Technical Report

UCAM-CL-TR-528
ISSN 1476-2986

Number 528

UNIVERSITY OF
CAMBRIDGE

Computer Laboratory

Measurement-based management of network resources

Andrew William Moore

April 2002

JJ Thomson Avenue
Cambridge CB3 0FD
United Kingdom
phone +44 1223 763500

http://www.cl.cam.ac.uk/

This technical report is based on a dissertation submitted by the author for the degree of Doctor of Philosophy to the University of Cambridge.

Technical reports published by the University of Cambridge Computer Laboratory are freely available via the Internet:

http://www.cl.cam.ac.uk/TechReports/

Series editor: Markus Kuhn

ISSN 1476-2986

Summary

Measurement-Based Estimators are able to characterise data flows, enabling improvements to existing management techniques and access to previously impossible management techniques. It is the thesis of this dissertation that in addition to making practical adaptive management schemes, measurement-based estimators can be practical within current limitations of resource.

Examples of network management include the characterisation of current utilisation for explicit admission control and the configuration of a scheduler to divide link-capacity among competing traffic classes. Without measurements, these management techniques have relied upon the accurate characterisation of traffic — without accurate traffic characterisation, network resources may be under or over utilised.

Embracing Measurement-Based Estimation in admission control, Measurement-Based Admission Control (MBAC) algorithms have allowed characterisation of new traffic flows while adapting to changing flow requirements. However, there have been many MBAC algorithms proposed, often with no clear differentiation between them. This has motivated the need for a realistic, implementation-based comparison in order to identify an ideal MBAC algorithm.

This dissertation reports on an implementation-based comparison of MBAC algorithms conducted using a purpose built test environment. The use of an implementation-based comparison has allowed the MBAC algorithms to be tested under realistic conditions of traffic load and realistic limitations on memory, computational resources and measurements. Alongside this comparison is a decomposition of a group of MBAC algorithms, illustrating the relationship among MBAC algorithm components, as well as highlighting common elements among different MBAC algorithms.

The MBAC algorithm comparison reveals that, while no single algorithm is ideal, the specific resource demands, such as computation overheads, can dramatically impact on the MBAC algorithm's performance. Further, due to the multiple timescales present in both traffic and management, the estimator of a robust MBAC algorithm must base its estimate on measurements made over a wide range of timescales. Finally, a reliable estimator must account for the error resulting from random properties of measurements.

Further identifying that the estimator components used in MBAC algorithms need not be tied to the admission control problem, one of the estimators (originally constructed as part of an MBAC algorithm) is used to continuously characterise resource requirements for a number of classes of traffic. Continuous characterisation of traffic, whether requiring similar or orthogonal resources, leads to the construction and demonstration of a network switch that is able to provide differentiated service while being adaptive to the demands of each traffic class. The dynamic allocation of resources is an approach unique to a measurement-based technique that would not be possible if resources were based upon static declarations of requirement.

Acknowledgements

I would like to thank my supervisors, Simon Crosby and Ian Leslie, for encouragement and valuable advice during the course of my research.

I am grateful to many people for their cajoling, enthusiasm, encouragement and assistance.

At the risk of missing-out people whom I should not, I thank current and former members of the Systems Research Group, including Richard Black, Austin Donnelly, Tim Granger, Steven Hand, Rebecca Isaacs, Richard Mortier, Ian Pratt, Kerry Rodden and Neil Stratford.

My thanks to my colleagues in the Computer Laboratory, notably Neil Dodgson, Margaret Levitt, Kona McPhee and Lewis Tiffany.

My support from college has been greatly appreciated — my thanks in particular to Margaret Cathie, David Greaves, Paul Hewitt and Andy Hopper.

My thanks to members of my old department, none of whom ever doubted this would all come to fruition. Thanks in particular to Jim Breen, Eryn Glover, Grant Hampson, Tony McGregor and Henry R. Wu.

I am grateful to the researchers who have willingly answered my questions and discussed my proposals, notably Ian Graham, Matt Grossglauser, Richard Gibbens, Suigh Jamin, Frank Kelly, Peter Key, Ed Knightly, Jingyu Qiu, Scott Shenker, Jonathan Smith and Gary Walley.

I am indebted to Ian Leslie, Steven Hand and Rebecca Isaacs who have read and commented on earlier drafts of this dissertation, and extend special thanks to Ralphe Neill and Philippa Baines who proof-read the final text.

This work was supported by scholarships from the Computer Laboratory, Corpus Christi College and the Cambridge Commonwealth Trust. My thanks to these orgainisations for making it possible for me to embark upon this work.

Finally, I thank my family and friends geographically near and far for their love and support.

Contents

List of Figures

List of Tables

Glossary

AAL	ATM Adaptation Layer
AAL5	ATM Adaptation Layer 5
ABR	Available Bit Rate
AC	Admission Control
ADSL	Asymmetric Digital Subscriber Line
AR	Acceptance Region
ATM	Asynchronous Transfer Mode
BE	Best Effort
CAC	Connection Admission Control
CBP	Complete Buffer Partitioning
CBR	Constant Bit Rate
CBQ	Class-Based Queueing
CDF	Cumulative Density Function
CE	Certainty Equivalence
CLR	Cell Loss Ratio
COPS	Common Open Policy Service
CORBA	Common Object Request Broker Architecture
DQLT	Dynamic Queue Length Threshold
EWMA	Exponentially Weighted Moving Average
FIFO	First In First Out
FSC	Fair Service Curve
GPS	Generalised Processor Sharing
HTTP	Hyper-Text Transfer Protocol
ICT	Inter-Cell Time
IFP	Interrupted Fluid Process
IP	Internet Protocol
ISP	Internet Service Provider
ITU	International Telecommunications Union
JPEG	Joint Photographic Experts Group
LAN	Local Area Network
LPF	Lowest Priority First
LRD	Long Range Dependence
MBAC	Measurement-Based Admission Control
MBE	Measurement-Based Estimator
MBM	Measurement-Based Management
MLU	Mean Line Utilisation
MPEG	Moving Picture Experts Group
MPLS	Multiprotocol Label Switching
MP3	MPEG standard for digital audio compression
MSS	Maximum Segment Size

MTU	Maximum Transmission Unit
OS	Operating System
PBS	Partial Buffer Sharing
PCR	Peak Cell Rate
PFQ	Packet Fair Queueing
PGPS	Packet-by-packet Generalised Process Sharing
PLR	Proportional Loss Rate
POTS	Plain Old Telephone Service
PSTN	Public Switched Telephone Network
PVC	Permanent Virtual Circuit
QoS	Quality of Service
RED	Random Early Discard
REM	Rate Envelope Multiplexing
RPC	Remote Procedure Call
RSM	Rate Statistical Multiplexing
RSVP	Resource ReSerVation Protocol
RTP	Real-time Transport Protocol
RTT	Round Trip Time
SCED	Service Curve Earliest Deadline
SCGF	Scaled Cumulative Generating Function
SCR	Sustainable Cell Rate
SLA	Service Level Agreement
SLS	Service Level Specification
SNMP	Simple Network Management Protocol
SONET	Synchronous Optical NETwork
TCP	Transmission Control Protocol
TM	Traffic Management
UBR	Unspecified Bit Rate
UDP	User Datagram Protocol
VBR	Variable Bit Rate
VC	Virtual Circuit
VCI	Virtual Channel Identifier
VLAN	Virtual Local Area Network
VP	Virtual Path
VPI	Virtual Path Identifier
VPN	Virtual Private Network
WAN	Wide Area Network
WFQ	Weighted Fair Queueing
WF^{2}Q	Worst-Case Weighted Fair Queueing
WF^{2}Q+	Worst-Case Weighted Fair Queueing Plus
WRR	Weighted Round Robin
WWW	World Wide Web

Nomenclature of Chapter 4

E	estimate of *effective bandwidth*
X,$\hat{X}$	link utilisation sample
s	current time
τ	indivisible period of measurement
K, k	traffic classes
N, n	number of calls
p	peak rate
s	sustained rate
r	ratio of s to p
b	mean burst size
B_T	maximum burst size
T,t	a number of intervals τ
q	queue length
T_h	flow holding time
σ	standard deviation (commonly of measurement x)
$\bar{x}$	mean of x

Chapter 1

Introduction

This dissertation investigates two aspects of measurement-based network management. Firstly, it reports on a unique evaluation of Measurement-Based Admission Control (MBAC) algorithms using a purpose built test environment. Secondly, it outlines the design and implementation of a measurement-based scheme offering differentiated service provision in a network switch.

Network management techniques have long been of interest to the networking research community. The management of networks involves both the control of data flowing through a network and maintaining the general well-being of the network. An example of network management is in maximising utilisation of the network while maintaining service guarantees to network users.

This dissertation restricts itself to two network management techniques. The first of these is a study of a subset of admission control techniques. Admission Control (AC) attempts to make best use of the finite link-capacity across a network by admitting a new flow of data into a network if the new flow can be accommodated without impacting upon the guarantees made to existing flows of data. The second network management technique is the partitioning of transmission and buffer resources among two or more classes of traffic using a common transmission path.

Both of the network management techniques assessed in this dissertation are measurement-based. This permits prior characterisation of the traffic that is minimal or incorrect without impacting upon the performance of the ongoing characterisation derived from measurements. The first of the two techniques is an investigation of MBAC algorithms which allow the best utilisation of network resources with minimal prior characterisation of new flows. However, there have been many proposed MBAC algorithms. This has motivated the need for an implementation-based comparison in order to identify an ideal MBAC algorithm.

The use of an evaluation environment based upon an experimental test-rig allows a comprehensive investigation into the performance and behaviour of AC algorithms. The use of this test environment allows the AC algorithm to be placed under realistic conditions of traffic and network, while being subject to realistic limits on memory, computational resource and access to measurements.

Measurement-Based Estimators also allow new techniques for network management to be derived. The division of resources, such as link-capacity, is a network management task required when several differentiated classes of traffic must share a common link. A Measurement-Based Estimator is able to compute the resource requirements of each class of traffic based upon measurements of line utilisation alone. In addition to requiring minimal prior characterisation of the traffic classes, a task that is not always possible for some traffic types, a measurement-based technique can adapt to changes in the requirements. This is a significant advantage over existing class differentiation systems that must either make a static commitment at the time they are configured or offer no service differentiation at all.

The following section provides a more detailed motivation for measurement-based management of network resources.

1.1 Motivation

The motivation for the management of networks is two-fold. Firstly, network resource is finite and as such it must be shared among competing users. For the purposes of this dissertation, the principle resource of a network is the link-capacity between network nodes. Network nodes are both the endpoints of a network and the intermediate places a flow of data may pass between. However, link-capacity alone is not the only resource for which management schemes are proposed in this dissertation. A transmission path, the link between network nodes is a resource with a finite capacity. The transmission path often has a significant cost associated with it. A transatlantic cable or the Local Area Network (LAN) cables around a modern office each have associated costs, both for the original installation and for its ongoing maintenance. Thus, for network management, the first motivating task becomes the maximisation of utilisation of a network resources thereby obtaining best value from the finite resources.

The second motivation for the management of networks is the supply of a particular quality of resource to a user. Network Quality of Service (QoS) may be a guarantee of a quantity of link-capacity to a user, although this is not the only form of QoS guarantee. Aside from resource QoS, other guarantees may be that a network provides a reliable permanent data-path between source and destination. To provide this service the network management system commonly needs to manage redundant paths as well as the mechanisms for determining path failure and recovery. A further example of such network management is in the planning of a network. The QoS in this case may be that the network will grow (in capacity) sufficiently so as to continue to provide its users with the resource they require. Clearly, network management may take a number of different forms.

Network management depends upon the reliable prediction of network traffic, both current network traffic, in order to determine current network allocations, and predictions of future network traffic so as to plan for future requirements. For almost a century, the planning of capacity in the principle network of the day, the telephone network, was able to use predictable models of traffic behaviour. However, the growth in networks carrying data has meant recent studies (e.g. [Paxson95, Crovella97, Feldmann98b]) have revealed that the traffic of data networks and telephone networks no longer follow reliable and easily understood models of behaviour.

The models of network traffic have become complicated, incomplete and, in some cases, impossible to formulate. This predicament provides the motivation for the Measurement-Based Management of network resources. Measurement of the activity on a link reveals the utilisation. The utilisation characteristics of the traffic present on the link are revealed without a specific, previous (*a priori*) characterisation of the traffic being known. The measurement-derived characteristics can then be used in place of a model of the traffic. If an objective of a network management scheme is to maximise utilisation, the scheme may aggregate new flows of data into a common link. A model-driven scheme would compute the individual flow requirements and add new flows as the capacity allows. In contrast, a measurement-driven scheme would measure the aggregate utilisation and new flows could be admitted if capacity still remained. This describes a simple Measurement-Based Management scheme that may remove the need for a network traffic to have an *a priori* model of characterisation for each flow.

