to exist. LRD models can parsimoniously model the correlation structure of the traffic, using the mean, variance and Hurst parameter, even if correlations at asymptotically large lag are not directly being used.

Also, a very relevant study to this discussion is Grossglauser and Bolot's research [Grossglauser and Bolot 1999] on the concept of a "correlation horizon" for the impact of self-similar traffic on network performance. In this study, it was demonstrated that the amount of correlation required for performance evaluation depended on the time scales specific to the system and that the impact of losses becomes neglible beyond the correlation horizon. The time scale of a system is a function of the maximum buffer size, and once the correlation horizon is determined for the system, any model which captures the correlation structure up to the correlation horizon can be used. Ryu and Elwalid came to similar conclusions in their paper [Ryu and Elwalid 1996]. Thus, for developing the controller, long-range dependent models are used for the background traffic due to the parsimonious representation of the data.

4.3.3 Defining the Prediction Compensation Algorithm

Returning to the design of the control mechanism, the concept of a prediction compensation algorithm is introduced, which uses connections with lower latency to compensate for larger prediction errors in the rate calculations for the high latency connections. Firstly, it should be noted that the global minimum is not necessarily achieved by optimising the controllers in Equation (4.8) for each of the controlled connections. This can be observed by the inequality:

$$\mathbf{E}\left[(\sum_j X_j)^2\right] \leq \sum_j \mathbf{E}\left[X_j^2\right], \tag{4.11}$$

particularly true when the variables are correlated, as is the case when dealing with long-range dependent variables. An operating point is thus sought which occurs closer to the true global minimum.

By considering the system as a whole, what is recognizable is that larger prediction errors will occur for the rate calculations for those connections with large round-trip delay, since the prediction horizon is greater. However, in terms of calculating feedback information, lower latency connections are allocated available bandwidth (for a particular future time slot) after high latency connections have been allocated bandwidth. Assuming that rate feedback information is stored at the network node in order to police the arrival rates of the controlled connections, prior feedback information is used in calculating explicit rates for low latency connections to compensate for the more significant prediction errors incurred in the transmission rates from sources with connections having longer round trip delay.

In defining the prediction compensation algorithm, the concept of the *partial controlled* bandwidth is required when the explicit rate is being calculated for a particular source, labelled j for instance. Recall that the sources have been ordered according to their round-trip delay. So when the rate information for source j is determined for the specific time slot $k + \delta_j$, all the desired rates for the sources i, where $i < j$, have been calculated previously. Hence, the partial controlled bandwidth is defined for the rate calculation of source j as the available bandwidth which has already been allocated for that particular time slot:

$$\partial U_j(k + \delta_j) = \sum_{i=1}^{j-1} ER_i(k - \delta_i + \delta_j) \tag{4.12}$$

This rationale assumes that all the longer-latency sources transmit at a rate which is equivalent to the explicit rate information that is returned to them. The source rates may actually be lower, due to the connection being constrained by congestion elsewhere in the network or the source not having enough data to send at the specified rate. However, for simplicity of modelling it is assumed that the sources are persistent and are always able to utilise any bandwidth allocated to them. Also congestion at the network node under consideration is assumed to be the constraining factor in the transmission rates of the sources—standard assumptions when developing analytical models for ABR rate control [Benmohamed and Meerkov 1993, Kolarov and Ramamurthy 1999].

After predicting the amount of bandwidth that should be allocated to source j using (4.9), the node can now also determine what the value of the partial controlled bandwidth should be for the fair share of the bandwidth available to the sources $i < j$ to be fully utilised. This quantity is defined as the optimal partial controlled bandwidth:

$$\partial U_j^*(k + \delta_j \mid R_b(m),\ m \in [k - M + 1, k]) = \sum_{i=1}^{j-1} \frac{\omega_i}{\omega_j} u_j(k). \tag{4.13}$$

Optimality is linked to the calculation of the source rate j for time slot $k+\delta_j$, and is not necessarily the true optimum when the system actually reaches that instance in time. However, ∂U_j^* has a lower error associated with it than does the partial bandwidth ∂U_j since the prediction process for source j has a shorter prediction horizon δ_j than all the previous predictions for sources $i < j$. This improvement can be incorporated into the distribution of the available bandwidth by adjusting the rate calculation for source j with a prediction compensation component:

$$\begin{aligned}\Delta u_j(k) &= \frac{\omega_j \left[\partial U_j^*(k+\delta_j) - \partial U_j(k+\delta_j)\right]}{\sum_{i=j}^{N} \omega_i}, \\ &= \frac{\sum_{i=1}^{j-1} \omega_i u_j(k) - \omega_j \sum_{i=1}^{j-1} ER_i(k - \delta_i + \delta_j)}{\sum_{i=j}^{N} \omega_i}.\end{aligned} \tag{4.14}$$

where $\delta_i \geq \delta_j$, $i = 1, \ldots, j-1$. The deviation of the partial controlled bandwidth from its optimal value (determined at the rate calcalation for source j) is distributed among the remaining sources $j \leq i \leq N$ according to their relative weight. The reason for this is that rate calculations for sources $i > j$ will have smaller prediction error, so it is undesirable to over-compensate early in the control process only to have to make significant compensations in the opposite direction for later sources, resulting in a potentially oscillatory effect.

Specifically, an expression for the explicit rate (ER) value for elastic source j is:

$$\begin{aligned}
ER_j(k) &= u_j(k) + \Delta u_j(k) \\
&= u_j(k) + \frac{\sum_{i=1}^{j-1} \omega_i u_j(k) - \omega_j \sum_{i=1}^{j-1} ER_i(k - \delta_i + \delta_j)}{\sum_{i=j}^{N} \omega_i} \\
&= \frac{u_j(k) - \omega_j \sum_{i=1}^{j-1} ER_i(k - \delta_i + \delta_j)}{\sum_{i=j}^{N} \omega_i} \\
&= \frac{\omega_j \left[\rho C - \hat{R}_b(k + \delta_j) - \sum_{i=1}^{j-1} ER_i(k - \delta_i + \delta_j) \right]}{\sum_{i=j}^{N} \omega_i} .
\end{aligned} \tag{4.15}$$

To simplify the notation, the summation of weights are defined as $W_j = \omega_j + \omega_{j+1} + \ldots + \omega_N$. The explicit rate value is now given by:

$$ER_j(k) = \frac{\omega_j}{W_j} \left[\rho C - \hat{R}_b(k + \delta_j) - \sum_{i=1}^{j-1} ER_i(k - \delta_i + \delta_j) \right] . \tag{4.16}$$

It should be noted that $W_1 = 1$ since the summation of all the weights ω_j is equal to one. This algorithm is labelled PERC+ to differentiate it from the PERC algorithm that only uses the predictive kernel given in Equation (4.9).

PERC+ has been derived for the purpose of compensating for prediction errors that result from the various round trip delays. However, it also distributes a more significant proportion of the instantaneous variations in the available bandwidth to connections with lower latency. This is achieved in conjunction with the prediction process, and applies the principle defined by Hu *et al.* [Hu *et al.* 1998]. This principle states that dynamically available bandwidth should be allocated to connections that can use it in a timely manner. In Hu's research, the sources are simply differentiated into two categories (with short and long latency connections), and a single round-trip delay threshold determining which category the sources belong to. The algorithm proposed here, however, incorporates the explicit values of round trip delays into the control algorithm, in contrast to the coarse granularity in terms of delay differentiation and bandwidth distribution of Hu *et al.*.

4.4 NETWORK PERFORMANCE OF THE RATE CONTROL MECHANISM

The performance of the predictive rate control mechanisms proposed in Section 4.3 is demonstrated in this section through an analytical study and also by simulating the system using actual data. Analytical results for queueing systems with feedback are notoriously difficult to obtain. A simpler approach is used which still sheds significant light on the performance of the system. This approach involves a comparison of the total input load arriving at the congested node with the desired target utilisation ρ of the outgoing link. By analysing the moments of this load, it is demonstrated that the predictive rate control mechanism (which includes the prediction compensation algorithm, ie PERC+) reduces the variance of the errors in bandwidth allocation around the target utilisation level. The performance of other filtering techniques for traffic measurements are also compared with PERC+. These analytical results are verified through simulations which compare the performance these algorithms with PERC+ using actual traffic data, such as the *Star Wars* MPEG data set.

4.4.1 Analytical Performance of the Prediction Compensation Algorithm

In this section, the performance of the prediction compensation algorithm, as incorporated within PERC+ and specified in Equation (4.16), is analytically derived and compared with the basic explicit rate algorithm, PERC, which only uses the predictive kernel in Equation (4.9). This analysis demonstrates the advantage of including the proposed prediction compensation algorithm within the control mechanism.

The background traffic is represented as the stationary process

$$R_b(k) = \mu_b + X(k), \tag{4.17}$$

which separates the mean μ_b of the process from the stochastic bandwidth fluctuations about the mean, modelled by the random variable $\mathbf{X}$. The covariance structure γ of $\mathbf{X}$ is equivalent to the structure of $\mathbf{R}_b$, but $\mathbf{X}$ has zero mean. The predictions of the background traffic is thus:

$$\begin{aligned} \hat{R}_b(k+\delta) &= \mu_b + \hat{X}(k+\delta) \\ &= \mu_b + G'_\delta \mathbf{X}(k) \end{aligned} \tag{4.18}$$

where $G_\delta = \Gamma^{-1}\gamma_\delta$ is the predictor used in the kernel of PERC, and $\mathbf{X}(k)$ is the vector of stored traffic measurements that have been mean-shifted using the process mean μ_b (Equation (4.17)).

4.4.1.1 System Analysis of PERC

Firstly, the system with uncompensated explicit rate calculations, utilising the PERC algorithm, is considered. There, the explicit rate information is only generated using the prediction kernel (4.9):

$$\begin{aligned} ER_j^{\mathbf{u}}(k) &= \omega_j \left[\rho C - \hat{R}_b(k+\delta)\right] \\ &= \omega_j \left(\rho C - \mu_b\right) - \omega_j \hat{X}(k+\delta_j). \end{aligned} \tag{4.19}$$

The superscript $^{\mathbf{u}}$ refers to the uncompensated system that is being analysed. The total input load $I^{\mathbf{u}}(k)$ arriving at the network node consists of the high priority background traffic and the arrival rates of all the controlled sources as follows:

$$\begin{aligned} I^{\mathbf{u}}(k) &= U^{\mathbf{u}}(k) + R_b(k) \\ &= \sum_{j=1}^{N} ER_j^{\mathbf{u}}(k-\delta_j) + R_b(k) \\ &= \rho C + X(k) - \sum_{j=1}^{N} \omega_j \hat{X}_{\delta_j}(k), \end{aligned} \tag{4.20}$$

where $\hat{X}_{\delta_j}(k)$ is the predicted value of $X(k)$ with a δ_j-step prediction horizon. Taking the expected value of the input arrival rate,

$$\mathbf{E}\left[I^{\mathbf{u}}\right] = \rho C, \tag{4.21}$$

it is observed that the desired link utilisation is achieved. Performance is measured by the variance of the input load, which is calculated as follows:

$$\begin{aligned} Var(I^{\mathbf{u}}) &= \mathbf{E}\left[(X(k) - \sum_{j=1}^{N} \omega_j \hat{X}_{\delta_j}(k))^2\right] \\ &= \mathbf{E}\left[(X(k) - \sum_{i=1}^{N} \omega_i G'_{\delta_i}\mathbf{X}(k-\delta_i))(X(k) - \sum_{j=1}^{N} \omega_j G'_{\delta_j}\mathbf{X}(k-\delta_j))'\right] \\ &= \sigma_b^2 - \sum_{i=1}^{N} \omega_i \mathbf{E}\left[X(k)\mathbf{X}'(k-\delta_i)G_{\delta_i}\right] - \sum_{j=1}^{N} \omega_j \mathbf{E}\left[G'_{\delta_j}\mathbf{X}(k-\delta_j)X(k)\right] \\ &\quad + \sum_{i=1}^{N}\sum_{j=1}^{N} \omega_i\omega_j \mathbf{E}\left[G'_{\delta_i}\mathbf{X}(k-\delta_i)\mathbf{X}'(k-\delta_j)G_{\delta_j}\right] \\ &= \sigma_b^2 - 2\sum_{j=1}^{N} \omega_j \gamma'_{\delta_j}\Gamma^{-1}\gamma_{\delta_j} + \sum_{i=1}^{N}\sum_{j=1}^{N} \omega_i\omega_j \gamma'_{\delta_i}\Gamma^{-1}\Gamma_{\delta_i\delta_j}\Gamma^{-1}\gamma_{\delta_j}, \end{aligned} \tag{4.22}$$

where the matrix $\Gamma_{\delta_i\delta_j}$ is defined by autocovariance function of the process $\mathbf{X}$:

$$\Gamma_{\delta_i\delta_j} = \mathbf{E}\left[\mathbf{X}(k-\delta_i)\mathbf{X}'(k-\delta_j)\right]$$
$$= \begin{bmatrix} \gamma(\delta_i-\delta_j) & \gamma(\delta_i-\delta_j+1) & \dots & \gamma(\delta_i-\delta_j+M-1) \\ \gamma(\delta_i-\delta_j-1) & \gamma(\delta_i-\delta_j) & \dots & \gamma(\delta_i-\delta_j+M-2) \\ \vdots & \vdots & \ddots & \vdots \\ \gamma(\delta_i-\delta_j-M+1) & \gamma(\delta_i-\delta_j-M+2) & \dots & \gamma(\delta_i-\delta_j) \end{bmatrix}, \tag{4.23}$$

with $i,j = 1,\dots,N$.

4.4.1.2 System Analysis of PERC+

The performance of the system which uses the prediction compensation algorithm, incorporated in PERC+, is now analysed. The explicit rate values are calculated using Equation (4.15), which is an algorithm based on recursion. To determine the variance of the total input load for the compensated system, this rate calculation needs to be expressed in terms of the variable $\mathbf{X}$, as follows:

$$ER_j^{\mathrm{c}}(k) = \omega_j\left(\rho C - \mu_b\right) - \frac{\omega_j \hat{X}_{\delta_j}(k+\delta_j)}{W_j} + \sum_{\substack{i=1\\ j\geq 2}}^{j-1} \frac{\omega_j\omega_i\hat{X}_{\delta_i}(k+\delta_j)}{W_{i+1}W_i}, \tag{4.24}$$

where the superscript $^{\mathrm{c}}$ denotes the system using prediction compensation. The proof of this result uses an inductive argument as follows. The case for $j=1$ (the connection with the largest delay) is a special case, which is handled separately:

$$\begin{aligned} ER_1(k) &= \omega_1\left(\rho C - \hat{R}_b(k+\delta_1)\right) \\ &= \omega_1\left(\rho C - [\mu_b + \hat{X}_{\delta_1}(k+\delta_1)]\right) \\ &= \omega_1(\rho C - \mu_b) - \omega_1\hat{X}_{\delta_1}(k+\delta_1). \end{aligned} \tag{4.25}$$

The case for $j=2$ also needs to calculated:

$$\begin{aligned} ER_2(k) &= \frac{\omega_2}{W_2}\left[\rho C - \hat{R}_b(k+\delta_2) - ER_1(k-\delta_1+\delta_2)\right] \\ &= \frac{\omega_2}{W_2}\left[\rho C - \mu_b - \hat{X}_{\delta_2}(k+\delta_2) - \omega_1(\rho C-\mu_b) - \omega_1\hat{X}_{\delta_1}(k+\delta_2)\right] \\ &= \frac{\omega_2(1-\omega_1)(\rho C-\mu_b)}{W_2} - \frac{\omega_2\hat{X}_{\delta_2}(k+\delta_2)}{W_2} + \frac{\omega_2\omega_1\hat{X}_{\delta_1}(k+\delta_2)}{W_2} \\ &= \omega_2(\rho C-\mu_b) - \frac{\omega_2\hat{X}_{\delta_2}(k+\delta_2)}{W_2} + \frac{\omega_2\omega_1\hat{X}_{\delta_1}(k+\delta_2)}{W_1W_2}.. \end{aligned} \tag{4.26}$$

In this derivation the identity $W_1 = 1$ twice has been used twice. Next, it is assumed that (4.24) holds for all sources $j > 1$. The explicit rate value for the subsequent source $j+1$ is calculated as:

$$
\begin{aligned}
&ER_{j+1}(k) \\
&= \frac{\omega_{j+1}}{W_{j+1}} \left[\rho C - \hat{R}_b(k+\delta_{j+1}) - \sum_{i=1}^{j} ER_i(k-\delta_i+\delta_{j+1}) \right] \\
&= \frac{\omega_{j+1}}{W_{j+1}} \Bigg[\rho C - \mu_b - \hat{X}_{\delta_{j+1}}(k+\delta_{j+1}) \\
&\quad - \sum_{i=1}^{j} \left\{ \omega_i(\rho C - \mu_b) - \frac{\omega_i \hat{X}_{\delta_i}(k+\delta_{j+1})}{W_i} + \sum_{\substack{m=1\\ i\geq 2}}^{i-1} \frac{\omega_i \omega_m \hat{X}_{\delta_m}(k+\delta_{j+1})}{W_{m+1}W_m} \right\} \Bigg] \\
&= \frac{\omega_{j+1}}{W_{j+1}} \Bigg[\left(1 - \sum_{i=1}^{j} \omega_i\right) (\rho C - \mu_b) - \hat{X}_{\delta_{j+1}}(k+\delta_{j+1}) \\
&\quad + \sum_{i=1}^{j} \left\{ \frac{\omega_i \hat{X}_{\delta_i}(k+\delta_{j+1})}{W_i} - \sum_{\substack{m=1\\ i\geq 2}}^{i-1} \frac{\omega_i \omega_m \hat{X}_{\delta_m}(k+\delta_{j+1})}{W_{m+1}W_m} \right\} \Bigg] \\
&= \frac{\omega_{j+1}}{W_{j+1}} \Bigg[W_{j+1}(\rho C - \mu_b) - \hat{X}_{\delta_{j+1}}(k+\delta_{j+1}) + \\
&\Bigg\{ \tfrac{\omega_1 \hat{X}_{\delta_1}(k+\delta_{j+1})}{W_1} \\
&- \tfrac{\omega_2 \omega_1 \hat{X}_{\delta_1}(k+\delta_{j+1})}{W_2 W_1} \quad + \tfrac{\omega_2 \hat{X}_{\delta_2}(k+\delta_{j+1})}{W_2} \\
&- \tfrac{\omega_3 \omega_1 \hat{X}_{\delta_1}(k+\delta_{j+1})}{W_2 W_1} \quad - \tfrac{\omega_3 \omega_2 \hat{X}_{\delta_2}(k+\delta_{j+1})}{W_3 W_2} \quad + \tfrac{\omega_3 \hat{X}_{\delta_3}(k+\delta_{j+1})}{W_3} \\
&\vdots \\
&- \tfrac{\omega_{j-1} \omega_1 \hat{X}_{\delta_1}(k+\delta_{j+1})}{W_2 W_1} \quad - \tfrac{\omega_{j-1} \omega_2 \hat{X}_{\delta_2}(k+\delta_{j+1})}{W_3 W_2} \quad \cdots \quad + \tfrac{\omega_{j-1} \hat{X}_{\delta_{j-1}}(k+\delta_{j+1})}{W_{j-1}} \\
&- \tfrac{\omega_j \omega_1 \hat{X}_{\delta_1}(k+\delta_{j+1})}{W_2 W_1} \quad - \tfrac{\omega_j \omega_2 \hat{X}_{\delta_2}(k+\delta_{j+1})}{W_3 W_2} \quad \cdots \quad - \tfrac{\omega_j \omega_{j-1} \hat{X}_{\delta_{j-1}}(k+\delta_{j+1})}{W_j W_{j-1}} \quad + \tfrac{\omega_j \hat{X}_{\delta_j}(k+\delta_{j+1})}{W_j} \Bigg\} \Bigg] \\
&= \omega_{j+1}(\rho C - \mu_b) - \frac{\omega_{j+1} \hat{X}_{\delta_{j+1}}(k+\delta_{j+1})}{W_{j+1}} + \frac{\omega_{j+1}}{W_{j+1}} \sum_{i=1}^{j} \frac{\omega_i \hat{X}_{\delta_i}(k+\delta_{j+1})}{W_i} \left(1 - \frac{1}{W_{i+1}} \sum_{m=i+1}^{j} \omega_m \right) \\
&= \omega_{j+1}(\rho C - \mu_b) - \frac{\omega_{j+1} \hat{X}_{\delta_{j+1}}(k+\delta_{j+1})}{W_{j+1}} + \frac{\omega_{j+1}}{W_{j+1}} \sum_{i=1}^{j} \frac{\omega_i \hat{X}_{\delta_i}(k+\delta_{j+1})}{W_{i+1} W_i} \left(W_{i+1} - \sum_{m=i+1}^{j} \omega_m \right) \\
&= \omega_{j+1}(\rho C - \mu_b) - \frac{\omega_{j+1} \hat{X}_{\delta_{j+1}}(k+\delta_{j+1})}{W_{j+1}} + \frac{\omega_{j+1}}{W_{j+1}} \sum_{i=1}^{j} \frac{\omega_i \hat{X}_{\delta_i}(k+\delta_{j+1})}{W_{i+1} W_i} \cdot W_{j+1} \\
&= \omega_{j+1}(\rho C - \mu_b) - \frac{\omega_{j+1} \hat{X}_{\delta_{j+1}}(k+\delta_{j+1})}{W_{j+1}} + \sum_{i=1}^{j} \frac{\omega_{j+1} \omega_i \hat{X}_{\delta_i}(k+\delta_{j+1})}{W_{i+1} W_i},
\end{aligned}
\tag{4.27}
$$

which shows that (4.24) holds for all $2 \leq j \leq N$ for any number of sources N.

Once the non-recursive expression for the ER values has been derived, the total input traffic load for the prediction compensated system can be calculated, using a similar reasoning:

$$
\begin{aligned}
I^{c}(k) &= R_b(k) + U^{c}(k) \\
&= R_b(k) + \sum_{j=1}^{N} ER^{c}_{j}(t-\delta_j) \\
&= \mu_b + X(k) + \sum_{j=1}^{N} \left\{ \omega_j(\rho C - \mu_b) - \frac{\omega_j \hat{X}_{\delta_j}(k)}{W_j} + \sum_{\substack{i=1 \\ j\geq 2}}^{j-1} \frac{\omega_j \omega_i \hat{X}_{\delta_i}(k)}{W_{i+1} W_i} \right\} \\
&= \mu_b + X(k) + (\rho C - \mu_b) \sum_{j=1}^{N} \omega_j + \\
&\quad \begin{Bmatrix}
-\frac{\omega_1 \hat{X}_{\delta_1}(k)}{W_1} & & & & \\
+\frac{\omega_2\omega_1 \hat{X}_{\delta_1}(k)}{W_2 W_1} & -\frac{\omega_2 \hat{X}_{\delta_2}(k)}{W_2} & & & \\
+\frac{\omega_3\omega_1 \hat{X}_{\delta_1}(k)}{W_2 W_1} & +\frac{\omega_3\omega_2 \hat{X}_{\delta_2}(k)}{W_3 W_2} & -\frac{\omega_3 \hat{X}_{\delta_3}(k)}{W_3} & & \\
\vdots & & & & \\
+\frac{\omega_{N-1}\omega_1 \hat{X}_{\delta_1}(k)}{W_2 W_1} & +\frac{\omega_{N-1}\omega_2 \hat{X}_{\delta_2}(k)}{W_3 W_2} & \cdots & -\frac{\omega_{N-1} \hat{X}_{\delta_{N-1}}(k)}{W_{N-1}} & \\
+\frac{\omega_N\omega_1 \hat{X}_{\delta_1}(k)}{W_2 W_1} & +\frac{\omega_N\omega_2 \hat{X}_{\delta_2}(k)}{W_3 W_2} & \cdots & +\frac{\omega_N\omega_{N-1} \hat{X}_{\delta_{N-1}}(k)}{W_N W_{N-1}} & +\frac{\omega_N \hat{X}_{\delta_N}(k)}{W_N}
\end{Bmatrix} \\
&= \rho C + X(k) + \sum_{j=1}^{N-1} \frac{\omega_j \hat{X}_{\delta_j}(k)}{W_j} \left(-1 + \frac{1}{W_{j+1}} \sum_{i=j+1}^{N} \omega_i \right) - \frac{\omega_N \hat{X}_{\delta_N}(k)}{W_N} \\
&= \rho C + X(k) + \sum_{j=1}^{N-1} \frac{\omega_j \hat{X}_{\delta_j}(k)}{W_j} \left(-1 + \frac{1}{W_{j+1}} \cdot W_{j+1} \right) - \frac{\omega_N \hat{X}_{\delta_N}(k)}{\omega_N} \\
&= \rho C + X(k) - \hat{X}_{\delta_N}(k)
\end{aligned}
\tag{4.28}
$$

The expected value of this compensated system is

$$
\mathbf{E}\left[I^{c}\right] = \rho C, \tag{4.29}
$$

and the variance is equivalent to the variance of the prediction error (3.50) for the predictor with smallest prediction horizon:

$$
\begin{aligned}
Var(I^{c}) &= \mathbf{E}\left[(X(k) - \hat{X}_{\delta_N}(k)) \right] \\
&= \sigma_b^2 - \gamma'_{\delta_N} \Gamma^{-1} \gamma_{\delta_N}.
\end{aligned}
\tag{4.30}
$$

This is a surprisingly simple result, considering that it only depends on the performance of the predictor for the connection with the smallest round trip delay. This holds because the prediction errors for the longer latency connections have been successively absorbed by the control information returned to shorter latency connections until the connection with smallest round trip delay is reached. From the perspective of the congested node, the system appears to be only dealing with a single controlled source with a connection delay equal to the smallest latency in the system. It should be noted that this "virtual" source appears to have a transmission rate with a variance multiplied by a factor of N, compared with any other actual source in the system. However, since this virtual source is located such that the round trip delay is the minimum amongst the collection of actual sources, the overall variance of the input traffic is significantly reduced, as compared with the system that does not use prediction compensation.

4.4.1.3 Analytical Performance Comparison

The performance of the PERC+ can be directly quantified using the relative variance of the input arrival load, which gives Theorem 1.

Theorem 1 *The relative variance of the input arrival load to the system using the prediction compensation algorithm (PERC+) compared with the system without prediction compensation is:*

$$Rel.\,Var. = \frac{\sigma_b^2 - \gamma'_{\delta_N}\Gamma^{-1}\gamma_{\delta_N}}{\sigma_b^2 - 2\sum_{j=1}^{N}\omega_j\gamma'_{\delta_j}\Gamma^{-1}\gamma_{\delta_j} + \sum_{i=1}^{N}\sum_{j=1}^{N}\omega_i\omega_j\gamma'_{\delta_i}\Gamma^{-1}\Gamma_{\delta_i\delta_j}\Gamma^{-1}\gamma_{\delta_j}}. \qquad (4.31)$$

Using fractional Gaussian noise as a stochastic model for the background traffic, the surface of the relative variance, as the number of rate-controlled sources and the Hurst parameter is varied, is shown in Figure 4.2. As additional rate-controlled sources are added to the system, the connection associated with the new source has a round-trip delay which is proportional to the total number of sources. Thus, the range of round-trip delays increases proportionally as the number of sources increases. For simplicity, the available bandwidth is shared fairly between all sources at the node, which means that the weights assigned to each connection are equal for a given number of connections.

Two main conclusions can be drawn from the figure. Firstly, there is no performance gain from incorporating the prediction compensation algorithm with traffic which has Hurst parameter close to 0.5. The first conclusions follows from the independence of future traffic levels on previous measurements (after any short-range dependence has been taken into account), so that predictions with smaller horizons have no greater accuracy than predictions required for connections with greater round-trip delay. The second conclusion is that the advantage of using the prediction compensation algorithm

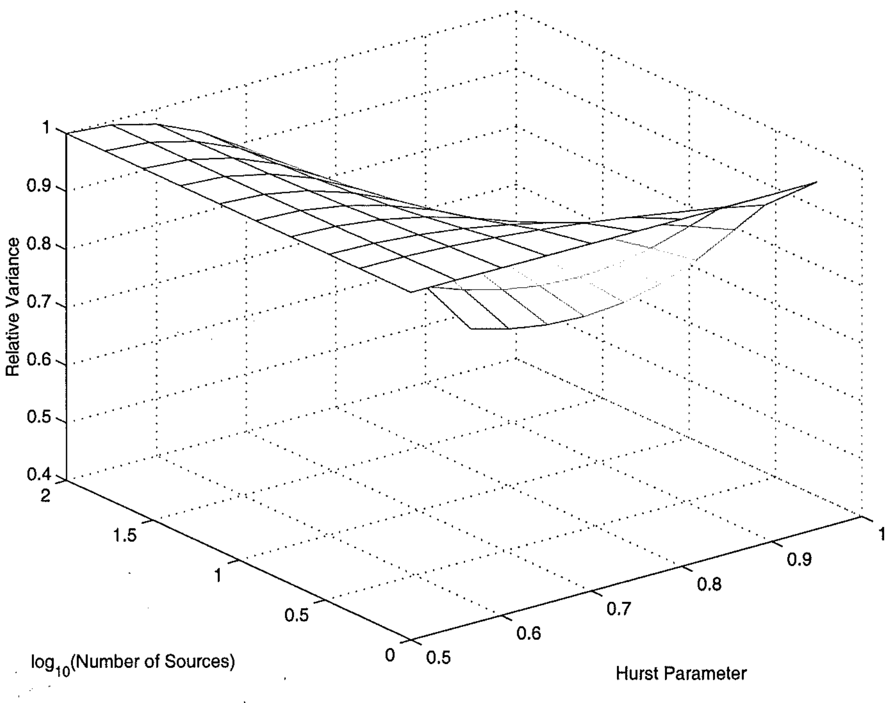


Figure 4.2 The performance of the prediction compensation algorithm.

PERC+ becomes more significant as the self-similarity of the background network traffic increases the range of round-trip delays.

4.4.1.4 Performance Comparison with General Filtering Techniques

The performance of the complete control mechanism including the prediction compensation algorithm can also be compared with other schemes based on filtering measurements of the background traffic. A scheme which uses low-pass filters has been proposed by Zhao *et al.* [Zhao *et al.* 1997], who develop an ABR control mechanism based on using a Hamming window. The motivation for using low-pass filtering has arisen from the research work of San-qi Li [Li *et al.* 1995], who show that network performance is primarily dominated by the low frequency behaviour of the input traffic below a cutoff frequency ω_c. The performance of this approach is compared with PERC+ in the following analysis. For generality, an arbitrary filter is represented by the symbol $\bar{G}$. The filtered traffic variable is then given by:

$$\begin{aligned}\bar{R}_b(k) &= \bar{G}'\mathbf{R}_b(k) \\ &= \bar{G}'\left(\mu \cdot \mathbf{1} + \mathbf{X}(k)\right) \\ &= \mu_b \bar{G}'\mathbf{1} + \bar{G}'\mathbf{X}(k).\end{aligned} \tag{4.32}$$

The explicit rate information which is returned to source j is derived from this filtered variable as follows:

$$\begin{aligned} ER_j(k) &= \omega_j \left[\rho C - \bar{R}_b(k)\right] \\ &= \omega_j \left[\rho C - \mu_b \bar{G}'\mathbf{1} - \bar{G}'\mathbf{X}(k)\right]. \end{aligned} \tag{4.33}$$

The total input load on the congested network node is, as before, the multiplexed high priority traffic and the total input rates from the controlled sources:

$$\begin{aligned} I^{\mathrm{g}}(k) &= R_b(k) + U(k) \\ &= R_b(k) + \sum_{j=1}^{N} ER_j(k-\delta_j) \\ &= \mu_b + X(k) + \sum_{j=1}^{N} \omega_j \left[\rho C - \mu_b \bar{G}'\mathbf{1} - \bar{G}'\mathbf{X}(k-\delta_j)\right] \\ &= \rho C + \mu_b - \mu_b \bar{G}'\mathbf{1} + X(k) - \sum_{j=1}^{N} \omega_j \bar{G}'\mathbf{X}(k-\delta_j). \end{aligned} \tag{4.34}$$

From this result, the mean input load is found to be

$$\mathbf{E}\left[I^{\mathrm{g}}\right] = \rho C + \mu_b - \mu_b \bar{G}'\mathbf{1}. \tag{4.35}$$

Thus, to achieve the target utilisation ρ of the link capacity the filter has the constraint $\bar{G}'\mathbf{1} = 1$. In other words, the filter coefficients must sum to unity. The variance of the total input load can also be calculated, using a similar approach to that used to derive Equation (4.22):

$$\begin{aligned} Var(I^{\mathrm{g}}) &= \mathbf{E}\left[\left(X(k) - \sum_{j=1}^{N} \omega_j \bar{G}'\mathbf{X}(k-\delta_j)\right)^2\right] \\ &= \sigma_b^2 - 2\sum_{j=1}^{N} \omega_j \gamma'_{\delta_j} \bar{G} + \sum_{i=1}^{N}\sum_{j=1}^{N} \omega_i \omega_j \bar{G}' \Gamma_{\delta_i \delta_j} \bar{G}, \end{aligned} \tag{4.36}$$

where the matrix $\Gamma_{\delta_i \delta_j}$ is defined in (4.23). The general measure of performance in the form of the relative variance is defined in Theorem 2.

Theorem 2 *The relative variance of the input arrival load to the system using PERC+ compared with the system using an arbitrary filter $\bar{G}$ is:*

$$Rel.\,Var. = \frac{\sigma_b^2 - \gamma'_{\delta_N}\Gamma^{-1}\gamma_{\delta_N}}{\sigma_b^2 - 2\sum_{j=1}^{N} \omega_j \gamma'_{\delta_j}\bar{G} + \sum_{i=1}^{N}\sum_{j=1}^{N} \omega_i\omega_j \bar{G}'\Gamma_{\delta_i\delta_j}\bar{G}} \tag{4.37}$$

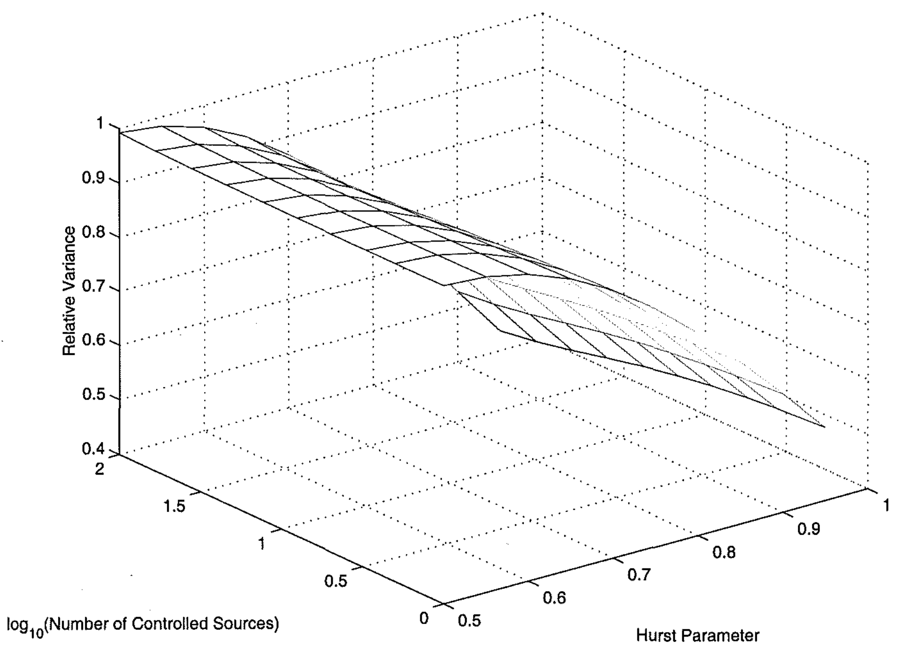


Figure 4.3 Comparison between performance of the controller using an averaging filter and the system using PERC+.

The sample mean is the simplest filter which can be used, with $\bar{G} = \frac{1}{M} \cdot \mathbf{1}$, which has a rectangular window such that the sum of filter coefficients is 1. This filter has the advantage of having minimal computational expense and can be easily implemented within an ABR rate control mechanism in a switch. In Figure 4.3 the performance of a traffic controller, using this averaging filter, is compared with PERC+. The surface shows that PERC+ performs significantly better as the Hurst parameter increases, regardless of the number of controlled sources within the system. Figure 4.3 emphasises the advantage of using the predictive rate control algorithm when the high priority traffic is known to be highly self-similar in nature.

From a signal processing perspective, using a rectangular window in the filtering process for the background traffic measurements is not ideal. This is due to the well-known effect called the Gibbs phenomenon, wherein a discontinuity that occurs at the edge of the filter in the time domain. In the frequency domain, this results in significant side lobes in the spectrum of the filter. The Gibbs phenomenon can be suppressed by using different windowing techniques. A standard window used in FIR filter design is the Hamming window, which has the expression:

$$w_k = 0.54 + 0.46 \cos(\frac{k\pi}{M}), \tag{4.38}$$

where M is the length of the filter. Zhao and San-qi Li [Zhao *et al.* 1997, Li *et al.* 1995] use this type of filter, and the performance of a Hamming window filter is compared with PERC+ in Figure 4.4. While there is slight variation in the relative variance as

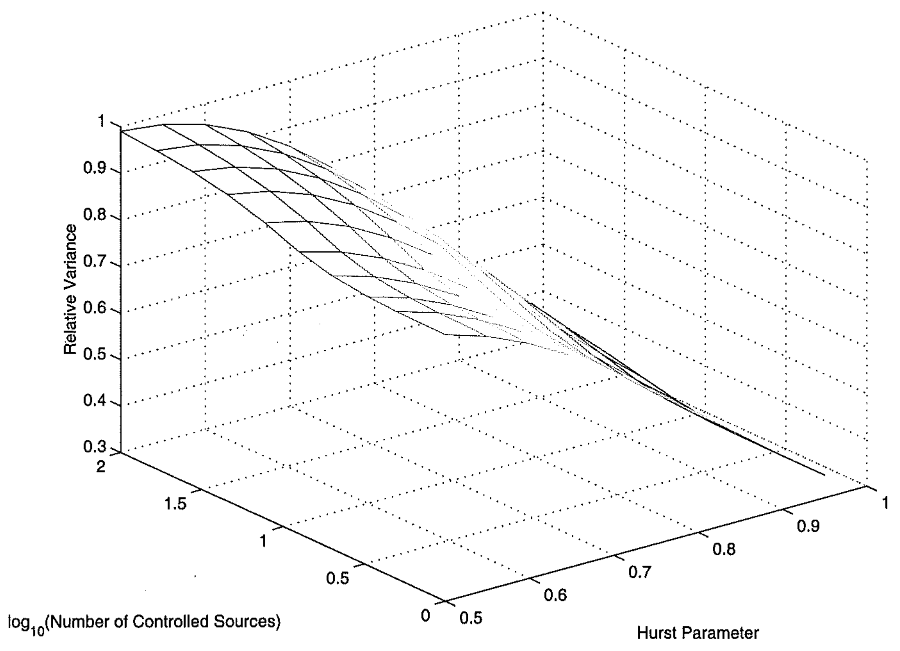


Figure 4.4 Comparison between performance of the controller using an Hamming window FIR filter and the system using PERC+.

the number of sources increases, the conclusion is that PERC+ achieves better network performance as the Hurst parameter increases due to incorporation of self-similarity within the control algorithm. In Figure 4.5, the effect of varying the cutoff frequency of the FIR filter is investigated. Figure 4.5 shows that a simple change in the cutoff frequency cannot improve the performance of the system using the low pass filter. The filter mechanism must incorporate the correlation structure of the long-range dependent traffic to achieve any performance gain.

4.4.2 Simulation Study of System Performance

4.4.2.1 Simulation Model

The simulation model that is used to study PERC+ is shown in Figure 4.6. This model includes a set of controlled sources and a source generating high priority traffic. Such sources are competing for bandwidth on a congested link which is returning explicit rate control information, regarding the available bandwidth on the link, to the controlled sources. Equal weights are assigned to the bandwidth allocated to each connection so as to achieve fair sharing at the node. The controlled sources each have their own round trip delay δ and have been ordered according to this delay.

It is anticipated that video, imaging and multimedia traffic, all of which include a high proportion of graphics and video clips, will become the traffic classes that dominate high priority network services. As a result, the dynamics of the individual sources and

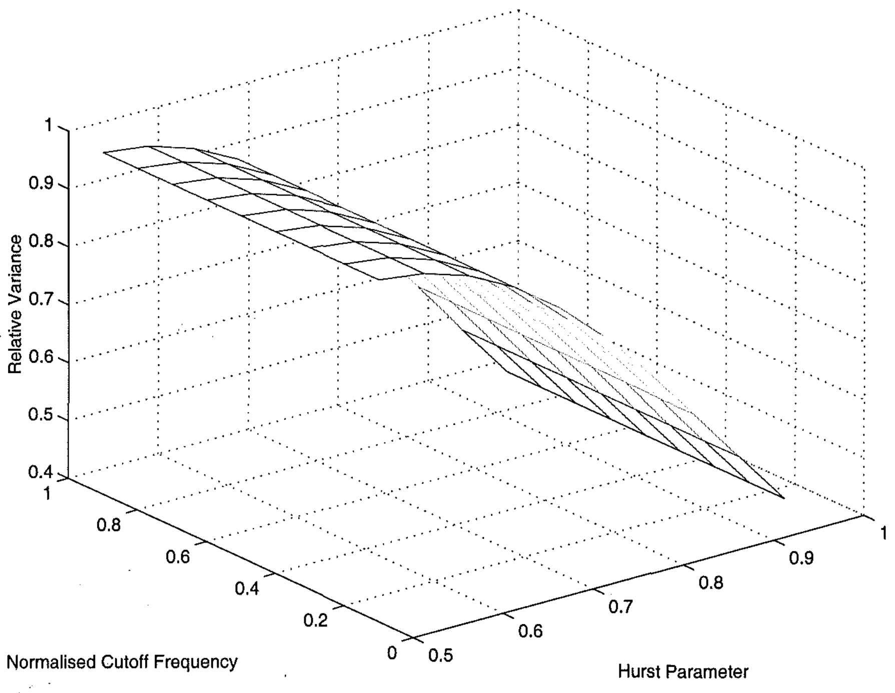


Figure 4.5 The effect of varying the cutoff frequency of the FIR filter, with Hamming windowing, on the comparative performance of the control system.

the queue at the network node is investigated using a data set that has been generated by aggregrating ten randomly shifted MPEG *Star Wars* video streams [Garrett and Willinger 1994]. The latter have been filtered to remove the Group of Picture (GOP) frame sequence [Chiruvolu and Sankar 1997, Liew and Tse 1998, Adas 1998]—a sequence that introduces a pattern into the correlation structure of the data set which is unnecessary in this study. The impact of the GOP structure of an MPEG stream is not the point of the study. Also it is unlikely that this level of granularity will be observed at high-speed backbone switches.