Using measurements can remove the need for the complicated, and incomplete models of network traffic. Measurement-based schemes can be used where no model was available. Modern traffic has been typified as difficult to model, therefore, measurements allow the management of networks carrying modern, evolving traffic.

Measurements of link utilisation, while common, need not be the only input to a management scheme. Measurement of other network parameters such as the probability of a buffer being full, the ratio in which new flows are admitted into a network, the end to end delay across a network, or the variation in delay for packets transmitted through a single buffer, are each examples of parameters the measurement of which would provide important input to the appropriate management scheme. That said, the majority of Measurement-Based Management schemes studied in this dissertation use measurements of line utilisation as input.

In addition to characterising traffic, measurement-based management schemes are able to dynamically adapt to changes in resource requirements. Such dynamic allocation of resources is simply not possible for environments where the *a priori* characterisation of network data will result in static resource allocation. Measurement-based resource management will adapt allocations as flow requirements change. Such adaptation is unique to a measurement-based approach.

With networks such as the Internet[1] not, by default, offering any explicit QoS, the need for networks to support service guarantees may be considered doubtful. However, the need for QoS has not disappeared because the default behaviour of common Internet services have no QoS associated with them.

Network management continues to offer QoS, whether as an assured connection of desired capacity (as in the telephone network), a bound on packet loss (as might be expected in a multi-service

[1] Whereas an internet may be considered the interconnection of two or more networks, the Internet specifically refers to the world-wide interconnection of networks of that name.

network), the uninterrupted connectivity of a redundant network architecture, or a network the demands of which are sufficiently easy to characterise so as to make its growth tractable. As an example, the Internet, a form of multi-service, does not offer the QoS of reliable end to end connectivity. However, the network supplying the underlying infrastructure on which the Internet runs may specify the high reliability of uninterrupted connectivity between each network node. Such a situation may be constructed for each of these examples of QoS and these form only a small part of the many variations under which network provision may be made.

The Internet is not QoS-less, and work by various Internet standards groups has lead to two significant approaches for the future provision of QoS in such networks. The INTSERV proposal [Braden97], offering per-flow guarantees through explicit admission control and resource reservation, was the earlier of these although potential complexity and unwieldy per-router overheads have made this approach less appealing. The alternative, DIFFSERV [Nichols99], does not use explicit admission control, and relies instead upon the classification of flows prior to entry into the network. Each class of traffic, (consisting of one or more flows) is supplied with a previously agreed amount of resource. In this way service guarantees are made to each traffic class. Currently the DIFFSERV approach is seen as more practical than INTSERV. However, determining the amount of resource that ought to be committed for each class of traffic continues to be an ongoing challenge of the DIFFSERV approach.

This section identifies that Measurement-Based Management is an approach that is able to offer significant improvements over non-measurement-based techniques, to operate with less information and to offer new, adaptive, services not possible under model-based management regimes.

1.2 Context

In this section the network architectures and technologies relevant to this dissertation are discussed. An evolution in network technologies and architecture has caused one resource management approach to be favored in place of another. This evolution is clear in the change of management for multi-service networks.

Multi-service networks may be considered as ones that support a variety of network traffic types. The network traffic types sharing a multi-service network may have similar or disparate demands upon the network resources of link-capacity or multiplexor buffer space. The modern Internet can be considered a multi-service network — although failing to provide differentiation in the service offered, the Internet carries a great variety of traffic types such as WWW traffic, audio-broadcasts, email and IP telephony traffic.

The provisioning of QoS guarantees (such as a fixed link-capacity or a particular packet-delay constraint) had been considered a problem solved in the POTS. In the telephone service, explicit AC is used to ensure that a new telephone call is connected if and only if all the resources it requires are available. As such the telephone network is a good example of both explicit AC and a connection-oriented network.

Success with the connection-oriented telephone service and an acknowledgement of the growing importance of a generic multi-service network has led to the consideration of Asynchronous Transfer Mode (ATM) networks as a multi-service solution (e.g. [Leslie93], [McAuley90]). A variety of reasons, including heavy-weight signalling protocols, inflexible standards bodies and ill-conceived decisions has meant that rather than being used end-to-end, poor market penetration sees a main role of ATM networks in the backbone networks of some providers. The failure of ATM networks, in the face of datagram oriented networks such as the Internet, has meant that the potential of explicit AC, and thus AC algorithms, has not been realised. While ATM-based multi-service networks will not be adopted, this does not imply that AC is not a suitable approach for a given application. As is illustrated below AC may have a place in the Internet and, given the large number of MBAC algorithms, a comparison is justified in order to determine the similarities and differences between these algorithms.

In contrast to ATM, the Internet protocols are based upon the IP [Postel81a] protocol, a datagram service for the datagram-oriented Internet network. While connection-oriented services are available in the Internet, commonly supplied using the TCP protocol [Postel81b] in combination with IP (e.g. TCP/IP), this service does not have explicit admission control. The advantages of offering QoS to data flows, even in the connectionless Internet, led to the Integrated Services (INTSERV) proposals, commonly exampled by its implementation by its use in combination with the RSVP [Braden97] signalling protocol. However, principally for reasons of scaling, INTSERV has not seen popular implementation and an alternative more in keeping with the connectionless

Internet was proposed.

The differentiated services (DIFFSERV) proposal [Nichols99] offers an approach to QoS provision based upon a small number of traffic classes sharing a common link. The DIFFSERV approach has been designed for a datagram-oriented network, an approach that assumes the class to which each packet belongs is carried with the packet. This means that a switch supporting differentiated service support under the DIFFSERV model need only allocate its resources of link-capacity and buffer-space among a small number of clearly defined classes. However, the disadvantage of the DIFFSERV model is the difficulty of providing QoS. For a DIFFSERV system that provides two distinct traffic classes, implementation is easy: a simple priority ordering will ensure that the first class will receive the resource it requires while the second class receives the remaining (left-over) resource. However, for allocation of resource for three or more classes allocation becomes more complex. A common approach has been the static division of network resource between competing traffic classes, however such an approach results in the ineffective use of those resources.

Precise resource allocation would require precise *a priori* characterisation of the demands of a class of traffic. A measurement-based approach need not have *a priori* characterisation of the traffic, instead a measurement-based approach continuously characterises traffic. Additionally, continuous characterisation allows such an approach to respond dynamically to changes in resource demand. It is in recognising the potential offered by a measurement-based approach to differentiated service networks, such as DIFFSERV, that the proposal of Chapter 6 is made.

While a network-wide INTSERV approach is no longer considered realistic, hybrid approaches that use attractive aspects of both INTSERV and DIFFSERV have been entertained (e.g. [Bernet98]). In such an environment the RSVP-based AC procedure will lend itself to improvements using MBAC algorithms.

1.3 Contribution

This dissertation investigates two examples of measurement-based network management thereby illustrating the application of Measurement-Based Estimators to network resource control. The contribution made to the application of such estimators in network resource control is summarised as follows:

- In order to provide realistic evaluation of measurement-based management, a purpose-built test environment is constructed.
- A comparison of MBAC algorithms is conducted by decomposing each using two different taxonomies. Firstly, that of estimator and admission policy; deconstructing each algorithm into these two components allows identification of common and unique elements. A second taxonomy recognises MBAC algorithms that take into account the random-variable nature of measurements.
- A unique, implementation-based comparison of MBAC algorithms is conducted using the evaluation environment. This evaluation is intended to identify an ideal MBAC algorithm or, failing this, to identify desirable qualities in each MBAC algorithm.
- An estimator is incorporated into a switch, providing differentiated service support for network services sharing a common data path. The use of an estimator also means the differentiated service support is adaptive to changing flow requirements. A report is made on an implementation of the switch supporting differentiated services.

The test-environment incorporates implementation of an algorithm controlling real traffic going through an actual switch. Any complexity in the implementation of this approach is compensated for by the removal of the error that can occur with a simulation. Through the use of a flexible traffic source able to generate both deterministic sources and those of real traffic, any MBAC algorithm is subject to the characteristics of real traffic. Using an actual implementation further limits any MBAC algorithm to actual measurement data, thereby eliminating a common approximation error that occurs in simulation. In a simulation, measurements of such properties as utilisation are not bound by limits in transfer capacity, or transfer rate between estimator and measurer. An implementation-based study removes this unbounded access to information and thus limits it as a source of error.

Providing a realistic comparison, an implementation-based assessment places each MBAC algorithm under limitations of memory, computation, and access to measurements. The exact memory, measurement and computation requirements also provide an additional point of comparison between the MBAC algorithms. Access to limited computation is expected to cause MBAC algorithm behaviour to vary from what is intended. In cases where algorithms need substantially more computation time than is available, certain MBAC algorithms are expected to fail outright.

Aside from AC, another approach to network resource management is to perform class-based differentiation. The approach reported in this dissertation presumes the classification of traffic into classes, with each class obtaining different resourcing. The previous section notes that the DIFFSERV network architecture takes this approach. Using an estimator to compute resource requirements for buffer capacity and buffer service rate, the variable demands of such a differentiated service architecture may be implemented. An appropriate estimator can compute the instantaneous resource demands and also adapt to changes in those demands. A network node offering such estimator-controlled differentiated service is thus able to adapt to unknown and changing requirements. Although the approaches presented here are intended to work for a variety of traffic classes, neither of them is a complete answer to the differentiated service problem.

1.4 Outline

The rest of this dissertation is organised as follows.

Chapter 2 provides fundamental background material to the Measurement-Based Management approach. This material includes a review of the evolution of network traffic and descriptions of the network traffic used as part of this study. The material further discusses a selection of approaches used in the management and control of networks and the use of measurements as part of a control process.

The test environment is introduced in Chapter 3. The design and construction of the test environment, a description of the environment's operation, and an evaluation of the environments operation, error margins, and performance limitations are reported in this chapter.

Chapter 4 presents a number of different MBAC algorithms, noting the fundamental premise upon which each is based, and comparing their algorithm structure. This chapter further decomposes each algorithm into policy and estimator allowing further comparison and contrast.

Results of an experiment based comparison of MBAC algorithms are presented in Chapter 5.

Following results of a comparison of MBAC algorithm performance, the interaction between an MBAC algorithm and timescales is studied along with the interaction between MBAC algorithm and implementation overheads.

Chapter 6 explains the need for a network element[2] that is able to support differentiated services. To this end, this chapter describes a switch-based implementation that uses an Measurement-Based Estimator to adapt to current flow demands while offering such a differentiated service. Experimental results are presented further supporting this approach.

Chapter 7 provides a summary of the results of this dissertation and offers directions for future work. Concluding remarks note that this dissertation provides workable implementations of measurement-based management schemes. Additionally, the concluding remarks restate how this dissertation has illustrated the application of measurement-based management schemes to a task not possible with traditional *a priori* characterisation.

[2] In this context the network element is a switch, a network switch being used for the implementation of Chapter 6. However, the network element need not specifically be a switch: manipulating data flows based upon *data-link* layer information, but may be a router: manipulating data flows at the *network* layer.

Chapter 2

Background

This chapter provides fundamental background information to the study of measurement-based management techniques. Section 2.1 starts with an overview of the evolution of network traffic including an introduction to core definitions of traffic properties. Section 2.2 presents the traffic sources used in this dissertation; justification of the selection of each traffic type is given alongside its description. Section 2.3 discusses a selection of approaches used in the management and control of networks. The subjects of Admission Control (AC), link-bandwidth scheduling and active buffer control are each core topics to this dissertation. Each of these topics receives study in greater detail as part of Section 2.3. Section 2.4 discusses the use of measurements as part of a control process — this topic is central to the comparison of MBAC algorithms in Chapters 4 and 5, and to the measurement-based mechanism proposed in Chapter 6.

2.1 Network Traffic

This section explores the evolution of network traffic and network traffic characterisation. The management and control of networks depends upon being able to compute a service demand. To use resources of link capacity and switch buffering effectively the demands of capacity and buffering need to be known quantities around which an appropriate network may be engineered. As network traffic has developed, its characteristics, including the demand it would place upon the network, have changed. The development of network traffic and the methods used to characterise it are charted through this section. Section 2.1.1 commences with traffic characterised using a link/source model. Following this, Section 2.1.2 presents the network level model: an improved approach for capturing network characteristics such as elastic traffic flows. Finally, Section 2.1.3 introduces several approaches that attempt more comprehensive characterisation of network traffic, through such approaches as behavioural models and fixed-point approximation.

2.1.1 Source Modelling

The source model evolved as part of the link/source model, an approach to resource management. In link/source modelling, a network provider uses a model of the source requirements, e.g. the *effective bandwidth* of the source, and a model of the transmission link, e.g. link-capacity and, if applicable, link buffering, to dimension a network. An example of application of link/source modelling is the POTS. In the POTS, a point-to-point telephone circuit requires 64 kbps of capacity for a constant rate source of traffic.[1] If a telecommunications provider has capacity for a 64 kbps channel between two endpoints then the phone circuit can be connected. It does not matter if the phone circuit runs on a pair of wires with only the capacity of one 64 kbps channel or a fibre optics pathway with the capacity of many thousands of 64 kbps channels. If the capacity for one channel is available from end to end the new circuit can be accepted. The source model for such telephone traffic is simple: a 64 kbps constant bit rate (CBR) source, this characterisation sees common use in the current telephone network (e.g. Chapter 8, [Hallsall96]).

A voice telephone conversation does not need to be represented as a 64 kbps CBR source. Speech can be considered as a variable data-rate source made up of talk-spurts and periods of silence — speech represented in this way is an ON-OFF source. Previous work (e.g. [Jaffe64, Brady68, Minoli79]) has shown that the ON-OFF source model of speech may be trivially represented as a 2-state ON-OFF Markov model.

Figure 2.1 illustrates a 2-state ON-OFF Markov model for voice. From [Brady69] and [Habib92] the mean periods for the talk and silence periods were derived as 650 ms and 352 ms respectively. Clearly, if a voice conversation is compressed to remove the periods of silence, less data needs to be transmitted between the endpoints.

[1] A 64 kbps channel of data carries one digitised telephone conversation.

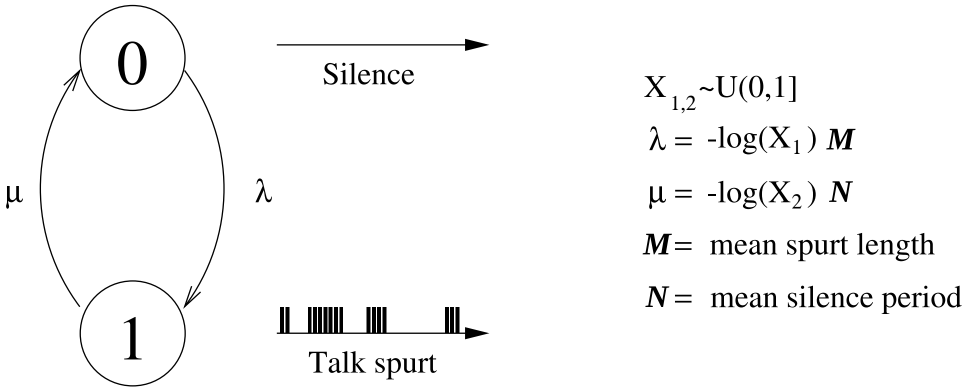

0
1
μ
λ
Silence
Talk spurt
$X_{1,2}$~U(0,1]
$\lambda =$ -log(X_1) ***M***
$\mu =$ -log(X_2) ***N***
M = mean spurt length
N = mean silence period

Figure 2.1: Markov 2-state ON-OFF generator for voice.

Compressed voice requires less network resource than uncompressed voice. A network provider can use this to maximise the utilisation of shared network resources, such as the fibre pathway carrying many thousands of voice connections. A fibre channel may carry more connections using compressed rather than uncompressed data. In the case of voice data, the improvement may be as much as a third more connections. Figure 2.2 illustrates the gain that may be made if sources can be statistically multiplexed. In this figure the peak requirements of the multiplexed source (A mux B) is significantly lower than the addition of the peak requirements of the two sources (A + B).

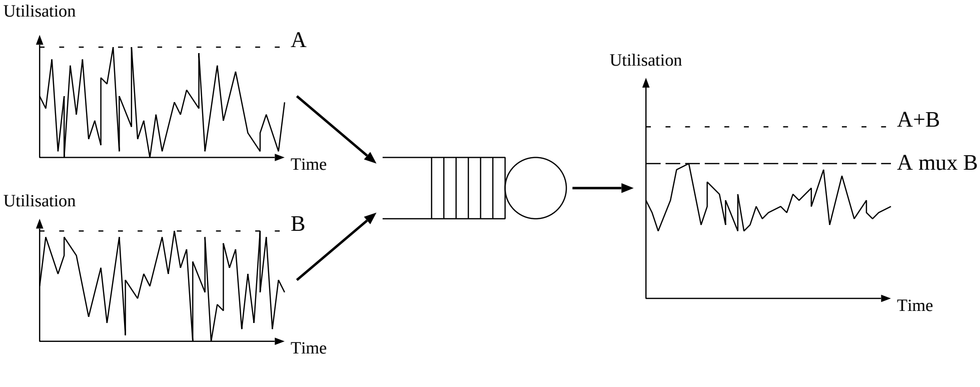


Figure 2.2: Gain from statistical multiplexing.

However, using compressed voice, the network provider no longer has a CBR source but a VBR source. The VBR voice model has three parameters rather than the one of CBR. For voice, the model is described using the mean talk period, the mean silence period, and the rate at which data is transmitted when the model is *talking* or ON. More commonly, these values are represented as the mean burst size: the mean talk period, the peak bit rate: the rate at which data is transmitted when the model is ON, and the sustained bit rate: which may be computed from the mean burst size and the mean silence period. As an example, the 64 kbps voice traffic, if compressed, still has a peak rate of 64 kbps but a sustained rate of approximately 23 kbps and a mean burst length of about 2.9 Kbytes.

The difficulty for the network provider becomes computing *what is the resource required by this traffic?* If a small quantity of lost data is acceptable, and the quantity of loss is computable, a network provider may conclude that the benefits of getting extra flows into a channel out weigh the impact of packet loss.

Traffic that may be represented as a Markov model such as the 2-state ON-OFF used here is attractive as such models are mathematically tractable and a considerable body of knowledge has evolved around their use. Using the VBR descriptions of traffic (peak rate, sustained rate, and mean burst size) proposals (e.g. [Guérin91, Buffet92]) enable a link to be dimensioned for a given number of VBR traffic sources. Such approaches will hold as long as the traffic sources may be represented using Markov models.

Long before silence suppression was an available compression technique, Markovian models were used to describe other aspects of the telephone network. The duration of a telephone call and the time between consecutive calls were presented as 2-state ON-OFF Markov models by [Erlang17] and [Molina27]. Using these models, network providers have been able to make the best use of telephone resources while providing a computable level of service to customers. The QoS a customer received was not measured as the chance of data lost in multiplexers but as the probability a new connection would be blocked — unable to be completed and the caller receiving a busy tone. The seminal work of [Erlang17] illustrated how the probability of blocking may be computed for a given combination of resource and demand. Once again, the link/model approach to the dimensioning of networks relies upon tractable models of demand, in this case the Markovian models to describe the properties of telephone calls.

However, telephone use has evolved along with the evolution in technology connected to the telephone networks. [Bolotin97] reported that the models proposed and used successfully for decades are no longer valid when applied to telephone networks when a percentage of the telephone calls carry computer data or facsimile transmissions, rather than voice calls.

Results reported in [Bolotin94] and [Bolotin97] revealed that the distribution of call duration has changed from an exponential decay to a decay with a heavy tail. This implies that, rather than following an exponential decay, flow-lifetimes are better represented by log-normal or Pareto distributions where, for a given mean, the variance tends towards infinity.

Results reported in [Bolotin97] concluded that one cause for change in behaviour is the rise in calls made to Internet Service Providers (ISPs). Examinations of ISP dial-up lines has supported this notion. In [Vicari00] results from trace data of a set of University dial-in lines are examined. Their work supports the observations of [Bolotin97] documenting how ISP call characteristics differ significantly from the voice-only telephone network. Interestingly, in a packet-based network such as the Internet, models of the traffic once assumed to be Markovian are now also best represented by heavy tail distributions [Paxson95, Crovella97, Feldmann98b].

Examples of VBR traffic that may be characterised by heavy tail distributions include video ([Garrett94]), Local Area Network (LAN) traffic ([Leland94]), and WWW transactions carried over TCP/IP (e.g. [Crovella97]). The reason that the heavy tail distribution more accurately characterises Internet traffic (or the duration of telephone calls) is because each of these sources, when multiplexed together, exhibit the properties of self-similarity and tend towards LRD.

Figure 2.3 illustrates how Poisson[2] and LRD traffic differs when examined over different timescales. Traffic with a Poisson distribution fluctuates in only one timescale and, hence, has a quantifiable variance but LRD traffic does not have one single timescale over which its variance can be quantified.

The expressions self-similar and LRD are used more-or-less interchangeably in much of the current networking literature although there is a strict difference in definitions. Any traffic source

[2] An example Poisson distribution is the exponential function central to the 2-state ON-OFF Markovian model.

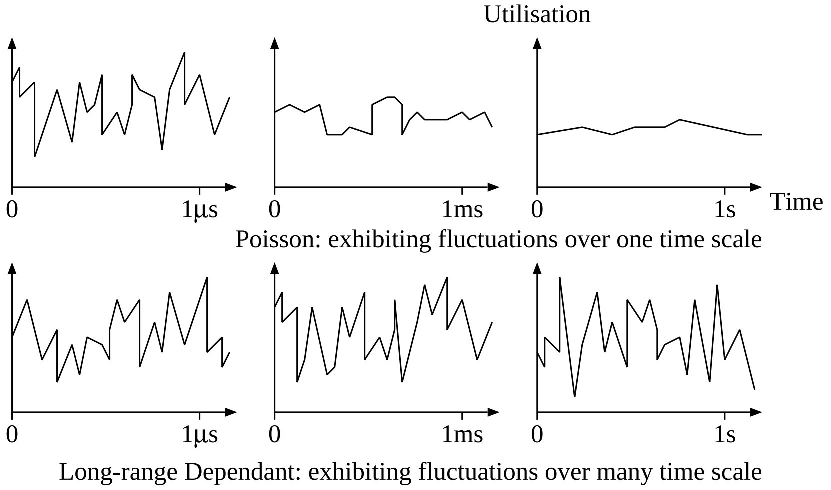


Figure 2.3: Poisson and Long-range Dependent Traffic over time

exhibits LRD if it exhibits significant correlations across arbitrarily large timescales. A strict definition of an LRD source is one that has a non-summable autocorrelation function. The simplest sources exhibiting LRD are self-similar processes which are specifically characterised by hyperbolically decaying autocorrelation functions. This will be represented by the 2nd-order moment; variance, tending towards infinity.

While self-similar traffic models possess LRD properties, LRD specifies a category of sources. Processes that are self-similar and those that are asymptotically self-similar are attractive to use as models of traffic with LRD characteristics because the long-range dependence can be characterised using a single parameter H, the Hurst variable. As a result, self-similar traffic models are the most common way of creating sources with LRD behaviour which has led to the two expressions being used interchangeably.

Despite considerable experimental data demonstrating LRD, the property was difficult to explain due to a lack of physical rationale. Systems that generate LRD observations do not have an intuitive mechanism for transmitting state-information over very long timescales. This phenomena was well documented for natural systems: e.g. the level of the Nile river at each of its dams, a process where the *memory* maintaining the state information about past behaviour is hard to imagine.

The Nile is not mentioned arbitrarily as a natural system exhibiting LRD. Following a study of the Nile dams, Hurst [Hurst51] documented techniques to establish whether a system was self-similar, thereby computing its Hurst parameter H. The technique he documented was used successfully to show that Ethernet traffic also possess self-similar properties in [Leland94]. Up until this work, the networking community had commonly used Poisson distributions and Markovian models.

Following the publication of this finding, [Paxson95] reported that WAN traffic may, in some cases, be modelled by Markovian models. But a number of the traces analysed did exhibit self-similarity. These conclusions were further confirmed in [Feldmann98b] who reported an analyses of WAN traces. Following an examination of WWW transfers logged from specially configured web-clients, it was reported in [Crovella97] that WWW traffic also exhibited self-similar behaviour. This final paper also attempted to explain the behaviour, reasoning that the observed characteristics were the result of the underlying distribution of transfer sizes combined with caching effects, the *user process* (a users reading and thinking time), and the multiplexing of many such transfers onto a LAN.