For the purpose of developing the predictors required in the control system, the parameters of a fractional ARIMA$(1, d, 0)$ model were estimated giving the following model:

$$\left(1 - 0.258z^{-1}\right) \Delta^{0.40}(R_b(k) - 3.74 \times 10^6) = \epsilon(k) \tag{4.39}$$

where $\{\epsilon(k)\}$ is a white noise process with zero mean and variance $\sigma_\epsilon^2 = 1.92 \times 10^{10}$. The switch has memory allocation for 64 samples of background traffic. In other words, the memory length of the digital predictors $M = 64$. The link capacity is set at 7.48Mbits/s, which means that the background traffic is using, on average, 50% of the link capacity, and the target utilisation of the system is 95%.

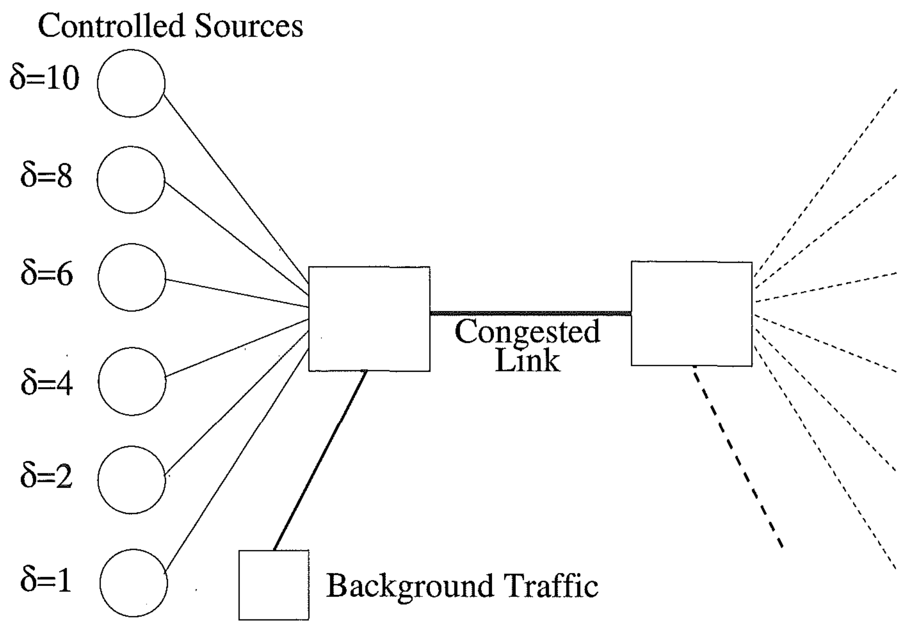


Figure 4.6 The network model for the simulation study.

4.4.2.2 Comparison of Source Transmission Rates

The transmission rates for sources controlled by PERC+ are shown in Figure 4.7. Two observations can be made from these simulation results—(1) the mean transmission rate is equivalent for each source (since the algorithm allocates the bandwidth fairly on average), however (2) the variability of the source rate increases as the round-trip delay of the connection decreases. Observation (2) follows because each connection has a predictor that is matched to round trip delay. The output for the predictors for high latency connections approach the sample mean. In contrast, predictors for low latency connections track the instantaneous bandwidth variations more closely, thus allowing more of the high frequency component of the traffic measurements to appear in the explicit rate information. The other reason for the higher variability is the action of the prediction compensation algorithm incorporated in PERC+. This algorithm adjusts the rate information for connections with low latency based on past control information for the connections with longer round-trip delay.

The dynamics of the individual controlled sources in a system using a rate control algorithm that incorporates a low-pass filter algorithm is shown in Figure 4.8. Figure 4.8 confirms that the network is returning identical rate information to all the sources regardless of their round-trip delay. As will be observed from other network performance parameters, rate information that is non-discriminating, with regards to delay, results in worse performance. This performance degradation is due to the fact that high latency connections will transmit at rates which are more "out-of-date", thus having more significant impact on the level of congestion within the network. In contrast, PERC+ allocates rapidly varying, instantaneous bandwidth to connections with shorter round-

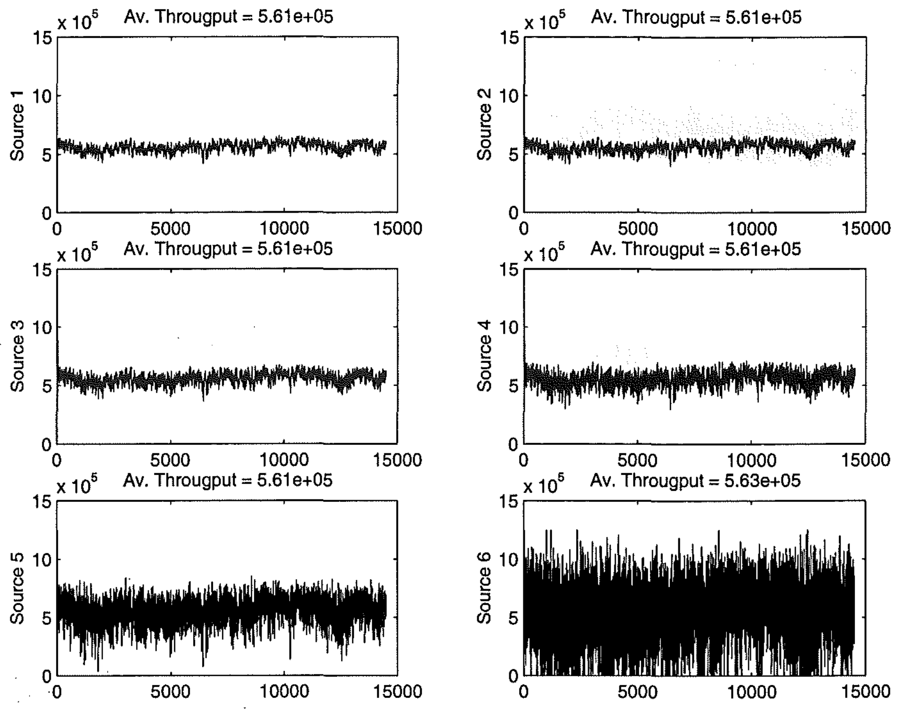


Figure 4.7 Source transmission rates for PERC+.

trip delay which can react faster and in time. On average, the high latency connections are allocated their fair share of the available bandwidth, but with less variable allocation as they cannot react fast enough to rapid changes that occur within the available bandwidth. These results are compared with the transmission rates of ERICA, as shown in Figure 4.9. ERICA also disregards the latency of the connection when it allocates the available bandwidth to connections, as no latency information is used in ERICA.

4.4.2.3 Comparison of Network Performance

Performance parameters of the network include the dynamics of the queue length at the network node and queueing delays experienced by cells, the utilisation of the outgoing link and the cell loss rate, which relate to the aggregate traffic load arriving at the node. Thus, the dynamics of the individual source rates are less significant, if their aggregate results in better network performance. The dynamics of queue length are compared in Figure 4.10 for three systems: PERC+, the system using the low-pass filter and the ERICA rate-control mechanism. These performance parameters are shown in Table 4.1.

It can be concluded that when the long-range dependence of uncontrolled, high priority traffic is incorporated into the control mechanism for controlled connections, there is significantly lower queue occupancy. The overall throughput has not, however, suffered, as can be seen from the link utilisation results which are roughly equivalent for all three systems. Since long queue lengths mean significant delays for the cells arriving

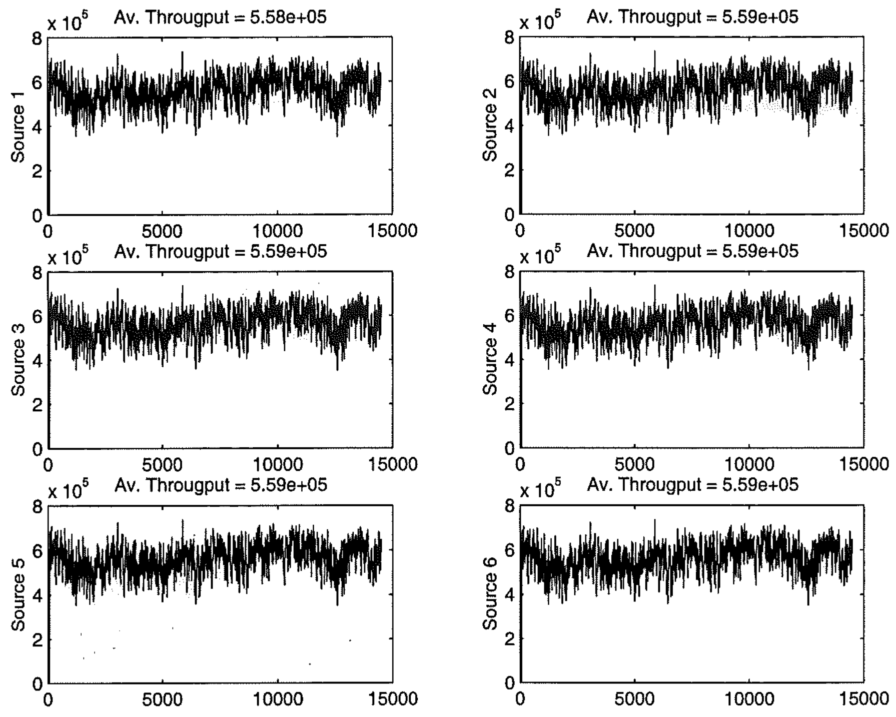


Figure 4.8 Source transmission rates for low-pass filtering algorithm.

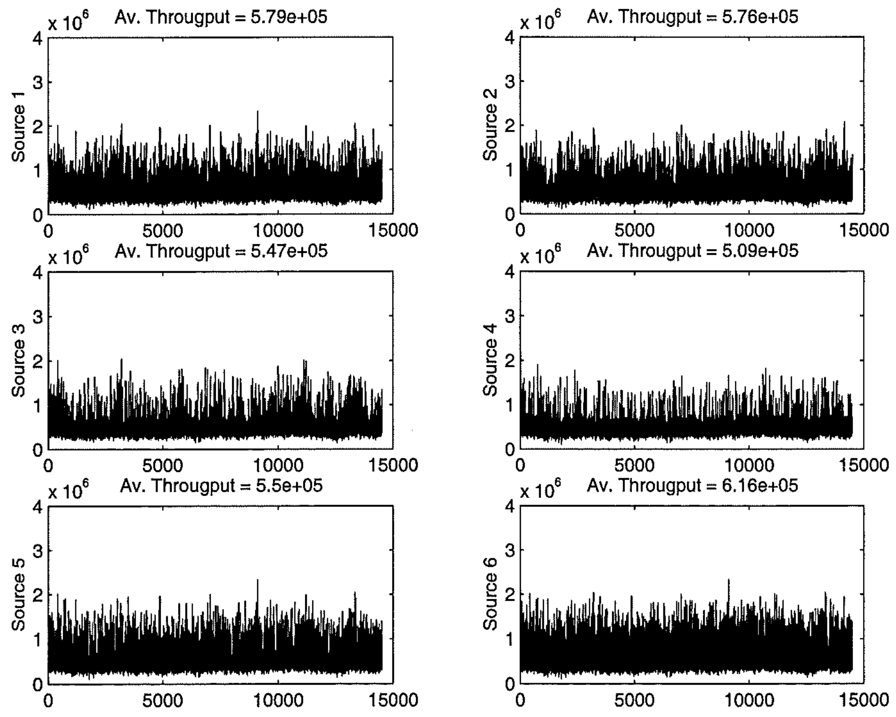


Figure 4.9 Source transmission rates for ERICA.

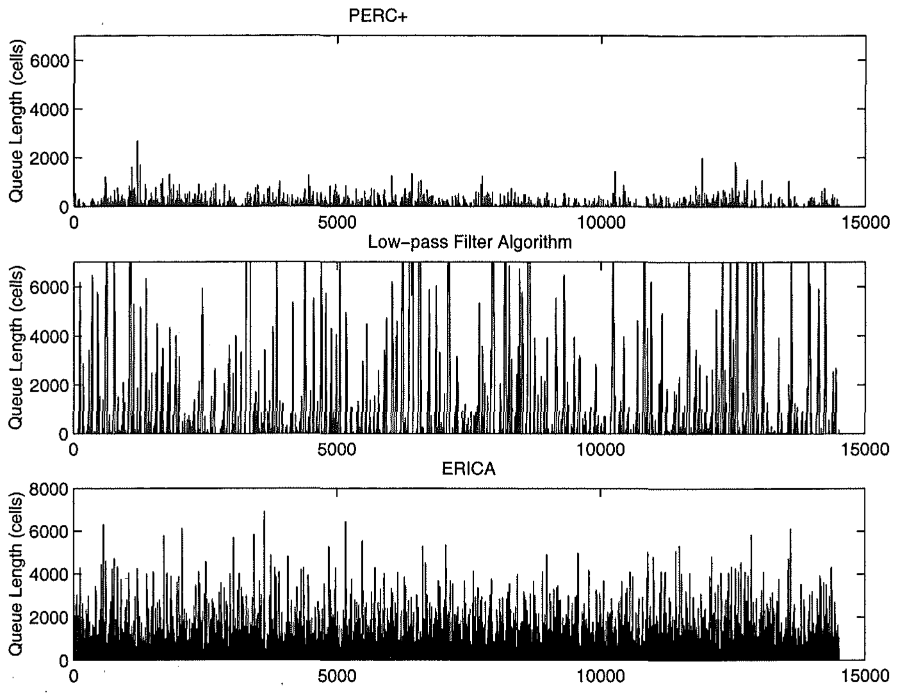


Figure 4.10 Comparison between the buffer requirements of PERC+, low-pass filter schemes and ERICA ($\rho = 0.9$)

at the node, PERC+ has effectively reduced queueing delays within the network.

For buffers with finite size, this results in lower cell loss rates, as is demonstrated in Figure 4.11. In the simulations which generated Figure 4.11, the buffer size was varied and cell loss rate was measured for different network utilisation levels. There is a fairly rapid decrease in cell loss rate as the buffer size is increased, in the case where the network uses PERC+. However, for the LPF scheme, the standard hyperbolic decrease in losses is observed. This demonstrates that PERC+ has effectively addressed one of the key problems which network engineers face with self-similar traffic, namely that sustained losses occur and that these losses do not diminish with a simple increase in the buffer size.

The *Star Wars* data set is often used as a typical VBR video data set [Garrett and Willinger 1994, Chiruvolu and Sankar 1997, Zhang and Yang 1997] which displays self-similarity. As such, it only represents one degree of burstiness as represented by its Hurst parameter. In fact, the performance of the proposed rate control algorithm over a range of Hurst parameters is of interest. To investigate a wider range of long-range dependence, self-similar data sets have been generated for Hurst parameters in the interval $H \in [0.5, 1.0)$ using the Random Midpoint Displacement algorithm [Crilly *et al.* 1991]. These data sets have been shifted and scaled so that they have the same mean and variance as the *Star Wars* data set. Only the Hurst parameter differs between each set. Then, using the same simulation model, PERC+ is compared with the low-pass filter algorithm and ERICA, using the performance metrics shown in Figures 4.12–

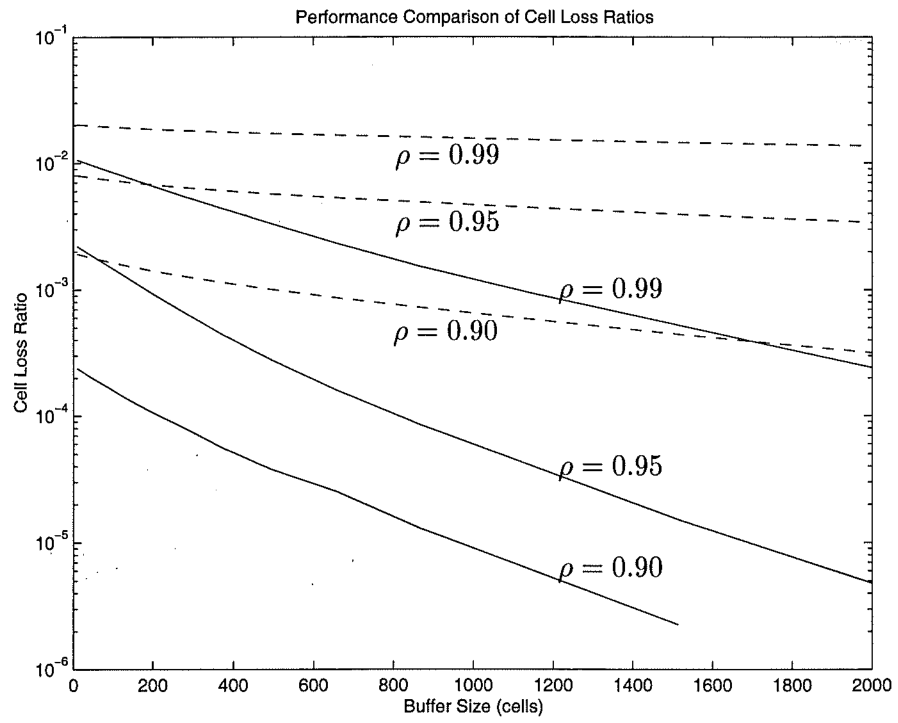


Figure 4.11 Comparison between the cell loss rates of PERC+ (solid) and low-pass filter scheme (dashed) for different link utilizations ρ.

Table 4.1 Comparison of the performance metrics for the different ABR algorithms.

Performance Parameter	**PERC+**	**LPF Algorithm**	**ERICA**
Queue Length Statistics			
Mean Length (cells)	30	1030	570
Maximum Length (cells)	2.68e3	2.62e4	6.91e3
Variance (cells^2)	1.39e4	6.49e6	8.49e5
Link Utilisation Statistics			
Mean Utilisation	0.948	0.947	0.944
Maximum Utilisation	1.00	1.00	1.00
Variance	0.0015	0.0039	0.0048

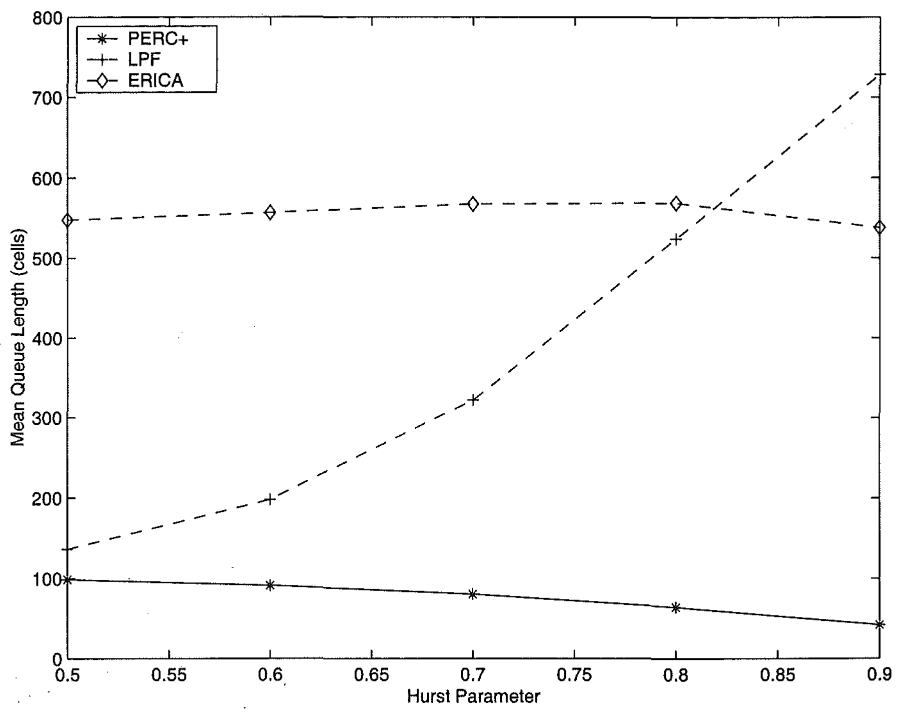


Figure 4.12 Comparing rate-control algorithms using mean queue length.

4.15.

As before, two different situations were investigated—the first being the system with infinite buffer capacity. From these simulations using an infinite buffer, the dynamics of the queue length can be investigated (Figure 4.12), with the utilisation of the link shown in Figure 4.13. What is observed from these graphs is that PERC+ consistently maintains a low buffer occupancy with insignificant variation in the link utilisation. In fact, the buffer requirements improve slightly as the Hurst parameter increases. In comparison, the low-pass filter algorithm has significantly increasing buffer requirements, and ERICA has reasonably long queue lengths but with little variation.

The other simulation scenario that has been investigated uses a finite buffer at the network node, with the cell loss ratio and the utilisation shown in Figures 4.14 and 4.15. PERC+ shows decreasing cell loss rates with link utilisations that are always higher than the other algorithms. The other algorithms show relatively high cell loss rates. It can then be concluded that, as the self-similarity of the network traffic varies, the predictive rate algorithm proposed in this chapter performs well in comparison with other rate control mechanisms.

4.4.2.4 Performance Gain Using Prediction Compensation Algorithm

In the analysis of the performance of PERC+ in Section 4.4.1.3, it has been demonstrated that the algorithm improves the performance of the system as both the range of round-

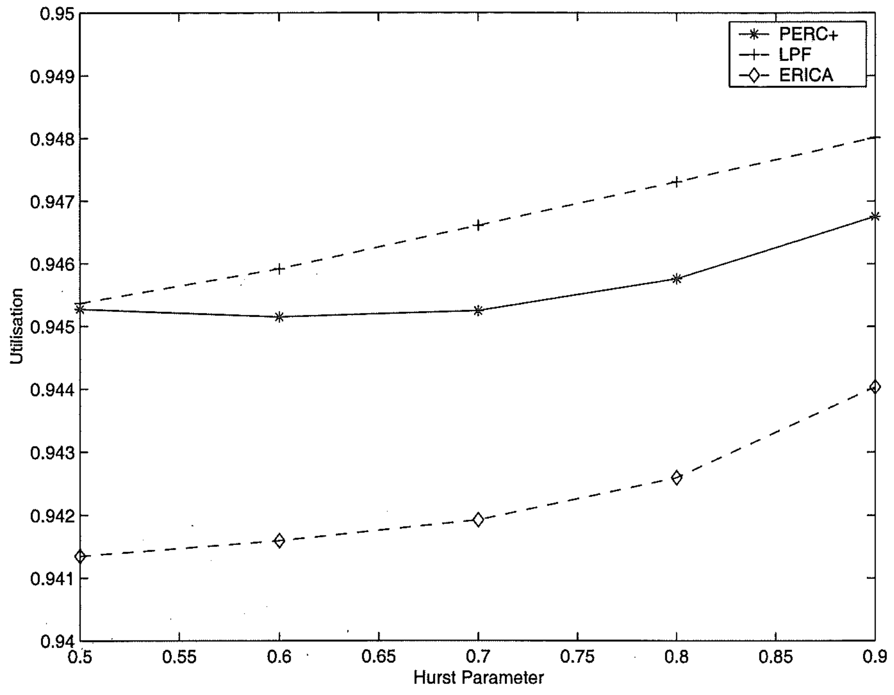


Figure 4.13 Comparing link utilisation for the rate-control algorithms.

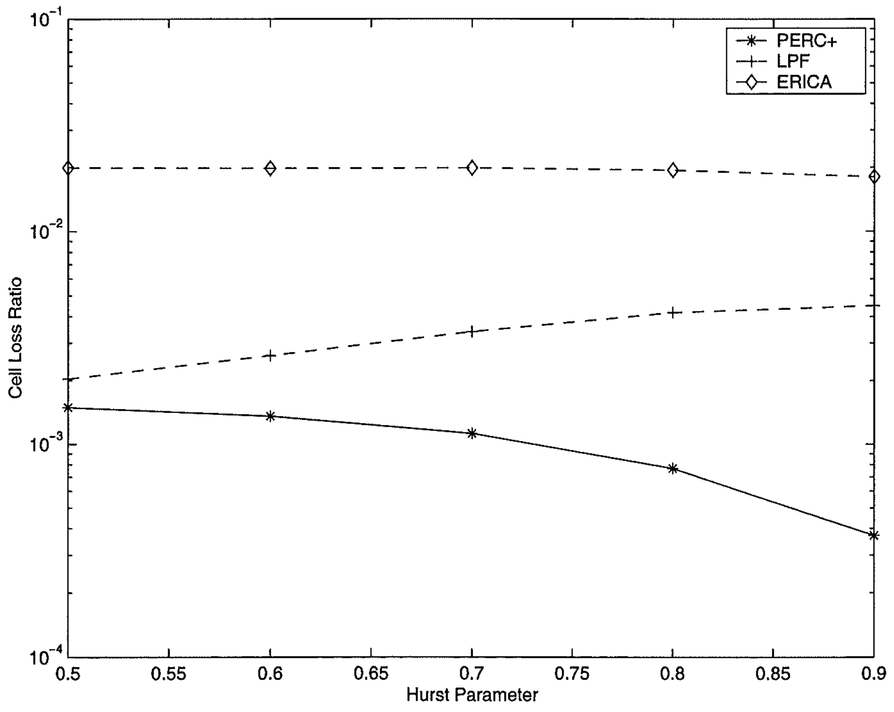


Figure 4.14 Comparing rate-control algorithms using cell loss ratio for a finite buffer size of 500 cells.

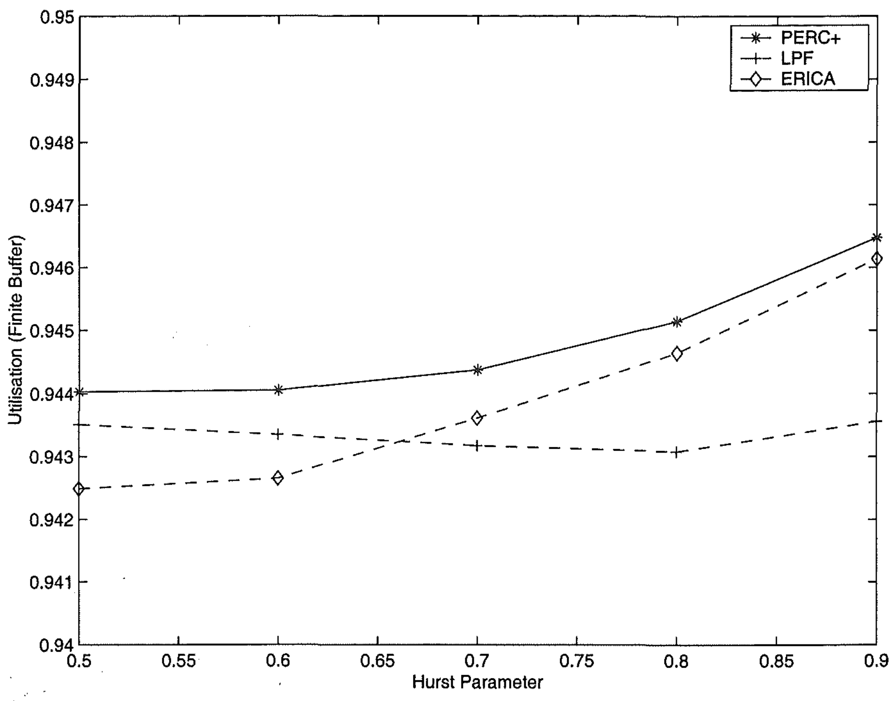


Figure 4.15 Comparing link utilisation for the rate-control algorithms with a finite buffer size.

trip delays and as the Hurst parameter increases (Figure 4.2). Improved performance results as the algorithm compensates for prediction errors, which occur in the rate calculations for high latency connections, by adjusting the rate information returned to lower latency connections. Also, the accuracy of predictions for these short delay connections increases as the correlations become more significant with increasing Hurst parameter.

This improvement in performance is also observed in simulations, as shown in Figure 4.16. In Figure 4.16, the maximum queue length at the network node is measured as the number of sources is increased. The increasing number of sources also means that the range of the round-trip delays is increasing as each additional connection has a round-trip delay greater than the maximum previous delay.

The performance of the system with the prediction compensation algorithm, as incorporated in PERC+, is compared with the system that only uses the predictive rate algorithm PERC in Figure 4.16. The simulation results show that PERC+ effectively maintains the maximum queue length at a fairly constant level for a wide range of round-trip delays; whereas the maximum length increases rapidly when the prediction compensation algorithm is not incorporated into the controller (ie PERC). The fairly constant maximum queue length for PERC+ is a result of the algorithm effectively clustering all connections into a single virtual connection with the smallest round-trip delay (see Equation (4.30)). Hence, additional connections with longer round-trip delays have minimal effect, since the network still only "observes" traffic originating from the same source with smallest delay.

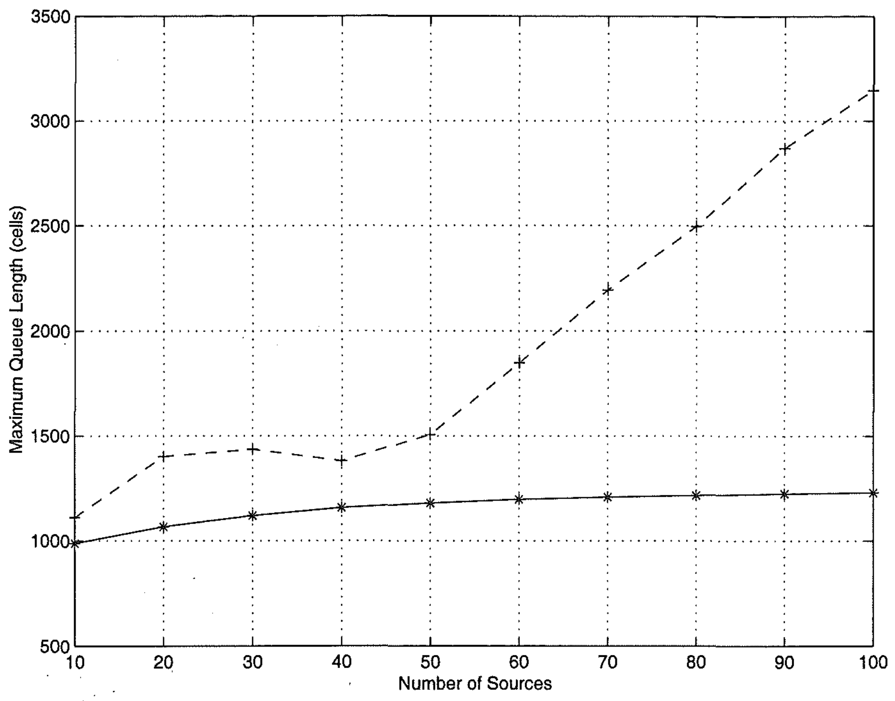


Figure 4.16 The performance of the prediction compensation algorithm as the number of sources is increased (the solid line shows the maximum queue length for PERC+, and the dashed line is for PERC).

4.5 IMPLEMENTATION ISSUES

Having demonstrated the advantages of using a rate control algorithm that is specifically designed for the self-similarity of network traffic, the specific issues related to implementing this mechanism within an actual network environment are now addressed. Since PERC+ predicts rate information based on the latency of a connection and the LRD of background traffic, implementation of the algorithm requires an estimation of the round-trip delay, an appropriate choice of a model for the background traffic and a method of estimating the parameters of this model. Each of these implementation issues are considered in turn, in the context of the ABR service in ATM networks.

4.5.1 Estimation of the Round-trip Delay

The rate control algorithm, as specified in Section 4.3, requires an estimate of the round-trip delay, as the predictors are generated based on this delay. At present, there is no mechanism for retrieving this information within ABR protocols. However, it is relatively straightforward to implement an estimation process for the delay. The TCP protocol in IP networking also requires an estimate of the round-trip time, necessary to set the retransmission timers to detect packet losses. Hence, the approach which is used in the TCP protocol can be employed. The main difference is that this information is now required by the network switches along the ABR connection, rather than at the source.

ATM networks make provision for the need for signalling regarding the management of the connection through the specification of the Resource Management (RM) cells. PERC+ requires special RM cells to be employed to traverse the connection, both during the setup of the connection and periodically during the connection. The ABR source uses these RM cells to estimate the round-trip delay. The RM cell is time-stamped by the source at transmission, and the time difference between the clock time and the time stamp gives the current measure of the round-trip delay. The source then estimates the delay as [Jacobson 1988]:

$$RTT = (1 - \alpha)RTT + \alpha\Delta T, \tag{4.40}$$

where RTT is the current estimate of the round-trip time and ΔT is the difference between the clock and time stamp on the most recent special RM cell. Since the predictors need not be updated every time the RTT is estimated, PERC+ would use a certain granularity to determine whether an ABR source needs to convey this new estimate to switches along the path of the ABR connection. This information can be conveyed to the switches using a special RM cell which updates the local predictors according to (3.47).

There is a tradeoff between prediction accuracy using frequently updated estimates of the delay and the computational expense of conveying the information to the switches and the subsequent updating of the predictors. However optimising the frequency of updating delay estimates is not studied in this thesis.

4.5.2 File Size Distributions

An important consideration is the use of PERC, or the ABR service in general, for the transport of small files. The heavy-tailed nature of file-size distributions has been highlighted earlier in this thesis in Sections 3.2.4 and 3.5.1. The high-variability of file sizes can lead directly to self-similarity that has been observed in network traffic. However, there is a tendency to concentrate on the fact that these heavy-tailed distributions lead to the transfer of large-sized files. In reality, a significant number of *small*-sized files occur within the Internet. These two types of file sizes have been affectionately labelled *elephants* and *mice*, respectively. In designing a best-effort service such as ABR which use mechanisms like PERC, the conventional assumption is that the source has a significantly large amount of data to send (ie. an "elephant") and that this can be represented by an infinite data source. Within data networks, most packets (and subsequently cells) originate from large-size files. However, in terms of the number of files transferred, most file transfers involve files of small size, or are "mice". Hence, it is important to consider the implications for the transfer of small file sizes.

The reason for considering the transfer of small sizes is that the information that rate-control mechanisms generate is delayed by the round-trip time before it is available at the source. As a result, it is out-of-date. For sources with an infinite amount of data to send, this information is the best information that is available and the source has sufficient time to process the information so that its transmission rate is optimal with regard to the network conditions. When transferring a small sized file, all packets may be transmitted within a few round-trip times, so out-of-date rate information can have greater impact on the transfer. If the rate information overestimates the available bandwidth, then a large number of packets will be lost, relative to the file size, and will need to be re-transmitted. Conversely, if the rate information underestimates the available bandwidth, the file transfer becomes unnecessarily slow. Both situations lead to inefficient use of network resources and performance is significantly degraded, relative to the file size.

Resolving this situation can be considered from two perspectives, firstly from the perspective of the network and then from the user's perspective. At an interior node within the network, small file transfers can be viewed as additive noise that disturbs the amount of available bandwidth. This is the approach taken by Kelly [Kelly 2000]. An interior node must filter out these fluctuations to estimate the long-term available bandwidth. Due to the heavy-tailed nature of file size distributions, this noise will be asymptotically self-similar. By incorporating small file transfers into the background traffic component given in Section 4.3, the design of PERC does not require further modification. Hence, the filtering mechanisms incorporated within PERC will be ideal from the perspective of the network.

From the perspective of the source, the rate information provided by PERC will not be optimal for small file transfers. However, it should be noted that this issue is not unique to PERC—any explicit rate algorithm used within the framework of the ABR service will face the same sub-optimality. Also, the problem with small file transfer has more relevance to TCP than ABR, since TCP uses slow start which restricts the transfer of the file unnecessarily. The interaction between TCP and ABR can be improved by using fast start techniques within TCP [Johansson *et al.* 1997]. ABR itself allows immediate access to a fair transmission rate as soon as the connection has been established. If this fair bandwidth allocation is still not considered sufficient, the solution does not necessarily lie in making significant changes to the structure and manner that explicit rate information is used by an ABR source. The intention of the ABR service is, as the name implies, to distribute available bandwidth, and rate control mechanisms need to be developed with that framework in mind. Instead, the solution lies in considering options that a user has before he/she commences packet transmission.

Firstly, minimum or peak cell rates for an ABR connection can be adjusted so that small file transfers can be achieved in less time. However, this is not an entirely satisfactory solution, since the purpose of the MCR is to provide bandwidth for connection management, and not necessarily to supply large, unfair bandwidth demands. Also, the transfer is still rate-controlled. Other options include subscribing to the nrt-VBR service, if a user requires priority treatment of packets within the network, or the UBR service, if losses can be detected and re-transmitted. In this way, higher transmission rates can be achieved so that small files can be transferred in less time. While these solutions shift the responsibility of transmission control to higher layers, they do not tamper with the objectives of the ABR service and remain completely within the purpose of ATM technology as a whole. Higher layers, that use TCP for example, can use large initial window sizes and trace packet losses. At the same time, the UBR service gives higher layers unspecified access to network bandwidth. This achieves an effective solution for transferring small files.

4.5.3 Robustness with Regards to Self-similar Model

Another issue in the implementation of the control algorithm is the specification of model that should be used as a basis for the predictor. The predictive kernel is based on a predictor which is constructed from the autocovariance structure of the background traffic. Using the concept of parsimonious modelling, the self-similarity of the background traffic is efficiently represented using a minimal number of parameters such as the Hurst parameter, variance, and mean. However, as shown in Chapter 3 there are a number of self-similar models. These basically differ in terms of the amount of SRD that is introduced into the model and the computational requirements for generating the autocovariance function specific to the model.

An appropriate model for developing predictors needs to be determined. In making this choice, the performance of the system is the key factor, and there are two aspects of the performance which need to be considered. Firstly, the selected model must not result in serious degradation of the network due to significant errors between the predicted and the true bandwidth requirements of the background traffic. Secondly (and just as importantly), the switches may need to update the predictors based on more recent estimates of the traffic parameters, which means there need to be computationally efficient methods for generating the autocovariance function of the chosen model. This allows predictors to be constructed rapidly.

4.5.3.1 Specifying the Self-similar Models

Three models are considered—the first model is fractional Gaussian noise, as the derivative of fractional Brownian motion, the canonical self-similar process. The second model considered is the fractional ARIMA$(1, d, 0)$ process, as the simplest model which includes short-range dependence. Finally, the autocovariance function can be generated directly from the estimates resulting from the Abry-Veitch estimator. The autocovari-

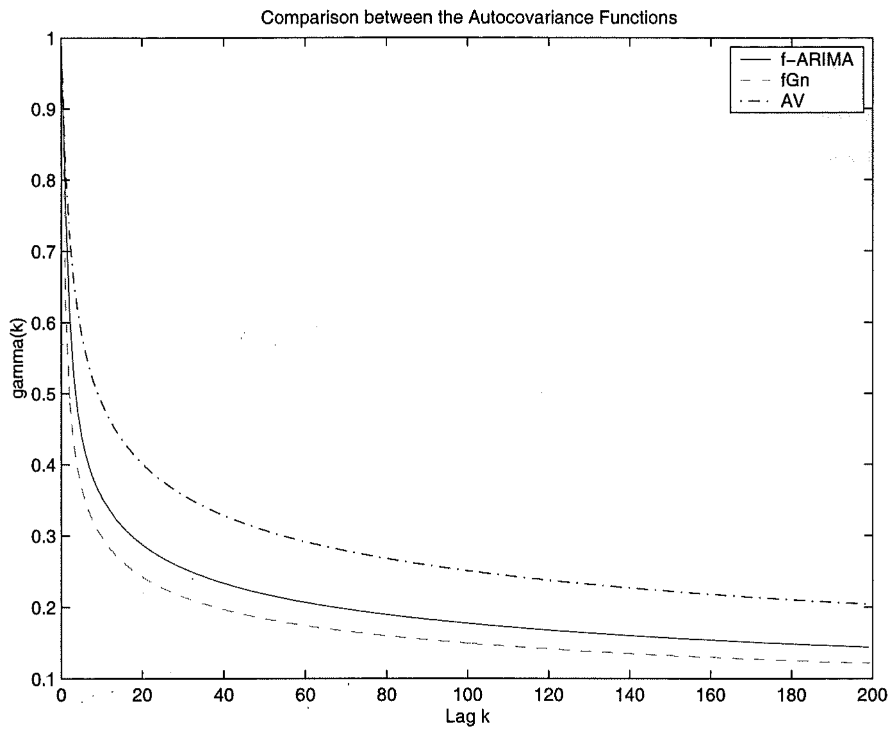


Figure 4.17 The autocovariance functions for the three different self-similar models—fractional Gaussian noise, fractional ARIMA$(1, d, 0)$ and the autocovariance function from the Abry-Veitch estimator.

ance functions for the first two models are given in (3.12) and (3.24), while the normalised autocovariance function from the Abry-Veitch estimator is represented by

$$\begin{aligned}\gamma(k) &= |k+1|^{-(1-\alpha)} \\ &\sim |k|^{-(1-\alpha)}, \text{ as } k \to \infty,\end{aligned} \tag{4.41}$$

which has the correct form as the lag tends to infinity, but is finite at the origin.