In addition to characterising traffic as heavy-tailed, LRD, or self-similar, other approaches have been taken to best characterise the traffic in order to create suitable models for the link/source approach. Multi-fractal analysis is a technique first introduced by Mandelbrot in the context of turbulence (e.g. [Mandelbrot74]) but has been adopted only recently into traffic analysis. A multi-fractal analysis was reported in [Riedi97] of traces of Internet traffic, and this work was extended in [Riedi98] to create a framework that supported multi-scale modelling of network traffic.

The analysis reported in [Feldmann98a] noted that WAN traffic can differ significantly from LAN traffic. The authors postulate that while LAN traffic is mono-fractal: exhibiting self-similarity, scaling behaviour, WAN traffic (ATM carriage of TCP/IP Internet traffic) exhibits a more complex scaling behaviour. They further postulate that this behaviour is best represented as a multi-fractal process. [Feldmann98a] then showed how cascades of (fractal) processes can be used to recreate the behaviour of WAN and LAN traffic.

Self-similarity is not a property unique to Internet traffic, a number of previous studies, notably

[Garrett94] reported the presence of self-similarity in VBR video traffic. Interestingly, [Ryu96] reported in studies of VBR video traffic carried via ATM, that the loss-rates are not adversely affected by the self-similarity properties and that Markovian models are sufficient for performance analysis. This conclusion was drawn by showing that short-term traffic correlations had a more significant effect on buffer behaviour for VBR video traffic than the effects of self-similarity.

Corroborating the findings of [Ryu96], [Grossglauser96] proposes the existence of an event-horizon beyond which the self-similar characteristics of traffic will not impact upon that time-scale. An example is the timescale determined by the buffer-size of the system — properties of traffic self-similar at timescales beyond this will have little impact at this timescale.

Clearly, the models of network traffic must develop as there is a maturing of analysis techniques and evolution of the network traffic itself. For traffic with more-complicated, less-tractable properties, the source model does not offer a network provider a sure means of establishing an answer to the earlier question: *what is the resource required by this traffic?* Common source models, whether Markovian or the heavy tailed Pareto, require a value for the sustained rate of that traffic flow. If a network provider is to provision resource, a new network user must be able to supply this value. While a sustained (mean) value may be computed from recorded traffic, predicting such a value is not plausible for many traffic sources. The VBR video data cannot be characterised until the traffic is generated and measured. For example, *a priori* characterisation is not possible if the data is generated using a camera viewing events in real-time. The computation of a mean burst size is similarly fraught with difficulty.

Further problems arise in a multi-hop network where the description of traffic on entry to the network may be accurate but the actions of multiplexing and demultiplexing as the traffic passes through several switches (or routers) may not be desired nor understood. If the source gets smoother as it passes through each switch then *effective bandwidth* based upon *a priori* declarations will be overly pessimistic. Additionally, [Crosby95] showed that burst-compression is a common phenomenon in multi-hop networks, making traffic increasingly bursty and therefore more demanding on network resources. Such experience further draws into doubt the usefulness of traditional traffic models for the accurate description of any more information than a flow's peak data rate.

Given the difficulty in applying a source model to satisfactorily characterise evolving traffic or to accurately characterise the elastic traffic carried through the Internet, an approach called the network model has arisen.

2.1.2 Network Modelling

Instead of a link/source model, what was needed was a model that could incorporate the adaptations a source may make interacting with a network. The notion that the link/source model is not suitable for characterising elastic traffic, such as a TCP/IP network, was supported in [Veres00]. Further, [Arvidsson99] illustrated the significant difference between the link/source modelling approach and simulations of the behaviour of adaptive traffic. By explicitly incorporating the effects of the closed-loop into the model, traffic generated in this manner will more preciously replicate the adaptive nature of traffic sources, such as those using TCP/IP that rely upon feedback from the network in the modification of their behaviour.

The link/source model approach may suit traffic sources that do not interact with or adapt to the current network's behaviour. However this open-loop approach disregards one of the major properties of some current traffic, e.g. TCP traffic in the Internet. The majority of Internet traffic is fundamentally adaptive in nature under the control of the TCP protocol (> 80% in 1985 [NLANR95], others use a similar figure [Hori98]). With an adaptive protocol such as TCP, the source behaviour cannot be disconnected from the network configuration (e.g. buffer management, network routing, or packet-scheduling) because the protocol interacts with the network. If the network configuration changes, so do the observations of network behaviour. As a result, any link/source model constructed may be valid for only the one observed configuration.

Further complicating this, in [Veres00] it was proposed that the TCP mechanism for congestion control is fundamentally chaotic in nature. This premise means that TCP-based networks will exhibit extreme sensitivity to initial conditions, unpredictable behaviour and odd periodicity.

Network-level models of the TCP process have been offered by several authors. Such a model must account for the fundamental operation of TCP, and this increases the complexity of the models enormously. A model was presented in [Mathis97] after analysing the macroscopic behaviour of TCP under light and moderate loss. The model constructed handles selective acknowledgements, a relatively recent addition to the commonly implemented TCP. Using such parameters as RTT, the MSS and the loss across the network, the paper has presented a fixed-point formula approximating the *effective bandwidth*.

[Padhye98] also reports on a behavioural model relating effective throughput and loss, taking particular note of the impact the timeout mechanism has on performance. The authors of [Padhye98] validated the network model against a wide range of platforms and TCP implementations. Building upon the work in [Padhye98], a network-model that concentrates upon short TCP connections was presented in [Cardwell00]. Starting with the assumption that TCP flows (such as those carrying the WWW) are short-lived this paper developed models to approximate the lifetime of short flows.

The network models of [Mathis97, Padhye98, Cardwell00] may be used in the development of TCP-friendly protocols to co-exist in the Internet (e.g. [Hori98, Padhye99, Floyd99]). However, each network model makes fundamental assumptions that the TCP congestion control is behaving in a periodic/predictable fashion. [Arvidsson99] proposed that this is in contradiction with the measurements and simulations of TCP traffic, and sought a resolution in a model that attempts to be network-wide yet achieves the same behaviour as the source models. A fundamental idea incorporated into the work of [Arvidsson99] is that cooperating TCP congestion control processes will together form a deterministic, albeit chaotic, system.

The TCP protocol is not the sole adaptive protocol in the Internet. A number of adaptive services are built upon the UDP protocol [Postel80] which offers a datagram service and reliable and adaptive transport mechanisms can be built upon it.

One of the few works on the use of wide-area UDP-based traffic is [Hori98], co-incidently, also one of the few works looking at traffic resulting from an MP3 audio server/client system (e.g. [Pan93]). MP3 is not limited to using UDP, and while network folklore indicates a common use of TCP by MP3 clients (to ensure some level of transport connectivity across the Internet), this may be disputed by another major investigation of audio traffic in the Internet [Mena00]. The work reported in [Hori98] is based upon the assumption that UDP is the principle carrier of real-time traffic, an assumption supported by [Mena00].

2.1.3 Fixed-Point and Behavioural Network Modelling

The previous sections on link/source and network models make clear that not only is traffic modelling an incomplete task but that it also involves the capture of complex, ill-understood behaviour. An alternative to computing the resource requirements of a particular traffic flow is to compute the behaviour of the total network by considering it to be in steady-state. If the network is assumed to be in steady-state and the network properties are assumed to be independent between links of the network, a fixed-point approach is possible. Such an approach allows the construction of an equation that can supply approximations of the network properties. This technique has been used by [Ross95] in circuit switched networks (e.g. POTS) to determine properties such as the blocking probabilities of new connection-attempts and the utilisation of links.

The example of fixed-point analysis is put forward in [Gibbens00], examining a DIFFSERV network offering QoS. The fixed-point analysis may be considered based upon a meta-network model, built upon underlying network and connection models. The fixed-point analysis reported in [Gibbens00] used TCP network models drawn from work reported in [Mathis97] and [Padhye98].

The TCP models of [Mathis97] and [Padhye98] are combined with the description of a buffer implementing strict priority-based class queueing and a model of the *thinning effect*, whereby traffic flow from one endpoint to another is reduced, due to loss, as it passes through successive buffers. These network models are then combined with a connection model constructed using measured values for the means of flows per class and per route for each of the network endpoints, and a Poisson distribution for flow arrivals. The approach of [Gibbens00] was able to illustrate that fixed-point modelling can provide a reliable prediction of the operating point of TCP networks by incorporating RTTs, end to end loss, and the number of user sessions. However, the fixed-point model is valid only if there is no significant traffic correlation between sources across the network. The example used in [Gibbens00] is a well-connected network and the authors noted that this assumption is valid in such a network where traffic aggregation may be treated as Markovian.

Interestingly, it is precisely this Markovian assumption that has been challenged (e.g. [Leland94], [Paxson95].) Despite this, [Gibbens00] illustrated that the method produces acceptable results depending not specifically upon the model but upon the mean characteristics independent of the distribution itself. The authors noted the infancy of this approach for TCP-based networks and state the importance of a thorough validation of their work.

An alternative to fixed-point analysis is the approach suggested in [Erramilli00], the phenomenological model. Phenomenological models capturing the interactions between network state and traffic flows can evolve from the work on link/source models. However, such ideas may be underestimating the role that the network user plays in a network's behaviour — many users would not keep trying to retrieve a WWW document if the speed of transfer was unusually slow. This sort of behaviour, as well the default actions of novice users (using browser default pages), the manner in which experienced users may retrieve many documents simultaneously if network loads are light, and the causes of diurnal patterns as well as the reaction of users to these patterns each constitute aspects that current traffic models do not attend. However, [Vicari00] has done some work on the change in behaviour of TCP as network speed has improved.

However, in common analysis techniques such as trace-driven analysis it may be difficult or impossible to separate the role of protocol and the role of user. Aside from recognising diurnal characteristics in traffic studies, each of the works discussed in the previous sections does not account for the impact that user behaviour, driven by the feedback of network behaviour, has upon the network behaviour. Clearly, the field of such user-behavioural models of traffic is noted here as one with significant potential contribution as part of the entire effort to model adaptive traffic such as TCP in particular and network traffic in general.

2.2 Traffic Used in this Study

Following the previous section illustrating the varied nature of network traffic, this section presents the various traffic types used in this dissertation. An objective in the selection of traffic has been to provide a representative sample of traffic that may be anticipated in the environments tested in Chapters 5 and 6.

Several types of sources are discussed in this section. Firstly, a set of sources based upon deterministic models whose characteristics are both easily simulated and well understood, a Markovian source model and a Pareto source model. Within this group of model-based sources are those representing the carriage of voice data, represented as both a CBR source and a silence-compressed source based upon a Markov model. The second source type is the result of the carriage of video stream data and represents a real-world network application. Against both of these controlled-load sources, neither of which is adaptive to network conditions, are several sources based upon Internet traffic behaviour.

Three Internet traffic sources are used in this study. The first is based upon traffic of a LAN environment, while the second reflects characteristics of traffic seen in the WAN environment. The final Internet source represents WWW transactions and is unique in that it is an elastic (adaptive) traffic source.

2.2.1 TP10S1 — 2-state ON-OFF Markov model

A deterministic source model is used both to make comparisons with theoretical results and to test predicated values for the AC algorithms. A source based upon a Markov model makes possible direct comparison between the experimental results and those gained from an AC reliant on purely deterministic traffic. In this way the theoretical results can be verified in a practical implementation.

TP10S1, is a 2-state ON-OFF Markov source, which emits at the peak rate when on. Using a peak rate of 10 Mbps, and a sustained rate of 1 Mbps, this source has a mean burst size of 25 packets (1325 octets). The Markov model generator is represented in Figure 2.4.

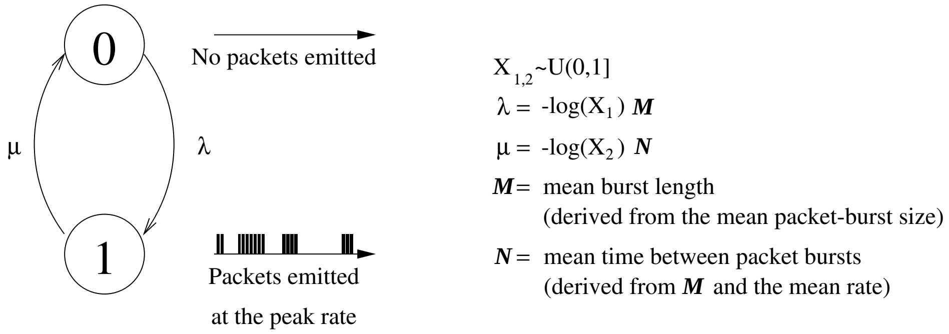

0
1
μ
λ
No packets emitted
Packets emitted
at the peak rate
$X_{1,2}$~U(0,1]
λ = -log(X_1) $\boldsymbol{M}$
μ = -log(X_2) $\boldsymbol{N}$
$\boldsymbol{M}$ = mean burst length
(derived from the mean packet-burst size)
$\boldsymbol{N}$ = mean time between packet bursts
(derived from $\boldsymbol{M}$ and the mean rate)

Figure 2.4: Markov 2-state ON-OFF generator.