The autocorrelation functions for these three models are compared in Figure 4.17 for an equivalent Hurst parameter value of 0.85. The fractional ARIMA$(1, d, 0)$ process has the autoregressive parameter $\phi = 0.3$. It is observed from Figure 4.17 that the autocorrelation functions are noticeably different. This would lead, one might assume, to significantly different performance results. The predictors resulting from these autocovariance functions are shown in Figure 4.18, where the similarity in the form of the predictors can be seen, with the only significant difference occuring in the predictor coefficients with index close to 1. Finally, the theoretical relative variance of the prediction errors for each predictor are compared in Table 4.2. While there is a difference in the theoretical performance of the different predictors, the difference is not significant. Thus, from a theoretical point of view it can be inferred that any model can be chosen without serious degradation in the performance of the system.

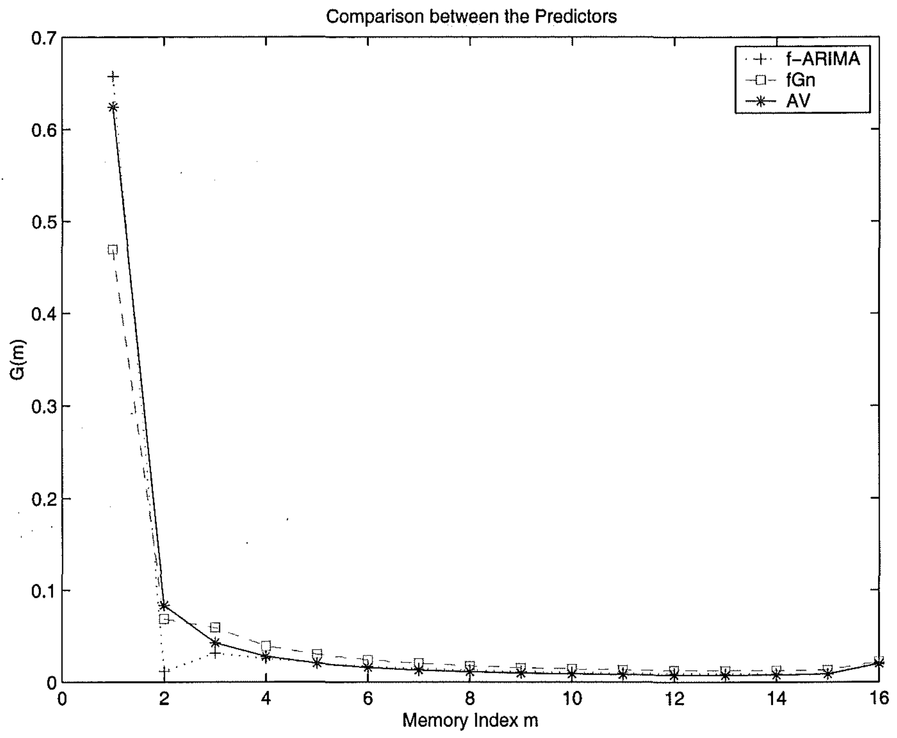


Figure 4.18 The predictors constructed from the three different self-similar models—fractional Gaussian noise, fractional ARIMA$(1, d, 0)$ and that from the Abry-Veitch estimator.

Table 4.2 Comparison of the relative variance for predictors based on different LRD models.

LRD Model	Relative Variance
Fractional ARIMA	0.41
Fractional Gaussian noise	0.57
Abry-Veitch estimates	0.32

Table 4.3 VBR data sets with the estimated parameters for the fractional Gaussian noise model, fractional ARIMA$(1, d, 0)$ process and the Abry-Veitch parameters.

	General Parameters		Model Specific Parameters				
			f-ARIMA			AV Parameters	
Data Set	H	μ	d	ϕ	σ_ϵ^2	α	c_γ
Star Wars	0.900	1.87e6	0.400	0.258	1.92e10	0.764	3.99e10
Mr. Bean	0.980	2.12e6	0.480	0.561	1.24e10	0.963	4.07e11
James Bond: Goldfinger	0.890	2.92e6	0.39	0.544	2.16e10	0.808	1.06e11
Jurassic Park	0.850	1.57e6	0.350	0.526	1.02e10	0.817	4.09e10
ATP Tennis Final 94	0.884	2.63e6	0.384	0.512	2.82e10	0.768	1.12e11
Soccer World Cup Final 94	0.791	3.26e6	0.291	0.649	4.12e10	0.581	1.25e11
Terminator II	0.838	1.13e6	0.338	0.294	1.06e10	0.675	1.34e10
VBR Combination Data Set	0.850	1.13e5	0.350	0.293	1.08e9	0.632	1.21e9

4.5.3.2 Simulation Comparision of Models

This inference is verified by considering a simulation of the system with a number of different VBR data sets [Garrett and Willinger 1994, Rose 1995] and subsequently comparing the performance of the above three models [Östring *et al.* 1999b]. As before, the data sets consist of aggregating randomly shifted versions of the original VBR trace, filtered to remove the GOP frame sequence. The parameters, which have been estimated for each data set, are given in Table 4.3. Predictors were constructed from each model, and the maximum queue length and the cell loss rates are shown in Figs. 4.19 and 4.20. While there are differences in the performance parameters for each of the different models, the performance is in the same order of magnitude. Hence, it can also be concluded that the model which is computationally most efficient can be used for developing PERC+. This allows the predictors to be updated in the minimal amount of time. In this case, utilising the autocovariance function from the Abry-Veitch estimator [Veitch and Abry 1999] is the most efficient approach in that predictors are developed directly from the estimated autocovariance function.

4.5.4 Real-time Parameter Estimation for Weakly Non-stationary Traffic

One final aspect of implementing PERC+ which is considered involves the use of real-time parameter estimates so that time-variations in the traffic can be accommodated. In this section, the assumption of stationarity for the background traffic model is relaxed, and adaptive methods of handling variations in the traffic parameters are investigated. More specifically, *weakly non-stationary* traffic is considered, where variations in parameters occur at a time scale that is greater than the time constant used in the control

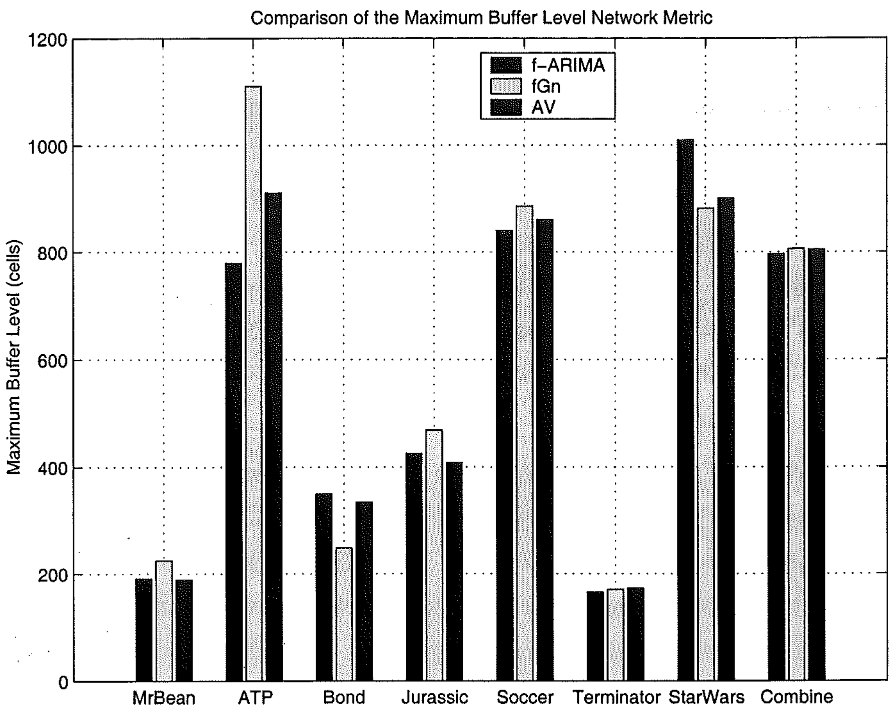


Figure 4.19 Comparison between the maximum queue lengths for the VBR data sets.

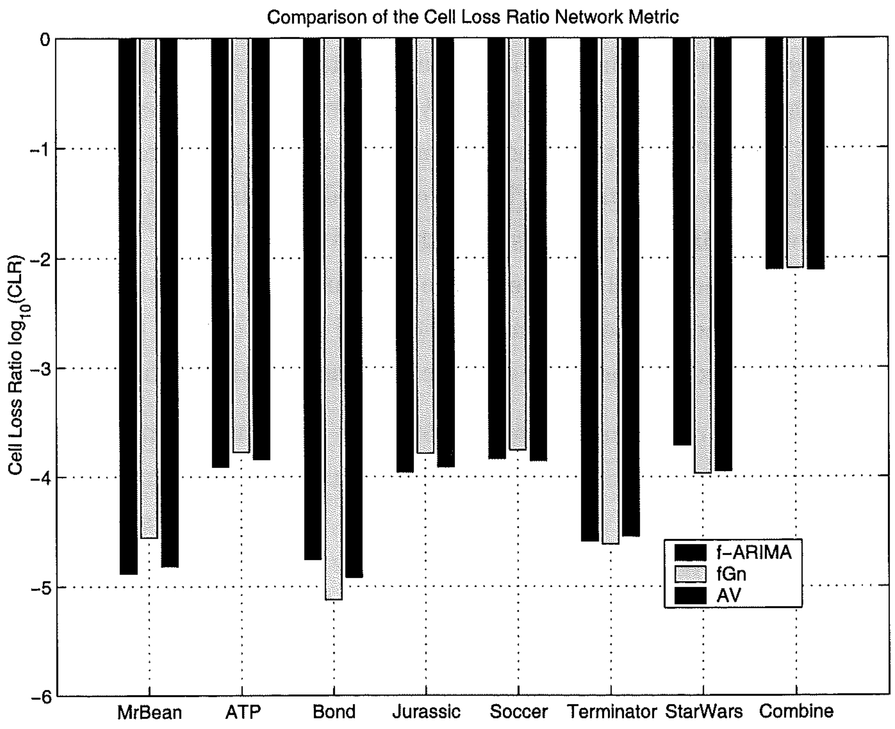


Figure 4.20 Comparison between the cell loss ratios for the VBR data sets.

algorithm.

4.5.4.1 Stationary versus Non-stationary Traffic Models

Before continuing with the discussion, however, the assumption of stationarity deserves further attention. By definition, long-range dependence is an asymptotic property which is not affected by short-term time variations but rather describes the trend in the relationship between correlations as the lag increases indefinitely. There are valid arguments for modelling data either as an LRD process or using a short-memory model with non-stationary mean [Klemeš 1974].

However, the modelling approaches, from a practical perspective, appear to differ in philosophy rather than in functionality. A key feature of LRD models is their practical value in parsimoniously modelling a process. In addition, estimators of the Hurst parameter based on the wavelet transform can disassociate non-stationarities in the mean and variance from the process of identifying long-range dependence within the data [Abry and Veitch 1998]. Thus, there are ways of isolating the non-stationarities that occur within the data, and as a result, assuming stationarity is a valid starting point.

4.5.4.2 Real-time Parameter Estimators

A simple adaptive version of the predictive rate-controller is considered, which utilises real-time estimates of the traffic parameters. Adaptive control is necessary in commercial implementations of traffic control mechanisms due to the time-variations that occur within the traffic. These non-stationarities are caused by admission of new connections (or completion of existing connections) which results in a change in the mean traffic level. There is also a potential change in the burstiness of the traffic due to the type of data being transferred or the transmission control protocol that a new connection is utilising. Hence the need to update the traffic parameters within the controller.

On-line estimation of the mean can be achieved using the standard sample mean. While LRD increases the variance of the mean estimates, the performance of the sample mean is close to optimal [Beran 1994] (see Figure 3.6), and it is very simple to implement as an on-line estimator. With regard to estimating the Hurst parameter and the variance, the real-time version of the Abry-Veitch estimator, which uses the wavelet transform to decorrelate the long-range dependence within traffic data, is incorporated [Abry and Veitch 1998]. Since the wavelet transform can be computed very efficiently using the fast pyramid algorithm, the on-line formulation of the estimator can done with relative ease [Roughan *et al.* 2000], and the estimator can be used for jointly estimating the variance and the Hurst parameter.

4.5.4.3 Simulation Study

The performance of an adaptive version of PERC+, which uses real-time measurements of the mean, variance and Hurst parameter, is compared with two other rate control approaches in Table 4.4—the low-pass filter algorithm considered previously in this thesis and ERICA [Kalyanaraman *et al.* 2000]. For adaptive PERC+, the mean is estimated every 2^{10} traffic samples, and the Hurst parameter and variance are updated every 2^{12} samples. These block sizes were chosen after considering the performance of the adaptive system for a range of different block sizes. Two data sets are used—the *Star Wars* data set which has already been discussed in this paper, and the *BC-pOct89* Ethernet trace, which is one of the original data sets used to demonstrate the self-similarity of network traffic [Leland *et al.* 1994]. This second data set is of particular interest in this study, since it has been noted by other researchers that the trace has a level shift in mean. As with the *Star Wars* set, ten copies of this trace that have been randonly shifted in time have been aggregated, which means that there are ten shifts in the mean in the final trace.

Two different simulation scenarios were considered, one with a buffer of infinite capacity and the other using a finite buffer with a maximum capacity of 100 cells. Comparing the performance of adaptive PERC+ with the other rate-control algorithms, it can be observed that the proposed algorithm has a higher performance for all of the parameters of interest. Even though the performance of ERICA comes close to PERC+ with respect to buffer occupancy, this comes at the cost of higher cell loss rates. Thus, it can be concluded that the adaptive PERC+ algorithm, which incorporates simple real-time parameter estimation techniques, will result in better control of elastic connections.

4.6 CONCLUSIONS

This chapter has presented the rationale and development of a rate control mechanism, labelled PERC+, which is targetted for the ABR service in ATM networks. The algorithm predicts explicit rate information for ABR sources using the LRD structure of background traffic. Due to the intrinsically self-similar nature of network traffic, PERC+ performs significantly better than other traffic control mechanisms. Using more accurate predictions based on LRD, the network achieves the target utilisation with less deviation. This is because the rate controlled connections are able to match the available bandwidth more closely. From a network perspective, there is less storage demand for cells that arrive when the output link bandwidth is exceeded resulting in lower queue occupancy. As a consequence, there are less cell losses since the queue is less likely to overflow. A well-known problem with long-range dependent traffic is that typically decay in cell losses follows a hyperbolic relationship with the size of the buffer. As a result, a linear increase in the buffer will not result in substantial decrease in loss rates.

Table 4.4 The performance parameters of the adaptive system using real-time estimation of traffic parameters, compared with the low-pass filter (LPF) algorithm and ERICA.

Data Set	Performance Parameters	Adaptive PERC+	LPF Algorithm	ERICA
Star Wars	Maximum Queue Length	2.60e3	2.620e4	6.91e3
	Mean Queue Length	33.2	1.03e3	569.0
	Queue Length Std.	136.2	2.55e3	921.1
	Utilization (infinite buffer)	0.946	0.947	0.944
	Cell Loss Rate	0.0016	0.0073	0.0263
	Utilization (finite buffer)	0.946	0.940	0.948
Ethernet	Maximum Queue Length	150.0	724.7	439.4
	Mean Queue Length	8.60	33.7	23.9
	Queue Length Std.	15.1	67.3	38.2
	Utilization (infinite buffer)	0.936	0.944	0.917
	Cell Loss Rate	9.40e-6	0.0027	0.0049
	Utilization (finite buffer)	0.936	0.941	0.952

However, since PERC+ incorporates the correlation structure of long-range dependence, the problem of slowly decreasing loss rates is counteracted. A number of implementation issues have been considered with regards to PERC+, including estimation of round-trip delay, selection of background traffic model and real-time estimation for an adaptive version of PERC+.

There are other fundamental aspects of PERC+ that need to be addressed in order to gain a deeper understanding of control in assocation with self-similar processes. These include the sensitivity of the algorithm to traffic parameters, in particular the Hurst parameter. Developing robust rate control, that has high performance even when there is uncertainty associated with traffic models, is also of interest. Finally, the influence of long-term correlations as opposed to the short-term correlations needs to be investigated. These issues are addressed in the following chapters.

Chapter 5

PARAMETER SENSITIVITY ANALYSIS

Self-similar traffic is fundamentally different from traditional Poisson models, so traffic management algorithms need to be considered with this type of input traffic. The impact of self-similarity on a network is that the network observes longer queue lengths on average, and greater losses than those predicted by Markovian models. These performance results for queues with self-similar input traffic have been substantiated by a number of research studies [Norros 1994, Addie *et al.* 1995, Heyman and Lakshmann 1996, Tsybakov and Georganas 1997].

Having demonstrated that the self-similar nature of traffic does have significant consequences on the performance of the network, it is logical to pursue research that considers a control algorithm when the input to the algorithm is a self-similar process and to design the algorithm so as to optimise the system. This not only means minimising the effect of self-similarity, but also taking advantage of the specific properties of self-similarity, as has been done in using the long-range dependence of traffic to produce more accurage predictions in rate-control mechanisms in the previous chapter.

Other research that has specifically considered self-similarity within network management algorithms includes Giordano's call admission control scheme [Giordano *et al.* 1996], Adas' queue management techniques based on active cell discard [Adas and Mukherjee 1995] and Chiruvolu's mechanism [Chiruvolu and Sankar 1997] which controls the transmission rate for VBR traffic. Finally, a significant contribution to this area is Tuan and Park's congestion control algorithm [Tuan and Park 1999, Park and Willinger 2000] which estimates network load at different time scales to improve the performance of linear increase, multiplicative decrease algorithms like TCP. This research work has shown that it is possible to incorporate the stochastic structure of self-similar traffic into network management protocols and gain quantifiable benefits from doing so.

While the principle of algorithm design, which incorporates self-similarity, has been demonstrated, it requires models which parameterise the burstiness of the traffic, and other aspects such as the average traffic level and the variance. The Hurst parameter is the index of self-similarity, and it captures distinctive qualities of self-similar traffic. Management algorithms that are designed using self-similar models for network traffic thus require estimates of the Hurst parameter from traffic measurements. Estimation techniques for the Hurst parameter have been outlined in Chapter 3.

However, an issue that has not been addressed previously in the research literature is the sensitivity of algorithms to estimates of the traffic parameters, particularly the Hurst parameter. Algorithms can be designed which do not make explicit use of the Hurst parameter (for example the multiple time scale control proposed by Tuan and Park [Tuan and Park 1999]). Here, algorithms are considered that have been specifically designed using the Hurst parameter. Heuristic approaches give single point estimates for the Hurst parameter which provide evidence as to whether measured traffic is self-similar. More sophisticated techniques give confidence intervals that can be used to give statistical proof of self-similarity. The details of these techniques can be found in Section 3.3.1. When a traffic control mechanism is developed for self-similar traffic, an estimate of the Hurst must be used. The final choice of the Hurst parameter value that is used in the design of the algorithm becomes a significant issue, because the performance of the algorithm will be affected by that choice. The sensitivity of PERC with regards to traffic parameters, including the Hurst parameter, variance and mean, is addressed in this chapter.

5.1 SENSITIVITY ANALYSIS OF THE RATE-CONTROL ALGORITHM TO TRAFFIC PARAMETERS

The three key traffic parameters that are required for developing the predictors in PERC are the Hurst parameter, variance and mean. Using these parameters, a fractional Gaussian noise model for the network traffic can be constructed. More sophisticated models, such as fractional ARIMA processes or aggregated ON/OFF sources with heavy-tailed distributions can be used. Of more fundamental interest, however, is the extraction of the fundamental parameters of the traffic and investigation of the sensitivity of the traffic to these parameters. It is assumed that a parameter estimate is available which is mismatched to the true parameter, and then the effect of this mismatch is determined analytically. These analytical results are also verified using simulation results.

5.2 HURST PARAMETER SENSITIVITY

5.2.1 Sensitivity Analysis

The Hurst parameter is the index of self-similarity for a stochastic process, and characterises the fundamental properties of this type of traffic. These properties include scale invariance of the distribution, long-range dependence within the autocorrelation function and a spectral density that obeys a power-law at the origin. PERC directly uses one of these properties, namely the long-range dependence of network traffic when the Hurst parameter is in the interval $(0.5, 1.0)$. Since correlations at large lag decrease more slowly than those for SRD processes, samples of previous traffic levels can be used to predict more accurately future utilisation levels of the network. Thus, the correlation structure forms the basis from which the algorithm is developed. If the true autocorrelation function $\gamma^*(k)$ for the traffic process is known, then the optimal linear predictor $\mathbf{G}^*_\delta$ is of the form:

$$\hat{X}_{k+\delta} = \mathbf{G}^{*T}_{\delta}\mathbf{X}_M, \tag{5.1}$$

where X_k is a covariance stationary stochastic process with zero mean, variance σ^2 and autocovariance function $\gamma^*(k)$; and $\mathbf{X}_M$ is the vector of stored traffic measurements $\{X_k, X_{k-1}, \dots, X_{k-M+1}\}'$. M is the memory-length of the predictor. The predictor $\mathbf{G}^*_\delta$ is generated by the matrix equation:

$$\Gamma^*\mathbf{G}^*_\delta = \gamma^*_\delta, \tag{5.2}$$

and performance of this optimal predictor is given by:

$$v^*_\delta = \gamma^*(0) - {\gamma^*}_\delta^T {\Gamma^*}^{-1} {\gamma^*}_\delta. \tag{5.3}$$

This theory was presented in Chapter 3 and used within PERC. If the model for the traffic is exact, then these results can be used to determine the overall performance of the network when a number of elastic connections are controlled by the rate-control algorithm.

5.2.2 Effect of Hurst Parameter Mismatch

In reality, absolute knowledge of traffic is not possible. The reasons for this include the estimation of the Hurst parameter from a finite set of measurements and the potentially time-varying nature of network traffic. Thus, consider the situation where the predictor is being developed from an estimated autocorrelation function $\hat{\gamma}(k)$ which is in fact mismatched to the true stochastic structure of the network traffic. The predictor would then be calculated as:

$$\hat{\Gamma}\hat{\mathbf{G}}_\delta = \hat{\gamma}_\delta. \tag{5.4}$$

Since the true correlation struture is given by γ^*, the performance of this mismatched predictor can be determined using the variance of the prediction errors as the performance metric:

$$\begin{aligned}\hat{v}_\delta &= \mathbf{E}\left[(R_b(k+\delta) - \hat{R}_b(k+\delta))^2\right] \\ &= \mathbf{E}[R_b(k+\delta)^2] - 2\mathbf{E}[R_b(k+\delta)\hat{R}_b(k+\delta)] + \mathbf{E}[\hat{R}_b(k+\delta)^2] \\ &= \gamma^*(0) - 2{\gamma^*}_\delta^T\hat{\Gamma}^{-1}\hat{\gamma}_\delta + \hat{\gamma}_\delta^T\hat{\Gamma}^{-1}\Gamma^*\hat{\Gamma}^{-1}\hat{\gamma}_\delta.\end{aligned} \tag{5.5}$$

A standard measure of the sensitivity of a stochastic control algorithm, with regards to a particular parameter, is to use the relative variance of the prediction errors (refer to [Åström and Wittenmark 1970]). In this case the sensitivity of the performance of the system implemented with the mismatched predictors is:

$$K_v = \frac{\hat{v} - v^*}{v^*}, \tag{5.6}$$

which gives Theorem 3 [Östring *et al.* 2000c].

Theorem 3 *The relative increase in the variance of the prediction errors resulting from a linear predictor which is mismatched to the stochastic structure of the background traffic is given by:*

$$K_v = \frac{\hat{\gamma}_\delta^T\hat{\Gamma}^{-1}\Gamma^*\hat{\Gamma}^{-1}\hat{\gamma}_\delta + {\gamma^*}_\delta^T{\Gamma^*}^{-1}{\gamma^*}_\delta - 2{\gamma^*}_\delta^T\hat{\Gamma}^{-1}\hat{\gamma}_\delta}{\gamma^*(0) - {\gamma^*}_\delta^T{\Gamma^*}^{-1}{\gamma^*}_\delta} \tag{5.7}$$

The sensitivity of the predictor can now be studied, with regards to the Hurst parameter estimate used to develop the predictor. The fractional Gaussian noise model is employed for this investigation because it parsimoniously models LRD, with an autocovariance function given by Equation (3.12). The relative increase in variance is calculated for pairs of $(H^*, \hat{H})$, which correspond to inaccurate estimates of the Hurst parameter, and the resulting surfaces are shown in Figures 5.1–5.3. These sensitivity surfaces have been generated for different prediction horizons δ.

Figures 5.1–5.3 reveal that, when the Hurst parameter is underestimated, there is an asymmetry in the sensitivity of the predictor, with rapidly increasing sensitivity as the estimated value decreases towards 0.5. There is a slight increase in the relative variance when the Hurst parameter is overestimated. However, this sensitivity is very low compared with situation when underestimation of the Hurst parameter occurs. As expected, the sensitivity is zero across the diagonal of the surface when the Hurst parameter estimate exactly matches the true value.

The significance of these results is that they add further weight to the argument that if network traffic is truly self-similar, then it is important that this property be taken into account in the design of traffic management algorithms so that the impact of

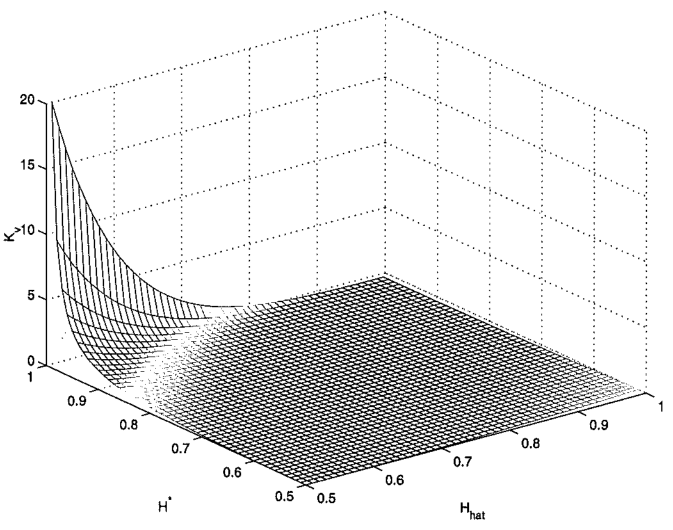


Figure 5.1 The sensitivity of linear predictors to Hurst parameter estimates using the fractional Gaussian model ($\delta = 1$).

the burstiness of the self-similar traffic can be avoided. Otherwise, the performance of the system will be seriously degraded.

5.2.3 Reasons for Asymmetric Sensitivity

The asymmetry in the sensitivity of the predictors, as shown in 5.1–5.3, requires an explanation for two reasons. Firstly, it will add more understanding to the nature of self-similar traffic and its interaction with network control mechanisms. Secondly, the design of such mechanisms can be enhanced so that the regions in the sensitivity surface where performance is particularly sensitive to the estimate of the Hurst parameter can be avoided. The surface suggests that it is more beneficial to err on the side of over-estimating the Hurst parameter, than underestimating it. Taken to the extreme, this would imply that the best choice of predictor would be one designed for traffic with Hurst parameter approaching the value 1.0, regardless of the true value.

This is an inappropriate choice because these predictors will be used to control individual microflows within the aggregate traffic load. Hence, even a small degradation in the performance of the control of any individual connection, where the actual predictor differs from the optimal predictor, can result in significant losses and delays for the aggregate traffic with a large numbers of connections. Better engineering solutions are investigated in Chapter 6 that ensure robust performance given a confidence interval for the Hurst parameter estimate.

To understand the interaction between a particular predictor and the self-similar traffic which it is attempting to predict, the structure of the predictor, which was introduced in Section 3.4.2, is investigated together with the correlation structure of the traffic, Section 3.2.2. Two extreme cases are considered, where the true Hurst parame-

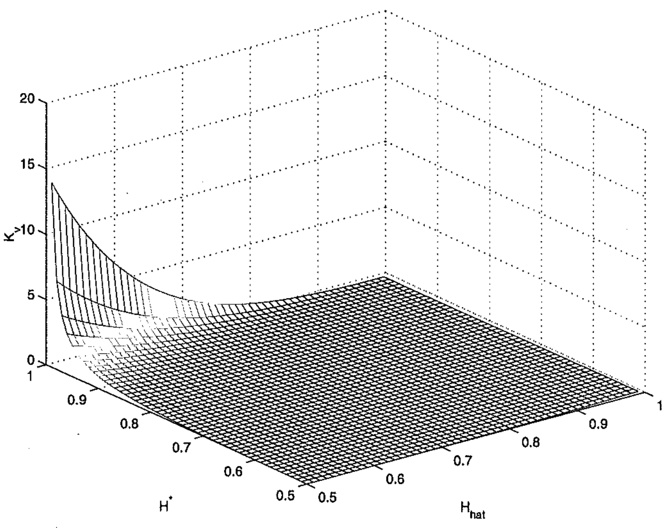


Figure 5.2 The sensitivity surface for prediction horizion $\delta = 2$.

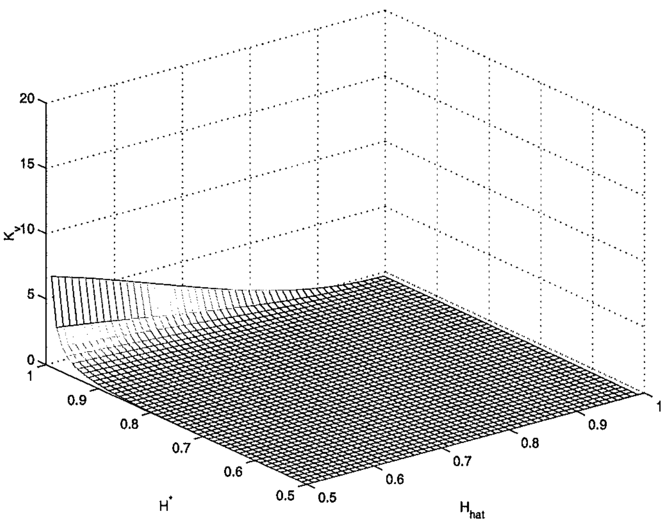


Figure 5.3 The sensitivity surface for prediction horizion $\delta = 32$.

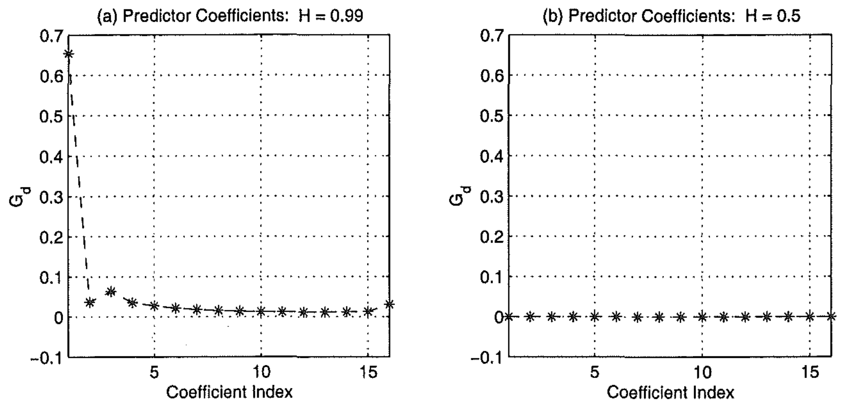


Figure 5.4 Predictors for the boundary Hurst parameter values: (a) $\hat{H} = 0.99$, (b) $\hat{H} = 0.5$.

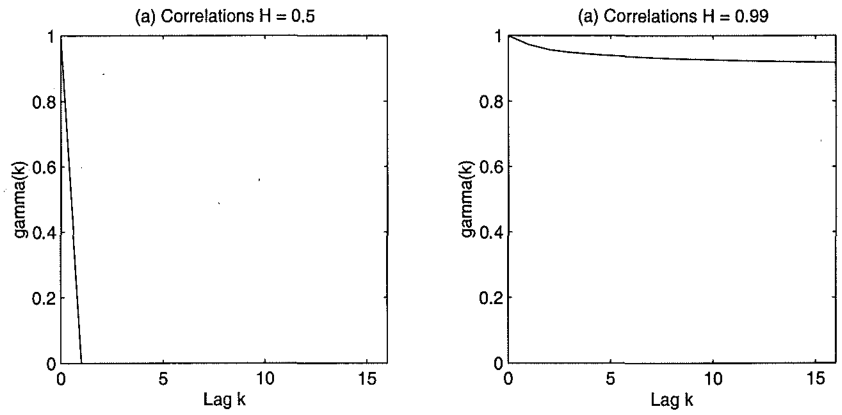


Figure 5.5 Correlation structure: (a) $H^* = 0.5$, (b) $H^* = 0.99$.

ter is 0.5 and 0.99 respectively but where the predictor is developed for the other case. These predictors are shown in Figure 5.4 and the correlation structure of the traffic is shown in Figure 5.5.

What can be observed in the first situation is that predictor gives significant weight to traffic measurements which have no effect on future traffic levels (the correlations are zero for any lag greater than zero). However, it should be recalled that the burstiness of the traffic with Hurst parameter of 0.5 is not as significant. Hence the overall impact of including weighted samples of the past traffic will not be as great. The second situation shows that the mismatched filter gives no weight to previous traffic samples, regardless of the fact that there is a significant amount of information in the previous samples regarding the future traffic level. This results in large prediction errors, and the performance of the system is reduced drastically.

5.2.4 Discussion on the Influence of SRD

While this discussion applies directly to self-similar processes and the sensitivity of predictors developed for these models, one aspect which has not considered is the extent that short-range dependent correlations dominate this sensivity. There have been a couple of significant research studies [Ryu and Elwalid 1996, Grossglauser and Bolot 1999] into how much impact long-range dependence has on the performance of the system, particularly on the losses that occur at a buffer of finite size. There is also evidence that the predictors developed for self-similar processes do not need to have infinite memory to achieve close to optimal performance, as shown in Chapter 3. Thus, it would also be expected that the sensitivity of the predictors would have some dependence on the short-term correlations.

This issue is discussed in more detail in Chapter 7 after robust control approaches have been considered. It should be noted that in relation to self-similar processes, the results presented in this section (Section 5.2) remain valid as the exact autocorrelation function have been used to develop the optimal predictor. This analysis has highlighted a particular aspect of the sensivity, and revealed further issues that need to be investigated.

5.2.5 Analysis of the Sensitivity of the System

The analysis in Section 5.2 addressed the sensitivity of a single predictor within a rate-control algorithm intended for a specific connection. However, a network system consists of a number of different rate-controlled connections competing with self-similar background traffic. Using the PERC, each source will be controlled by a predictor developed specifically for the connection's delay. The overall system performance, and sensitivity, is based on the interaction between these individual control actions.

This is particularly true when PERC+ is used, which includes the prediction compensation algorithm developed in Section 4.3.3. In this situation, the prediction errors from sources with higher latency are replaced with the errors from sources with lower latency. Hence the sensitivity of the system is more dependent on the sensitivity of the prediction errors for the connection that has the lowest round-trip delay. The sensitivities of both PERC and PERC+ are investigated.

5.2.5.1 Sensitivity of PERC

The system with uncompensated rate calculations is considered first. In this case, rate calculations are made without reference to rate information returned to other sources—more specifically sources which have greater round-trip delay. Hence, the total input rate to the network node is given by:

$$\begin{aligned} I^{\mathbf{u}}(k) &= U^{\mathbf{u}}(k) + R_b(k) \\ &= \sum_{j=1}^{N} ER_j^{\mathbf{u}}(k-\delta_j) + R_b(k) \\ &= \rho C + X(k) - \sum_{j=1}^{N} \omega_j \hat{X}_{\delta_j}(k), \end{aligned} \tag{5.8}$$

based on the same logic and notation as Equation (4.20). In this case, the predicted variables $\hat{X}_{\delta_j}$ result from using the mismatched predictors $\hat{G}_{\delta_j}$. These have been developed using an estimated autocorrelation structure $\hat{\gamma}$. To determine the sensitivity of this system to the Hurst parameter estimate, the variance of the input rate is calculated in a similar way to Equation (4.22):

$$\begin{aligned} Var(I^{\mathbf{u}}) &= \mathbf{E}\left[\left(X(k) - \sum_{j=1}^{N} \omega_j \hat{X}_{\delta_j}(k)\right)^2\right] \\ &= \sigma_b^2 - 2\sum_{j=1}^{N} \omega_j \gamma'_{\delta_j} \hat{\Gamma}^{-1} \hat{\gamma}_{\delta_j} + \sum_{i=1}^{N}\sum_{j=1}^{N} \omega_i\omega_j \hat{\gamma}'_{\delta_i} \hat{\Gamma}^{-1} \Gamma_{\delta_i\delta_j} \hat{\Gamma}^{-1} \hat{\gamma}_{\delta_j}, \end{aligned} \tag{5.9}$$

where the matrix $\Gamma_{\delta_i\delta_j}$ is defined in Equation (4.23). The system variance using optimal predictors is given in Equation (4.22), leading to the following theorem:

Theorem 4 *The relative increase in the variance of the input rate to a network node for the system using PERC, with predictors that are mismatched to the stochastic structure of the background traffic, is given by:*

$$K_v^{\mathbf{u}} = \frac{2\sum_{j=1}^{N} \omega_j \gamma'_{\delta_j} \Gamma^{-1}\left(\gamma_{\delta_j} - \hat{\gamma}_{\delta_j}\right) + \sum_{i=1}^{N}\sum_{j=1}^{N} \omega_i\omega_j \left(\hat{\gamma}'_{\delta_i} \hat{\Gamma}^{-1} \Gamma_{\delta_i\delta_j} \hat{\Gamma}^{-1} \hat{\gamma}_{\delta_j} - \gamma'_{\delta_i} \Gamma^{-1} \Gamma_{\delta_i\delta_j} \Gamma^{-1} \gamma_{\delta_j}\right)}{\sigma_b^2 - 2\sum_{j=1}^{N} \omega_j \gamma'_{\delta_j} \Gamma^{-1} \gamma_{\delta_j} + \sum_{i=1}^{N}\sum_{j=1}^{N} \omega_i\omega_j \gamma'_{\delta_i} \Gamma^{-1} \Gamma_{\delta_i\delta_j} \Gamma^{-1} \gamma_{\delta_j}} \tag{5.10}$$

where $K_v = \frac{\hat{v}-v^*}{v^*}$ as before, with v representing variance.

5.2.5.2 Sensitivity of PERC+

The other system that is being considered uses the prediction compensation algorithm, as transmission rates are controlled by PERC+. Prediction errors for connections with high latency are compensated for using the more accurate predictions for connections with smaller delays. The expression for the input rate for the specific network node in this situation is:

$$I^{\mathrm{c}}(k) = \rho C + X(k) - \hat{X}_{\delta_N}(k). \tag{5.11}$$

The lengthy derivation for this result parallels that given in Section 4.4.1 and is not reproduced here. Using the mismatched filter $\hat{G}_{\delta_N}$, the predicted variable is given by

$$\hat{X}_{\delta_N}(k) = \hat{G}'_{\delta_N}\mathbf{X}(k - \delta_N). \tag{5.12}$$

Finding the variance of the input rate for the system using PERC+ is straightforward:

$$\begin{aligned} Var(I^{\mathrm{c}}) &= \mathbf{E}\left[(X(k) - \hat{X}_{\delta_N}(k))\right] \\ &= \sigma_b^2 - 2\gamma^{*\prime}{}_{\delta}\hat{\Gamma}^{-1}\hat{\gamma}_\delta + \hat{\gamma}'_\delta\hat{\Gamma}^{-1}\Gamma^*\hat{\Gamma}^{-1}\hat{\gamma}_\delta. \end{aligned} \tag{5.13}$$

Noting that the optimal system variance is given by Equation (4.30), the following theorem can be defined:

Theorem 5 *The relative increase in the variance of the input rate to a network node for the system using PERC+, with predictors that are mismatched to the stochastic structure of the background traffic, is given by:*

$$K_v^{\mathrm{c}} = \frac{\hat{\gamma}'_{\delta_N}\hat{\Gamma}^{-1}\Gamma^*\hat{\Gamma}^{-1}\hat{\gamma}_{\delta_N} + \gamma'_{\delta_N}\Gamma^{-1}\gamma_{\delta_N} - 2\gamma^{*\prime}{}_{\delta_N}\hat{\Gamma}^{-1}\hat{\gamma}_{\delta_N}}{\sigma_b^2 - \gamma'_{\delta_N}\Gamma^{-1}\gamma_{\delta_N}}. \tag{5.14}$$

The sensitivity of each individual system to the estimated covariance structure is of interest if one is implementing that specific system. However in this research work, comparing the performance of the two systems is of interest. Previously, it has been shown that the system which uses PERC+ achieves higher performance in terms of smaller variance around the desired link utilisation. This follows because the system compensates for the prediction errors, by adjusting the explicit rate information for connections with lower latency based on the rate calculations for the connections with higher latency. As a result, the network node primarily observes a virtual source with prediction error due to smallest round-trip delay in the set of connections, rather than encountering a wide range of prediction errors from the heterogeneous collection of sources. However, it can be observed from Figures 5.1–5.3 that predictors with shorter prediction interval are more sensitive to inaccuracies in the Hurst parameter. Thus, the relative sensitivity of the two algorithms needs to be determined, as defined in the following theorem.