The behaviour of traffic from the source is based around the uniformly distributed random variable, X, and the traffic properties of the peak rate, sustained rate and mean burst size. The variable X in turn, is based on a uniform pseudo-random number generator. As a result, several traffic generators can have identical traffic properties of peak rate, sustained rate and mean burst size yet, through the use of different seeds to the pseudo-random number generator, the generators will create a stream of packets that will differ over time in the packet level characteristics of burst length and inter-burst spacing.

2.2.2 PP10S1 — 2-state ON-OFF Pareto model

PP10S1 is a 2-state ON-OFF source with the same peak-rate, sustained-rate and mean-burst-length as TP10S1. It differs with the distributions for burst length and inter-burst period taken from a Pareto distribution with a shape of 1.5. Previous studies (e.g. [Willinger95, Crovella97]) have noted that the aggregation of such a source produces traffic that exhibits self-similarity.

2.2.3 VP64S64 — Voice channel uncompressed

Representing a standard voice channel, this source is configured to send one 48 byte packet per burst at a constant rate, creating a CBR stream of data at 64 kbps.

2.2.4 VP64S23 — Voice channel with compression

While less common in current end-to-end applications, voice compression, often of the form of silence-suppression, is often used by inter-continental carriers of voice traffic or in the carriage of voice over the Internet. Compressed voice traffic is represented with a 2-state ON-OFF Markov model. Like TP10S1 and PP10S1, this model transmits at a constant, peak-rate while in the ON state and does not generate traffic in the OFF state. The time in ON and OFF states is independent and exponentially distributed. The peak-rate is 64 kbps, the sustained rate is $\approx$22.48 kbps and the mean-burst length is $\approx$23068 or about 60 packets (1325 octets in length). These values are derived from ON and OFF times of 352 ms and 650 ms from [Brady69, Habib92].

2.2.5 VP25S4 — Video data stream

The traffic representing a video data stream has been created from real video data.

VP25S4 is a controlled-load source used in this study to represent real-world, rather than deterministic, traffic based upon video data. This source represents the carriage of video stream data in packets within a stream with a pre-defined peak-rate, sustained-rate and both maximum and mean burst-sizes. The video source is created from a data file containing a sequence of integers. Each integer denotes the number of bits that results from a frame of the video compressed using MPEG 1 [Le Gall91].

The conversion of video data into a stream involves the conversion of large, periodic sets of data (each representing a video frame) into a variable number of packets. A number of approaches may be taken by a video codec. Several approaches are reviewed here.

One method would be to convert the whole frame into a large block of packets and to then transmit these packets at the peak-rate for the connection. Such a method has the advantage that the peak-rate can be specified. Figure 2.5 (a) gives a profile of the bytes per second that might occur in this encoding system. However, this method is not satisfactory. The main problem is that associated with packet burst sizes.

For a frame consisting of 75,000 bytes,[3] a conversion of this data into packets would create a single burst of 1,563 packets in length. A typical switch may carry buffers sized to cope with such a large burst from one source but not enough to cope with the multiplex of many such sources. In the experiments of Chapter 5 the buffer was 512 packets in length, and while the burst may be at only a percentage of the total link bandwidth, only a few connections of this type of traffic would quickly overflow the buffering capacity of a switch causing high loss in the encoded frames.

A second approach is to space the transmission of the packets of each burst out over the entire duration of each frame. Figure 2.5 (b) gives a profile of the bytes per second that might occur in this encoding system. This technique is commonly used when matching the output of a bursty device to a low bandwidth transmission path. It can be seen that while this would make the traffic more specific it requires significant buffer capacity at transmitter and receiver to space out the transmission and to reconstruct the frame as it is received.

[3] The value of 75,000 bytes is not an arbitrary selection, it is the mean frame size of one of the video sequences used in this study.

The final technique presented here would be to transmit the frame using a fixed number of pieces per frame. This system of dividing a frame into pieces occurs in commercially available codecs (e.g. [NRL96]). The traffic is bundled into pieces, called *slices*, with each slice corresponding to a region or Group-of-Blocks in the MPEG coding standard. Each slice includes an AAL5 [CCITT90a, CCITT90b] wrapper so each slice can be transmitted in a manner that allows error detection. The name, slice, is derived because each is a horizontal region of the original image. A slice technique is used in image transmission because it reduces the latency with which an image can be reconstructed; slices of the image can be reconstructed as each is received. Thus, the compression/decompression process can occur, pipelined with the transmission of an image frame.

In addition to being a common commercial technique, slices allow specification of the peak-rate for transmission of the blocks. Figure 2.5 (c) gives a profile of the bytes per second that might occur in this encoding system. This final method is the technique used in this study for video streams into traffic sources.

The analysis of an MPEG encoding of the Star Wars movie reported in [Garrett93] and [Garrett94], illustrated how the encoded data possessed many of the properties of LRD traffic such as self-similarity. Garrett made the encoding of the video available and, as a result, a wide number of studies have been conducted based upon traffic derived from this data stream. The VP25S4 has been created using the Star Wars traffic. The traffic is used to compute the size of each AAL5 encapsulated video slice and each video slice is transmitted at a peak-rate (approximately 25 Mbps) sufficient to ensure 30 frames per second (fps) transmission rate. A frame rate of 30 fps results in a sustained data transmission rate of approximately 4 Mbps.

2.2.6 RP10S1 — Internet LAN traffic

RP10S1 represents IP traffic recorded from a LAN. This traffic mix consists of a combination of TCP/IP sources carrying flows such as WWW transactions, UDP/IP sources carrying NFS traffic and an assortment of background flows such as time-protocol and name-server protocol. This source of traffic represents the network activity found in the internal, intranet, networks of an organisation. Comparison revealed good correlation between the properties of this source, the traffic recorded on local-area academic networks and the traffic recorded in various of the

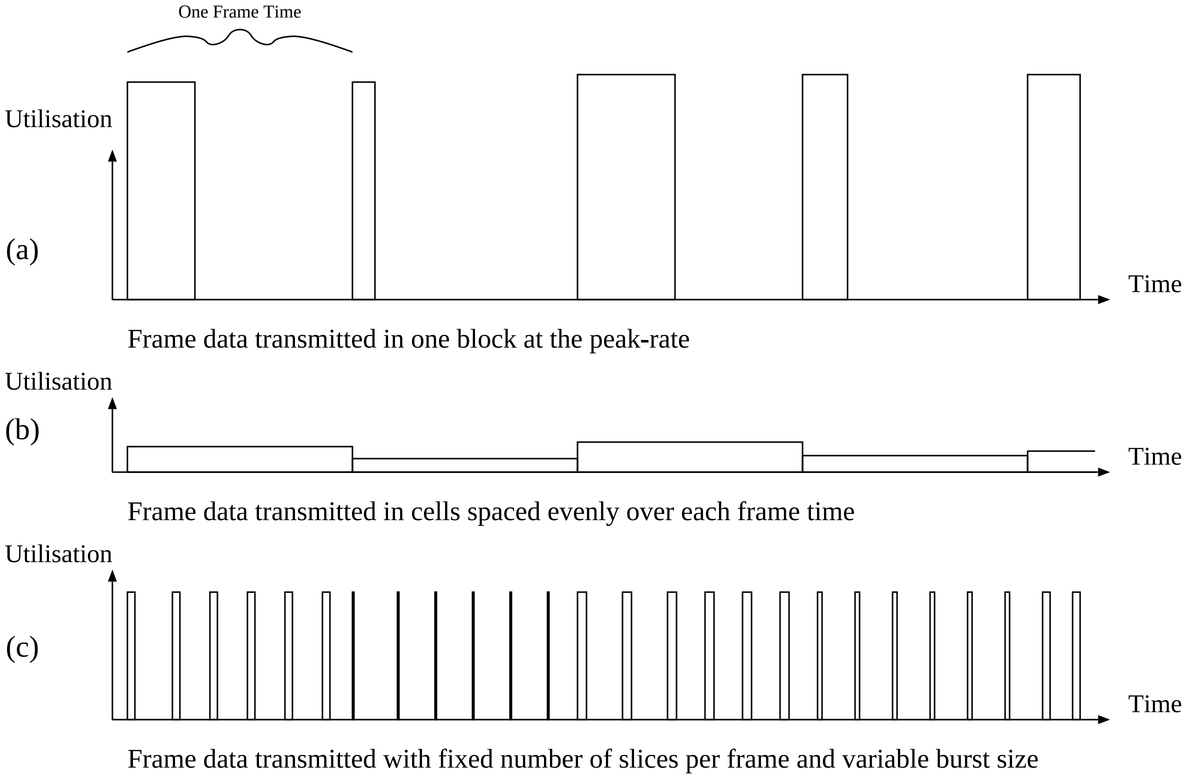

One Frame Time
Utilisation
(a)
Time
Frame data transmitted in one block at the peak-rate
Utilisation
(b)
Time
Frame data transmitted in cells spaced evenly over each frame time
Utilisation
(c)
Time
Frame data transmitted with fixed number of slices per frame and variable burst size

Figure 2.5: Methods for encoding frames into packets.

traces held at the Internet traffic repository [ACM99], such as the original Ethernet traces used in Willinger [Willinger95].

The source, RP10S1, is captured as a set of packet traces. While the adaptive nature of elastic IP traffic is not represented in a static trace of traffic, the statistical nature of the original traffic is captured. The statistical nature of the traffic such as the interactions between one flow and another and the subsequent effects on flows inside such an aggregate are each captured in a static-trace. Such traces have been used previously to good effect when the replication the statistical characteristics, such as scaling over multiple time-scales, is desirable [Vicari97, Crovella97].

The source has a peak rate and is intended to emulate switched 10 Mbps connections flowing over a campus-wide backbone network. The source exhibits a sustained rate of approximately 1 Mbps over the period represented in the trace.

2.2.7 EP6S480k — Internet WAN traffic

Representing a stream of IP traffic as would be found connecting sites across a wide-area network, the Internet traffic trace EP6S480k consists largely of a trace of WWW transactions along with supporting name-service operations. This source has also been validated against traces taken as part of the instrumentation and data collection from a UK ISP. It has also shown good comparison against the appropriate traces of traffic recorded in the [ACM99], in particular the traces of a university point-of-presence (POP) recorded at that site. A comparison reveals that the trace captures the first and second order statistical properties in addition to scaling effects over multiple time-scales. The traffic consists of an aggregate of traces of host traffic presenting a 6 Mbps peak rate. The source exhibits a sustained rate of approximately 480 kbps over the complete trace.

2.2.8 WP10S1 — Elastic WWW traffic

WP10S1 is an elastic source created using the generator described in Section 3.3.3.2. This source emulates the transaction profile of an aggregate of WWW traffic. This source may be considered as a multi-stage Markov chain. The parameters to the Markov chain are: Inter-page time: the time between the retrieval of consecutive pages in the one session,
Transactions per-page: number of objects on each web page,

Inter-transaction time: time between requests for objects part of a single page, and
Transaction size: the size of each object/transaction.

The configuration parameters, given in Table 2.1 are based upon the work presented in [Barford98] which also saw use in [Feldmann99].

Attribute	Distribution	Characteristics
Inter-page Time	Pareto	mean 10, shape 2
Transactions per page	Pareto	mean 3, shape 1.2
Inter-Transaction Time	Pareto	mean 0.5, shape 1.5
Transaction Size	Pareto	mean 12 kbytes, shape 1.2

Table 2.1: Probability distributions for transaction attributes

Such a source has the advantage of being representative of more recent traffic flows and of being adaptive and dynamic in nature — an aspect commonly eliminated from studies by the use of purely trace-driven traffic systems. Rather than the open-loop sources, of Sections 2.2.1 through 2.2.7 where there is no pathway for network-derived control of the sources, this source is a closed-loop with an explicit path for control feedback from the network to the source. In this way the source is fully adaptive to changing network conditions, in the same manner any TCP based traffic flow would be. This source is policed through a rate-limiter with a 10 Mbps peak rate and, like RP10S1, is intended to emulate switched 10 Mbps connections flowing over a campus-wide backbone network and onto the Internet. The source exhibits a sustained rate of approximately 1 Mbps for measured periods but this figure will adapt as the effects of multiplexing with other streams impact the precise behavior of this flow.

2.3 Network Control

This dissertation presents a detailed comparison of a class of AC algorithms: MBAC algorithms. Chapter 6 then illustrates a network element offering adaptive differentiated services through an adjustable scheduler and active buffer management. These are two approaches towards network control using three distinct timescales of control: connection or flow, packet burst, and packet scheduling. This section provides background to these three approaches to network control.

Section 2.3.1 explores network control methods based upon their timescale of operation. Particular attention is given to how network control methods overlap each other in their respective operating time frame as well as the interaction between control and traffic that may also exhibit multiple critical timescales.

Particular attention is given to the three control timescales: packet scheduling, traffic burst, and traffic session. The packet scheduling level is explored in Section 2.3.2, notably as it applies to the scheduler used in the implementation of Chapter 6.

An active buffer management scheme is also employed in Chapter 6 and the background of this control technique is given in Section 2.3.3.

Section 2.3.4 gives a background to the field of AC and MBAC algorithms. This section also summarises the comparisons of MBAC algorithms.

Finally, Section 2.3.5 summaries network control activities at timescales longer than that of the call or session.

2.3.1 Timescales

Decomposing a network control problem by timescale was first introduced in [Hui88]. At the smallest timescale, Hui noted that data on a transmission line was sent at the speed of the line. The network Hui described was a packet network and he referred to this as the *packet level* considered to be indivisible. An example of a control event at this timescale is the scheduling of coincident packet arrivals. Schedulers are discussed at greater length in Section 2.3.2.

Increasing the time period, Hui described the *burst level*. The burst level was the timescale that described a burst of packets being transmitted at the peak rate of the source. Using the voice source described earlier in Section 2.1.1 as an example, the data associated with a talk-spurt would be considered a burst. Control exhibited at the burst timescale may overlap the control exhibited by the scheduler, although buffering and buffer management are mechanisms unique to this timescale. Section 2.3.3 presents a selection of current resource control work at the burst timescale.

Expanding further time period, Hui described the *call level*, although a commonly used series of control mechanism exists between that of burst and of call. The RTT, the period of time required for information to be communicated from source to destination and back to source, is a useful timescale to categorise mechanisms such as TCP ([Postel81b, Nagle85, Lottor88, Stevens97, Mathis96]) and explicit congestion notification schemes proposed as alternatives to it [Ramakrishnan90, Floyd94, Ramakrishnan99, Key99, Gibbens99].

Section 2.3.4 summarises the significant quantity of literature that exists describing control techniques at the *call level*, the timescale over which flows of data are connected. A call example would be a connection in the telephone network. Calls also exist in other networks including the datagram oriented Internet. A flow of data using TCP/IP is clearly delineated by a connection set-up and tear-down. If an AC algorithm is intended to preserve network QoS, such as packet loss, it must exert control over the lower timescales of burst and packet. Clearly the *call level* cannot be so clearly delineated from the smaller timescales.

The *call level* is the largest timescale Hui discusses, although in this summary, larger control timescales are noted. The timescales of hours through days to months cover such activities as network design, deployment, operational issues such as maintaining redundancy and failure recovery, and methods to minimise diurnal behaviour such as peak-period pricing. The method of control for each of these larger timescales will overlap the smaller timescales. A network is designed with appropriate resource to offer a certain QoS — the actions of AC, buffer, or scheduler can control the distribution of that network's resource.

[Hui88] presented a decomposition of both traffic and control timescales. The assumption that there is independence or a clear delineation between either the timescales of traffic or of control must be brought into doubt. Section 2.1 noted how an evolution in traffic has given rise to inseparable timescales in the traffic covering many scales from those of multiplexing of packets through to the macro behaviour of flow aggregates. Unfortunately, the ability to decompose traffic into different time-frames is a fundamental requirement of many control mechanisms, e.g. AC algorithms.

An AC algorithm must exert control over QoS issues that are measured on the smallest of timescales (those of packet discard), yet it is a *blunt tool* able only to accept or reject new flow admissions. However, a precise conclusion is less certain, with several papers proposing either solutions to the decomposition problem or proposing that the multiple timescales of traffic does not cause as significant an impact as might have been first thought. [Grossglauser97b] proposed that a boundary between control timescales does exist but that it is dynamic, and further proposed a mechanism for computing the point of separation in the timescales based upon the size of the system.[4] [Ryu96] reported that the indistinct timescales of video traffic were of little consequence. They note that the loss-rates for an adequately configured switch were not adversely affected by the self-similarity properties and that Markovian models were sufficient for performance analysis. Finally, [Gibbens00], who used a fixed-point analysis of a network reported that the precise model of traffic was not as relevant as the 1st order statistical properties (e.g. the mean number of flows and the mean flow lifetime).

2.3.2 Packet Scheduling Level

Packet scheduling is the sharing of a transmission link among a number of users. The ideal theoretical scheduler, Generalised Processor-Sharing (GPS), is designed to enable a link to be shared among any number of users. By assuming that the input is infinitely divisible (using a fluid-flow model [Parekh93, Parekh94]) and that all sessions can be served simultaneously, a GPS server is defined to be work-conserving, where each session is guaranteed to receive a minimum service rate [Parekh93].

Various solutions note that the class of service disciplines, the Packet-Fair Queueing (PFQ) algorithms that evolved from the GPS policy, have two important and desirable properties. Regardless of the behaviour of other sessions it can a) guarantee instantaneous fair allocation of bandwidth among all backlogged sessions and b) it can make end-to-end guarantees of delay to a session as long as that session is constrained. Guarantees of fairness are important in the support of best-effort traffic [Shenker94, Wu98, Stoica98, Ng99] and the support of hierarchical link-sharing services [Clark92, Floyd95, Stoica97]; while the controlled-load service of INTSERV [Wroclawski97] or ATM Forum TM4 [ATMF95] require end-to-end guarantees [Parekh93, Parekh94].

Fair queueing, the principle of output bandwidth fair-sharing among multiple queues, was first

[4] [Grossglauser97b] considered that the size of a system was defined by the number of concurrent flows present.

developed in [Demers89, Keshav91]. The definition relied upon an idealised fluid-flow model. This model and its delay-bounds were significantly refined in [Parekh93, Parekh94] as GPS or Weighted Fair Queuing (WFQ). In the work of [Parekh93, Parekh94], Packet-by-packet GPS (PGPS) or PFQ is proposed as an approximation of GPS.

In GPS, each session i, is characterised by the weight ϕ_i, which is a real positive number. Each session receives service in proportion to its weight relative to the sum of the weights of the back logged sessions. Thus, at any given time t, the session i is allocated a fraction of the link bandwidth which is equal to $\frac{\phi_i}{\sum_{j \in B[t]} \phi_j} C$ where C is the bandwidth of the system and $B[t]$ is the set of sessions which have a positive backlog at time t. However, GPS is not realisable because it assumes that packets are infinitely divisible. PFQ attempts to emulate GPS by assigning a virtual finishing time to each packet. The finishing time is the moment when the packet would have departed if served by a GPS server and no packets arrived after this packet.

There are many algorithms proposed to achieve a near-PFQ performance, each with different trade-offs between accuracy, implementation complexity and facility [Demers89, Zhang91, Roberts94, Shreedhar95, Bennett96a, Bennett96b, Goyal96, Stiliadis96, Bennett97], yet it was observed in [Stephens99] that few real implementation exists that can support a large number of diverse sessions at high speed (100 Mbps and higher) while maintaining the important GPS properties of delay bounding and service fairness.

The reasons for difficulties of the PFQ implementation are that it requires both buffering on a per-session basis and a complicated service distribution among all sessions. The issue of complexity most clearly shows itself in the worst-case complexity bounds; worst-case complexity will directly impact upon the delay bounds any algorithm can offer. WFQ [Demers89, Keshav91] has a worst-case complexity function that is $O(N)$, while other algorithms have a complexity of $O(1)$ and $O(\log N)$ [Roberts94, Bennett96a, Bennett96b, Goyal96, Stiliadis96, Bennett97].

Further approaches have been made towards achieving a low complexity solution — by collecting sessions into groups of similar parameter [Rexford96] described how the complexity can be reduced. However, because of variation in the set of flows in each group, it cannot provide minimum bandwidth guarantees. A reduction in complexity is achieved by [Bennett97] using the observation that the complexity of a scheduler for fixed-size packets depends not upon the number of queues but on the number of rates.

Building from this insight, the Worst-Case Weighted Fair Queueing (WF^{2}Q) scheduler was described in [Bennett96b] as a better approximation of GPS. This is achieved by using the packet start time in the reference GPS system to determine the ordering of packet departures. An enhancement to this algorithm, the Worst-case Weighted Fair Queueing+ (WF^{2}Q+) scheduler, was presented in [Bennett96a]. This enhanced algorithm uses a system virtual function computed directly from the packet system rather than WF^{2}Q, which emulates the progress of the GPS. Apart from this difference, used to provide guarantees to flows in a hierarchy of queues and further reducing the complexity of actual implementation, the WF^{2}Q+ version maintains the delay-bound and the close emulation of GPS that are properties of the WF^{2}Q.

A review of router architectures [Kumar98] indicated that the PFQ scheduler and its variants have become the common algorithm providing support for both real-time and best-effort traffic. However, a well-known limitation of PFQ-style schedulers (such as WF^{2}Q+) is in the coupling of delay and bandwidth allocation. In these schedulers there is only one parameter, the per-flow weight, that specifies the resource allocation made to a flow. The value of the per-flow weight affects both the delay and the bandwidth properties of the flow. Under PFQ it is not possible to differentiate between two flows that have the same bandwidth requirements but different delay constraints without over-reservation.

A number of solutions to this drawback have been developed using the Service Curve approach of [Cruz92, Cruz95], such as Service Curve Earliest Deadline (SCED) [Sariowan95], and the Fair Service Curve (FSC) [Stoica97]. The Service Curve approach allows a separation of a flow's delay and bandwidth requirements, assigning service curves of different shapes to different flows — convex curves for flows without stringent per-packet delay bounds and concave curves for flows with tight per-packet delay bounds. The flexibility of FSC allows it to achieve higher resource utilisation for real-time traffic than PFQ. The work of [Stephens99] has shown how an extended version of the hierarchical structuring used in that paper to implement WF^{2}Q+ scheduling can be adapted to implement algorithms with the complexity of the FSC. However due to its complexity, implementations of this algorithm are still difficult.

Examining the timescale between packet-level scheduling and burst-level scheduling

[Courcoubetis97] among others [Roberts92, Chapter 8] noted the importance of packet rather than burst-level scheduling as well as an examination of the traffic conditions that cause one effect or the other to dominate the multiplexing buffer. The importance of timescales is clear, and other authors (e.g. [Grossglauser96]) have proposed that in the examination of specific traffic-induced effects critical horizons exist beyond which traffic-induced effects will have no effect upon the network level (e.g. scheduler, buffer) examined.

2.3.3 Burst Level

At burst timescales, two techniques offer control at this level of granularity. Scheduling (discussed in the previous section), and buffer management. Buffer management may cover a range of schemes from the active division of buffer capacity between users to the precise scheme to be used when space must be recovered from a set of shared buffers. An interest in buffer management has grown with the (perceived) need to decrease the congestion resulting from flow or congestion control schemes that rely upon packet loss as feedback. The TCP protocol is widely used and is an example of such a loss based feedback algorithm. Reacting to the successful transmission of packets,[5] a TCP session will increase its window size, thereby increasing the amount of data and acknowledgements that a TCP session can have in the network at that time. The window is decreased in size whenever loss of packets is detected as it is assumed that packet loss indicates congestion in the network. Such a mechanism for flow control may cause buffers between the TCP endpoints to remain at a high level of congestion.

An alternative use of buffer management is to share buffer resource in order to make service guarantees (e.g. packet loss). This section discusses buffer management schemes proposed to provide service guarantees.

Often resource management is performed to provide differing service levels to different classes of traffic. Several queue management mechanisms have been offered as approaches to improve class-based differentiation. A selection of these schemes is summarised below.

Lowest Priority First (LPF), also known as pushout [Lin91, Kroner91] uses strict prioritisation to

[5] Strictly speaking TCP will increase its window size upon successful receipt of a data segment, a data segment consists of a number of packets — this difference has little impact upon the description given here.

determine how queue resources will be used; when a packet needs to be dropped, it is removed from the lowest priority queue. This mechanism always guarantees ordering of priorities but has the potential for starvation of lower priorities in favour of higher ones. Additionally, until recent router work [Chao97, Suter99], the pushout technique was considered too expensive an operation to be performed in high-speed routers.