Theorem 6 *The normalised relative variance of the system that uses PERC+, compared with the system simply using PERC, where predictors are generated by a mismatched,* *estimated autocovariance structure of the background traffic, is defined as:*

$$K_s = \frac{\sigma_b^2 - 2{\gamma^*}'_{\delta}\hat{\Gamma}^{-1}\hat{\gamma}_\delta + \hat{\gamma}'_\delta\hat{\Gamma}^{-1}\Gamma^*\hat{\Gamma}^{-1}\hat{\gamma}_\delta}{\sigma_b^2 - 2\sum_{j=1}^N \omega_j {\gamma^*}'_{\delta_j}\hat{\Gamma}^{-1}\hat{\gamma}_{\delta_j} + \sum_{i=1}^N\sum_{j=1}^N \omega_i\omega_j\hat{\gamma}'_{\delta_i}\hat{\Gamma}^{-1}{\Gamma^*}_{\delta_i\delta_j}\hat{\Gamma}^{-1}\hat{\gamma}_{\delta_j}} \cdot \frac{\sigma_b^2 - 2\sum_{j=1}^N \omega_j {\gamma^*}'_{\delta_j}{\Gamma^*}^{-1}{\gamma^*}_{\delta_j} + \sum_{i=1}^N\sum_{j=1}^N \omega_i\omega_j{\gamma^*}'_{\delta_i}{\Gamma^*}^{-1}{\Gamma^*}_{\delta_i\delta_j}{\Gamma^*}^{-1}{\gamma^*}_{\delta_j}}{\sigma_b^2 - {\gamma^*}'_{\delta_N}{\Gamma^*}^{-1}{\gamma^*}_{\delta_N}}. \tag{5.15}$$

5.2.6 Peformance Evaluation Using Simulations

The asymmetry in the sensitivity of the control algorithms can be verified using simulations based on a network model that includes a set of ABR sources and non-adaptive, background traffic. The simulation model is the same one which was used to investigate the rate-control algorithm presented in Chapter 4, and is shown in Figure 4.6. Using this model, the sensitivity of the overall system to the Hurst parameter estimates can be determined. In addition, the interaction of self-similar traffic with elastic traffic, that is being controlled by predictive mechanisms that are mismatched to the structure of the background traffic, is investigated. Specifically, a number of estimation techniques are used to determine where the Hurst parameter of the traffic lies, but control mechanisms are implemented using Hurst parameters which lie across the interval $[0.5, 1.0)$ to observe the behaviour of the system.

The data sets that are used to investigate the sensitivity are shown in Table 5.1. These data sets are based on VBR data sets [Garrett and Willinger 1994, Rose 1995] which have been randomly shifted in time and then aggregated. As in the simulation study in Section 4.4.2, the original data sets were filtered to remove the correlation structure which is due to the GOP frame sequence. The Hurst parameter estimates for these data sets using variance-time analysis, R/S statistic analysis and Abry-Veitch estimation techniques are given in Table 5.1. Two additional data sets are included in this set, which are the *Combination VBR Data* and the *White Gaussian Noise* sets. The *Combination VBR Data* set is the aggregation of all the original data sets above it in the table, and the *White Gaussian Noise* set has the same mean and variance of the *Star Wars* data set. This data has, however, been generated by a normally distributed random number generator.

Comparing the estimates from the different estimation techniques, it is observed that while there is sufficient agreement between the techniques that self-similarity is evident in each of the aggregated VBR data sets; and that, for most cases, the 95% confidence interval generated from the AV estimator contains all three estimates, there is also sufficient discrepancy between the values for the Hurst parameter to leave uncertainity in the choice of the value when an ABR congestion control algorithm is being developed. To determine the effect of this uncertainity on the performance of the ABR system,

Table 5.1 Comparison between the Hurst parameter estimates for the variance-time (V-T), R/S statistic (R/S) and Abry-Veitch (AV) estimation methods (CI denotes confidence interval).

Data Set Source	**V-T Estimate**	**R/S Estimate**	**AV Estimate** with 95% CI
Star Wars	0.84	0.90	0.835 [0.774,0.897]
Mr. Bean	0.76	0.86	0.817 [0.634, 1.000]
James Bond: Goldfinger	0.89	0.86	0.851 [0.669, 1.034]
Jurassic Park	0.79	0.81	0.850 [0.6667, 1.033]
Terminator II	0.77	0.83	0.838 [0.732,0.943]
ATP Tennis Final 94	0.77	0.85	0.884 [0.778, 0.989]
Soccer World Cup Final 94	0.57	0.79	0.791 [0.685, 0.896]
Combination VBR Data	0.71	0.85	0.811 [0.629, 0.994]
White Gaussian Noise	0.47	0.51	0.495 [0.469, 0.522]

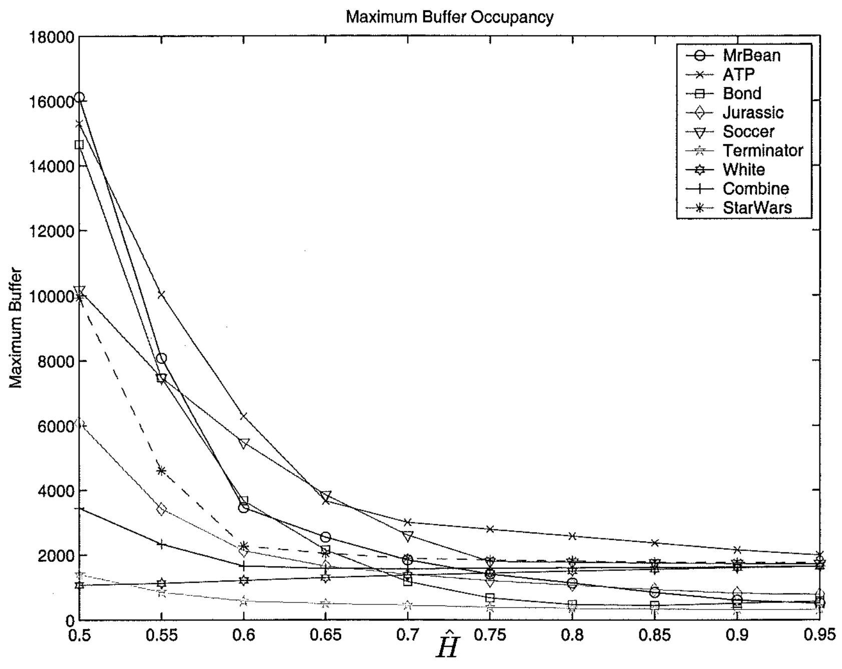


Figure 5.6 The maximum queue length against the Hurst parameter estimate.

predictors for the simulation model were developed across the entire interval of possible estimates, $\hat{H} \in [0.5, 1.0)$. The system was simulated using each predictor.

The simulation results are shown in Figures 5.6–5.8. Two performance metrics are used to test the sensitivity of the system to the Hurst parameter estimates: (1) the maximum queue length, for the situation where the congested node has infinite buffer capacity, and (2) the cell loss rate, where there is a limit on the queue length. It is observed from the simulation results that if the background traffic is self-similar, then a similar sensitivity to the estimate of the Hurst parameter is observed when the parameter is significantly underestimated. What this means is that the controllers have assumed that the traffic is better predicted using the mean. However this results in the performance of the network being reduced. If the Hurst parameter is estimated more accurately, the performance can be improved by an order of magnitude. The only data set where this is not observed is for the *White Gaussian Noise* data set, where there is a marginal decrease in performance as the Hurst parameter moves away from 0.5, which is the true value. These two observations correspond with the conclusions from the analytical study in Section 5.2.1. The observation of most serious consequence is that the sensitivity of the control algorithm increases when the Hurst parameter of a highly self-similar process is underestimated.

It should be noted that the algorithm's improved performance does not occur at the cost of reducing throughput of the rate controlled connections. It is possible for an algorithm to reduce queue lengths and cell losses within a network by suppressing the throughput of controlled connections. This leads to a deceptive perspective of perfor-

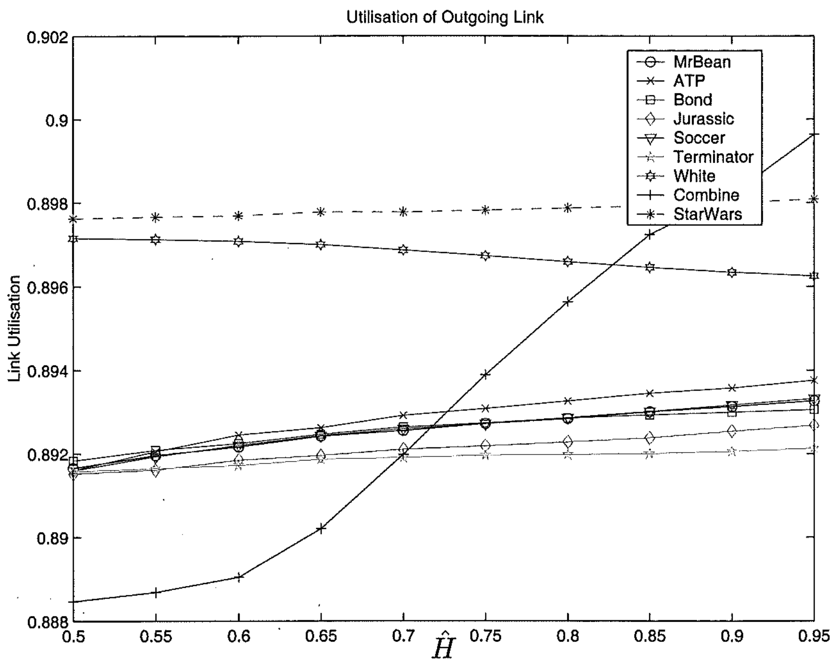


Figure 5.7 The utilisation of the output link with an infinite buffer.

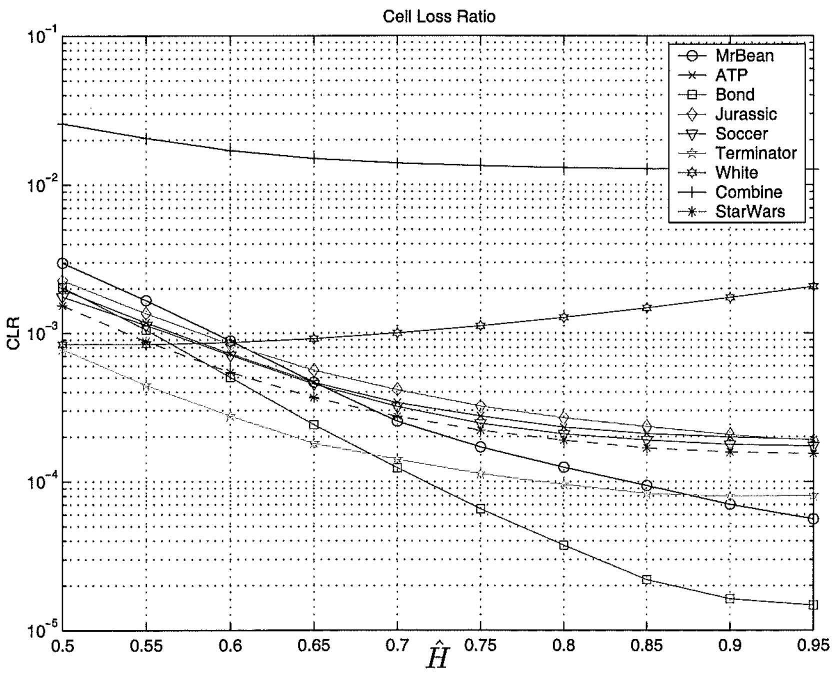


Figure 5.8 The Cell Loss Ratio, for the system with a finite buffer of maximum capacity 100 cells, versus the Hurst parameter estimate.

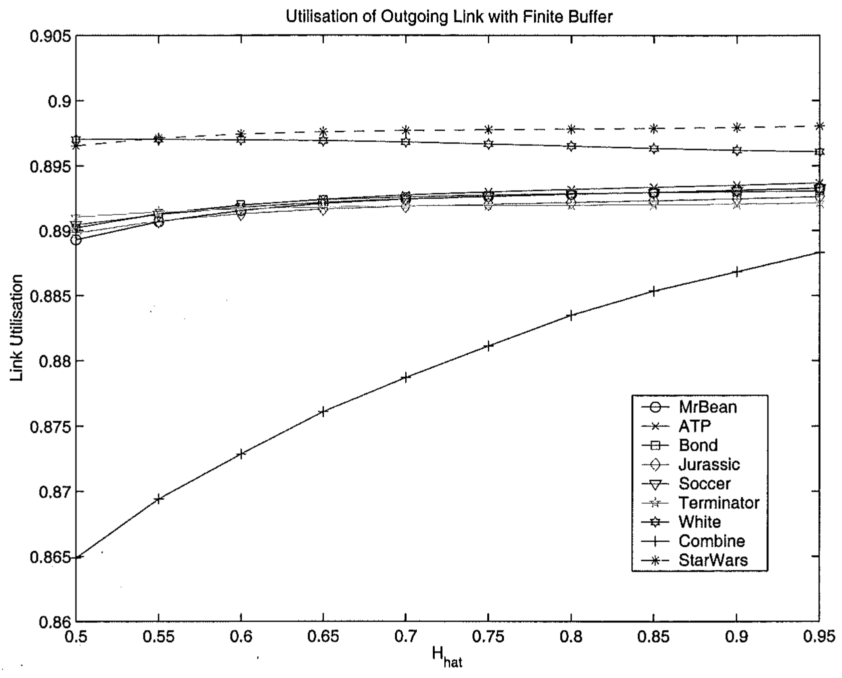


Figure 5.9 The utilisation of the output link with a finite buffer.

mance improvement. Instead, PERC+ maintains throughput of the ABR connections. This can be observed from Figures 5.7 and 5.9, where the link utilisation remains approximately constant as the queue lengths and cell losses are reduced.

While these simulation results do not prove the self-similarity of the data sets that were used as background traffic (this is not the purpose of this research), two main conclusions can be drawn. Firstly, significant performance gains can be achieved by incorporating the stochastic structure of LRD into rate-control mechanisms. This allows network management algorithms to utilise the available resources with smaller margins of error and still provide the QoS which has been guaranteed to the users. This is important within a network which is providing a service framework with different levels of QoS offered depending on the service that a user has subscribed to. This conclusion also implies the converse, that QoS will be degraded if self-similarity is not taken into account.

The second conclusion, is that the sensitivity surface suggests that robust performance can be achieved by an appropriate choice of the Hurst parameter for developing the predictors in PERC. This robust solution is based on optimising the performance of the system across the interval in which there is a high probability that the Hurst parameter lies. This is particularly important in the situation where online estimates of the Hurst parameter are being used. In this situation, wider confidence intervals can result because the parameter is being estimated reasonably frequently from smaller data sets. Rather than having the performance of a system change rapidly, values can be

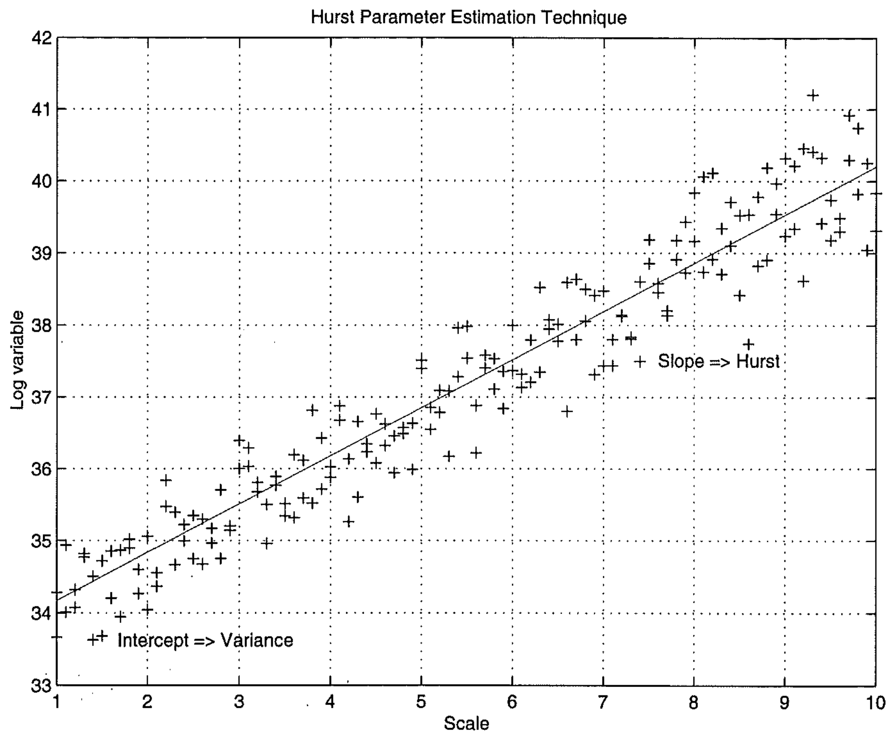


Figure 5.10 A generic joint parameter estimator for the variance and the Hurst parameters using linear regression.

chosen that result in more robust, stable performance. This approach is investigated in Chapter 6.

5.3 VARIANCE PARAMETER SENSITIVITY

Estimation of the variance parameter for a long-range dependent process is closely associated with the estimation of the Hurst parameter. Indeed, both parameters are estimated by a linear regression of the variance at different scales, where the Hurst parameter is the related to the slope and the variance of the process is related to the intercept. Thus, these parameters are estimated jointly from network traffic measurements [Veitch and Abry 1999], and the estimation of one parameter will affect the estimate of the other parameter. This interrelationship is shown in Figure 5.10 for a generic estimation technique, with the estimation of both parameters varied in Figure 5.11.

While the linear relationship that is fitted to the data defines the estimate of both the variance and the Hurst parameter for the resulting model, it is only the slope of this fit which determines the performance of the predictors which are the core of PERC. This is because the predictor coefficients are based on the correlation structure of the estimated model, and not on the absolute variance. The performance of the predictor is based on the use of the correlations within the traffic process to reduce the variance of future traffic levels of interest.

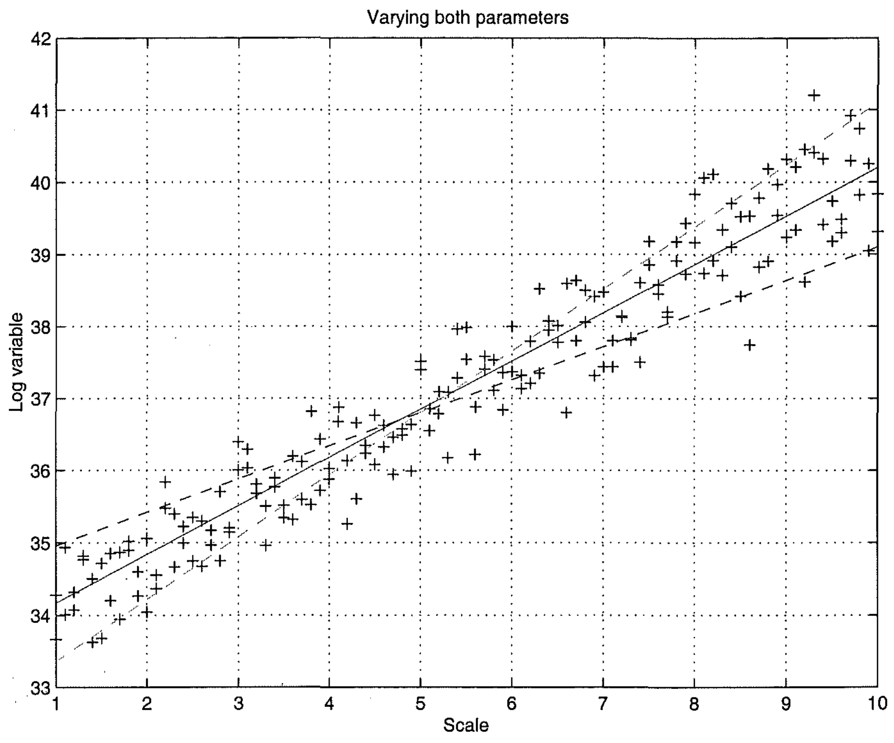


Figure 5.11 Potential linear fits to a data set, resulting in variations in parameter estimates.

However, the variance of the underlying noise process which "drives" the stochastic process is a fundamental limit to the performance of the predictor. Thus, the variance can be estimated incorrectly, but this estimate will not affect the variance of the predictor errors. It is the estimated correlation structure that will be able to effect the variance of the output of the predictor. This is demonstrated using the relative variance, Equation (5.7), for the situation where the autocorrelation function is mismatched only in terms of the variance parameter and not the Hurst parameter. These surfaces are given in Figures 5.12, 5.13 and 5.14; and show that the sensitivity of the predictors to inaccurate estimates of the variance is negligible.

5.4 MEAN PARAMETER SENSITIVITY

The final parameter which PERC requires is the mean traffic level. The development of the algorithm assumed that the mean was known, even if it was time-varying. Time variations in the mean are primarily due to changes in the number of connections currently using the network, and in general this is not known specifically. Hence, the mean parameter needs to be estimated as well. In Chapter 3, a general linear mean estimator was introduced and it was shown that the performance of the sample mean is very close to the optimal BLUE estimator. However, since the sample mean is computationally very simple to implement, it is the preferred estimation technique and it has been included

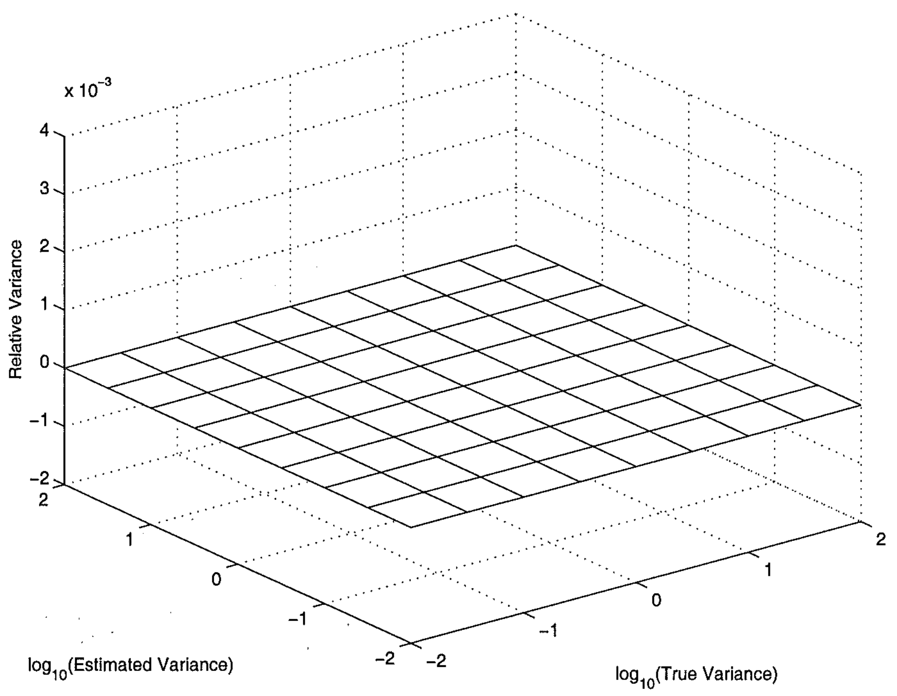


Figure 5.12 The sensitivity surface for the variance estimator, with the Hurst parameter $H = 0.5$.

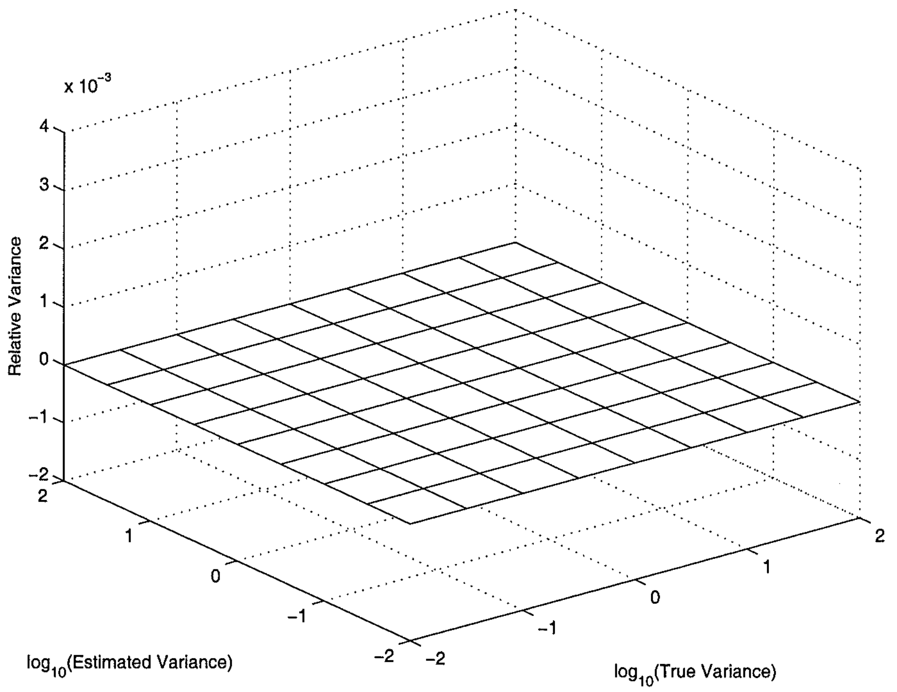


Figure 5.13 The sensitivity surface for the variance estimator, with the Hurst parameter $H = 0.75$.

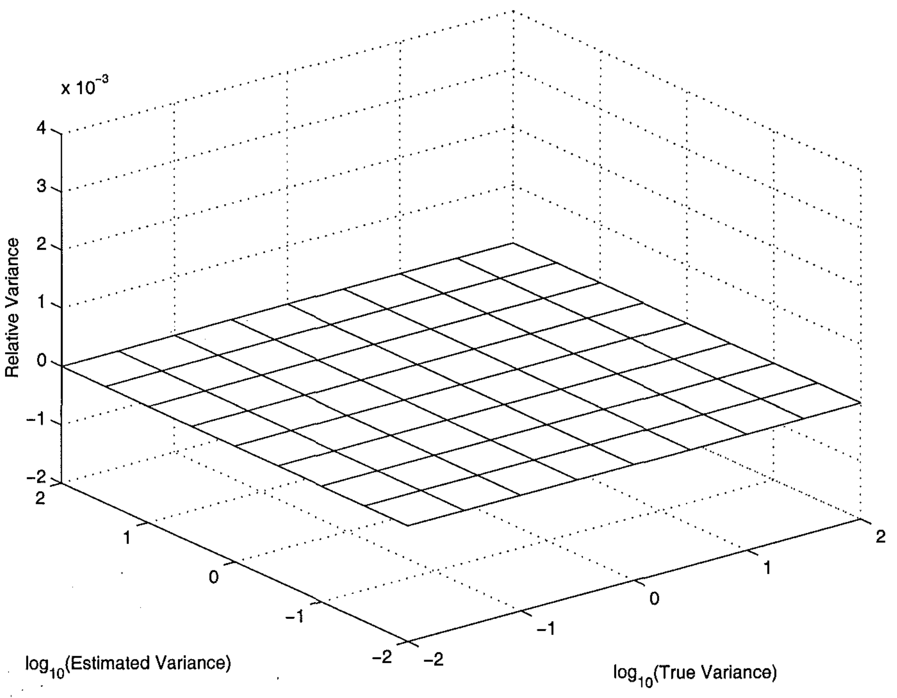


Figure 5.14 The sensitivity surface for the variance estimator, with the Hurst parameter $H = 0.99$.

in the on-line version of the rate-control algorithm in Chapter 4.

5.4.1 Sensitivity Analysis

In this section, the impact of an estimated mean parameter on the prediction of the background traffic is analysed. Due to the high performance of the sample mean, it is anticipated that the sensitivity of the predictors to the sample mean estimator will not be highly significant. This is shown in the following analysis.

The estimated mean can be represented as the true mean with an additive deviation error

$$\hat{\mu}_b(k) = \mu_b^* + \epsilon_\mu(k), \tag{5.16}$$

where $\epsilon_\mu(k)$ has zero mean, variance $\sigma^2_{\hat{\mu}}$ which is equivalent to the variance of the estimated mean and has long-range correlations due to the LRD of the original traffic. Using this estimate for the traffic mean, the predicted future traffic level becomes:

$$\begin{aligned}\hat{R}_b(k+\delta) &= \hat{\mu}_b + \mathbf{G}'_\delta\left[\mathbf{R}_\mathrm{b}(k) - \mathbf{1}\cdot\hat{\mu}_b\right] \\ &= \mu_b^* + \epsilon_\mu(k) + \mathbf{G}'_\delta\left[\mathbf{X}_\mathrm{b}(k) - \mathbf{1}\cdot\epsilon_\mu(k)\right],\end{aligned} \tag{5.17}$$

where $\mathbf{X}_\mathrm{b}$ has zero mean by mean-shifting the original traffic data $\mathbf{R}_\mathrm{b}(k)$. The expected value of $\hat{R}_b(k+\delta)$ is the mean of the traffic μ_b^*, since ϵ_μ and $\mathbf{X}_\mathrm{b}(k)$ both have zero mean. Also, the error in the expected mean can be written in terms of the variable X_b: $\epsilon_\mu(k) = w'\mathbf{X_b}(k)$. The variance of the prediction errors can be derived as follows:

$$\begin{aligned}
v_{\hat{\mu}} &= Var(\hat{R}_b(k+\delta) - R_b(k+\delta)) \\
&= \mathbf{E}\left[\left(\epsilon_\mu(k) + \mathbf{G}'_\delta\left[\mathbf{X_b}(k) - \mathbf{1}\cdot\epsilon_\mu(k)\right] - X_b(k+\delta)\right)^2\right] \\
&= \mathbf{E}\left[\left\{w'\mathbf{X_b}(k) + \mathbf{G}'_\delta\left[\mathbf{X_b}(k) - \mathbf{1}\cdot w'\mathbf{X_b}(k)\right] - X_b(k+\delta)\right\}\right. \\
&\quad \left.\cdot\left\{w'\mathbf{X_b}(k) + \mathbf{G}'_\delta\left[\mathbf{X_b}(k) - \mathbf{1}\cdot w'\mathbf{X_b}(k)\right] - X_b(k+\delta)\right\}'\right] \\
&= \mathbf{E}\left[w'\mathbf{X_b}(k)\mathbf{X_b}'(k)w\right] + \mathbf{E}\left[\mathbf{G}'_\delta(\mathbf{X_b}(k) - \mathbf{1}\cdot w'\mathbf{X_b}(k))(\mathbf{X_b}(k) - \mathbf{1}\cdot w'\mathbf{X_b}(k))'\mathbf{G}_\delta\right] + \mathbf{E}\left[X_b^2(k+\delta)\right] \\
&\quad + 2\mathbf{E}\left[w'\mathbf{X_b}(k)(\mathbf{X_b}(k) - \mathbf{1}\cdot w'\mathbf{X_b}(k))'\mathbf{G}_\delta\right] - 2\mathbf{E}\left[w'\mathbf{X_b}(k)X_b(k+\delta)\right] \\
&\quad - 2\mathbf{E}\left[\mathbf{G}'_\delta\left[\mathbf{X_b}(k) - \mathbf{1}\cdot w'\mathbf{X_b}(k)\right])X_b(k+\delta)\right] \\
&= w'\Gamma^{-1}w + \mathbf{G}'_\delta\mathbf{E}\left[(\mathbf{X_b}(k) - \mathbf{1}w'\mathbf{X_b}(k))(\mathbf{X_b}'(k) - \mathbf{X_b}'(k)w\mathbf{1}')\right]\mathbf{G}_\delta + \sigma_b^2 \\
&\quad + 2\mathbf{E}\left[w'\mathbf{X_b}(k)(\mathbf{X_b}'(k) - \mathbf{X_b}'(k)w\mathbf{1}')\right]\mathbf{G}_\delta - 2w'\gamma_\delta - 2\mathbf{G}'_\delta(\gamma_\delta - \mathbf{1}w'\gamma_\delta) \\
&= \sigma_b^2 + w'\Gamma^{-1}w + \gamma'_\delta\Gamma^{-1}\left(\Gamma - \Gamma w\mathbf{1}' - \mathbf{1}w'\Gamma + \mathbf{1}w'\Gamma w\mathbf{1}'\right)\Gamma^{-1}\gamma_\delta \\
&\quad + 2w'\left(\Gamma - \Gamma w\mathbf{1}'\right)\Gamma^{-1}\gamma_\delta - 2w'\gamma_\delta - 2\gamma'_\delta\Gamma^{-1}\left(\gamma_\delta - \mathbf{1}w'\gamma_\delta\right).
\end{aligned} \tag{5.18}$$

Since the relative increase in the variance of the prediction errors is

$$K_{\hat{\mu}} = \frac{v_{\hat{\mu}} - v^*}{v^*}, \tag{5.19}$$

Theorem 7 follows immediately.

Theorem 7 *The relative increase in variance of the prediction errors resulting from a linear predictor which utilizes an estimate of the traffic mean is given by:*

$$\begin{aligned}
K_{\hat{\mu}} = &\left\{\sigma_{\hat{\mu}}^2 + \gamma'_\delta\Gamma^{-1}\left(\Gamma - 2\Gamma w\mathbf{1}' + \mathbf{1}w'\Gamma w\mathbf{1}'\right)\Gamma^{-1}\gamma_\delta\right. \\
&+ 2\gamma'_\delta\Gamma^{-1}\left(\Gamma - \mathbf{1}w'\Gamma\right)w - 2w'\gamma_\delta - 2\gamma'_\delta\Gamma^{-1}\left(\gamma_\delta - \mathbf{1}w'\gamma_\delta\right) \\
&\left.+\gamma'_\delta\Gamma^{-1}\gamma_\delta\right\} / \left(\sigma_b^2 - \gamma'_\delta\Gamma^{-1}\gamma_\delta\right)
\end{aligned} \tag{5.20}$$

The sensitivity of the congestion control algorithms to the sample mean is analytically investigated in Figures 5.15 and 5.16 where the fractional Gaussian noise model has been used. Figure 5.15 shows the relative increase in variance for different Hurst parameter values as the length of data used to calculated the sample mean is increased. It can be seen that the sensitivity rapidly decreases and is, for all practical purposes, very close to zero with data lengths above $2^6 = 64$ samples. This data length is used for estimating the sample mean, and the relative increase in variance as the round-trip delay of the connections is increased is shown in Figure 5.16. Firstly, it should be ob-

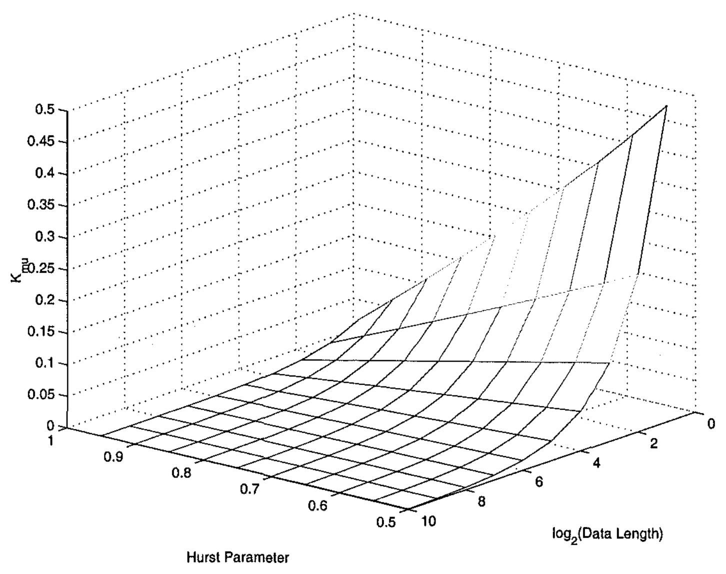


Figure 5.15 The sensitivity of linear predictors to the length of data used to calculate the sample mean.

served from the vertical axis (Figure 5.16) that the relative increase in the variance of the prediction errors is very small, with the maximum being below 0.06 even for relatively large round-trip delays of 64 sampling time units. Secondly, this maximum occurs approximately where the efficiency of the sample mean is minimum (see Figure 3.6).

The conclusion that can be drawn from this analytical study is that the effect of using the sample mean is negligible on the performance of the background traffic predictors when a reasonable number ($M > 64$) of traffic samples are used. Furthermore, sensitivity does not increase with round-trip delay either. With the increased width of the confidence intervals, which are associated with using the sample mean, it is not initially obvious why the predictors are relative insensitive to the sample mean. However, it can be explained firstly by referring to the efficiency of the sample mean, which is theoretically very high, and secondly by the long-range dependence of the background traffic, which has the effect of increasing the correlation between the sample mean and future traffic levels. This means that the sample mean reflects the long-term trends of the traffic levels more accurately and hence results in a very similar performance compared with predictors that have access to the true traffic mean.

If the mean is non-stationary, the variance in the estimation of the mean will increase particularly in intervals where the mean is changing significantly. During these transitional periods, the prediction errors will increase significantly. However, the analysis in this section is valid for finite intervals where the mean is stationary, for example, situations where there are level changes in the mean [Duffield and Whitt 1997]. PERC will only become sensitive to mean estimates if transitions occur too frequently, or the actual mean changes continuously during the estimation process.

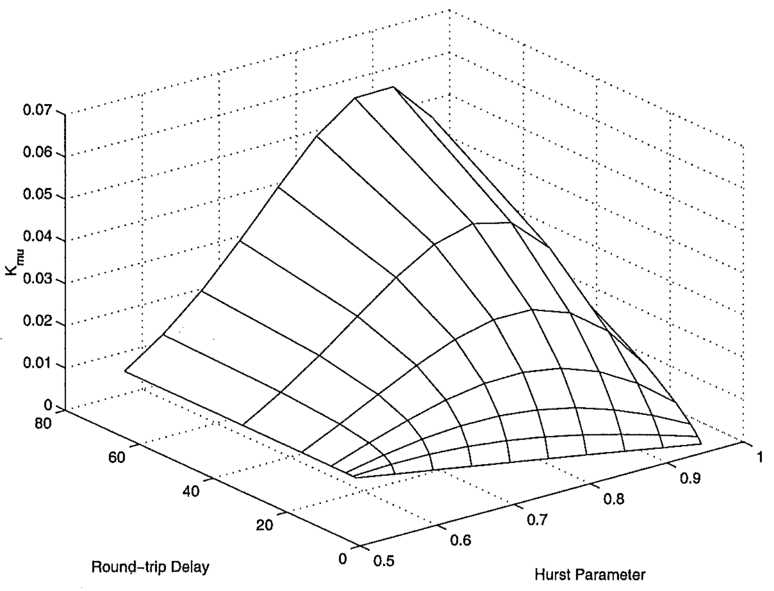


Figure 5.16 The variation in the sensitivity of linear predictors incorporating the sample mean with increasing round-trip delay of the rate-controlled connections.

5.4.2 Peformance Evaluation Using Simulations

Using the same simulation model as shown in Figure 4.6, the sensitivity of the system to the sample mean is investigated. The length of data, that is used to calculate the sample mean, is plotted on the horizontal axis in Figures 5.17 and 5.18. The parameters, maximum queue length and cell loss ratio, are used to measure the performance of the system using these mean estimates. The cell loss ratio has been calculated for a buffer with a maximum queue length of 100 cells.

As can be seen from Figures 5.17 and 5.18, there is no significant change as the length of data used to calculate the sample mean is increased. In the case of the buffer with infinite memory resources, maximum queue length does vary with the data length, but there is no clear trend in the variations and the performance of the system remains within the same order of magnitude as the sample mean length is increased. For the situation with a finite buffer, the relative constancy of performance, in terms of cell loss ratio, is even more apparent across all the sample mean lengths. This means that the traffic mean can be estimated from relatively short lengths of data. The non-stationarities in the mean can be tracked, as long as the mean does not change significantly or frequently within the interval in which the sample mean is calculated. The significance of this result is that there is little loss in performance when the sample mean is used as an estimate for the traffic mean.

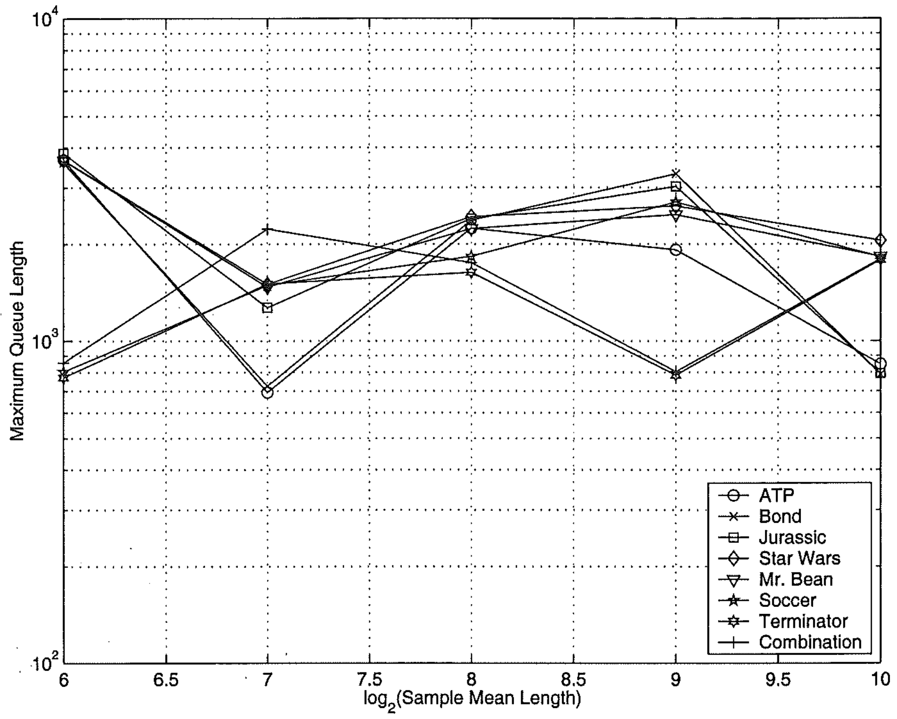


Figure 5.17 The maximum queue length at the node versus the length of data used to estimate the sample mean.