A simple scheme that performs precise partitioning of resources, Complete Buffer Partitioning (CBP) [Lin91] allocates a proportion of the total buffer space based upon the priority of the traffic class. However, [Dovrolis00] noted that this scheme has the disadvantage that the mechanism for sizing the buffers is complicated by the relationship between the buffer size, the traffic load of each priority and the delays in the service mechanism in place in the scheduler. Additionally, the scheme is very sensitive to the precise settings of the queue lengths when it is desirable to select a particular loss rate.

Improving the effective use of unused resources, Partial Buffer Sharing (PBS), [Kroner91] and [Matsufuru00] used relative thresholds for each priority of traffic. However, while the system is simpler that CBP, it retains many of the same tradeoffs. To ensure preservation of priorities, this scheme is sensitive to the relationship between the load of each traffic type and, when configured to achieve a particular set of relative loss rates, it is hard to tune because it is highly sensitive to the actual threshold values used [Dovrolis00]. Proponents of a relative-priority scheme [Dovrolis00] concluded that the primary drawback of both CBP and PBS schemes is the inability to dynamically adapt (partition sizes or threshold values) to changing load conditions.

While RED [Floyd93] receives discussion below as a mechanism more commonly identified with TCP/IP, Multi-Class RED [Sahu99] and RIO [Clark98], deserve mention as mechanisms for providing service differentiation. Both schemes can be considered to be built upon PBS; using per-class thresholds to provide differentiated service to a number of classes of flows. In such schemes the threshold adapts to changing traffic-loads. Multi-Class RED involves the use of a token bucket to pre-categorise packets at ingress with interior network nodes using different drop profiles (probability curves) for each class of traffic flow. RIO also uses packet classification at ingress but packets marked IN or OUT of profile are compared against a simpler two-drop-profile scheme at each interior network node. Being considered as special case PBS schemes, both RIO and Multi-Class RED carry with them the same drawbacks of PBS; difficulty controlling the loss rate differentiation between classes.

Proportional Loss Rate (PLR) was proposed as an approach that disconnected the two buffer management tasks, deciding when a packet-drop should occur and deciding from which class the packet should be dropped [Dovrolis00]. This scheme maintains loss differentiation by combining a running estimate of the average loss-rate per-class with the weighting of loss for each class. This allows computation of drop probability for each class. [Dovrolis00] noted issues when dealing with non-stationary traffic and offered two alternative drop mechanisms — one that is accurate in the long-term and simple to implement but that may be less adaptive to changing loads, and an alternative PLR mechanism that copes with non-stationary traffic through the use of a mechanism that maintains a history of loss events for each traffic class.

Using a buffer acceptance algorithm alone rather than with a specified scheduler, [Guérin98] showed that it is possible to provide limited bandwidth guarantees to some flows. One advantage of this is allowing control of the sharing of left-over bandwidth — as compared with leftover bandwidth in scheduler-based systems where the allocation cannot be easily adjusted without changing the scheduling mechanism itself. However, such a scheme may be prone to difficulties adapting to changing traffic load and be difficult to adjust to a precise loss-rate.

In the context of ATM per-VC queueing, [Choudhury96a] described a Dynamic Queue Length Threshold (DQLT) buffer acceptance algorithm, where the (discard) threshold of any port at any instant in time is proportional to the amount of unused buffer space in the switch. With arrivals blocked whenever a port's queue length exceeds the threshold, the DQLT scheme can adapt to changing traffic conditions; if a lightly-loaded queue becomes active, its queue will grow and the total free-buffer space will shrink. As the free-buffer space shrinks the thresholds will be reduced on all queues and any (previously) heavily-loaded queues will be penalised. This scheme has the benefit of being robust but configuration of the tuning parameters is still the subject of further work [Arpaci00].

Clearly, the area of resource control and service differentiation through buffer management has contributed many papers to the literature although few of these schemes are implemented in current switch or router hardware. One reason for this is that such schemes have special demands on the hardware (e.g. Multi-Class RED, RIO, DQLT) or have yet to demonstrate ease of configuration (e.g. PLR, PBS, CBP, RIO). However, such schemes show promise and as switch and router designs evolve their availability is certain.

2.3.4 Session Level

Admission Control

In a connection-oriented network, a session is the period of time between a connection being set up and then torn down. This definition becomes less rigid when applied to connectionless networks carrying connection-oriented protocols. However, for protocols such as TCP/IP, the concept of a flow lifetime delineated by set up and tear down still exists. For session-oriented systems such as the POTS network, AC is the standard mechanism for resource management — a new connection is not admitted into the network unless the capacity is available at every point through which that connection must pass. AC is a preventive traffic control which consists of only admitting new traffic source if its QoS, as well as that of the already accepted sources, can be guaranteed. AC may be considered the principle resource management mechanism at the session timescale. However, AC does not incorporate any intrinsic differentiation of service. Typically, all flows admitted by an AC are subject to the same QoS constraints and same resource. The exception to this is discussed along with acceptance regions below where a combination of scheduling and AC can offer service differentiation for different classes of traffic. In this section, admission control will be explored along with a coverage of survey and comparison papers in this area.

In admission control, an *a priori* description may be used to describe the characteristics of a new admission. The AC algorithm uses this description along with the parameters of traffic already admitted to make its admission decision. Traffic sources may only declare their peak-rate requirements although a variable source may declare other additional information such as a mean rate and information about the bursts of traffic. Using this information the AC algorithm will attempt to ensure a high utilisation of network resources through efficient statistical multiplexing.

A series of comprehensive surveys of static AC algorithms were presented in [Perros96], [Elsayed99], and as recently as [Knightly99]. In these papers the authors considered a large range of AC algorithms that, in order to give best performance, required each new flow to declare as much information about itself as possible; peak-rate being the absolute requirement but more usually sustained-rate (mean-rate) and burst-size as well. Being static AC algorithms, these techniques only use the declared parameters and are unable to adapt to unpredictable changes in resource demand. Section 2.1.1 notes the difficulty faced when computing *a priori* characterisation of traffic sources is required, and subsequently the performance of static AC will suffer when presented with under declared traffic.

MBAC algorithms

The approach of MBAC was driven by both the need to allow only simple characterisation of new traffic flows that are attempting admission and an ability to adapt to the changing traffic.

There have been a wide variety of MBAC algorithms, few with common theoretical backgrounds. Despite that, many of the algorithms do have common elements such as overall structure, the design of the estimator, and the admission policy. In this section a summary intended to highlight the range of MBAC algorithms is given and, following this, the common survey papers in the field are discussed, each offering a unique approach to the comparison of MBAC algorithms.

Two important classifying criteria have been introduced into the field of MBAC algorithms. The first used by [Jamin97a] and [Tse99] is the separation of an MBAC algorithm between an admission criterion and a measurement procedure. The admission criterion determines, based upon the traffic characteristics of current and new flows, whether or not to accept a new flow. The measurement procedure calculates the required traffic characteristics from the current flows, possibly in combination with an *a priori* traffic descriptor of the new flow.

This dissertation adopts a similar policy of separation. An algorithm's admission decision (policy) is treated separately from an algorithm's estimation of traffic requirements. This approach is explored further in Section 4.1.1, decomposing the policy and estimator components for ten MBAC algorithms, illustrating the manner in which policy is common to several MBAC algorithms.

Another element of classification is whether an algorithm exhibits Certainty Equivalence (CE) or not. CE describes an MBAC algorithm that substitutes a measurement-based estimation of traffic requirements in place of a traffic requirement derived from *a priori* traffic description. The concept of CE was introduced in [Grossglauser97b]. MBAC algorithms that are CE make use of a static AC algorithm, inserting measured quantities rather than *a priori* known traffic descriptors. The measured quantities are then assumed to be equivalent to declared/actual parameters.

An advantage of the CE model is that it gives an architecture equivalent to the static AC algorithms. All that is required is a simple substitution process for the method by which traffic parameters (and therefore requirements) are calculated. It transpires that many MBAC algorithm can be considered as using a CE model although CE-based MBAC algorithms will tend towards over-admittance [Grossglauser97b]. This is because MBAC algorithms based upon the CE model do not take into account uncertainty in the measurements themselves. The handling of measurement-based estimations of the traffic characteristics as if these values were as equally valid as traffic descriptors given *a priori* may, therefore, lead to a degrading of the MBAC algorithm's ability to honour the QoS to which it is committed.

The ability to reuse static AC algorithms has meant many MBAC algorithms are based upon the CE model (e.g. [Floyd96, Casetti96, Gibbens97, Jamin97c, Jamin97b, Jamin97c, Droz97] and [Lewis98]). However, this does not mean the problem with measurement uncertainty has been left unconsidered. A common approach used to overcome the disadvantage introduced by this error is to compensate with a conservative measurement procedure; examples of this approach include [Floyd96, Gibbens97, Jamin97c]. A slight variation is given in [Casetti96], an algorithm that adjusts its behaviour on the basis of the traffic offered.

The MBAC algorithms from [Grossglauser97b, Tse99, Gibbens95, Key95, Courcoubetis95] and [Duffield99a] depart from the CE model. Each of these algorithms recognises the overload that results from the occurrence of misleading measurements. Both [Grossglauser97b, Tse99] and [Duffield99a] attempted to characterise the error made when using the CE method. In [Grossglauser97b, Tse99] weak assumptions were made about the asymptotic regime; a heavy load and a single flow contributing only a small part of the total link capacity.

[Duffield99a] also assumed that a heavy load is present and that a single flow has a small activity relative to the link capacity. This is combined with assuming loss is rare and that large deviation theory is applicable. One of the assumptions required for large deviation theory to be valid is the independence of measurement samples and an absence of correlation between samples.

The approach of [Courcoubetis95] discussed the use of virtual buffers to increase sampling and reduce variance of measure-only methods. Primarily a heavy theoretical treatment of the problem, this approach may work when measurements are under-determined.

Discussed at greater length in Section 4.4.3, the approaches of [Gibbens95] and [Key95] use two techniques to overcome the error and poor performance of the CE model; firstly a Bayesian prior on the call statistics serves to smooth-out the fluctuation in successive memoryless estimates, observations are weighted by a fixed prior. Secondly, a policy whereby new flows of a particular type (class) are not accepted if one has been rejected until a flow of that class has left the system, has the effect of countering overly high rates of new flow attempts.

Comparison of MBAC algorithms

In contrast to the general field of AC algorithms and despite there having been substantial literature introducing MBAC algorithms, only a few comparisons of these algorithms exist. As a primary contribution of this dissertation is a comparison of MBAC algorithms, it is appropriate to review the related literature of the field. In addition to few comparisons, previous literature comparing MBAC algorithms has commonly restricted itself to a comparison based on limited criteria such as the resulting line utilisation and overall packet loss. Several comparisons have noted that each MBAC algorithm achieves near identical utilisation for a given total packet loss rate, [Knightly98] reported this result while conducting a comparison of a new MBAC algorithm with those of [Floyd96, Jamin97a], and the comparison of [Breslau00], based upon a wider simulation of several MBAC algorithms [Jamin97c, Floyd96, Gibbens97, Lewis98, Knightly98] both drew this conclusion.

Additionally, [Breslau00] concluded that none of the algorithms was capable of accurately meeting set loss targets and that the tuning parameters inherent in each MBAC algorithm could be considered as *uncalibrated knobs*.

The comparison reported in [Jamin97b] revealed that the algorithms they considered share a common structure [Jamin97b, Eq. 14]:

$$\hat{\nu} < f(\cdot)\mu - g(\cdot) \tag{2.1}$$

where μ is the link capacity, $\hat{\nu}$ is the measured load and $f(\cdot)$ and $g(\cdot)$ are respectively functions of source's reserved rate and the number of admitted sources. Interestingly in this common structure the CE model is clear, particularly when compared with the decision criteria of a sample algorithm such as the Measured Sum MBAC algorithm of Section 4.4.5.

The survey reported in [van den Berg00] reviewed admission control and through a simulation, analysed two MBAC schemes, an algorithm from [Jamin97a] with an algorithm reported in [Brichet97]. Using both artificial model sources (e.g. Markovian ON-OFF sources) and traces of real traffic (an unknown amount of Internet traffic carried over ATM-based networks), a comparison was conducted between each MBAC and results gained using the static AC approach from [Elwalid95]. The algorithm from [Elwalid95], described in Section 4.2.12, is a static AC algorithm which computes an *effective bandwidth* estimate of traffic from a set of *a priori* parameter declarations. Comparison with this algorithm allows contrast between MBAC algorithms and static AC algorithms. The conclusions of [van den Berg00] were that a network offering integrated services requires a combination of both AC based upon *a priori* declaration of traffic parameters (for traffic with strict requirements) and AC based upon measurement-based estimation (for traffic with more flexible QoS requirements.) Additionally, the authors noted the problems of CE-based algorithms and then use two MBAC approaches, neither of which address the drawbacks of the CE model.