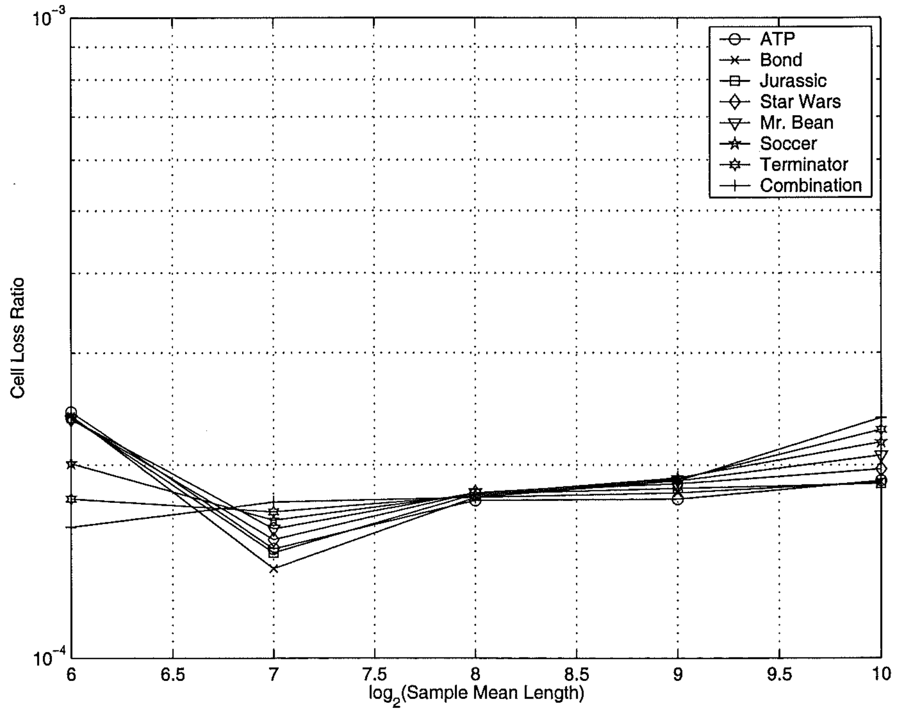


Figure 5.18 The Cell Loss Ratio, for the system with a finite buffer of maximum capacity 100 cells, versus the length of data used to estimate the sample mean.

5.5 DISCUSSION

In this chapter, the rate control algorithm PERC+ that is presented in Chapter 4 has been shown to be most sensitive to the Hurst parameter, among the three principle traffic parameters including Hurst parameter, variance and mean. This is not surprising, since the predictors that form the core of the rate control algorithm use the correlation structure of background traffic to predict transmission rates for ABR sources. It is the Hurst parameter that captures the structure of the autocovariance function for traffic measurements. In contrast, variance reflects the absolute power of the background traffic, and any performance gain achieved through prediction arises from using the correlation structure. Finally, the mean traffic level is well characterised by the sample mean estimate, as has been shown previously [Beran 1994], and simulation results confirm that good performance can be achieved using the sample mean.

Any study on sensitivity, particularly with regards to a certain parameter, which happens to be the Hurst parameter in this case, leads to the question of whether algorithms can be designed so that the sensitivity is reduced or avoided. Due to the particular form of sensitivity, which is an asymmetry towards underestimation of the Hurst parameter, robustness can be achieved and this is the study of the following chapter, Chapter 6. The aim of the approach proposed in Chapter 6 in achieving robustness is to determine a solution that results in close-to-optimal performance independent of uncertainty that occurs in estimating the Hurst parameter.

The sensitivity analysis reported in this chapter highlights the dependence of network performance on the estimation of the Hurst parameter. Parameter estimates that underestimate the true Hurst parameter results in higher variablity in the transmission rates of controlled sources, and subsequently longer queue lengths and greater cell losses. It should be noted that a limitation of only using the Hurst parameter to characterise the autocovariance function is that the influence of any SRD is not accounted for. The results remain valid for any exactly self-similar process which can be parsimoniously modelled using the Hurst parameter, variance and mean. However, further work involves investigating the influence of SRD and LRD on the performance of predictive rate-control mechanisms. This area will be addressed in Chapter 7.

Chapter 6

DESIGNING RATE CONTROL FOR ROBUST PERFORMANCE

Uncertainty in the exact nature of network traffic is inevitable. This is because there is an inherent randomness in the microscopic processes which give rise to the aggregate traffic patterns observed within the network. The arrival patterns of users entering the network, and amount of bandwidth or the length of the connection all vary significantly and unpredictably. This has been very successfully modelled using stochastic processes which capture the inherent randomness of network traffic. Poisson models are adequate for describing telephone connections, and have been used to design networks for many years [Schwartz 1987, Kleinrock 1975]. Quality of service involved minimising the blocking probabilities obtained from Poisson models. As has been presented in this thesis, many applications that will use an integrated network generate traffic that is LRD. These applications include data, multi-media and video, and it is important to consider a heterogeneous mixture of traffic that occurs within a network that provides a wide range of services.

However, stochastic models cannot capture all of the uncertainty that occurs in network traffic. A single model with well-defined moments and correlation structure will inevitably become obsolete due to the time-variations in the traffic. In addition, within an integrated services network, traffic of a fundamentally different nature may proceed to utilise network resources. Thus, a traffic management algorithm that has been designed for a particular type of traffic may easily become mismatched with the traffic that it encounters within the network. This will lead to increased delays and losses within the network, unless the algorithm has been designed to be robust to the changes that may occur. It follows that meeting the QoS requirements of network traffic means that algorithms have to be designed in the wider context of variations that occur in time and in the nature of network traffic. Even if the traffic is relatively time-invariant, estimating traffic parameters involves a certain amount of uncertainty, where confidence intervals give a measure of that uncertainty. The accuracy of estimation techniques depends on the amount of data that is used for estimating the parameters, and if on-line approaches are used with smaller blocks of data being used by estimators, the uncertainty can increase.

Accommodating modelling uncertainty within a control system can be approached using robust control techniques. The purpose of robust control is to incorporate the uncertainty within the design of the controller so that the system will perform as expected, regardless of changes that occur within the system [Green and Limebeer 1995]. There is much theoretical work on robust control [Green and Limebeer 1995], with H_∞-control being the foundational theory behind it. The basic principle behind this approach is that a bound on the model uncertainty is identified, and then the control design is optimised for this bound. The controller is thus designed for the worst case situation, which ensures that the controller performs within expectation for any other situation it may encounter.

The results on the sensitivity of the rate-control algorithms in Chapter 5 suggested that robust control designs could also be achieved for predictive control algorithms for long-range dependent traffic. The design of predictive rate control algorithm for robust performance is investigated in this chapter.

6.1 ROBUST PREDICTION USING MAXIMUM ENTROPY

The traditional approach to robust prediction is to specify the uncertainty of the system by a set of possible models, define an appropriate metric and then optimise across this set in such a way that the solution is maximally degenerate [Frieden 1985]. In other words, the worst-case situation is considered and then the predictor design proceeds by optimising for this situation. Since the theory is relevant and an appropriate link to the approach that will be proposed in this chapter, the theory underpining the approach is introduced [Poor 1999].

6.1.1 Defining Model Uncertainty

A zero-mean random variable X_k is considered which is defined on the measurable space $(\Omega, \mathcal{F})$. The correlation structure of X_k is defined by

$$E[X_{k+n}X_k] = \gamma(n), \text{ for } |n| = 0, 1, \ldots, p, \tag{6.1}$$

which is specified only up to lag p. The correlation structure can be used to form the positive definite correlation matrix Γ. The uncertainty is defined by allowing the covariance segment γ to range within the convex set $\mathcal{C}$ of p-lag covariance segments. Now, the minimax game at the fixed point k can be considered:

$$\min_{\hat{X}\in\mathcal{F}_k} \max_{P\in\mathcal{P}_c} \int_\Omega |\hat{X} - X_{k+\delta}|^2 dP, \tag{6.2}$$

where P is a model that has correlation structure given by (6.1). There exists a saddlepoint solution $(\hat{X}^*, P^*)$ that minimises the mean square prediction error, with the minimum being given by

$$\epsilon^*(\gamma) = \min_{\hat{X} \in \mathcal{F}_k} \int_\Omega |\hat{X} - X_{k+\delta}|^2 dP^*, \tag{6.3}$$

for the least favourable covariance segment γ^*

$$\gamma^* = \arg\max_{\gamma \in \mathcal{C}} \epsilon^*(\gamma). \tag{6.4}$$

P^* is the autoregressive model of order p having the autocovariance segment γ^*. The actual predictor is a linear predictor of length p and is given by the solution of (3.47). This can be interpreted as the best predictor for the worst case of uncertainty in the set of models for the random variable. Since the predictor has been optimised for the worst case of model uncertainty, the mean square prediction errors for all the other cases will be lower. This is similar in concept to H_∞-control approach for the design of robust controllers.

6.1.2 Aspects of Robustness

Since linear predictors are being considered, the minimax problem can be specified in a more restricted sense as

$$\min_{g \in \mathcal{G}} \max_{\gamma \in \mathcal{C}} \int_\Omega |g'X|^2 dP \tag{6.5}$$

where $\mathcal{G}$ is the set of vectors of length $(p+1)$ where the first component is unity. The mean square prediction error in this instance is given by

$$\epsilon(g, \gamma) = g'\Gamma g = \int_\Omega |g'X|^2 dP. \tag{6.6}$$

In the frequency domain, this approach to developing the robust predictor results in maximising the spectral entropy

$$S(f) = \frac{1}{2\pi} \int_{-\pi}^{\pi} \log f_P(\lambda) d\lambda. \tag{6.7}$$

The process is then sufficiently constrained to lie within the class of models that fit the criteria that the autocovariance segment lies in the set $\mathcal{C}$.

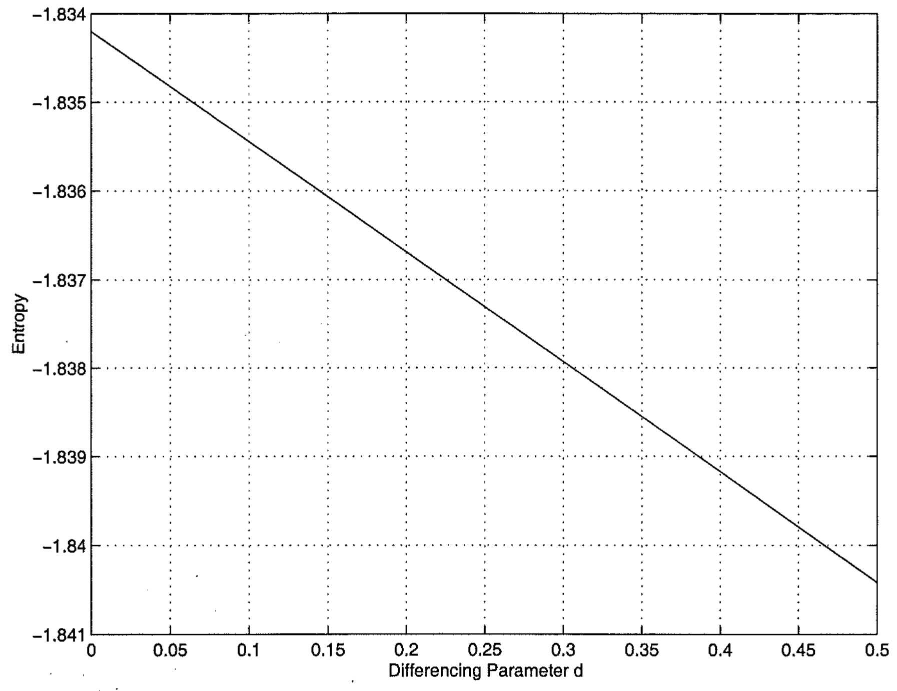


Figure 6.1 Entropy for the fractional ARIMA$(0, d, 0)$ process as the parameter d is varied.

6.2 LIKELIHOOD AND MAXIMUM PERFORMANCE

There is difficulty in applying the approach of maximising spectral entropy to the robustness problem for LRD processes. Firstly, the convexity of a set of long-range dependent autocovariance functions is difficult to prove. Secondly, even if the entropy of simple long-range dependent processes is considered, using the entropy of fractional ARIMA $(0, d, 0)$ and $(1, d, 0)$ processes as examples as shown in Figures 6.1 and 6.2, it can be observed that the entropy is maximised when the differencing parameter is 0. Converting this to the Hurst parameter, the classic robustness approach says that the robust solution occurs when the Hurst parameter is 0.5. This appears to completely contradict the study into the sensitivity of the control mechanisms to the Hurst parameter in Section 5.2. Specifically, the worst sensitivity was observed when the Hurst parameter was chosen to be 0.5 (Section 5.2.1).

A note before continuing—many readers who are familiar with entropy of discrete random variables, for example in the context of information theory [Haykin 1994], may wonder why entropy is negative in Figures 6.1 and 6.2. In this case, negative entropy results from LRD processes being continuous random variables, which can have either positive or negative entropy [Poor 2000]. The sign depends specifically on the scale of the process, and the only constraint on the range is that entropy must be less or equal to the log of the power in the process (via Jensen's inequality [Mansuripur 1987]). In contrast, scaling the magnitude of discrete random variables has no effect on the probabilities of the variables, so that entropy is always positive. However, the concept

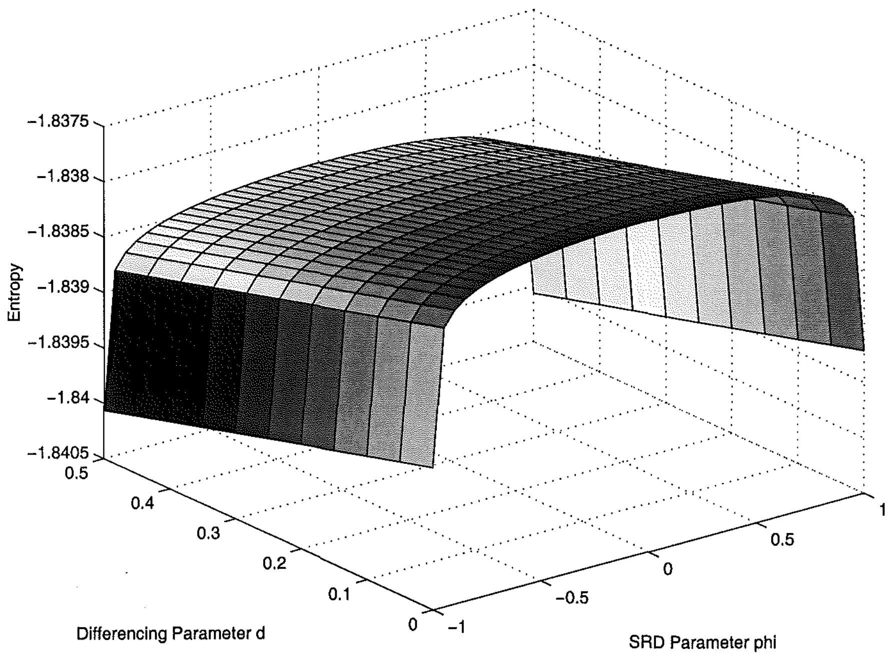


Figure 6.2 Entropy for the fractional ARIMA$(1, d, 0)$ process as the parameters d and ϕ are varied.

of maximising entropy has the same meaning in either case, so maximum entropy still implies the greatest uncertainty.

6.2.1 Analysing the Maximum Entropy Solution

The apparent contradiction, between the maximum entropy solution and the sensitivity results in Chapter 5, is resolved when the true nature of maximum entropy algorithms are understood [Frieden 1985]. These algorithms give the optimal solution when the user has maximum ignorance regarding the biases within the model structure. They actually give the maximally degenerate solution. While this is the optimal solution given maximum ignorance, it is not the solution which will result in the best network performance for all possible values of the Hurst parameter.

Robustness in this sense is guaranteed by a lower bound on the performance of the system, which is achieved by optimizing for the worst case situation. This is shown more clearly in Figure 6.3 which shows the absolute variance of the prediction error for predictors which are mismatched to the fractional Gaussian noise process that they are attempting to predict. The absolute variance was determined in Chapter 5, and is given in Equation 5.5. Robust prediction determines the model which has most uncertainity, and then optimises performance based on this model. The model with the greatest amount of uncertainity has Hurst parameter $H^* = 0.5$ and by minimising the variance of the prediction for this case, the prediction error variance for all other Hurst parameter values is guaranteed to be lower.

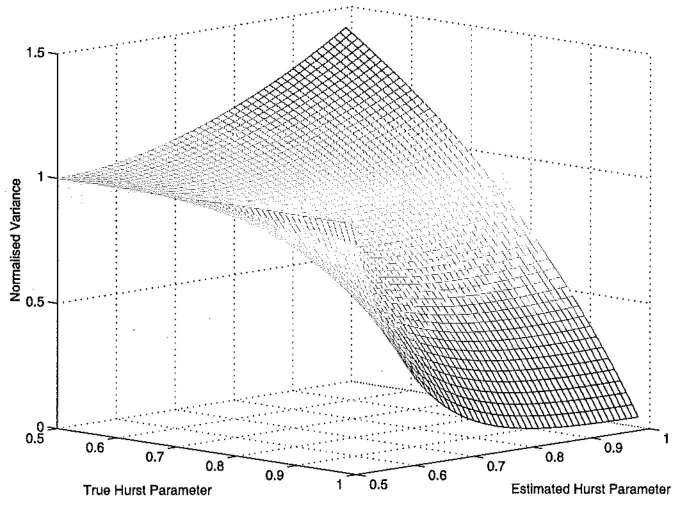


Figure 6.3 The absolute variance of the prediction errors for the fractional Gaussian model.

However, this does not ensure that the system performance is the best across the range of model uncertainties. In fact, if the Hurst parameter is closer to 1.0 significantly better performance could be achieved. In the very apt words of Frieden—"maximum entropy not only gives the wrong answer, it gives an answer that is very far from right" [Frieden 1985]! Thus, there is a trade-off between achieving robustness, in the sense of optimising for the greatest uncertainty, and achieving best performance. It would seem fairly obvious that in the context of network management that one should aim for robust performance [Zhou *et al.* 1994, Bu and Sznaier 1997], rather than robustness itself.

6.2.2 Robustness with the Likelihood Function

It is proposed that, for LRD processes, the use of entropy as the metric, by which robustness is determined, be replaced by the frequency domain likelihood function [Östring *et al.* 2000b]. This is the metric that is used within the Whittle's estimator which defines the proximity of a model estimate to the actual process itself, represented by a data set. The likelihood function can be maximised by minimising the following function [Beran 1994]:

$$L_W(\theta) = \frac{1}{2\pi}\left[\int_{-\pi}^{\pi} \log f(\lambda;\theta) d\lambda + \int_{-\pi}^{\pi} \frac{I(\lambda)}{f(\lambda)} d\lambda\right] \tag{6.8}$$

which is a comparative measure between a parametric spectral model $f(\lambda;\theta)$ and the true spectral density $I(\lambda)$. It can be observed from Equation (6.8) that the first term in this function is the spectral entropy. Equation (6.8) is used to test the closeness of a parametric model to the observed periodogram, with the best model maximising the likelihood function. The function thus measures the spectral distance between the observed spectral density and the one given by the model.

By discretising the frequency spectrum with

$$\lambda_{j,m} = \frac{2\pi j}{m}, \tag{6.9}$$

$j = 1, \dots, m^*$ and m^* is the integer part of $(m-1)/2$, the function $L_W(\theta)$ of Equation (6.8) can be approximated by [Beran 1994]:

$$\tilde{L}_W = \frac{1}{\pi}\left[\sum_{j=1}^{m^*} \log f(\lambda_{j,m};\theta)\frac{2\pi}{m} + \sum_{j=1}^{m^*} \frac{I(\lambda_{j,m})}{f(\lambda_{j,m};\theta)}\frac{2\pi}{m}\right]. \tag{6.10}$$

The function $\tilde{L}_W$ is shown in Figure 6.4, where the LRD parameter is the fractional differencing parameter d. Figure 6.4 shows that the behaviour of the function is asymmetric across the parameter, with the function noticeably increasing as the true value of the parameter d is underestimated. This indicates the divergence between the estimated spectral density and the true spectral density is greater in this region. Thus, in terms of performance, it is desirable to use a spectral density where the function $\tilde{L}_W$ is minimised. This conclusion is in agreement with the investigation into the sensitivity of the congestion control algorithm to the Hurst parameter (which is directly related to the differencing parameter) [Östring *et al.* 2000c]. In this research, the variance of prediction error has been used as a metric of the sensitivity, arriving at a similar result. This is understandable, since variance is equivalent to integrating the spectral density. Robustness in this sense is achieved by choosing the Hurst parameter value that achieves the minmax value for the function $\tilde{L}_W$ across the interval of possible true Hurst parameter values.

The robustness of this approach can be explained in the following way, using Figure 6.3. First, consider the maximum entropy solution. This approach seeks to find the case with the worst uncertainty, which is when the true Hurst parameter is 0.5, and then gives the robust solution as the optimal predictor for this case. Thus, the robust solution uses an estimated Hurst parameter also of value 0.5. It can be observed from Figure 6.3 that the performance for all Hurst parameters is bounded using this result, though the performance can be far from optimal, for example when one considers traffic with true Hurst parameter approaching 1.0. Label the normalised variance bound of the maximum entropy solution as $V(H_{0.5})^*$.

It can be noted that the performance for traffic with true Hurst parameter $H = 0.5$ is always worse for all predictors developed from estimated Hurst parameters in the range $\hat{H} \in [0.5, 1.0)$. If the variance bound for this worst case situation is relaxed to $V(H_{0.5})^* + \epsilon$, then it is possible to shift the performance of a robust predictor closer to the optimal performance for traffic with other Hurst parameter values, while still maintaining knowledge of the performance of the worst case. The maximum likelihood

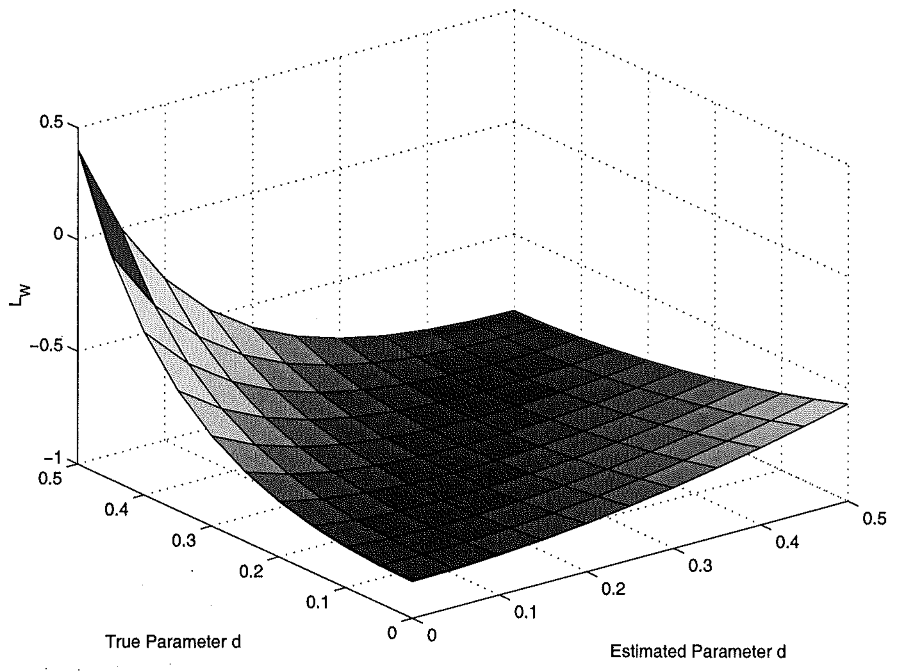


Figure 6.4 The function $\tilde{L}_W$ for the fractional ARIMA$(0, d, 0)$ process as the parameter d and is varied against its estimated value.

approach optimises this situation with the variance bound $V(H_{0.5})^* + \epsilon^*$ such the relative performance of all predictors is close to optimal. This is due to the fact that the maximum likelihood function includes a measure of the comparative proximity of a solution to the optimal solution, while the maximum entropy simply measures absolute performance. It should be emphasized that the maximum likelihood solution is still robust, as the variance of all predictors will be bounded by the well-defined bound $V(H_{0.5})^* + \epsilon^*$. The difference is that the *relative performance*, with respect to the optimal performance, of the predictor is significantly proved for all traffic with Hurst parameters in the range $H \in [0.5, 1.0)$.

6.3 SIMULATION STUDY

6.3.1 Simulation Setup

The performance of this algorithm is demonstrated using simulations. The simulation model is the same one which used in Chapter 4, as shown in Figure 4.6 in Chapter. A number of ABR connections are competing with a VBR source for bandwidth of a congested link. The control algorithm for ABR sources uses optimal prediction of the VBR source, and returns the ER values to the ABR sources. The VBR data has been generated by aggregating randomly shifted MPEG video traces [Garrett and Willinger 1994, Rose 1995], which have confidence intervals for the Hurst parameter as given

Table 6.1 Confidence Intervals (CI) for the Hurst Parameter for the VBR data sets.

VBR Data Set	95 % CI
Star Wars	(0.774, 0.897)
Mr. Bean	(0.634, 1.000)
James Bond: Goldfinger	(0.669, 1.034)
Jurassic Park	(0.667, 1.033)
Terminator II	(0.732, 0.943)
ATP Tennis Final 94	(0.778, 0.989)
Soccer World Cup Final 94	(0.685, 0.896)
White Gaussian Noise	(0.469, 0.522)

in Table 6.1. The Abry-Veitch estimator was used to generate these estimates.

6.3.2 Performance Comparison

Firstly, it is assumed that there is no *a priori* information regarding the self-similarity of the background traffic entering the congested node. The ABR rate control mechanisms is simply designed by maximizing the entropy and compare this with the proposed method of optimising the likelihood function. Thus, standard controllers are used for all the VBR traffic, with the Hurst parameter of 0.5 used for the maximum entropy case, and 0.7842 obtained for minimising the maximum value of the function $\tilde{L}_W$. The results are shown in Figures 6.5 and 6.6, which clearly indicate that the proposed robust control performs significantly better than the maximum entropy controller. Also, robust control performs close to the optimal controller for each data set, as observed in Figures 6.5 and 6.6. The only exception is the Gaussian case, but the performance of the proposed robust controller is comparable with both the maximum entropy controller and the optimal controller.

The case where estimates of the Hurst parameter have been obtained is also investigated, so that the controller is optimised over the region specified by the confidence interval, rather than the whole surface of the likelihood function. The simulation results are shown in Figures 6.7 and 6.8, and the conclusion that can be drawn is that the system for which the likelihood function is maximised performs better than the system where entropy has been maximised.

6.3.3 Varying System Parameters

The performance of the system has been investigated more extensively for three of the data sets, the *Star Wars*, *Jurassic Park* and *Gausssian* data sets by varying the link utilisation and comparing the buffering requirements for the different control algorithms. The robust solutions have been generated in the situation where no *a priori* information

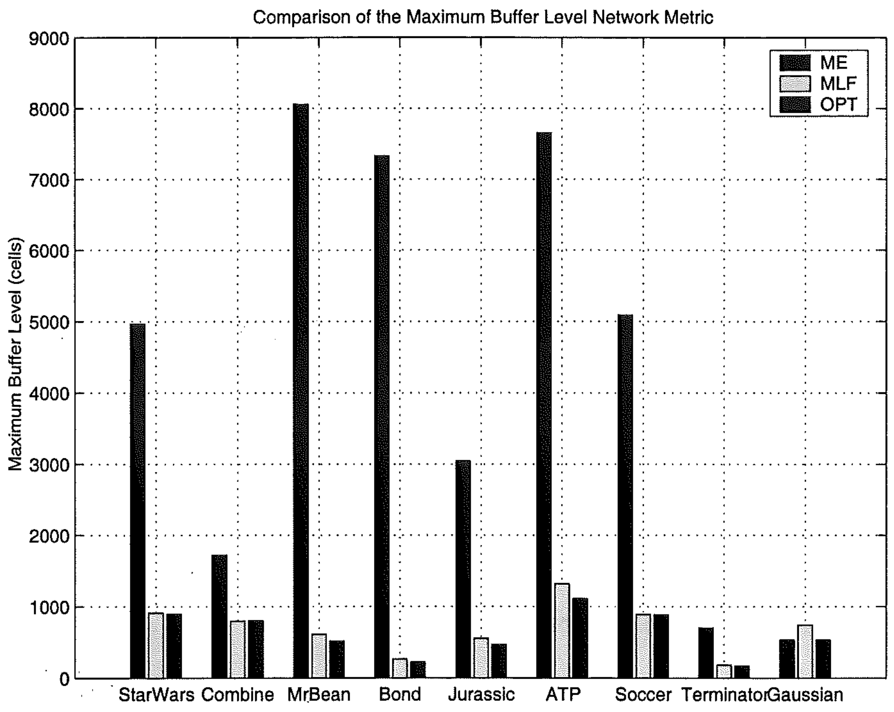


Figure 6.5 Comparison of the maximum queue lengths of the robust controllers for maximum entropy (ME) and the maximum likelihood (MLF) with no *a priori* information. The (OPT) case is for the controller designed using the optimal Hurst parameter estimate for the specific data set.

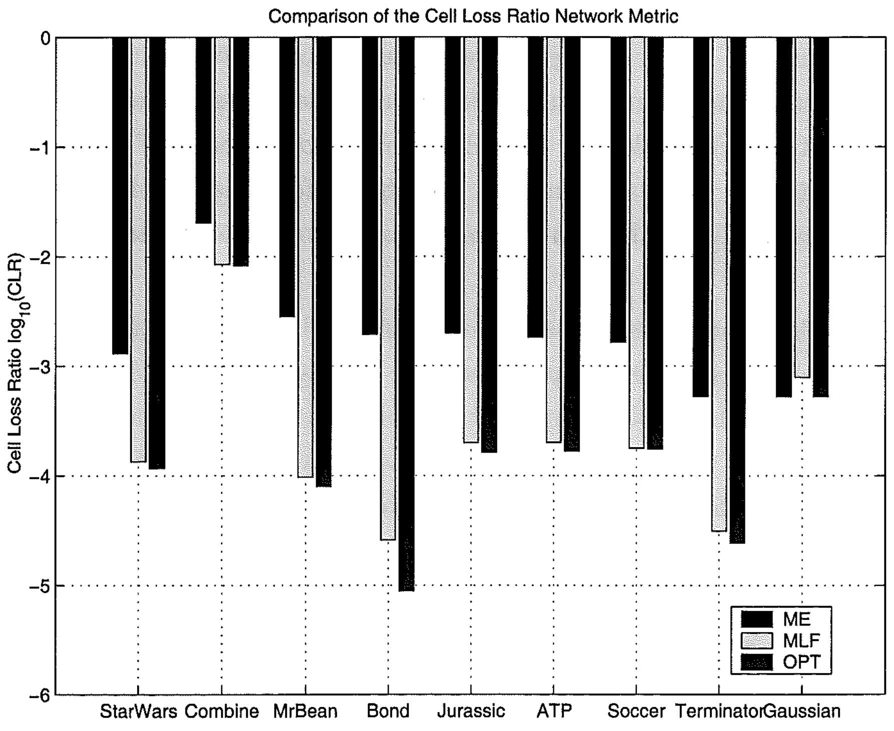


Figure 6.6 Comparison of the cell loss ratios of the robust controllers for maximum entropy (ME) and the maximum likelihood (MLF) with no *a priori* information. The (OPT) case is for the controller designed using the optimal Hurst parameter estimate for the specific data set.

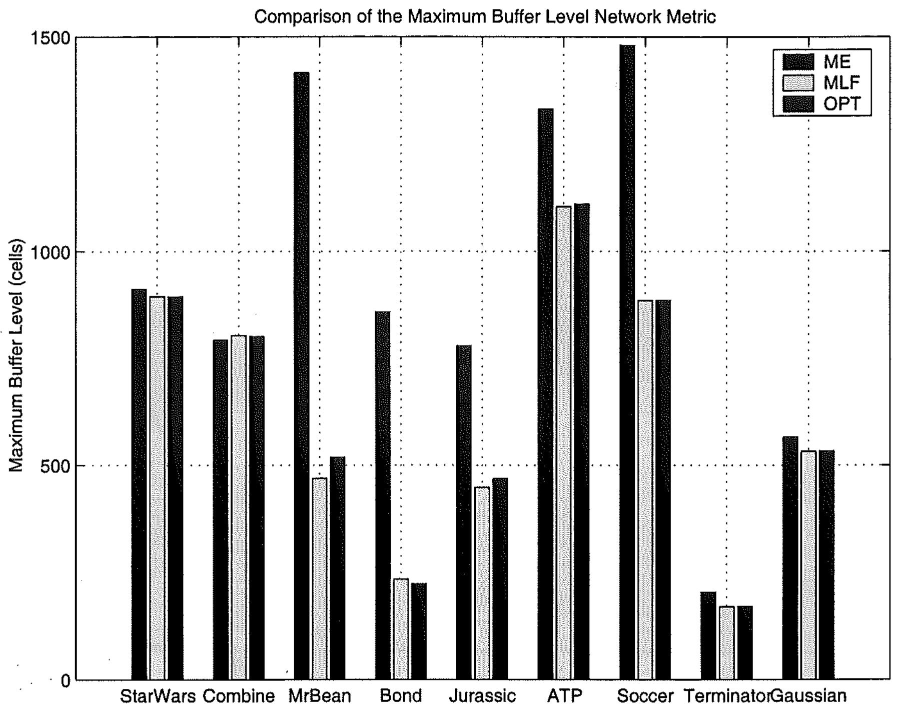


Figure 6.7 Comparison of the maximum queue lengths of the robust controllers for maximum entropy (ME) and the maximum likelihood (MLF) using CI from Table 6.1. The (OPT) case is for the controller designed using the optimal Hurst parameter estimate for the specific data set.

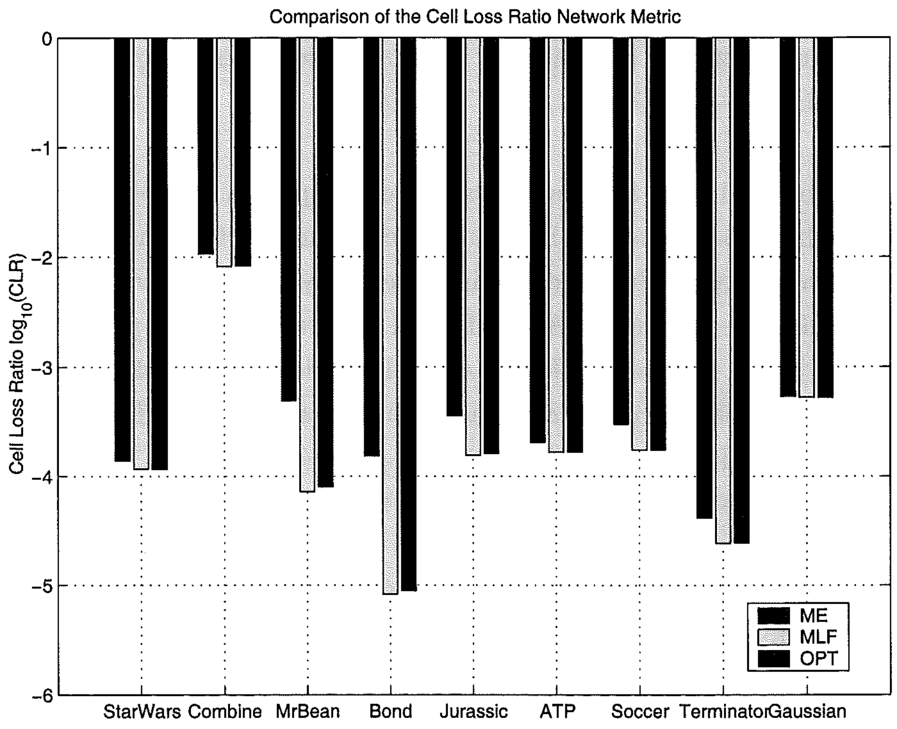


Figure 6.8 Comparison of the cell loss ratios of the robust controllers for maximum entropy (ME) and the maximum likelihood (MLF) using CI from Table 6.1. The (OPT) case is for the controller designed using the optimal Hurst parameter estimate for the specific data set.

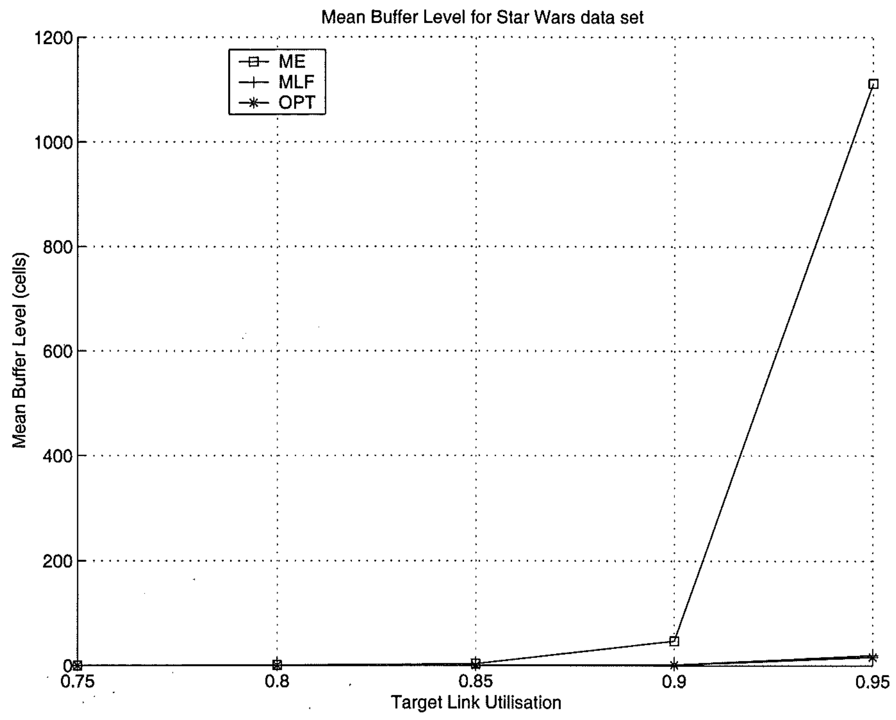


Figure 6.9 Performance Curves for Star Wars: the mean buffer occupancy as the target link utilisation is varied.

is available regarding the value of the Hurst parameter. Cell loss rates for finite buffers are compared, as the size of the buffer is varied. These performance curves are shown in Figures 6.9–6.16.

It can be observed from the graphs that robust approach proposed in this chapter, which uses the likelihood function, is close to the optimal performance for a wide range of networking conditions, while the maximum entropy solution performs much worse as the link utilisation increases. Also, the cell loss rates for the proposed approach drops off much faster, together with the optimal solution. It should be noted that the performance curves for the robust approach, which maximises the likelihood function, and the optimal solution overlap in Figure 6.9. At the same time, the link utilisation is maintained at the same level for all the controllers, where the utilisation curves in Figures 6.10, 6.13 and 6.16 are identical for all three algorithms. Thus the proposed robust control has performed significantly better, and is more robust in the sense that it has high performance regardless of the data set used.

The only situation where the proposed robust algorithm does not perform as well as the maximum entropy solution is for the Gaussian data set. However, the difference between the performance of the maximum likelihood solution and the optimal solution is minimal. Hence, the simulations have verified the alternative approach.

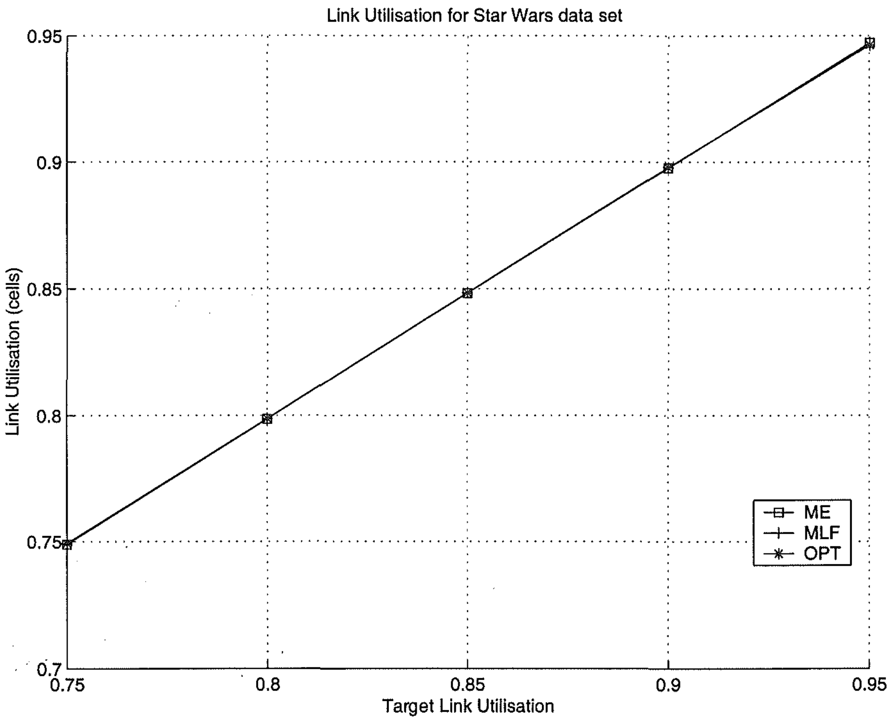


Figure 6.10 Performance Curves for Star Wars: comparing the link utilisation achieved for the different algorithms.

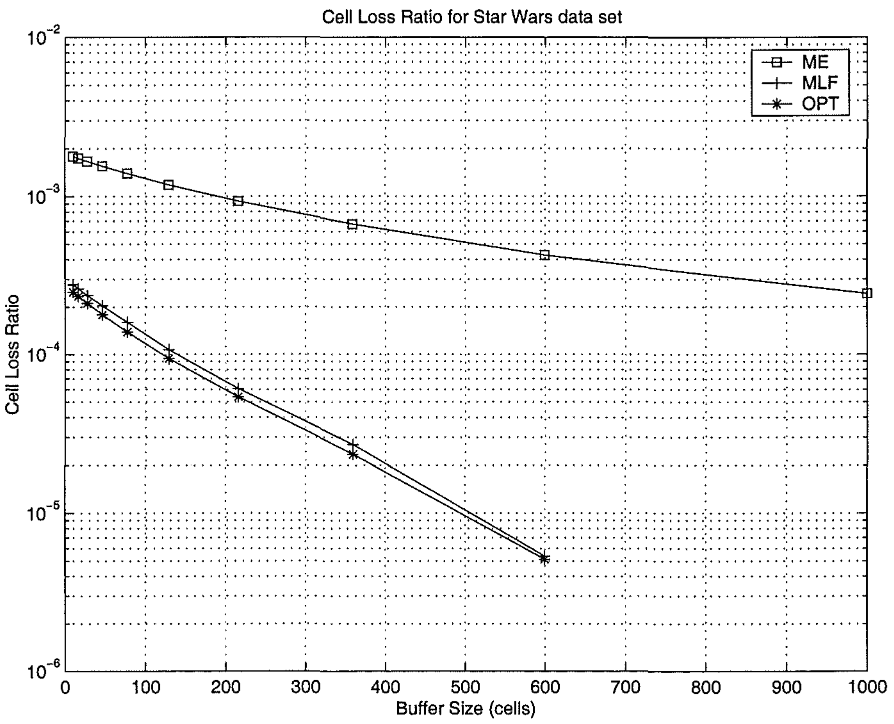


Figure 6.11 Performance Curves for Star Wars: the cell loss ratio as the size of the buffer is varied.

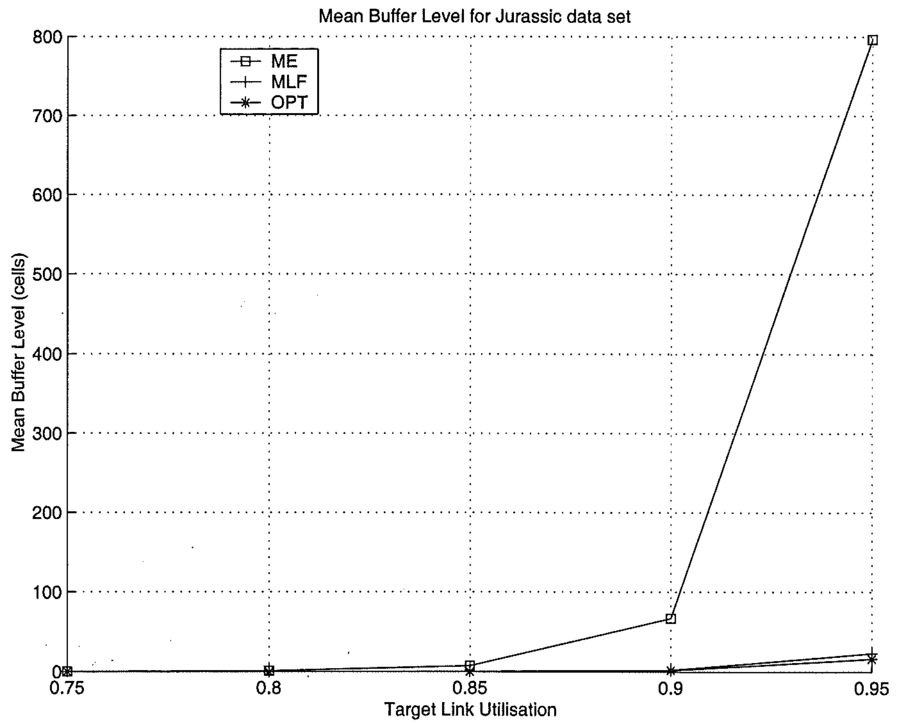


Figure 6.12 Performance Curves for Jurassic Park: the mean buffer occupancy as the target link utilisation is varied.

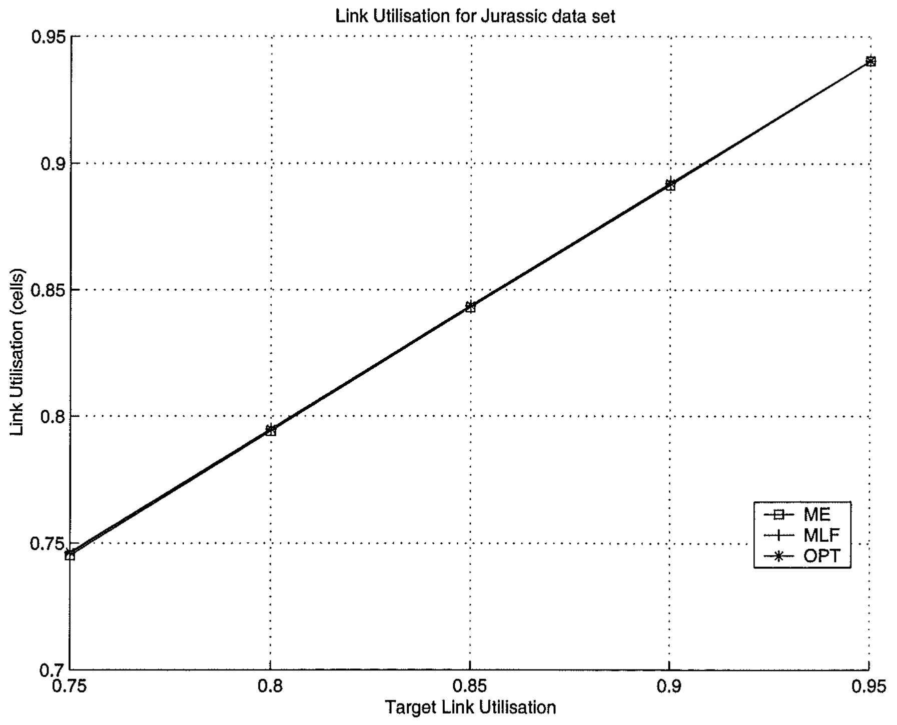


Figure 6.13 Performance Curves for Jurassic Park: comparing the link utilisation achieved for the different algorithms.

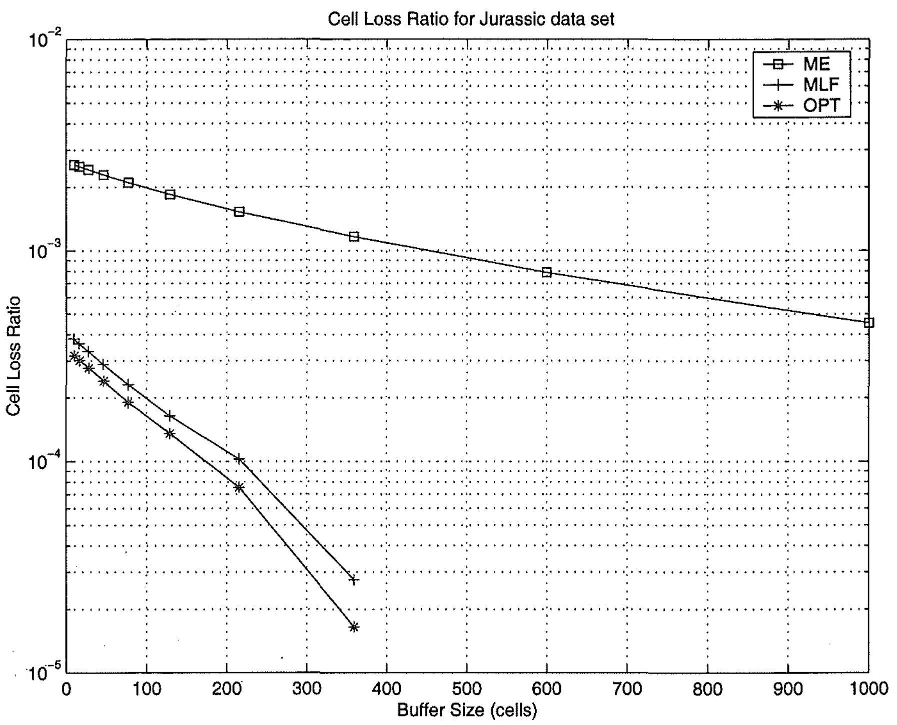


Figure 6.14 Performance Curves for Jurassic Park: the cell loss ratio as the size of the buffer is varied.

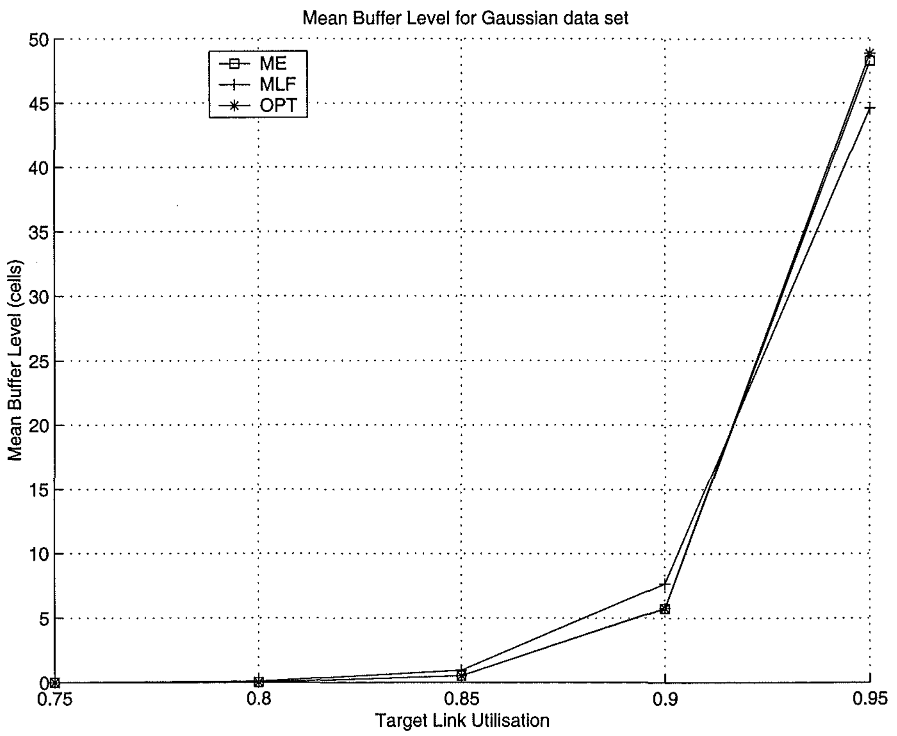


Figure 6.15 Performance Curves for the Gaussian data set: the mean buffer occupancy as the target link utilisation is varied.

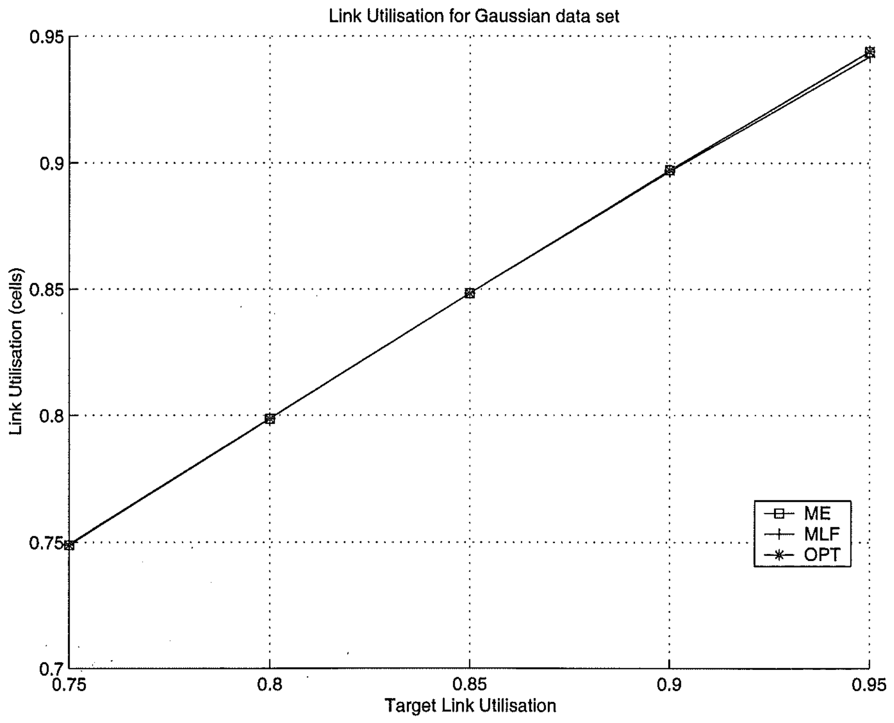


Figure 6.16 Performance Curves for the Gaussian data set: comparing the link utilisation achieved for the different algorithms.

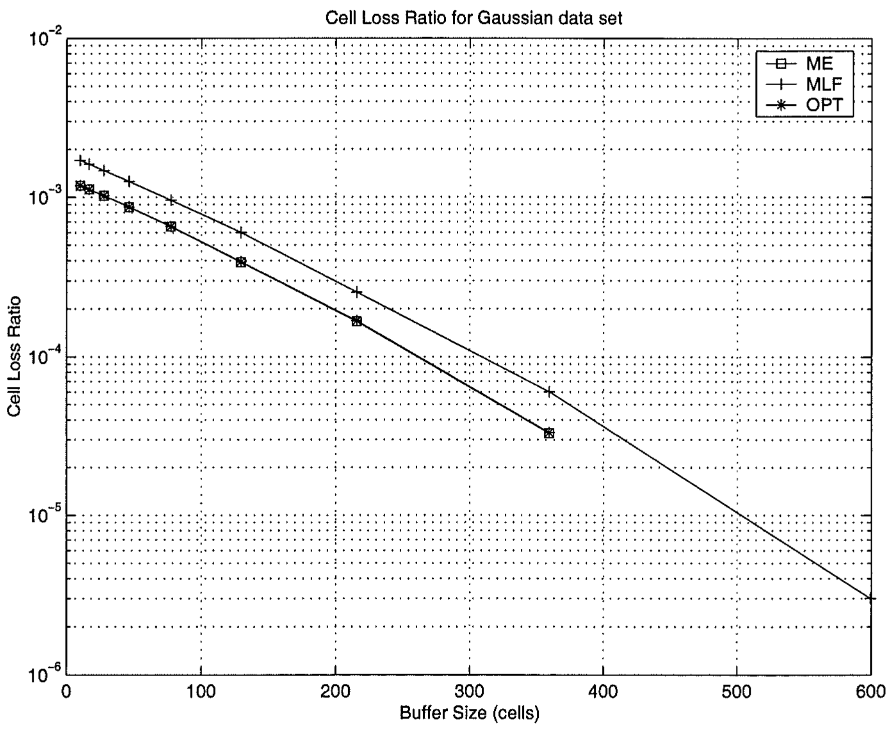


Figure 6.17 Performance Curves for the Gaussian data set: the cell loss ratio as the size of the buffer is varied.

6.4 CONCLUSIONS

In this chapter, the issue of designing the predictive rate control algorithm, PERC, in the situation where there is uncertainty in the Hurst parameter estimates has been addressed effectively. This uncertainty can be caused by time-varying traffic, where parameter estimates become obsolete, or by inadequate data which results in significantly wider confidence intervals. The approach has been to re-employ the likelihood function, in place of spectral entropy, to more accurately reflect the performance of PERC+ in the presence of model uncertainty. This has fascilitated the development of a simple, robust approach to uncertainty in LRD models where the system performance remains close to optimal for a wide range of data sets and system conditions. This complements the investigation in Chapter 5 into the sensitivity of PERC with regards to traffic parameters, as the analytical results regarding sensitivity have been used to enhance the robustness of PERC.

Chapter 7

THE INFLUENCE OF LONG-RANGE DEPENDENCE

The focus in this thesis, up to this point, has been on explicitly incorporating the stochastic structure of LRD into rate control algorithms. Chapter 4 provides a thorough description and performance evaluation of such an algorithm, PERC+, for ABR control in ATM networks, while Chapter 5 addresses the issue of the sensitivity of this algorithm to traffic parameters. This research suggested the possibility of robust algorithms, which have been investigated in Chapter 6. Overall, the thesis has included results which demonstrate the performance improvement using predictive rate control algorithms that are specifically designed for LRD processes.

However, it is important to step back and consider other basic objectives of reactive traffic management. The ability to predict traffic patterns within a network is one of the fundamental requirements of network design and management. A balance is required between achieving high network utilization and providing quality of service, and this requires accurate prediction. At the same time, predictors need to be simple and scalable so that they can be implemented within nodes. Traffic management, in the context of LRD, may be possible using a sophisticated predictor, but in reality simple protocols will succeed rather than sophisticated ones. Thus research must strive to simplify, while maintaining performance levels.

In addition, a deeper understanding of the factors that contribute to the performance of PERC is of interest. While the fact that the LRD property of network traffic is beyond dispute [Leland *et al.* 1994, Beran *et al.* 1995, Crovella and Bestavros 1997], there is still significant debate regarding the impact of LRD on network performance. This stems from the fact that asymptotic queueing performance is invariably analysed, where buffer overflow probabilities are evaluated as buffer sizes approach infinity [Norros 1994, Tsybakov and Georganas 1997, Jelenković 2000]. Instead, in realisable networks, buffers are of finite size, which significantly reduces the impact of LRD such that losses can be predicted reasonably well using short-term correlations [Ryu and Elwalid 1996, Grossglauser and Bolot 1999]. The constraint of finite buffer sizes means that the correlation structure is only meaningful up to a particular correlation horizon, which is dependent on the size of a buffer.

This raises the question of how much influence LRD has on predicting network traffic, as opposed to using short-term correlations [Östring and Sirisena 2001]. This issue has been addressed by Gripenberg and Norros [Gripenberg and Norros 1996] for the prediction of fractional Brownian motion. They show that, for a particular prediction interval, the predictor window (which specifies how much history of an fBm process is required) only needs to be of the same order of magnitude as the required predictor interval. Hence, the performance of the predictor is only really dependent on a short history of the process and associated short-term correlation structure. The same issue, with regards to predictive traffic management algorithms, is investigated in this chapter.

7.1 ANALYSIS OF FGN PREDICTORS

To determine the prediction window that is effective in achieving optimal performance, Gripenberg and Norros uses the relative variance for a predictor with finite window versus the predictor with infinite window. When the relative variance reaches the lower bound of 1.0, a predictor with finite window length is achieving near-optimal performance. When discrete-time prediction is required, infinite window predictors cannot be used, as this would require the inversion of a matrix of infinite dimension. However, the performance of a predictor, as its window size is increased by orders of magnitude, can be observed.

In Figure 7.1, the autocorrelation function of fGn is plotted as the Hurst parameter increases. The correlations decay much more slowly for Hurst parameters greater than 0.5, as is expected for a long-range dependent process. This means that the current sample of the fGn process has significant dependence on previous samples of the process and this dependence does not rapidly depreciate the further one goes into the history of the process. Thus, it would be assumed that predictors based on this structure would also place significant weight on past samples.

However, this is not the case, as can be observed from Figure 7.2. In Figure 7.2, the predictor with horizon $\delta = 1$ has been plotted as the Hurst parameter increases. It is apparent that the weight on the most recent samples do increase, but there is little increase in the weights for samples further back in the history of the process. Thus, the predictors themselves appear to be short-range dependent.

Of course, visual inspection of the predictor weights is not conclusive for determining the performance of predictors. Even weights that approach zero can still have significant impact on the cumulative result, as can be observed by non-summable correlations of LRD. Thus, to effectively analyse the performance of the predictors, the variance of the prediction errors is used, as shown in Figure 7.3. In Figure 7.3, the Hurst parameter and the length of the predictor have been varied to observe their effect on the prediction error variance.

Firstly it should be noted that there is a significant improvement in the performance

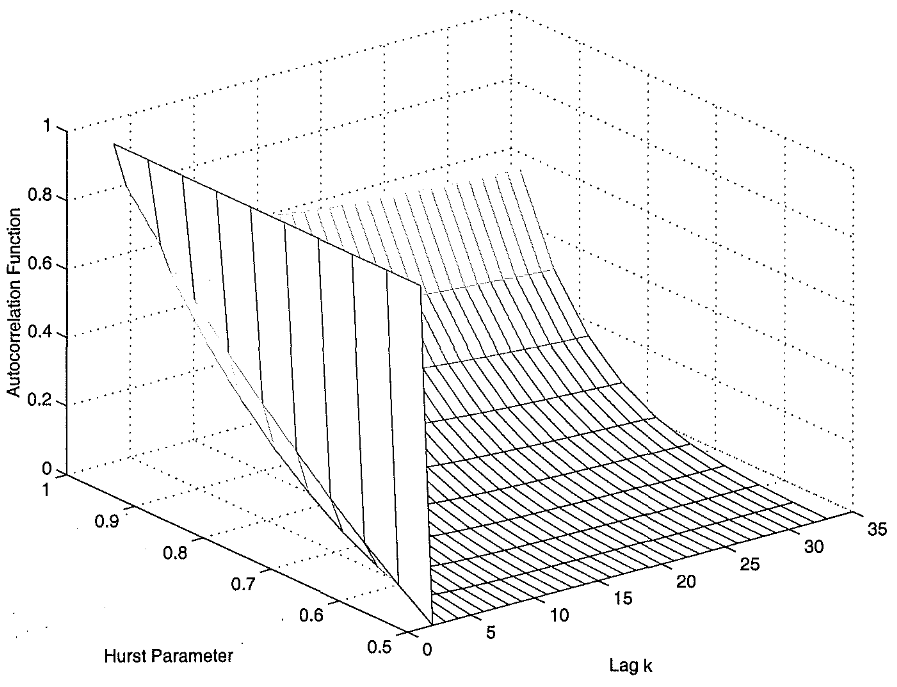


Figure 7.1 The autocorrelation function as the Hurst parameter varies.

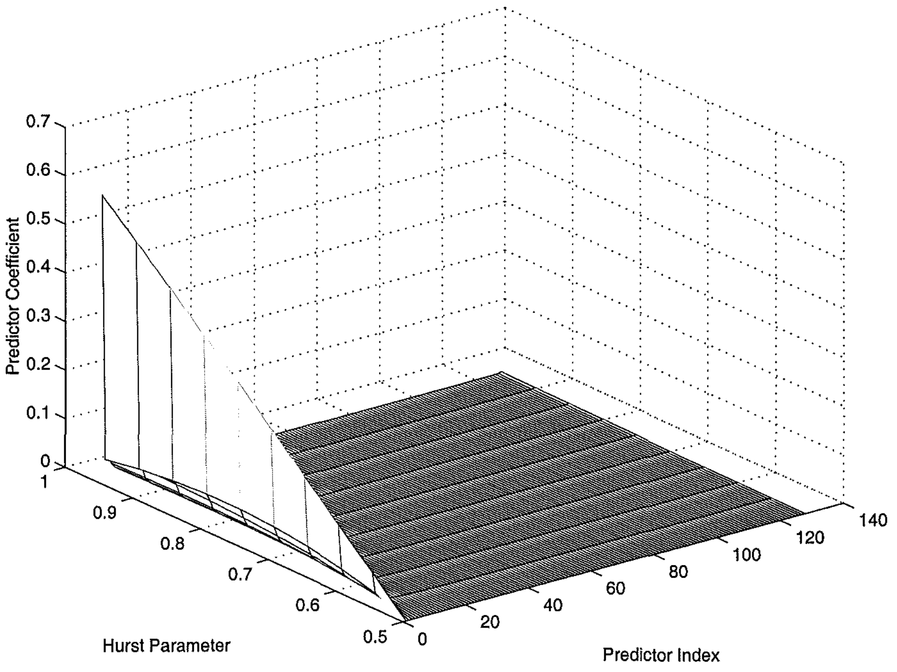


Figure 7.2 The linear predictor coefficients as the long-range dependence is increased, with the prediction interval $\delta = 1$.

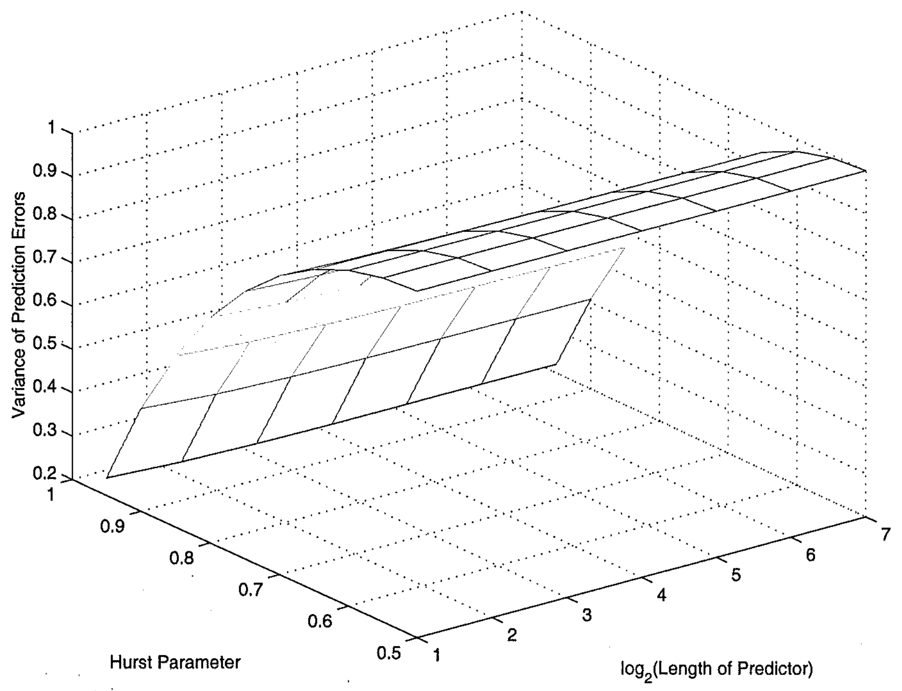


Figure 7.3 The variance of the prediction errors as the Hurst parameter and length of the parameter increases, with the prediction interval $\delta = 1$.

as the Hurst parameter increases, which agrees with the result by Gripenberg and Norros [Gripenberg and Norros 1996]. However, if one moves along the axis with increasing predictor length, there is little improvement in performance. This is further evidence that incorporating more of the process history and correlation structure does not influence the performance of the predictors. Another parameter of interest is the prediction interval δ, and it could be speculated that the influence of long-term correlations becomes more apparent the further into the future one attempts to predict. However, this is not the case, as shown in Figure 7.4, where there is insignificant change in the variance of the predictor even as the predictor length increases. The variance of the prediction errors increases as the prediction interval δ increases, but this cannot be effectively reduced by increasing M, the predictor length. Overall, the evidence points to the fact that it is the *short-term correlations* within the structure of fractional Gaussian noise that dominate the performance of the predictors and not the long-term correlations.

7.2 REVISITING SRD AND LRD

7.2.1 Definitions

As has already been introduced in Chapter 3, long-range dependence refers to a correlation structure that decays at a rate much slower than the exponential decrease that occur in the correlations of a short-range dependent process. Thus, for a stationary LRD

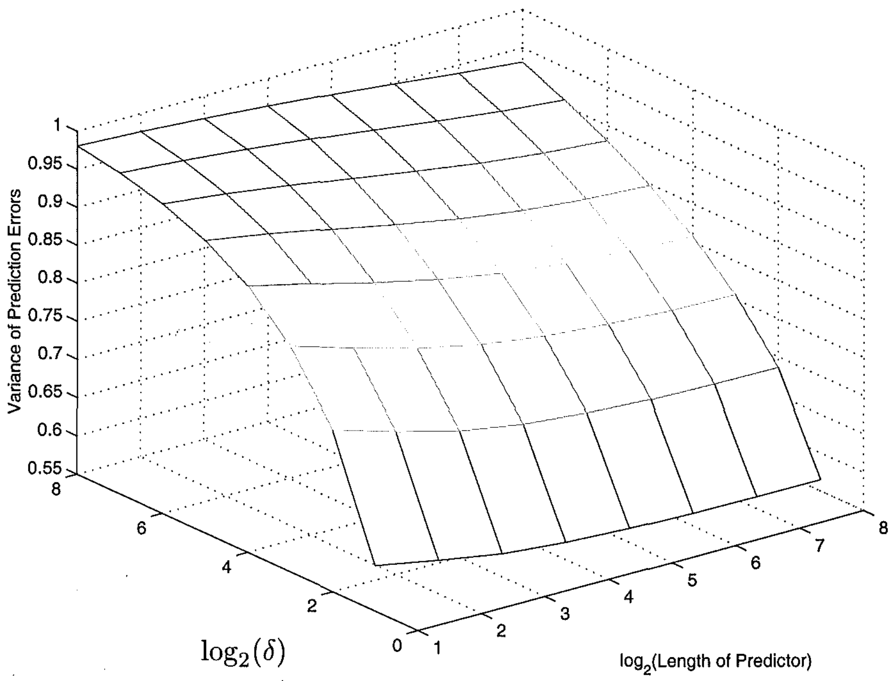


Figure 7.4 The variance of the prediction errors as the prediction interval and length of the parameter increases, for a typical Hurst parameter of 0.85.

process with autocovariance function $r(k)$ [Beran 1994],

$$r_{LRD}(k) \sim k^{2H-2}, \text{ as } k \to \infty, \tag{7.1}$$

which means that the correlations are non-summable, $\sum_{k=0}^{K} r(k) \to \infty$ as $K \to \infty$. While this defines the asymptotic rate of decay of the correlation structure, autocovariance functions of this type need to be generated, and this can be done using the autocovariance function of fractional Gaussian noise:

$$r(k) = \frac{1}{2}\left[(k+1)^{2H} - 2k^{2H} + (k-1)^{2H}\right]. \tag{7.2}$$

If the notation of the second central difference operator $\nabla^2(f(k)) \equiv f(k+1) - 2f(k) + f(k-1)$ [Ryu and Elwalid 1996] is used, then the correlation structure of fGn can be expressed more succinctly as:

$$r(k) = \frac{1}{2}\nabla^2\left(k^{2H}\right). \tag{7.3}$$

It should be noted that this is an *exactly self-similar* process. There are other *asymptotically self-similar* processes that have correlation structure which does not follow this specific autocovariance function for all $k > 0$, but where Equation (7.1) still holds for asymptotically large lag.

In contrast, a short-range dependent process would have an autocovariance function that is summable. SRD is, however, more difficult to specify exactly, since there are numerous examples of strictly decreasing absolute functions that are negligible after some finite lag. The degenerate example is the autocovariance function for an independent process, where $r(k) = 0$ for $k \neq 0$. Another example is a truncated LRD autocovariance function where $r(k) = 0$ for $k > K$ where K is a finite integer, which is summable since the function is zero beyond lag K. However, an SRD function which decays to zero in a structured way is desired. For SRD to hold, the autocovariance function must decay exponentially:

$$r_{SRD}(k) = a^k, \tag{7.4}$$

where $|a| < 1$. Since the correlations become negligible at suitably large lag, the autocovariance function is summable.

7.2.2 A Flexible SRD/LRD Model

To study the influence of long-term correlations, as opposed to short-term correlations, on the prediction of of LRD processes, a flexible model is required, where the amount of LRD and SRD can be varied as desired. This could be achieved by utilising fractional ARIMA processes, which include both SRD and LRD components within the model. However, these processes are computationally expensive to generate and do not lead to an intuitive understanding of the factors that influence good performance. As a result, the approach suggested by Ryu and Elwalid [Ryu and Elwalid 1996] for introducing variable short-range dependence into the correlation structure of fGN is used.

Ryu and Elwalid [Ryu and Elwalid 1996] use a compound model that includes a discrete autoregressive process (AR) and a fractal Binomial noise driven Poisson process. The resulting autocorrelation function can be simplified as:

$$r(k) = \omega \cdot \frac{1}{2}\nabla^2\left(k^{2H}\right) + (1-\omega)\cdot a^k \tag{7.5}$$

where ω is a weight that determines the contribution made by the LRD and SRD components. This model allows the short-term correlations to be varied, as shown in Figure 7.5. In Figure 7.5, the Hurst parameter is 0.9 while a varies from 0.7 to 0.99. The parameter ω is held at 0.5 so that the LRD and SRD components contribute equally to the autocovariance function at zero lag.

The relative increase in variance of the prediction error as the short-range dependence parameter a is increased can be calculated and compared with the base case of only having the LRD component, ie $a = 0$. The relative performance of the predictors is shown in Figure 7.6 as the Hurst parameter and short-range dependence parameter are

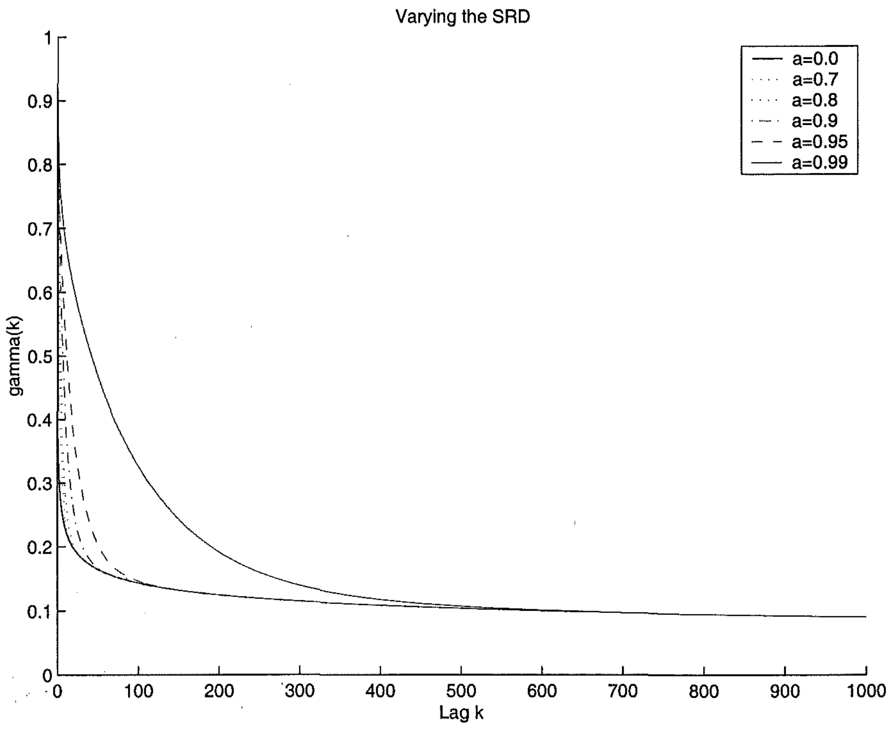


Figure 7.5 Varying the short-range correlations within a long-range dependent correlation structure.

varied. This shows that additional short-range dependence does reduce the performance of the predictor, if the short-term correlations are not taken into account. This effect actually increases with the Hurst parameter.

7.3 NETWORK SIMULATIONS WITH LRD DATA

The conclusions reached in the previous section are investigated using actual data within network simulations. For this, two standard data sets, that are commonly used to study LRD, are employed: (1) the *Star Wars* data set [Garrett and Willinger 1994] and the Bellcore Ethernet data set *BC-pAug89* that was used in the original study on self-similarity of Ethernet traffic [Leland *et al.* 1994]. The reason for this selection of data sets is that LRD originating from fundamentally different network applications can been investigated. Hence, the results are valid for more general LRD processes, rather than specifically applying to one type of source. These data sets are used to investigate the performance of PERC+, with predictors based on SRD and LRD being compared. Averaging filters are also used for comparison purposes.

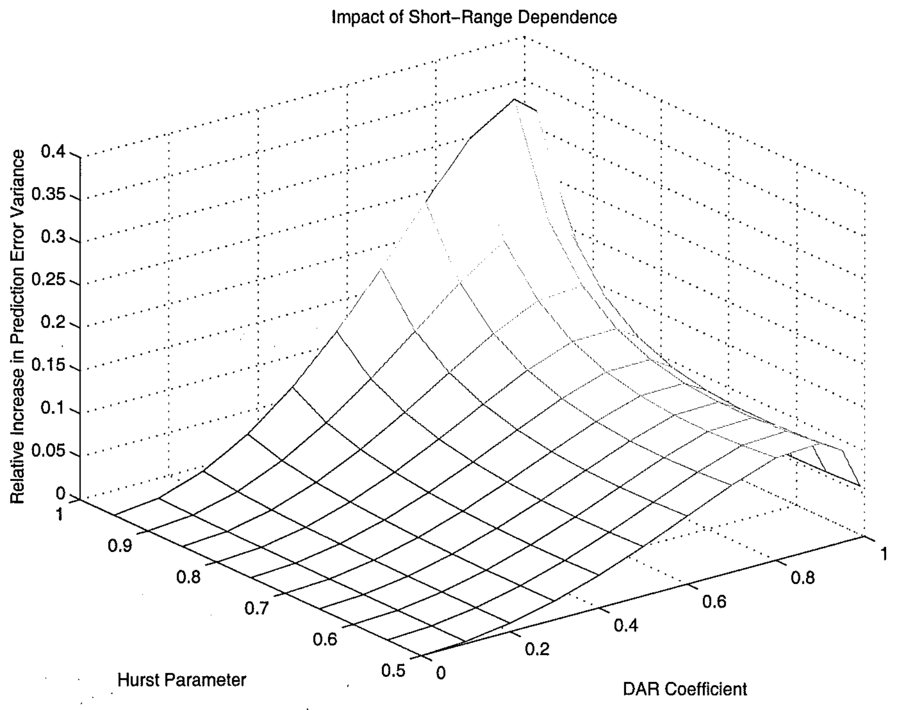


Figure 7.6 The change in the performance of the predictors as more short-range dependence is introduced.

7.3.1 LRD Data Analysis

It has been demonstrated previously that the *Star Wars* data set has correlations that do not decay rapidly as the lag increases [Garrett and Willinger 1994], thus confirming that the data is long-range dependent. Estimates of the Hurst parameter for this set are in the region $H \in (0.84, 0.90)$, showing that the LRD component within the data set is fairly significant. The autocorrelation function derived from an exactly LRD process, and matched with long-range dependence in the data, is compared with the estimated correlations within the data set in Figure 7.7. Significant correlations for lags much greater than zero are indeed observed. It should be noted once again that the Group of Picture (GOP) sequence has been filtered out of the data set, since the LRD nature of the data is of interest and not the correlation structure specific to MPEG encoding.

In Figure 7.8, the figure zooms into the short-term correlation region, and a single AR parameter is matched to the first correlation. It can be observed from both figures that the SRD model achieved by this decays much more rapidly than the true correlation structure of data. However, the SRD model is well-matched to the data for short-term correlations, though it rapidly diverges form the estimated autocorrelation function. These models can be used to develop predictors that utilise long-term correlations, as specified by the LRD model. The performance of these LRD predictors can then be compared with predictors which match the data for the short-term correlations. From this work the influence of LRD on traffic prediction can be determined, and subsequently

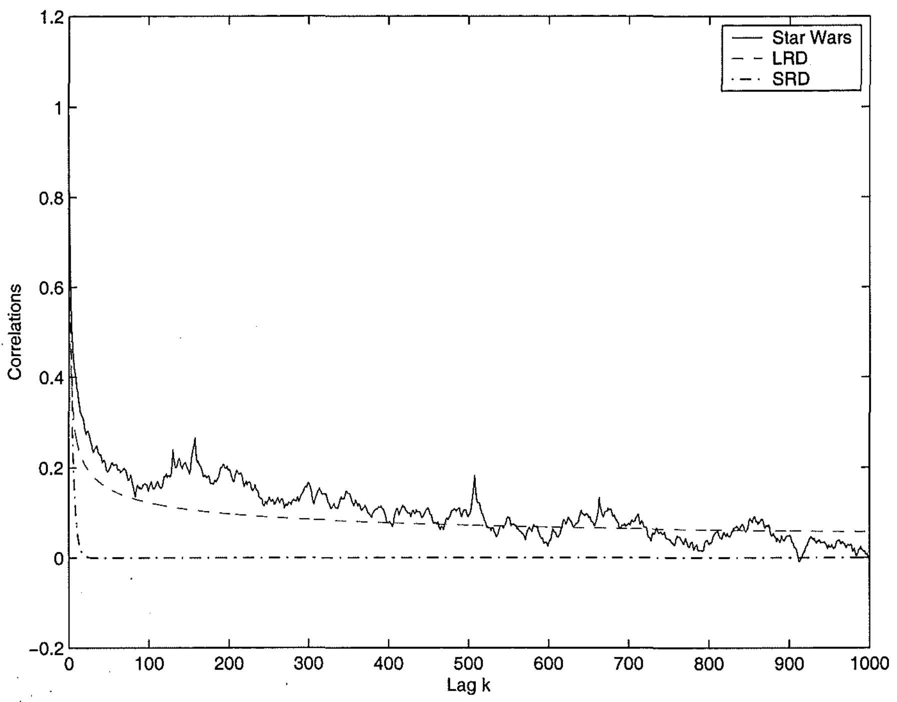


Figure 7.7 The long-term correlations within the *Star Wars* data set and closeness of model fits to the data.

methods of simplying the predictive rate control algorithms can be discovered.

The autocorrelation function for the Bellcore *BC-pAug89* Ethernet data set is shown in Figures 7.9 and 7.10. Likewise with the *Star Wars* data, the Ethernet data has been studied and shown to have long-range dependence, with typical Hurst parameters in the region $H \in (0.85, 0.95)$ [Leland *et al.* 1994]. The estimated autocorrelation function is shown in Figures 7.9 and 7.10, which reveals that there is an underlying periodic structure to the data, shown by the oscillations that occur in the estimation. This cannot be as easily removed as the GOP frame sequence in MPEG traffic, since the exact form of this periodic structure is not known. However, LRD and SRD models can be fitted to the function so that either the short-term or long-term trends in the correlations can be well-represented. Predictors can then be developed from these models, as before.

7.3.2 Simulation Performance

7.3.2.1 Simulation System Model with PERC+

With regards to controlling elastic traffic, there is an inherent delay before a source receives this information and makes the appropriate changes to its rate, thus prediction is an implicit requirement within the rate-control algorithm. Explicit rate information is returned, in the framework of the ABR service in ATM networks, ie using PERC+. The simulation model consists of a single congested link that is returning information

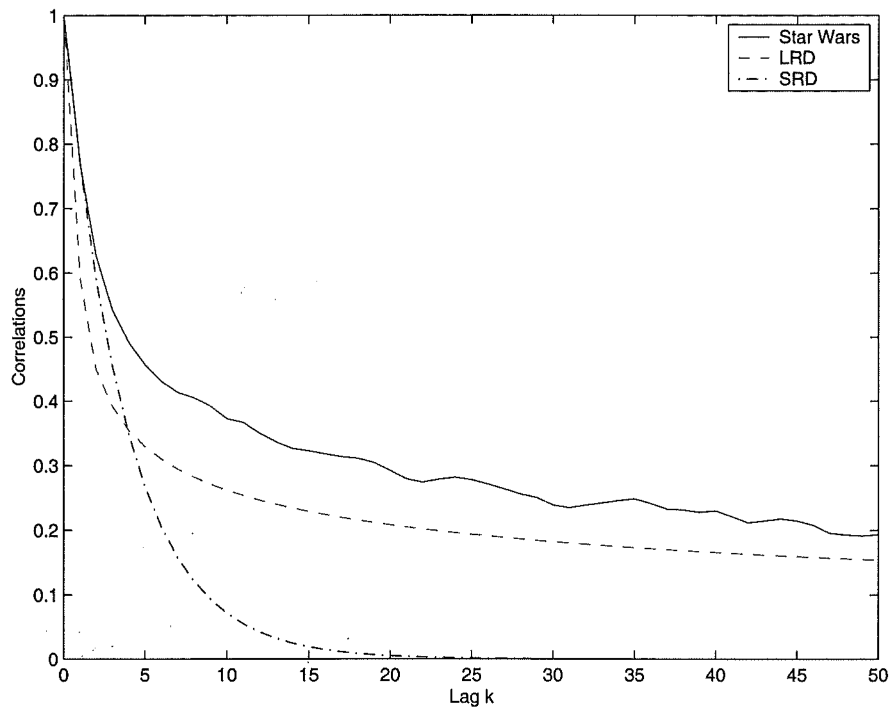


Figure 7.8 Short-term correlations in the *Star Wars* data set.

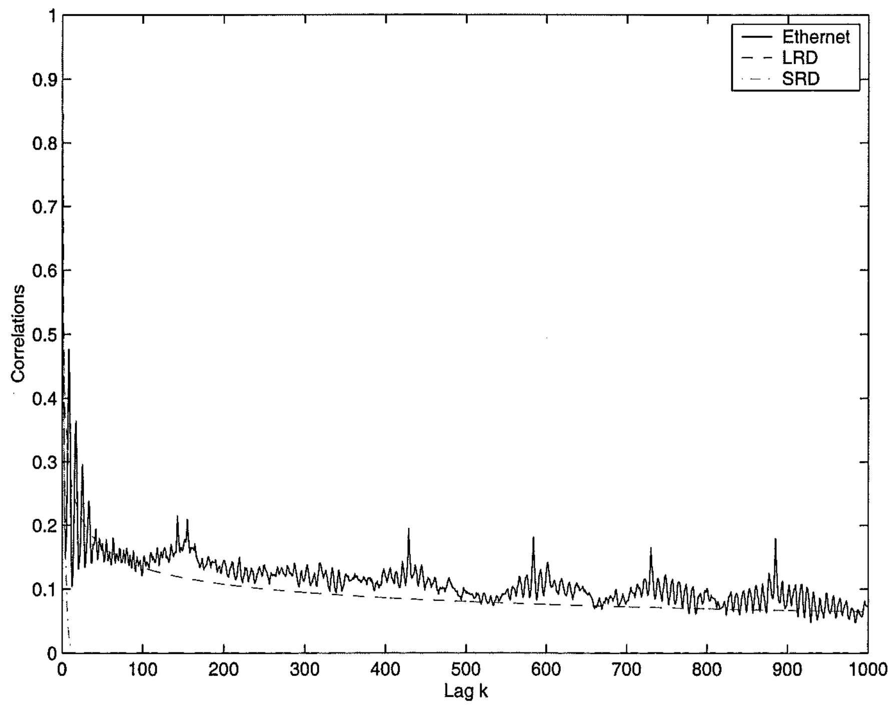


Figure 7.9 The long-term correlations within the *Ethernet* data set and the closeness of model fits to the data.

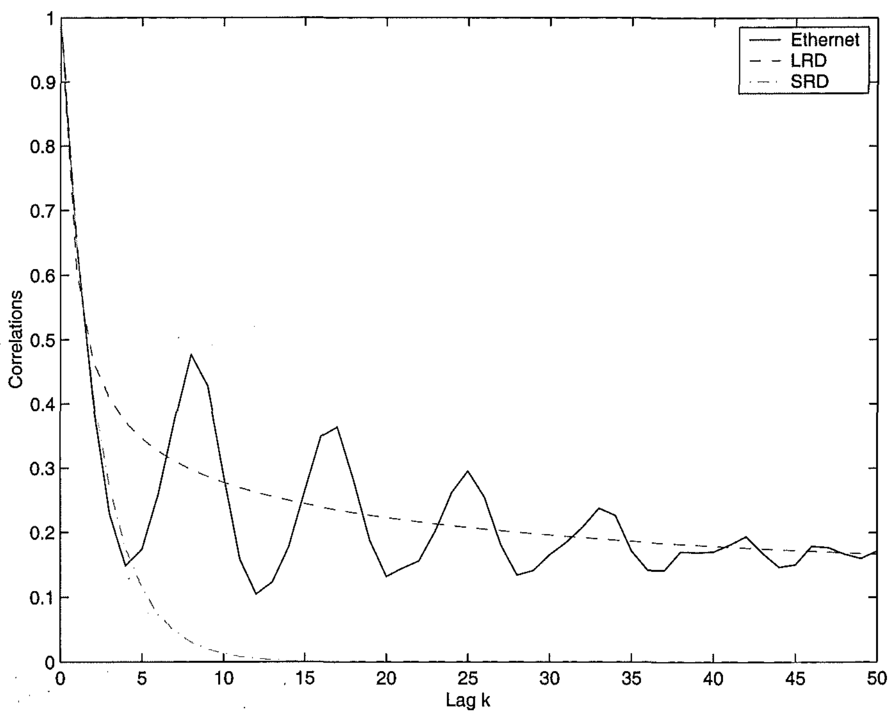


Figure 7.10 Short-term correlations in the *Ethernet* data set.

to a set of six sources with heterogeneous delays $\delta = \{1, 2, 4, 6, 8, 10\}$, as shown in the simulation model in Figure 4.6 in Chapter 4. The link is serving higher priority real-time traffic, represented by a data set generated by aggregating ten copies of an LRD data set that have been randomly shifted in time, using either the *Star Wars* data set or the Bellcore Ethernet data set. The switch distributes the remaining available bandwidth fairly between the elastic connections. This is achieved using predictors based on the long-range dependence of the background traffic and the round-trip delay of the elastic connection.

7.3.2.2 Simulation Performance Comparison

Firstly, the performance of the system is considered using the *Star Wars* data set. Four different filtering techniques are used for the rate control algorithm, including a predictor generated directly from the estimated autocorrelation function for the data set, an LRD and SRD fit to the data, and finally a low-pass filter based on the rate control algorithm proposal Zhao *et al.* [Zhao *et al.* 1997] which has been considered previously in Section 4.4.2. The trends of a number of different performance parameters are shown in Figures 7.11–7.14. In particular, the mean queue length is shown in Figure 7.11 as the target link utilisation is varied. At the same time, the actual utilisation that is achieved is plotted in Figure 7.12. Cell loss rates, when the bottleneck link has a buffer of finite size, are plotted in Figure 7.14, while link utilisations achieved for the system with finite buffers

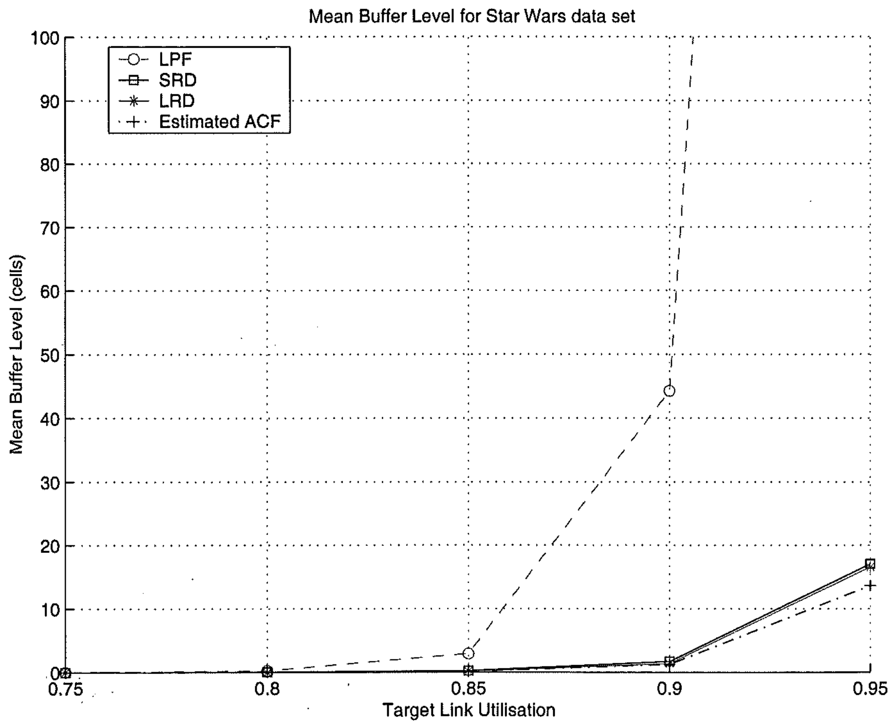


Figure 7.11 Mean buffer occupancy for *Star Wars*, comparing predictors using LRD vs SRD.

is shown in Figure 7.14. It should be noted that link utilisations are virtually identical for all filtering techniques, which rules out the pathological situation where rate control algorithms simple improve performance by reducing throughput.

Firstly, it can be observed that predictions based on the estimated autocorrelation function for the data set result in the best performance, both in terms of queue occupancy when the buffer has infinite capacity (Figure 7.11), and lower cell loss rates when the buffer size is finite (Figure 7.13). While predictors developed from the estimated autocorrelation function do not necessary achieve optimal performance (as can be seen from Figures 7.21 and 7.22), these predictors can certainly be used as benchmarks for comparing the performance of the other three prediction techniques. In particular, predictors based on LRD and SRD fits to the data set can be compared.

Performance curves for LRD and SRD predictors are also shown in Figures 7.11–7.14. It is particularly striking that the performance of predictors that only use short-term correlations within the data set is very close to the performance of predictors based on the LRD fit. While the curves for LRD predictors are closer to the performance of predictors using the estimated autocorrelation function, the performance improvement from using LRD predictors compared with simply using SRD predictors is only marginal. In comparison with the rate control algorithm that uses a low pass filter, the disparity between performance curves is insignificant. The poor performance of averaging techniques has been noted previously (Section 4.4.2.3).

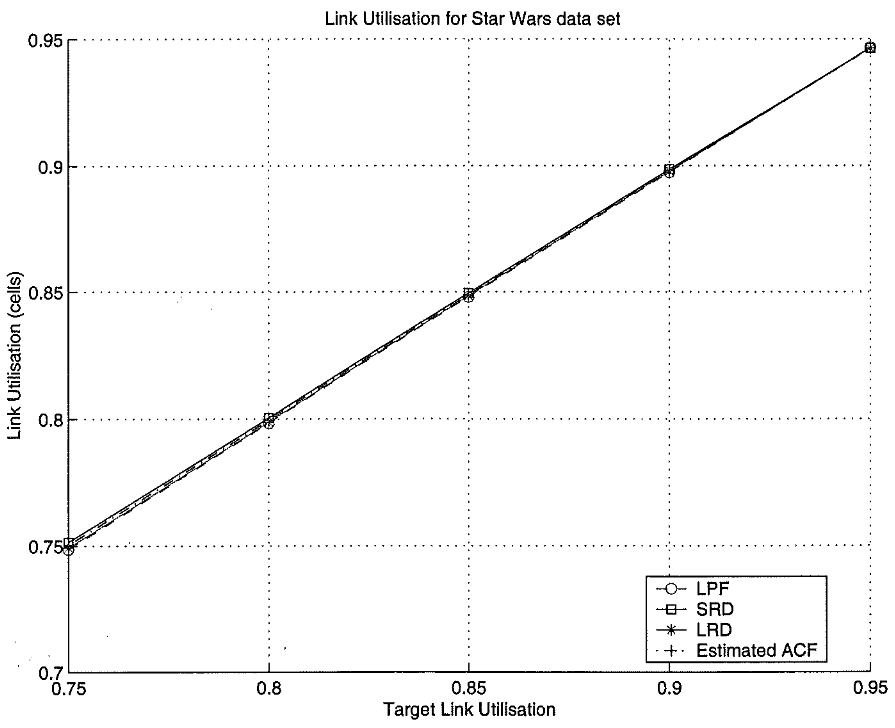


Figure 7.12 Link utilisation for *Star Wars*, with infinite buffer, comparing predictors using LRD vs SRD.

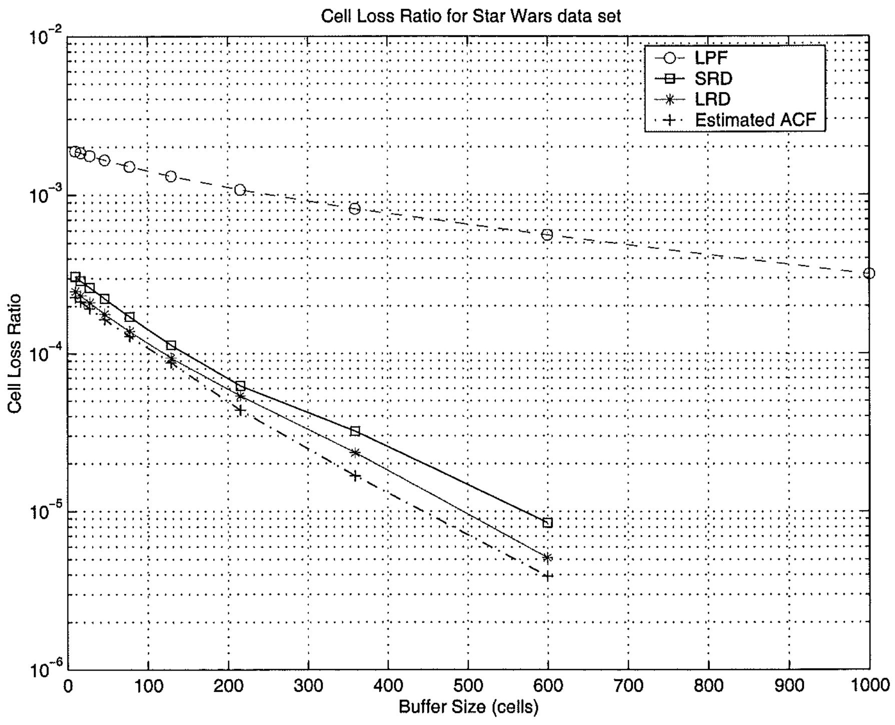


Figure 7.13 Cell loss rates for *Star Wars*, comparing predictors using LRD vs SRD.

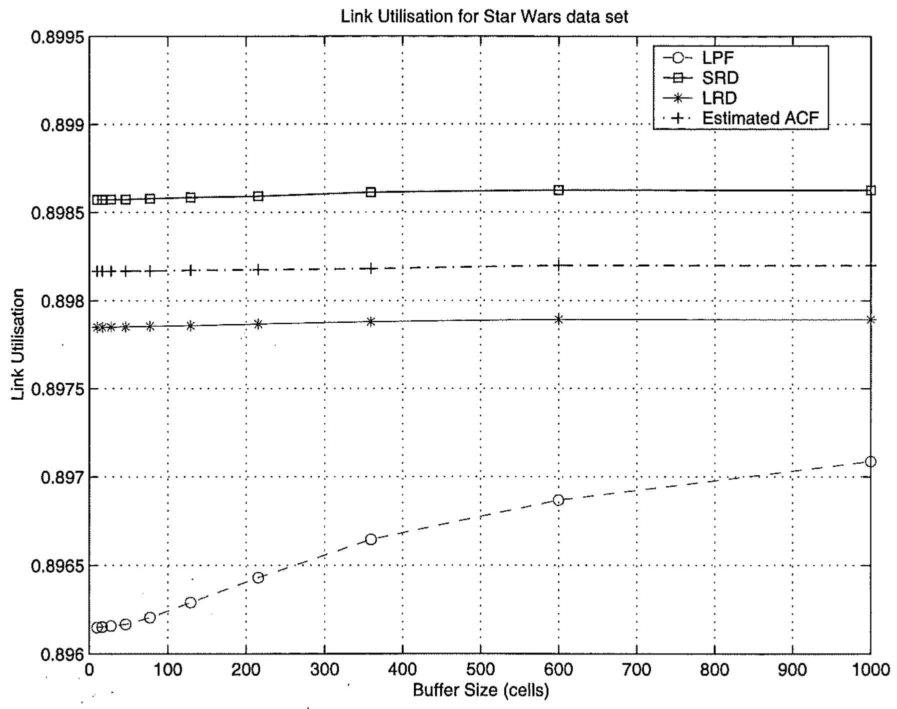


Figure 7.14 Link utilisation for *Star Wars*, with finite buffers, comparing predictors using LRD vs SRD.

The performance curves when the Bellcore Ethernet *BC-pAug89* data set are plotted in Figures 7.15–7.18. Once again, the estimated autocorrelation function produces the predictor that has the highest performance, while the low-pass filter performs the worst. The link utilisations curves are virtually identical, which means that it is the ability of the rate control algorithm to track variations in the underlying background traffic that leads to the algorithm's performance. Finally, the difference between LRD predictors and SRD predictors is very small, and the performance of both types of predictors are much closer to predictors based on the estimated autocorrelation function.

It can be observed, from Figure 7.15, that mean buffer levels for the SRD predictor for the Ethernet data set diverge from the LRD predictor faster than for the *Star Wars* data set. This can be attributed to the unique form of the autocorrelation function for the Ethernet data set, as shown in Figure 7.10. In this case, the oscillations that occur in the autocorrelation function mean that an SRD model fit rapidly becomes inconsistent with the actual correlations that occur within the data. This does not occur as rapidly for the smoother correlation function for the *Star Wars* data set (see Figure 7.8). However, this does not detract from the observation that SRD predictors achieve a performance that is only marginally lower than LRD predictors, given the same data set.

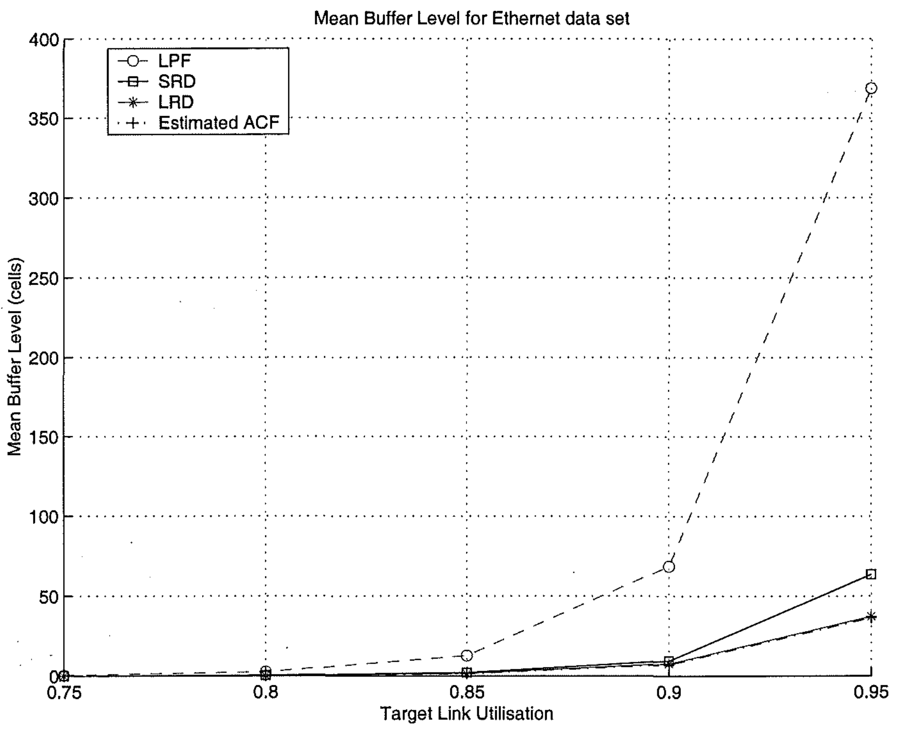


Figure 7.15 Mean buffer occupancy for *Ethernet*, comparing predictors using LRD vs SRD.

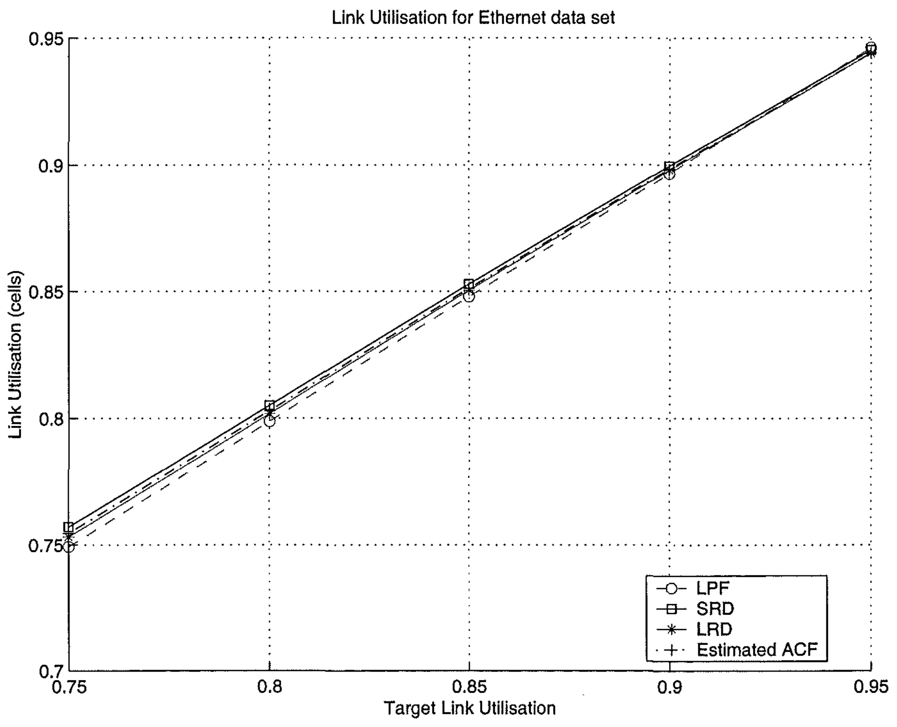


Figure 7.16 Link utilisation for *Ethernet*, with infinite buffer, comparing predictors using LRD vs SRD.

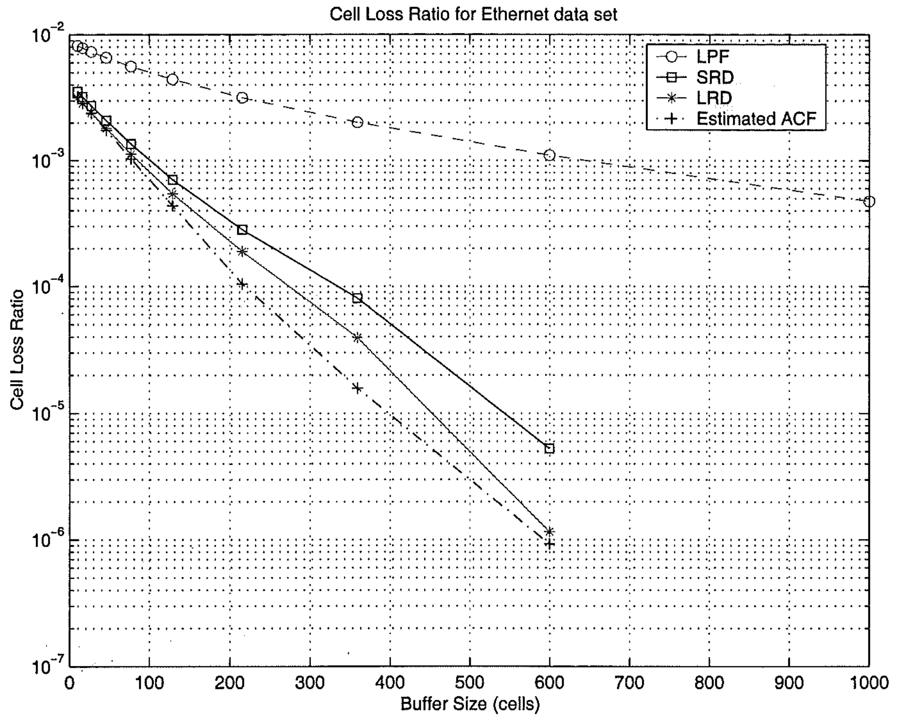


Figure 7.17 Cell loss rates for *Ethernet*, comparing predictors using LRD vs SRD.

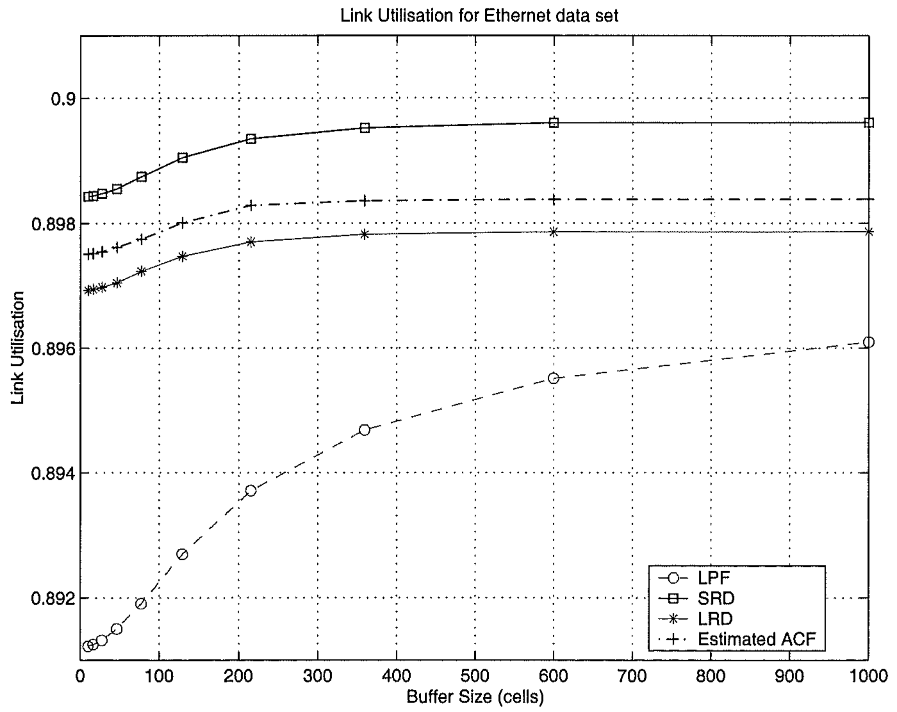


Figure 7.18 Link utilisation for *Ethernet*, with finite buffers, comparing predictors using LRD vs SRD.

7.3.2.3 Effect of Predictor Length

It has been demonstrated, using variance of prediction errors, that the performance of a predictor based on an fGn model does not improve significantly as the length of the predictor increases (see Figure 7.4). Instead, most of the performance gain, for a particular prediction interval, is achieved using predictors of relatively short length. This agrees with the analysis of Gripenberg and Norros [Gripenberg and Norros 1996], who show that the amount of history of a process that is required to achieve near optimal performance is approximately equivalent to the prediction interval. As a result, storing significant data for a particular flow of traffic to be used in predicting future traffic levels does not significantly improve the performance of the predictor, even if the traffic is LRD.

To complete this study into the influence of LRD compared with using SRD, the PERC+ system is simulated with varying predictor lengths. The performance curves for the *Star Wars* data set are shown in Figures 7.19 and 7.20, while the same performance parameters, for simulations with the Bellcore Ethernet data set, are plotted in Figures 7.21 and 7.22. In these simulations, four filtering techniques are used to generate the explicit rate information that is returned to the ABR sources. The estimated autocorrelation function is used, together with LRD and SRD model fits to the data, to produce predictors for the background traffic. In addition, a low-pass filter algorithm is included for comparison.

Two main conclusions can be drawn from Figures 7.19–7.22. Firstly, increasing the length of the predictors does not signficantly improve the performance of predictors based on either the estimated correlation function or LRD models of the data. While in the case of the *Star Wars* data set, the trend is a slight improvement in performance when the estimated autocorrelation function is used, as compared to the *Ethernet* data set where a slight degradation occurs, the variation in performance parameters cannot be described as significant. With regards to the SRD predictors, there is no variation as the predictor length is increased. This is to be expected, since the SRD predictor only uses short-term data and its associated correlation structure. Any further data that is stored regarding the traffic process is effectively disregarded by the predictor.

The other conclusion that can be drawn from Figures 7.19–7.22 is that increasing the length of the low-pass filter reduces the performance of its associated rate control algorithm. This is because the filter is averaging over a longer interval of data, which removes more of the high-frequency content of the traffic. Hence low-pass filtering is less effective in distributing dynamically available bandwidth to the ABR sources.

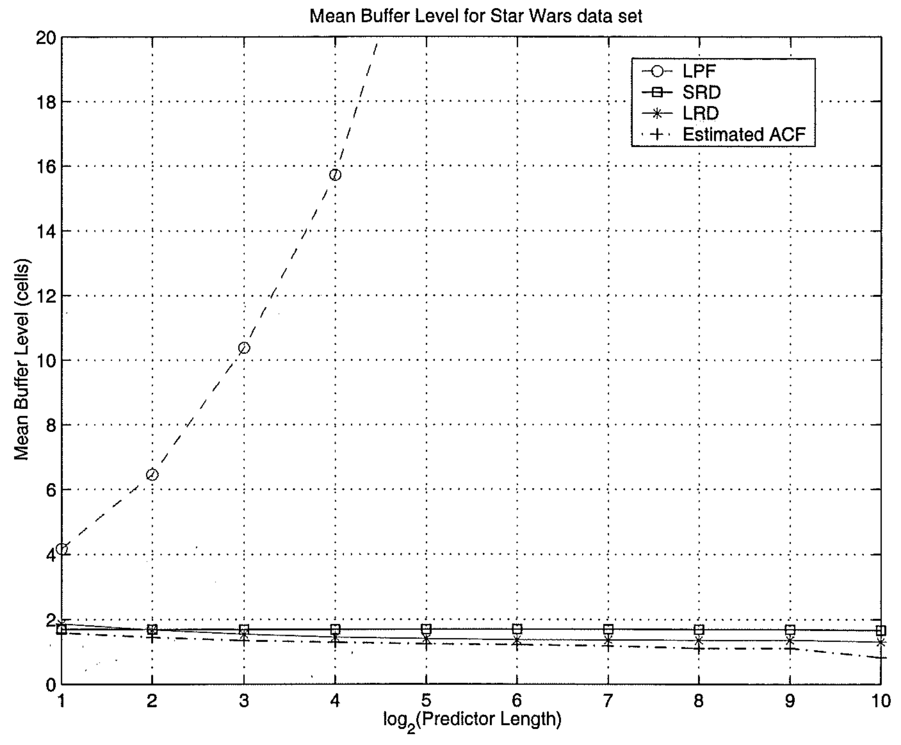


Figure 7.19 Variation in mean buffer occupancy for *Star Wars*, as the predictor length is increased.

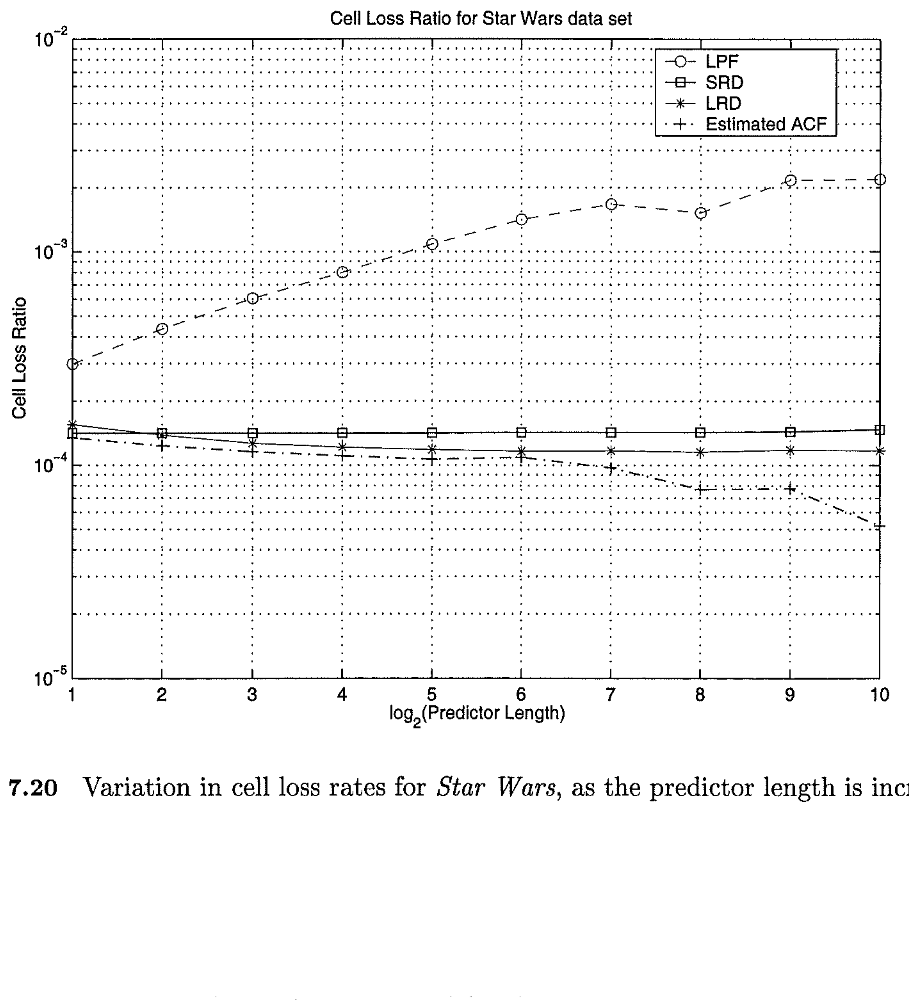


Figure 7.20 Variation in cell loss rates for *Star Wars*, as the predictor length is increased.

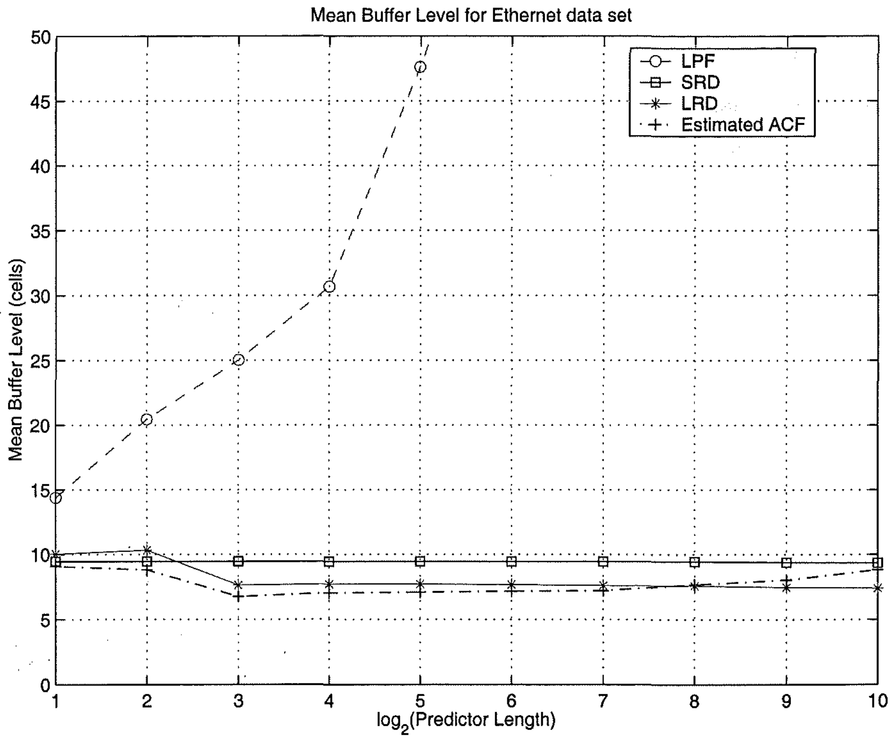


Figure 7.21 Variation in mean buffer occupancy for *Ethernet*, as the predictor length is increased.

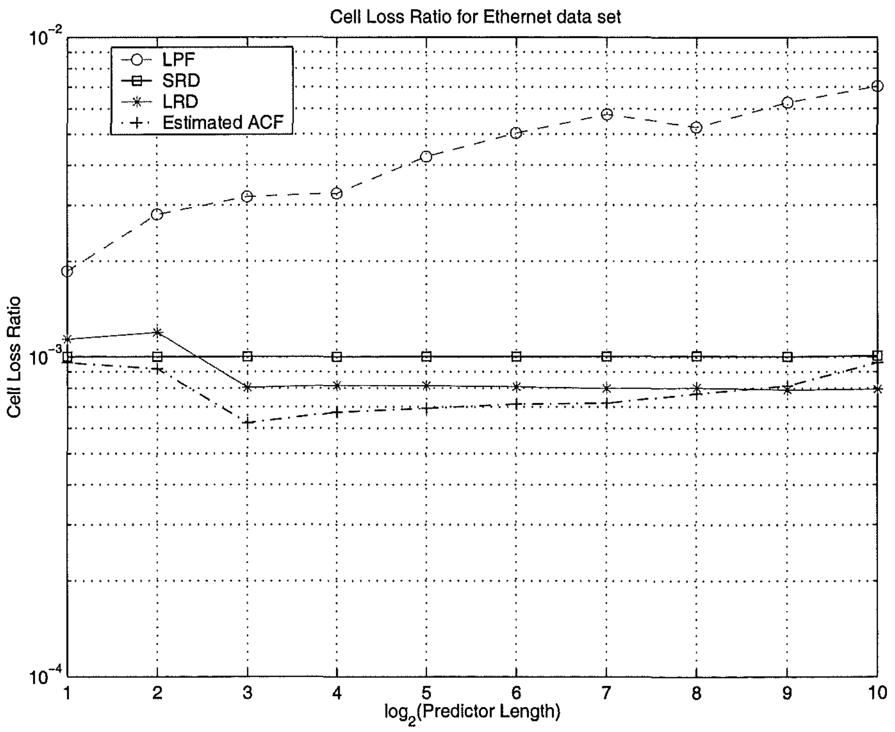


Figure 7.22 Variation in cell loss rates for *Ethernet*, as the predictor length is increased.

7.3.3 Simulation Evaluation

The reason for the effectively equivalent performance of predictors using estimated autocovariance functions, LRD or SRD models, is due to the dynamics of self-similar processes and the process of predicting these dynamics. Self-similar traffic is fundamentally bursty, which means that there is a high probability that if the current traffic load is heavy, then the traffic load in the "near" future will also be heavy. The converse also holds, where light traffic loads have a high probability of being followed by another light load. This property has been utilised by Tuan and Park [Tuan and Park 1999], who base their rate control algorithm on this non-degenerate structure of LRD traffic predictions. In contrast, the expected level of a traffic process with Hurst parameter $H = 0.5$ is given by the mean of the process, so that prediction degenerates to simply estimating the mean traffic level.

The estimated autocorrelation function and the LRD and SRD model fits all capture the correlations between a current traffic level and traffic levels in the subsequent time slot, and hence predictions generated by predictors based on these models have a high degree of accuracy. In contrast, averaging techniques, like low-pass filtering background traffic, place less emphasis on the previous state of the traffic and a higher emphasis on the mean traffic level. Hence, averaging filters do not capture the dynamic nature of LRD traffic as well, and result in poorer performance.

7.4 CONCLUSIONS

In this chapter, the impact of LRD on the performance of predictors of LRD traffic has been investigated, versus the use of short-term correlations. In particular, the theorectical performance of predictors for fGn processes has been investigated, and it has been shown that long-term correlations have only marginal influence on predictions. This has been demonstrated through simulations using two distinct data sets. The conclusion that is drawn from these results is that high performance can be achieved using the short-range correlation structure in data.

Firstly, it should be noted that LRD models for the data still out-perform ones which only take short-range correlations into account. This is because LRD follow the autocorrelation function more closely for large lag. Thus, if determining optimal performance of rate control algorithms, like PERC+, is the objective then the long-term correlations that occur in background traffic need to be considered when PERC+ is implemented.

However, this chapter has suggested ways in which rate-control mechanisms can be simplified, and yet still achieve good performance. Estimating a complete autocorrelation function, or even the Hurst parameter, can be computationally expensive and require significant storage space (depending on the estimation technique used). Instead, using a limited number of the short-term correlations, high-performance predictors can be developed. This reduces the complexity of implementing algorithms, such as PERC+, for explicit rate control of elastic connections.

Chapter 8

CONCLUSIONS

Over the last few years, the service philosophy of networking has broadened from providing a single service to multiple services that are integrated within the same network. Services, such as telephony, television or data transport, have previously been handled by dedicated networks, and management within these networks has been designed specifically with the particular requirements of the traffic transported. However, the trend is for these applications to be transported using a common network that handles traffic according to the class of application which is generating the traffic.

One of the key challenges is to be able to accommodate the wide variety of requirements that each application using the network. In this way, QoS is ensured. Real time applications, such as telephony and video, need specific bounds on the delay and jitter through the network, while the performance of data applications, such as *ftp* and *telnet*, depends on packet loss. Losses for data applications necessitates retransmission of packets, which simply adds more load to the network. These QoS requirements have been incorporated in the development of ATM networks and the IP DiffServ framework, two of the current approaches to providing differentiated service networking [Black 1995, Onvural 1995, Blake *et al.* 1998].

An important traffic management approach that is fundamental to these networks is reactive control which adapts to overload situations within the network. This approach is exemplified by best-effort services, like the ABR service in ATM and the service offered by TCP in IP networks, and is intended for data applications that can adjust their transmission rates depending on feedback from the network. This type of service achieves two objectives—firstly, it allows users to maintain connectivity to the network and throughput is achieved via multiplexing, rather than by blocking calls like in fixed bandwidth networks. Secondly, it allows the network to increase utilisation of its resources by distributing available bandwidth to sources that can adjust to feedback. Feedback can involve simple binary information, such as the detection of packet loss or setting specific bits in packet headers, as in ATM cells or TCP packets. Alternatively, it can involve explicit rate information, as is specified for the ABR service in ATM networks. Whatever the mechanism, the principle is that sources need to respond in a predictable manner so that the network can be steered toward its optimal operating point.

In this thesis, reactive control mechanisms for broadband networks are considered that have been designed in consideration of the nature of high priority network traffic—specifically, the self-similarity that has been observed in a wide variety of types of traffic. Self-similar traffic has an inherent burstiness that exists for a wide range of scales. This burstiness does not diminish with aggregation or multiplexing. Hence, many of the traditional ways of solving traffic management problems are not effective in achieving the QoS requirements of applications using broadband networks. Thus, a logical step is to include the structure of self-similar traffic into traffic management algorithms. A rate control mechanism has been proposed within this thesis that incorporates the LRD characteristics in network, and its performance and properties have been analysed extensively. This analysis includes comparison with other rate-control mechanisms, the sensitivity of the algorithm with regards to estimates of traffic parameters and robust designs for rate control. Finally the influence of using long-term correlations has been investigated, as compared with using short-term correlations in the data.

8.1 PREDICTIVE RATE CONTROL

The rate control mechanism proposed in this thesis incorporates the long-range dependence of the background traffic into the feedback mechanism for the ABR service. Long-range dependence means that significant correlations exist within traffic levels, and these correlations do not decay exponentially fast. Thus, by storing measurements of the background traffic bandwidth requirements and using an appropriately designed filter, the available bandwidth at the network node can be predicted more accurately. The prediction interval depends on the round-trip delay of the individual ABR connection. This information is used to calculate the rate allocations for the ABR source, which is entered into the ER field of the stream of RM cells on route back to the source. This algorithm has been labelled PERC.

In addition to this basic filtering process, a prediction compensation algorithm has been proposed that adjusts the rate allocations for ABR sources with smaller round-trip delays based on the rate allocations previously made to sources with higher delay—resulting in the enhanced algorithm labelled PERC+. Bandwidth predictions for future time slots have already been made for sources with higher delay, yet these predictions have greater errors. These errors can be partially absorbed by adjusting the explicit rate calculations for sources with lower latency. Chapter 4 has shown that this approach has significantly better performance than other rate-control mechanisms such as ERICA, especially when traffic is long-range dependent. Performance gains include lower cell losses and smaller average queue lengths, resulting in a traffic management scheme that closely matches adaptive transmission rates to available bandwidth within a network.

In addition, real-time parameter estimation has been incorporated into PERC+ so that time-varying traffic can be accommodated.

Once the performance advantages of using PERC+ have been established, sensitivity of the algorithm to traffic parameters is an important investigation, as reported in Chapter 5. Self-similar processes can be modelled parsimoniously by the Hurst parameter, mean and variance parameter. Investigating the sensitivity of PERC+ with regards to these parameters shows how the performance of the system will be affected should the estimate of the traffic parameter be either incorrect, or if it changes in the course of time. Each parameter has been considered in turn, and it has been shown that there is significant sensitivity to the Hurst parameter estimate in the region when the parameter was signficantly underestimated. This means that if the traffic control algorithm is specifically designed for traffic with Hurst parameter close to 0.5, and the actual background traffic is LRD with greater Hurst parameter, the performance of the system will suffer significantly. In contrast, it was shown that there is little sensitivity with regards to the mean estimate, if the sample mean is used. This agrees with the known result that the sample mean is close to optimal for LRD processes [Beran 1994]. In addition, PERC+ is insensitive to the variance parameter estimated for the background traffic. The predictors are sensitive to Hurst parameter estimates because they are developed using the correlation structure of LRD traffic, and it is the Hurst parameter which defines this structure, not the mean or the variance.

Another aspect in the design of high-performance rate control algorithms is achieving robustness. Robust controllers ensure that the performance of the system does not deteriorate seriously even when there is uncertainty in traffic parameters. The asymmetric nature of the sensitivity of PERC to Hurst parameter estimates implies that robust design is achievable. The traditional approach to robust prediction is to optimise for the greatest model uncertainty, which then enforces a lower bound on the performance. Hence, the *absolute performance* of the traditional robust predictor will be higher for any other traffic model within the set of possible models. Furthermore, robust predictors provided by this approach maximise the entropy of the system. However, in consideration of the sensitivity analysis, it has been shown that maximising spectral entropy results in designing a predictor using Hurst parameter $H = 0.5$, which does not lead to high performance for other values of the Hurst parameter. Instead, the choice of the robust predictor has been approached by maximising the likelihood function, which determines the proximity of the selected model to the actual model. In this way, the *relative performance* is optimised, so that the performance of the robust control solution will be close to the performance of the optimal predictor for the entire range of Hurst parameters, and hence traffic conditions. This has been demonstrated via simulations in Chapter 6.

The fundamental aspects of the performance of PERC are of interest, and in Chapter 7 considered the contributions that long-term correlations have towards the performance of the algorithm as compared with using short-term correlations. This work was motivated by a couple of significant studies that investigated the limits on the length of the correlation function that impacts network performance [Ryu and Elwalid 1996, Grossglauser and Bolot 1999]. By investigating the structure of predictors for LRD as the length of the predictors are varied and the prediction interval is increased, it has been shown that short-term correlations dominate the performance of the predictor and that long-term correlations have only marginal effect on improving the performance of PERC. Predictors based on LRD still out-perform SRD predictors, but predictive control algorithms can be simplified by using predictors based only on short-term correlations.

In conclusion, the focus of this thesis has been the investigation of high performance predictive rate control algorithms where elastic connections have been competing with self-similar traffic for network resources. This has involved investigating predictor designs that result in high throughput for the sources, while at the same time optimising performance parameters such as queue lengths and cell loss rates. Matching the construction of these predictors with self-similar traffic has included not only the explicit use of the LRD structure of the traffic, but also determining sensitivity of predictors with regards to intensity of the self-similarity and designing robust algorithms which can handle parameter uncertainties. Since high performance is a key goal, including computational and implementation complexity, the influence of long-term correlation has been compared using short-term correlations, and this study leads to possible simplifications to predictive rate control. In essence, this thesis represents a thorough study of rate control in a self-similar networking environment.

8.2 FURTHER WORK

A number of different research avenues follow on from the work described in this thesis. A very interesting concept is that of developing an adaptive controller based entirely within the wavelet domain. As has been mentioned, wavelets are a natural tool in the analysis of self-similar processes. Since the wavelet transform effectively maps the long-range dependence that occur within samples the time domain into short-range dependence within wavelet coefficients [Flandrin 1992, Veitch and Abry 1999], it is possible to design a simple predictor that predicts coefficients at each level of the wavelet transform. Then, by inverting the transform, future traffic levels can be predicted. This can be formulated as an adaptive control technique because the short-range dependence at each level of the wavelet transform can be tuned, using parameter estimates that are updated as more coefficients become available [Roughan *et al.* 1998]. This appears to be fundamental research in information theory that has not been addressed previously.

Currently, the transmission protocol which is most widely used is TCP. As has been outlined in this thesis, a number of flavours of TCP have been proposed, with new modifications addressing different aspects of TCP performance. An interesting problem is how to design the window dynamics of TCP, when it is faced either with transporting files with heavy-tailed distributions, or when it encounters self-similarity within the network. Tuan and Park [Tuan and Park 1999] have proposed a multiple time scale control algorithm which adapts the aggressiveness of a linear increase, multiplicative decrease algorithm so that the predictability of self-similar traffic is utilised. However, further work remains in this area, since it is possible to enhance protocols like TCP Vegas, that compare actual throughput with expected throughput, so that higher performance is achieved when background traffic is self-similar. In addition, there has been a flurry of research work studying the advantages of using active queue management, using the concept of ECN, to improve the performance of TCP within the Internet [Bu and Towsley 2000, Misra *et al.* 2000, Kunniyur and Srikant 200]. An interesting area of research would be the development of a simple queue management algorithm that cooperates with ECN-enabled TCP connections in the situation where traffic arriving at the queue is self-similar.

To transfer the performance advantages of PERC from the ATM arena to the IP community, the inherent components of the approach needs to be distilled and then applied to active queue management. PERC incorporates two elements: the use of long term traffic trends, as represented by traffic mean estimates, together with short term variations, whose effect is scaled with respect to the round-trip delay of a particular connection. This multiple time scale decomposition parallels the work by Tuan and Park [Tuan and Park 1999], and also Kunniyur and Srikant [Kunniyur and Srikant 200]. Specifically, Kunniyur and Srikant identify TCP's response to losses as a short time scale control action, and couple this with an adaptive marking scheme that adapts to system dynamics with a longer time scale. However, they do not utilise the unique stochastic structure that accompanies long-range dependence, and simply adjust marking probabilities based on a fixed step-size and a gradient. The approach of PERC could be incorporated into active queue management by estimating conditional probabilities for traffic levels and then predicting future long-term trends, in a manner that parallels Tuan and Park's work, though transferring some functionality from the edge into the interior of the network. This would then utilise the predictability of LRD traffic to adapt ECN marking schemes. Based on the work presented in this thesis and Tuan and Park's work, this could lead to improvements in network performance.

Another concept which is closely related to this work is an efficient estimator for the packet arrival rate from a self-similar source. The DiffServ framework requires meters located within edge routers to determine the rate of microflows from packet arrivals. Typically this is done using token buckets, where tokens are available when packets arrive below a specified rate. The depth of the token bucket allows for short bursts of traffic above this rate. However, meters that use a token buffer implementation are biased towards TCP connections with short round-trip delay. Other methods have been proposed, such as the time-sliding window approach [Clark and Fang 1998] which basically implements an averaging mechanism for estimation. It has been shown that these are more suitable for estimating the rates of TCP connections in heterogeneous networks.

However, recent studies have shown that TCP itself is a cause for self-similarity within network traffic [Feldmann *et al.* 1999, Feng *et al.* 2000, Veres and Boda 2000] and, furthermore, can propagate long-range dependence throughout the network [Veres *et al.* 2000]. Thus, appropriate rate estimators are required at the edges that reflect the true transmission rate of a TCP source. These estimators need to be based on the window dynamics of TCP and the self-similar structure of TCP's transmission rate, so that throughput achieved by the source matches the agreed DiffServ SLA. It is envisaged that simple techniques that use a mixture of an average and a slope (a differential between subsequent packet arrivals) will have good performance, effectively using the short-term correlations within the network traffic.

From a broader perspective, these research problems only touch the surface of a much deeper field, the control of self-similar processes. There has been a significant amount of theoretical work in the areas of identifying self-similar models and estimating their parameters. Also, their effect on queueing systems has been reasonably well-studied. However, the theory of controlling these processes and the impact of self-similarity on control systems is still in its infancy, as has been noted by Park [Park and Willinger 2000]. Further progress in this research field is required so that a global understanding of systems, such as the Internet which handles multiple types of traffic using protocols like TCP, can be gained.

REFERENCES

ABRY, P. AND VEITCH, D. (1998), 'Wavelet analysis of long-range-dependent traffic', *IEEE Transactions on Information Theory*, Vol. 44, No. 1, January, pp. 2–15.

ADAS, A. (1997), 'Traffic models in broadband networks', *IEEE Communications*, Vol. 35, No. 7, July, pp. 82–89.

ADAS, A. (1998), 'Using adaptive linear prediction to support real-time VBR video under RCBR network service model', *IEEE/ACM Transactions on Networking*, Vol. 6, No. 5, October, pp. 635–644.

ADAS, A. AND MUKHERJEE, A. (1995), 'On resource management and QoS guarantees for long range dependent traffic', *Proceedings of INFOCOM '95*, pp. 779–787.

ADDIE, R., ZUKERMAN, M. AND NEAME, T. (1995), 'Fractal traffic: Measurements, modelling and performance evaluation', *Proceedings of INFOCOM '95*, pp. 977–984.

ALTMAN, E. AND BAŞAR, T. (1998), 'Multiuser rate-based flow control', *IEEE Transactions on Communications*, Vol. 46, No. 7, July, pp. 940–949.

ALTMAN, E., BOCCARIA, F., BOLOT, J., NAIN, P., BROWN, P., COLLANGE, D. AND FENZY, C. (1995), 'Analysis of the TCP/IP flow control in high-speed wide-area networks', *Proceedings of the 34th Conference on Decision and Control*, December, pp. 368–373.

ALTMAN, E., AVRACHENKOV, K. AND BARAKAT, C. (2000), 'A stochastic model of TCP/IP with stationary random losses', *Proceedings of SIGCOMM '00*, August.

ARULAMBALAM, A., CHEN, X. AND ANSARI, N. (1996), 'Allocating fair rates for available bit rate service in ATM networks', *IEEE Communications*, Vol. 34, No. 11, November, pp. 92–100.

ÅSTRÖM, K. AND WITTENMARK, B. (1970), *Introduction to Stochastic Control*, Academic Press, New York.

ÅSTRÖM, K. AND WITTENMARK, B. (1997), *Computer-Controlled Systems Theory and Design*, Prentice Hall, New Jersey, 3rd ed.

ATHURALIYA, S., LAPSLEY, D. AND LOW, S. (2000), 'An enhanced random early marking algorithm for internet flow control', *Proceedings of INFOCOM '00*, March, pp. 1425–1434.

BARAKAT, C., ALTMAN, E. AND DABBOUS, W. (2000), 'On TCP performance in a heterogeneous network: A survey', *IEEE Communications*, Vol. 38, No. 1, January, pp. 40–46.

BARTON, R. AND POOR, H.V. (1988), 'Signal detection in fractional gaussian noise', *IEEE Transactions on Information Theory*, Vol. 34, No. 5, September, pp. 943–959.

BENMOHAMED, L. AND MEERKOV, S. (1993), 'Feedback control of congestion in packet switching networks: The case of a single congested node', *IEEE/ACM Transactions on Networking*, Vol. 1, No. 6, December, pp. 693–708.

BENMOHAMED, L. AND MEERKOV, S. (1997), 'Feedback control of congestion in packet switching networks: The case of multiple congested nodes', *International Journal of Communication Systems*, Vol. 10, pp. 227–246.

BENMOHAMED, L. AND WANG, Y. (1998), 'A control-theoretic ABR explicit rate algorithm for ATM switches with per-VC queueing', *Proceedings of INFOCOM '98*, Vol. 1, March, pp. 183–191.

BERAN, J. (1994), *Statistics for Long-memory Processes*, Chapman & Hall, New York.

BERAN, J., SHERMAN, R., TAQQU, M. AND WILLINGER, W. (1995), 'Long-range dependence in variable-bit-rate video traffic', *IEEE Transactions on Communications*, Vol. 43, No. 2/3/4, Feb/Mar/April, pp. 1566–1579.

BLACK, U. (1995), *ATM: Foundation for Broadband Networks*, Prentice Hall, Inc., New Jersey.

BLAKE, S., BLACK, D., CARLSON, M., DAVIES, E., WANG, Z. AND WEISS, W. (1998), 'An architecture for differentiated services', *IETF RFC2475*.

BOLOT, J. AND SHANKAR, A.V. (1990), 'Dynamical behaviour of rate-based flow control mechanisms', *ACM Computer Communication Review*, April, pp. 35–49.

BONALD, T. (1999), 'Comparison of TCP Reno and TCP Vegas: Efficiency and fairness', *Performance Evaluation*, Vol. 36-37, No. 1-4, pp. 307–332.

BONOMI, F. AND FENDICK, K. (1995), 'The rate-based flow control framework for the Available Bit Rate ATM service', *IEEE Network*, Mar/April, pp. 25–39.

BOORSTYN, R., BURCHARD, A., LIEBEHERR, J. AND OOTTAMAKORN, C. (2000), 'Effective envelopes: Statistical bounds on multiplexed traffic in packet networks', *Proceedings of INFOCOM '00*, March, pp. 1223–1232.

BOX, G., JENKINS, G. AND REINSEL, G. (1994), *Time Series Analysis: Forecasting and Control*, Prentice-Hall Inc., New Jersey, 3rd ed.

BRAKMO, L. AND PETERSON, L. (1995), 'TCP Vegas: End to end congestion avoidance on a global internet', *IEEE Journal on Selected Areas in Communications*, Vol. 13, No. 8, October, pp. 1465–1480.

BROCKWELL, P. AND DAVIS, R. (1991), *Time Series: Theory and Methods*, Springer-Verlag, New York, 2nd ed.

BU, J. AND SZNAIER, M. (1997), 'Robustness analysis with mixed performance objectives', *Proceedings of CDC*, December, pp. 446–451.

BU, T. AND TOWSLEY, D. (2000), *Fixed Point Approximations for TCP Behavior in an AQM Network*, Technical Report TR 2000-43, Department of Computer Science, University of Massachusetts, Amherst, MA 01003, July.

CHEN, T., LIU, S. AND SAMALAM, V. (1996), 'The available bit rate service for data in ATM networks', *IEEE Communications*, May, pp. 56–71.

CHIRUVOLU, G. AND SANKAR, R. (1997), 'An approach towards resource management and transportation of VBR video traffic', *Proceedings of ICC '97*, pp. 550–554.

CLARK, D. AND FANG, W. (1998), 'Explicit allocation of best-effort packet delivery service', *IEEE/ACM Transactions on Networking*, Vol. 6, No. 4, August, pp. 362–373.

CRILLY, A.J., EARNSHAW, R.A. AND JONES, H. (1991), *Fractals and Chaos*, Springer-Verlag, New York.

CROVELLA, M. AND BESTAVROS, A. (1997), 'Self-similarity in world wide web traffic: Evidence and possible causes', *IEEE/ACM Transactions on Networking*, Vol. 5, No. 6, December, pp. 835–846.

CRUZ, R. (1991), 'A calculus for network delay, part I: Network elements in isolation', *IEEE Transactions on Information Theory*, Vol. 37, No. 1, January, pp. 114–131.

CRUZ, R. (1995), 'Quality of service guarantees in virtual circuit switched networks', *IEEE Journal on Selected Areas in Communications*, Vol. 13, No. 6, August, pp. 1048–1056.

DELLACHERIE, C. AND MEYER, P. (1982), *Probabilities and Potential*, North Holland, Amsterdam.

DOVROLIS, C., STILIADIS, D. AND RAMANATHAN, P. (1999), 'Proportional differentiated services: Delay differentiation and packet scheduling', *Proceedings of SIGCOMM '99*, August.

Duffield, N. and Whitt, W. (1997), 'Control and recovery from rare congestion events in a large multi-server system', *Queueing Systems*, Vol. 26, pp. 69–104.

Fall, K. and Floyd, S. (1996), 'Simulation-based comparisons of Tahoe, Reno and SACK TCP', *Computer Communiction Review*, Vol. 26, No. 3, July, pp. 5–21.

Fang, W., Seddigh, N. and Nandy, B. (2000), 'A time sliding window three colour marking', March. IETF Internet Draft *draft-fang-diffserv-tc-tswtcm-01.txt.*

Feldmann, A., Gilbert, A., P., H. and Willinger, W. (1999), 'Dynamics of IP traffic: A study of the role of variability and impact of control', *Proceedings of SIGCOMM '99*, August.

Fendick, K. (1996), 'Evolution of controls for the available bit rate service', *IEEE Communications*, November, pp. 35–39.

Feng, W., Tinnakornsrisuphap, P. and Philp, I. (2000), 'On the burstiness of the TCP congestion-control mechanism in a distributed computing system', *Proceedings of ICDCS'00*, April.

Flandrin, P. (1992), 'Wavelet analysis and synthesis of fractional brownian motion', *IEEE Transactions on Information Theory*, Vol. 38, pp. 910–917.

Floyd, S. (1994), 'TCP and explicit congestion notification', *ACM Computer Communications Review*, Vol. 24, No. 5, October, pp. 10–23.

Floyd, S. and Henderson, T. (1999), 'The NewReno modification to TCP's fast recovery algorithm', *IETF RFC 2582*, April.

Floyd, S. and Jacobsen, V. (1993), 'Random early detection gateways for congestion avoidance', *IEEE/ACM Transactions on Networking*, Vol. 1, No. 4, August, pp. 397–413.

Floyd, S. and Jacobson, V. (1995), 'Link-sharing and resource management models for packet networks', *IEEE/ACM Transactions on Networking*, Vol. 3, No. 4, August, pp. 365–386.

Floyd, S., Handlye, M., Padhye, J. and Widmer, J. (2000), 'Equation-based congestion control for unicast applications', *Proceedings of SIGCOMM '00.*

Frieden, B.R. (1985), 'Dice, entropy and likelihood', *Proceedings of the IEEE*, Vol. 73, No. 12, December, pp. 1764–1770.

Garrett, M. and Willinger, W. (1994), 'Analysis, modelling and generation of self-similar VBR video traffic', *Computer Communications Review Proceedings of ACM SIGCOMM*, Vol. 24, No. 4, August, pp. 269–280.

GIBBENS, R. AND KELLY, F. (1997), 'Measurement-based connection admission control', *15th ITC Proceedings*, June.

GIBBENS, R.J. AND KELLY, F.P. (1999), 'Resource pricing and the evolution of congestion control', *Automatica*, Vol. 35, pp. 1969–1985.

GIORDANO, S., PAGANO, M., PANNOCCHIA, R. AND RUSSO, F. (1996), 'A new call admission control scheme based on the self similar nature of multimedia traffic', *Proceedings of ICC '96*, March, pp. 1612–1618.

GOVINDAN, R. AND REDDY, A. (1997), 'An analysis of internet inter-domain topology and router stability', *Proceedings of INFOCOM '97*, April, pp. 851–858.

GREEN, M. AND LIMEBEER, D. (1995), *Linear Robust Control*, Prentice-Hall Inc., New Jersey.

GRIPENBERG, G. AND NORROS, I. (1996), 'On the prediction of fractional brownian motion', *Journal of Applied Probability*, Vol. 33, pp. 400–410.

GROSSGLAUSER, M. AND BOLOT, J. (1999), 'On the relevance of long-range dependence in network traffic', *IEEE/ACM Transactions on Networking*, Vol. 7, No. 5, October, pp. 629–640.

GROSSGLAUSER, M. AND TSE, D. (1999), 'A framework for robust measurement-based admission control', *IEEE/ACM Transactions on Networking*, Vol. 7, No. 3, June.

HASSAN, M., SIRISENA, H. AND BREEN, J. (1996), 'Design and analysis of a congestion control architecture using a proportional feedback controller', *Proceedings of ATNAC*, December, pp. 607–612.

HAYKIN, S. (1994), *Communication Systems*, John Wiley & Sons, Inc., New York.

HEINANEN, J. AND GUERIN, R. (1999), 'A two rate three color marker', *IETF RFC2698*, September.

HEINANEN, J., BAKER, F., WEISS, W. AND WROCLAWSKI, J. (1999), 'Assured forwarding PHB group', *IETF RFC2597*, June.

HENGARTNER, U., BOLLIGER, J. AND GROSS, T. (2000), 'TCP Vegas revisited', *Proceedings of INFOCOM '00*, March.

HEYMAN, D. AND LAKSHMANN, T. (1996), 'What are the implications of long-range dependence for VBR-video traffic engineering?', *IEEE/ACM Transactions on Networking*, Vol. 4, No. 3, June, pp. 301–317.

HOSKING, J. (1981), 'Fractional differencing', *Biometrika*, Vol. 68, No. 1, pp. 165–176.

HU, L.C., TSAI, W.K., KIM, Y. AND IYER, M. (1998), 'Temporal flow control for ABR traffic management in integrated networks', *Proceedings of GLOBECOM '98*, November.

HUANG, S. AND CAMBANIS, S. (1978), 'Stochastic and multiple wiener integrals for gaussian processes', *The Annals of Probability*, Vol. 6, No. 4, pp. 585–614.

JACOBSEN, V., NICHOLS, K. AND PODURI, K. (1999), 'An expedited forwarding PHB', *IETF RFC2598*, June.

JACOBSON, V. (1988), 'Congestion avoidance and control', *Proceedings of SIGCOMM '88*, August, pp. 314–329.

JAFFE, J.M. (1981), 'Bottleneck flow control', *IEEE Transactions on Communications*, Vol. 29, July, pp. 954–962.

JELENKOVIĆ, P. (2000), *Self-similar Network Traffic and Performance Evaluation*, John Wiley & Sons, Inc., New York, Chap. Asymptotic Analysis of Queues with Subexponential Arrival Processes, pp. 249–268.

JOHANSSON, P., WEDLUND, E. AND KARLSSON, J. (1997), 'Interaction between TCP flow control and ABR rate control', *IEEE ATM Workshop '97*, May, pp. 455–464.

KALYANARAMAN, S., JAIN, R., FAHMY, S., GOYAL, R. AND VANDALORE, B. (2000), 'The ERICA switch algorithm for ABR traffic management in ATM networks', *IEEE/ACM Transactions on Networking*, Vol. 8, No. 4, February, pp. 87–99.

KELLY, F. (1999), 'Mathematical modelling of the internet', *Fourth International Congress on Industrial and Applied Mathematics*, July.

KELLY, F.P. (2000), 'Models for a self-managed internet', *Submitted to Royal Society.*

KELLY, F.P., ZACHARY, S. AND ZIEDINS, I.B. (Eds.) (1996), *Stochastic Networks: Theory and Applications*, Vol. 4 of Royal Statistical Society Lecture Notes, Oxford University Press, Oxford, UK, Chap. Notes on Effective Bandwidths, pp. 141–168.

KELLY, F.P., MAULLOO, A.K. AND TAN, D.K.H. (1998), 'Rate control for communication networks: Shadow prices, proportional fairness and stability', *Journal of the Operational Research Society*, Vol. 49, No. 3, March, pp. 237–252.

KLEINROCK, L. (1975), *Queueing Systems*, Vol. 1, Wiley, New York.

KLEMEŠ, V. (1974), 'The Hurst phenomenon: A puzzle?', *Water Resources Research*, Vol. 10, No. 4, pp. 675–688.

KOLAROV, K. AND RAMAMURTHY, G. (1999), 'A control-theoretic approach to the design of an explicit rate controller for ABR service', *IEEE/ACM Transactions on Networking*, Vol. 7, No. 5, October, pp. 741–753.

KUNG, H. AND MORRIS, R. (1995), 'Credit-based flow control for ATM networks', *IEEE Network*, Mar/April, pp. 40–48.

KUNNIYUR, K. AND SRIKANT, R. (200), *A Time Scale Decomposition Approach to Adaptive ECN Marking*, Technical Report, Department of Electrical and Computer Engineering and Coordinated Science Laboratory, University of Illinois, Urbana, IL 61801, September.

LA, R. AND ANANTHARAM, V. (2000), 'Charge-sensitive TCP and rate control in the internet', *Proceedings of INFOCOM '00*, March, pp. 1166–1175.

LAKSHMAN, T., MADHOW, U. AND SUTER, B. (1997), 'The performance of TCP/IP for networks with high bandwidth-delay products and random loss', *IEEE/ACM Transactions on Networking*, Vol. 5, No. 3, June, pp. 336–350.

LELAND, W., TAQQU, M., WILLINGER, W. AND WILSON, D. (1994), 'On the self-similar nature of ethernet traffic (extended version)', *IEEE/ACM Transactions on Networking*, Vol. 2, No. 1, February, pp. 1–15.

LI, S., CHONG, S. AND HWANG, C. (1995), 'Link capacity allocation and network control by filtered input rate in high-speed networks', *IEEE/ACM Transactions on Networking*, Vol. 3, No. 1, February, pp. 10–25.

LIEW, S. AND TSE, D. (1998), 'A control-theoretic approach to adapting VBR compressed video for transport over a CBR communications channel', *IEEE/ACM Transactions on Networking*, Vol. 6, No. 1, February, pp. 42–55.

LOW, S. AND LAPSLEY, D. (1999), 'Optimization flow control I: Basic algorithm and convergence', *IEEE/ACM Transactions on Networking*, Vol. 7, No. 6, December, pp. 861–874.

LOW, S., PETERSON, L. AND WANG, L. (2000), 'Understanding TCP Vegas: Theory and practice', *Preprint.*

MANDELBROT, B. AND VAN NESS, J. (1968), 'Fractional brownian motions, fractional noises and applications', *SIAM Review*, Vol. 10, No. 4, October, pp. 423–437.

MANSURIPUR, M. (1987), *Introduction to Information Theory*, Prentice-Hall, Inc., Englewood Cliffs, New Jersey.

MATHIS, M. AND MAHDAVI, J. (1996), 'Forward acknowledgement: Refining TCP congestion control', *Proceedings of SIGCOMM*, August.

MATHIS, M., SEMKE, J. AND MAHDAVI, J. (1997), 'The macroscopic behavior of the TCP congestion avoidance algorithm', *ACM Computer Communications Review*, Vol. 27, No. 3, July.

MAZUMDAR, R., MASON, L. AND DOULIGERIS, C. (1991), 'Fairness in network optimal flow control: Optimality of product forms', *IEEE Transactions on Communications*, Vol. 39, pp. 775–782.

METZ, C. (1999), 'IP QoS: Traveling in first class on the internet', *IEEE Internet Computing Magazine*, March/April, pp. 84–88.

MISRA, V., GONG, W. AND TOWSLEY, D. (2000), 'Fluid-based analysis of a network of AQM routers supporting TCP flows with an application to RED', *Proceedings of SIGCOMM '00*, August.

MO, J. AND WALRAND, J. (2000), 'Fair end-to-end window-based congestion control', *IEEE/ACM Transactions on Networking*, Vol. 8, No. 5, October, pp. 556–568.

NORROS, I. (1994), 'A storage model with self-similar input', *Queueing Systems*, Vol. 16, pp. 387–396.

ONVURAL, R. (1995), *Asynchronous Transfer Mode Networks*, Artech House Inc., Norwood MA, 2nd ed.

ÖSTRING, S. AND SIRISENA, H. (2001), 'The influence of long-range dependence on traffic prediction', *Accepted for ICC'01*, June.

ÖSTRING, S., SIRISENA, H. AND HASSAN, M. (1997), 'Optimization of a rate feedback control with feedforward for traffic management in ATM networks', *Proceedings of APCC '97*, Vol. 3, December, pp. 1551–1555.

ÖSTRING, S., SIRISENA, H. AND HUDSON, I. (1999a), 'Dual dimensional ABR control scheme using predictive filtering of self-similar traffic', *Proceedings of ICC '99*, Vol. 3, June, pp. 129–134.

ÖSTRING, S., SIRISENA, H. AND HUDSON, I. (1999b), 'Robustness of ABR congestion control algorithms with regards to self-similar traffic models', *IEEE LAN/MAN'99 Workshop*, November, pp. 112–115.

ÖSTRING, S., SIRISENA, H. AND HUDSON, I. (2000a), 'Rate control of elastic connections competing with long-range dependent network traffic', *Accepted for IEEE Transactions on Communications.*

ÖSTRING, S., SIRISENA, H. AND HUDSON, I. (2000b), 'Robust ABR control for uncertainties in long-range dependent traffic', *Accepted for post-IEEE LAN/MAN'99 Publication.*

ÖSTRING, S., SIRISENA, H. AND HUDSON, I. (2000c), 'Sensitivity of ABR congestion control algorithms to hurst parameter estimates', *Proceedings of IFIP-TC6 Networking 2000*, May, pp. 36–48.

OZVEREN, C., SIMCOE, R. AND VARGHESE, G. (1994), 'Reliable and efficient hop-by-hop flow control', *Proceedings of ACM SIGCOMM'94*, September, pp. 89–100.

PADHYE, J., FIROIU, V., TOWSLEY, D. AND KUROSE, J. (2000), 'Modeling TCP throughput: A simple model and its empirical validation', *IEEE/ACM Transactions on Networking*, Vol. 8, No. 2, April, pp. 133–145.

PAREKH, A. AND GALLAGHER, R. (1993), 'A generalized processor sharing approach to flow control in integrated services networks: The single-node case', *IEEE/ACM Transactions on Networking*, Vol. 1, No. 3, June, pp. 344–357.

PAREKH, A. AND GALLAGHER, R. (1994), 'A generalized processor sharing approach to flow control in integrated services networks: The multiple node case', *IEEE/ACM Transactions on Networking*, Vol. 2, No. 2, April, pp. 137–150.

PARK, K. AND TUAN, T. (2000), 'Performance evaluation of multiple time scale tcp under self-similar traffic conditions', *ACM Transactions on Modeling and Computer Simulation*, Vol. 10, No. 3, pp. 1–26.

PARK, K. AND WILLINGER, W. (Eds.) (2000), *Self-similar Network Traffic and Performance Evaluation*, John Wiley & Sons, Inc., New York.

PARK, K., KIM, G. AND CROVELLA, M. (1996), 'On the relationship between file sizes, transport protocols, and self-similar network traffic', *Proceedings of IEEE International Conference on Network Protocols*, pp. pp. 171–180.

PAXSON, V. AND FLOYD, S. (1995), 'Wide area traffic: The failure of poisson modelling', *IEEE/ACM Transactions on Networking*, Vol. 3, No. 3, June, pp. 226–244.

POOR, H.V. (1999), 'Maximum entropy and robust prediction on a simplex', *IEEE Transactions on Information Theory*, Vol. 45, No. 4, May, pp. 1150–164.

POOR, H.V. (2000), 'Private communication', December. Via email.

QIU, J. AND KNIGHTLY, E. (1999), 'Inter-class resource sharing using statistical service envelopes', *Proceedings of INFOCOM '99*, March, pp. 1404–1411.

RAMAKRISHNAN, K. AND NEWMAN, P. (1995), 'Integration of rate and credit scheme for ATM flow control', *IEEE Network*, Mar/Apr, pp. 49–56.

REN, H. AND PARK, K. (2000), 'Toward a theory of differentiated services', *Proceedings of IEEE/IFIP International Workshop on Quality of Service*, pp. 211–220.

RITTER, M. (1998), *Modeling of Flow Control Mechanisms for the Available Bit Rate Service*, PhD thesis, Institut für Informatik, Universität Würzburg, Am Hubland, D-97074 Würzburg, Germany, January.

ROHRS, C. AND BERRY, R. (1997), 'A linear control approach to explicit rate feedback in ATM networks', *Proceedings of INFOCOM '97*, April, pp. 277–282.

ROHRS, C., BERRY, R. AND O'HALEK, S. (1996), 'Control engineer's look at ATM congestion avoidance', *Computer Communications*, Vol. 19, pp. 226–234.

ROSE, O. (1995), 'Statistical properties of MPEG video traffic and their impact on traffic modelling in ATM systems', *Proceedings of 20th Conference on Local Computer Networks 1995*, October, pp. 397–406.

ROUGHAN, M. AND ERRAMILLI, A. (2001), 'Network performance for TCP networks Part I: Persistent sources', *Submitted to INFOCOM 2001.*

ROUGHAN, M., VEITCH, D. AND ABRY, P. (1998), 'On-line estimation of the parameters of long-range dependence', *Proceedings of GLOBECOM '98*, November.

ROUGHAN, M., VEITCH, D. AND ABRY, P. (2000), 'Real-time estimation of the parameters of long-range dependence', *IEEE/ACM Transactions on Networking*, Vol. 8, No. 4, August, pp. 467–478.

RYU, B. AND ELWALID, A. (1996), 'The importance of long-range dependence of VBR video traffic in ATM traffic engineering: Myths and realities', *Proceedings of SIGCOMM*, pp. 3–14.

SAHU, S., NAIN, P., TOWSLEY, D., DIOT, C. AND FIROIU, V. (2000), 'On achievable service differentiation with token bucket marking for TCP', *ACM Sigmetrics.*

SAMORODNITSKY, G. AND TAQQU, M. (1994), *Stable Non-Gaussian Random Processes: Stochastic Models with Infinite Variance*, Chapman & Hall, New York.

SANG, A. AND LI, S.Q. (2000), 'A predictability analysis of network traffic', *Proceedings of INFOCOM '00*, March, pp. 342–351.

SCHWARTZ, M. (1987), *Telecommunication Networks: Protocols, Modelling and Analysis*, Addison-Wesley Publishing Company, Reading MA.

SCHWARTZ, M. (1996), *Broadband Integrated Networks*, Prentice Hall, Inc.,, New Jersey.

STEVENS, W. (1994), *TCP/IP Illustrated: The Protocols*, Vol. 1, Addison-Wesley Publishing Company, Reading MA.

TAQQU, M., WILLINGER, W. AND SHERMAN, R. (1997), 'Proof of a fundamental result in self-similar traffic modelling', *Computer Communication Review*, Vol. 27, pp. 5–23.

THOMPSON, K., MILLER, G. AND WILDER, R. (1997), 'Wide-area internet traffic patterns and characteristics', *IEEE Network*, Vol. 11, No. 6, November/December, pp. 10–23.

TSYBAKOV, B. AND GEORGANAS, N. (1997), 'On the self-similar traffic in ATM queues: Definitions, overflow probability bound and cell delay distributions', *IEEE/ACM Transactions on Networking*, Vol. 5, No. 3, June, pp. 397–409.

TUAN, T. AND PARK, K. (1999), 'Multiple time scale congestion control for self-similar network traffic', *Performance Evaluation*, Vol. 36-37, No. 1-4, pp. 359–386.

TUAN, T. AND PARK, K. (2000), 'Multiple time scale redundancy control for QoS-sensitive transport of real-time traffic', *Proceedings of INFOCOM '00.*

VEITCH, D. AND ABRY, P. (1999), 'A wavelet based joint estimator of the parameters of long-range dependence', *IEEE Transactions on Information Theory*, Vol. 45, No. 3, April, pp. 878–898.

VERES, A. AND BODA, M. (2000), 'The chaotic nature of TCP congestion control', *Proceedings of INFOCOM '00*, March, pp. 1715–1723.

VERES, A., KENESI, Z., MOLNAR, S. AND VATTAY, G. (2000), 'On the propagation of long-range dependence in the internet', *Proceedings of SIGCOMM*, August.

WALRAND, J. AND VARAIYA, P. (1996), *High-Performance Communication Networks*, Morgan Kaufmann Publishers, Inc., San Franciso, 2nd ed.

WHITE, P. AND CROWCROFT, J. (1997), 'The integrated services in the internet: State of the art', *Proceedings of the IEEE*, Vol. 85, No. 12, December, pp. 1934–1946.

WILLINGER, W., TAQQU, M., SHERMAN, R. AND WILSON, D. (1997), 'Self-similarity through high-variability: Statistical analysis of ethernet LAN traffic at the source level', *IEEE/ACM Transactions on Networking*, Vol. 5, No. 1, January, pp. 71–96.

XIAO, X. AND LIONEL, M.N. (1999), 'Internet QoS: A big picture', *IEEE Network*, March/April, pp. 8–18.

YANG, C. AND REDDY, A. (1995), 'A taxonomy for congestion control algorithms in packet switching networks', *IEEE Network*, Vol. 9, July/August, pp. 34–45.

ZHANG, H. AND YANG, O. (1997), 'Design of robust congestion controllers for ATM networks', *Proceedings of INFOCOM '97*, April, pp. 302–309.

ZHAO, Y., LI, S.Q. AND SIGARTO, S. (1997), 'A linear dynamic model for design of stable explicit-rate ABR control scheme', *Proceedings of INFOCOM '97*, April, pp. 283–292.

ZHOU, K., GLOVER, K., BODENHEIMER, B. AND DOYLE, J. (1994), 'Mixed $\mathcal{H}_2$ and $\mathcal{H}_\infty$ performance objectives I: Robust performance analysis', *IEEE Transactions on Automatic Control*, Vol. 39, No. 8, August, pp. 1564–1574.