A study presented in [Shiomoto99] reviewed a handful of MBAC algorithm approaches [Saito91, Tedijanto93, Gibbens95, Li95, Shiomoto95, Dziong97]. This survey presented a summary of the base ideas of each algorithm. The MBAC algorithms are contrasted using the criteria of the implementation complexity (estimated from each algorithm's design), bandwidth efficiency (based upon whether the algorithm accounts for buffering effects), and the form of the basic computation (whether based upon calculations of *effective bandwidth* or loss-ratio). [Shiomoto99] does not conduct an operational comparison and only notes in passing the effects of traffic self-similarity. However, this paper provides an interesting comparative contribution.

Interactions between Session Level and Smaller Timescales

As noted in the introduction to this section, control at a timescale smaller than that of a flow lifetime involves buffer and schedule management as well as the various schemes of self-regulation in elastic flows. However, prior to discussing these schemes, it is worth elaborating on the union of scheduling and admission-control put forward as the concept of a *schedulable region*.

As was mentioned earlier, given a single queue-service discipline, AC alone can offer only one QoS (e.g. one combination of loss probability and delay boundary) to all flows regardless of the traffic type. Schedulable regions were introduced in [Hyman91] as a way of improving the performance of ATM networks. Each traffic class has a particular QoS associated with it, and the *schedulable region* defines a surface describing where combinations (numbers of) of traffic connections of each traffic class are able to maintain a desired overall QoS commitment. Central to this scheme is the desire to separate packet and flow level characterisation. The schedulable region put forward is done so in conjunction with packet scheduling methods — the authors of [Hyman91] noting that the packet scheduling is mandatory.

Further work in [Hyman93] proposed a mechanism for distribution of admission control and scheduling along with the dynamic computation of the admissible surface [Lazar91], emphasising the principle of separation that exists between scheduler and admission control.

Scheduling regions are also referred to as admission regions because they define when combinations of traffic may be admitted into a system, while still maintaining a desired set of QoS guarantees. Admission regions are discussed further in the outline of several of the implemented MBAC algorithms in Section 4.4. Although, without specific scheduler support/control these MBAC algorithms use a common QoS for all flows.

2.3.5 Beyond the Session Level

Network management at timescales beyond that of session or call include the timescales of deployment and service renegotiation. As well as deployment, this timescale includes the operational management of networks, encapsulating activities such as network load balancing, failure recovery, and maintenance. It is clear that such tasks are critical to the ongoing operation of any management network.

A number of tools, often using measurement-derived information, are used to assist the necessary trend analysis. For example, [McGregor00] presents a system for visualizing data from logs and measurements, while [Caceres00] describes an early version of the Netscope tool. Purpose-built for IP networks, Netscope combines network routing information with measurement data (e.g. flow duration, or aggregate utilisation) to provide information for management and problem-resolution.

Each of these examples are designed for operation over the periods of days or months, although the demand on such tasks has grown to cover both the largest and smallest extents of the timescale range beyond the session level. For example, requirements for maintaining continual redundancy may force a long-term management tool such as those mentioned above, to be continually rebalancing flows in a network to ensure an appropriate quantity of redundancy is maintained. Another example is Service Level Agreements (SLAs), which may be deployed or renegotiated over a wide range of timescales. For the renegotiation of such SLAs, a measurement-based scheme may seem an obvious choice for providing the SLA supplier and user with the ideal quantities.

Chapter 6 presents a form of dynamic renegotiation that illustrates the application of a measurement-based estimator to this problem. With an increased difficulty in modelling modern traffic behaviour and an increased demand on network management across all timescales, application of measurement-based techniques become a clear approach the management of each timescale will incorporate.

2.4 Measurement

The previous section has attempted to highlight some of the difficulties with schemes for resource-management. In many cases the particular management mechanism is bound to a certain time-frame and, while the relationship between a particular timescale and a particular management scheme cannot be relied upon, appropriate construction, using more pessimistic assumptions, can lead to schemes for managing network traffic both in spite of and through the use of the timescales which are present.

The interaction of measurement and control timescales requires particular attention. If an adaptive algorithm for bandwidth allocation operating at a large timescale relies upon utilisation measurements made over too-small a period, interaction between the traffic and the adaptive routing algorithm may occur. The precise interaction is difficult to predict but may take the form of over or under provisioning of resource or of oscillations between different provisioned quantities. The solution may be quite obvious, such as in this example where the characterisation period of the bandwidth allocator needs to be sufficiently long so as to correctly characterise the traffic. However, faced with complex underlying traffic, any system using a measurement-based estimator will require careful attention to the critical parameters of both estimator and traffic.

In addition to interactions between the timescales of different levels of resource management, there are assumptions about the measurements which are themselves intrinsic to the construction of many measurement-based resource management schemes. At the simplest level the measurement type and period derived characterisation should be appropriate for the management task: e.g. the day-long mean utilisation may be of limited use in an AC of voice connections. Along with measurements over too long a period, measurements taken over too-short a period may not prove useful either.

Measurements must suit the purpose to which they will be put; even among a common management approach there can be variation, Chapters 4 and 5 explore the manner in which several different types of measurements are used by AC algorithms. However, having the right measurement for the task is not the only issue of measurements; often overlooked is that measurements themselves have statistical properties and may be considered random variables in their own right.

In Section 2.3.4 the notion of CE in AC algorithms was noted as where MBAC algorithms use an Measurement-Based Estimator (MBE) as a drop-in replacement for the estimate of current demand in a static AC algorithm. However, for this substitution to be valid, special attention needs be paid to the characteristics of measurements themselves. In [Tse99], discussion was made of the uncertainty of measurements and how, in turn, MBAC algorithms must account for this uncertainty as part of the admission process. The measurements required as essential input into the estimators of MBAC introduce issues unique to the MBE — not raised for the static AC algorithms commonly replaced through CE.

Thus special problems include the time taken to communicate the measurements to the MBE, the choice of measurement period and the nature of the measurement. Figure 2.6 illustrates the last one of these. This figure shows how an AC algorithm that relies on *instantaneous utilisation* will receive a measurement that is out of date by on average half of the measurement period. Figure 2.6 illustrates regular utilisation measurements computed periodically by an implementation.

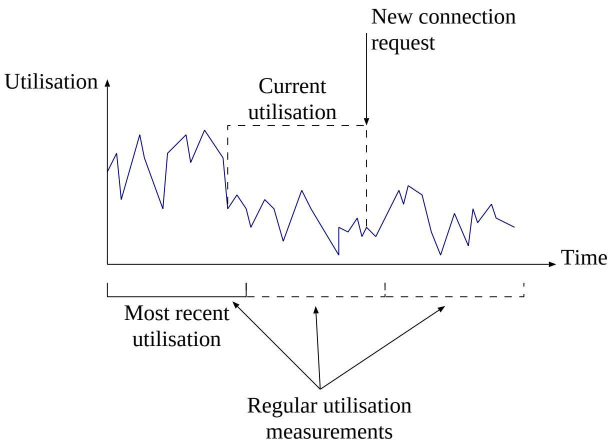


Figure 2.6: Instantaneous and periodic measurements of utilisation.

When the AC algorithm requires a utilisation measurement, the actual current utilisation may be significantly different to the most recent measurement of utilisation provided by the implementation. The use of instantaneous utilisation measurements is a requirement common to a number of the AC algorithms presented in Chapter 4 (e.g. AC-ST, AC-MS, and AC-CB). Yet, in each of these algorithms, no mention is made of the problem or of the error it would introduce.

For a homogeneous traffic flow that displays Markovian statistical properties, the situation of Figure 2.6, the accidental rejection or admittance of new flows might be expected to create an error that displays normal distribution with minimum and maximum values being the minimum and maximum values of utilisation measurable in any single measurement period.

However, how such an error actually affects any particular algorithm is less well understood. A common assumption of authors is to ignore the problem and inherently to assume that the long term effect will be negligible. Such a conclusion about this effect being negligible may be because the system, in the case of the AC-ST, appears to have an equal likelihood of admitting a flow attempt as it has of rejecting a new flow attempt. In reality this is not the case.

In Table 2.2 (extracted from Table 5.4) the simple threshold algorithm (AC-ST) is used to illustrate the dramatic effect variance can have upon the performance of an AC. The most significant errors occur when the measurement period is 3.94 ms. As stated on Page 190, assuming a normal distribution of error, a mean of 92.89 Mbps, and a standard deviation of 17.90 Mbps implies

	Period	Mean	Coefficient of Variation	95% Confidence Interval
E-ST with P-TO	131.2 ms	78.2	7.2	4.2×10^{-2}
	3.94 ms	92.7	3.4	2.5×10^{-2}

Table 2.2: Mean flows-in-progress statistics for algorithm/policy combinations.

as many as 1 in 10 measurements were at or below 70 Mbps. With every measurement below the threshold, flows would have been (erroneously) admitted and, because there is no memory of either these erroneous admissions or memory of the significant variance error, the system documented in Table 5.4 was in chronic overload.

Figure 2.7 illustrates how traffic measurements can vary with measurement period. This figure was made by measuring 70 continuous flows of the Markovian traffic source TP10S1 using several measurement periods. The TP10S1 traffic source, being based upon exponential distributions of burst size and inter-burst delay, does display the characteristics whereupon the measurements follow a normal distribution. Note that the longer the measurement period, the better the sample of the flow — the less variance the measurements exhibit. The selection of measurement period introduces a significant effect on the variance of each measurement but the answer does not simply lie in increasing the measurement period.

A discussion was reported in [Claffy93] of issues of sampling in the context of network traffic characterisation. They conduct a comparison between event (packet) driven measurements and those strictly time-driven and conclude that packet-driven measurements provided a result that most closely approached the characteristics of the original traffic. The importance of this result cannot be underestimated, but is difficult to confirm, as the implementation of many measurement systems, including the measurement-systems providing data for the MBEs studied in Chapter 5, are based on time-driven sampling. Figure 2.6 forms a useful illustration of sampling error that results from a time-driven sampling approach.

In addition, the assumption that the traffic is either homogeneous or displays appropriate statistical properties is also a dangerous one to make. Figure 2.8(a) and 2.8(b) show how both the individual and the multiplex of three traffic flows with similar peak and mean rate character-

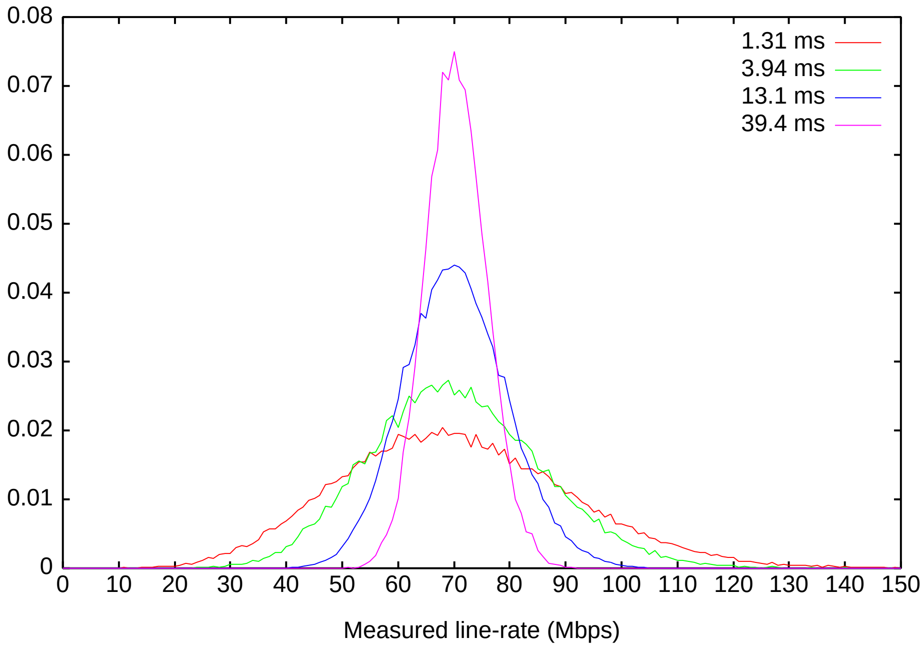
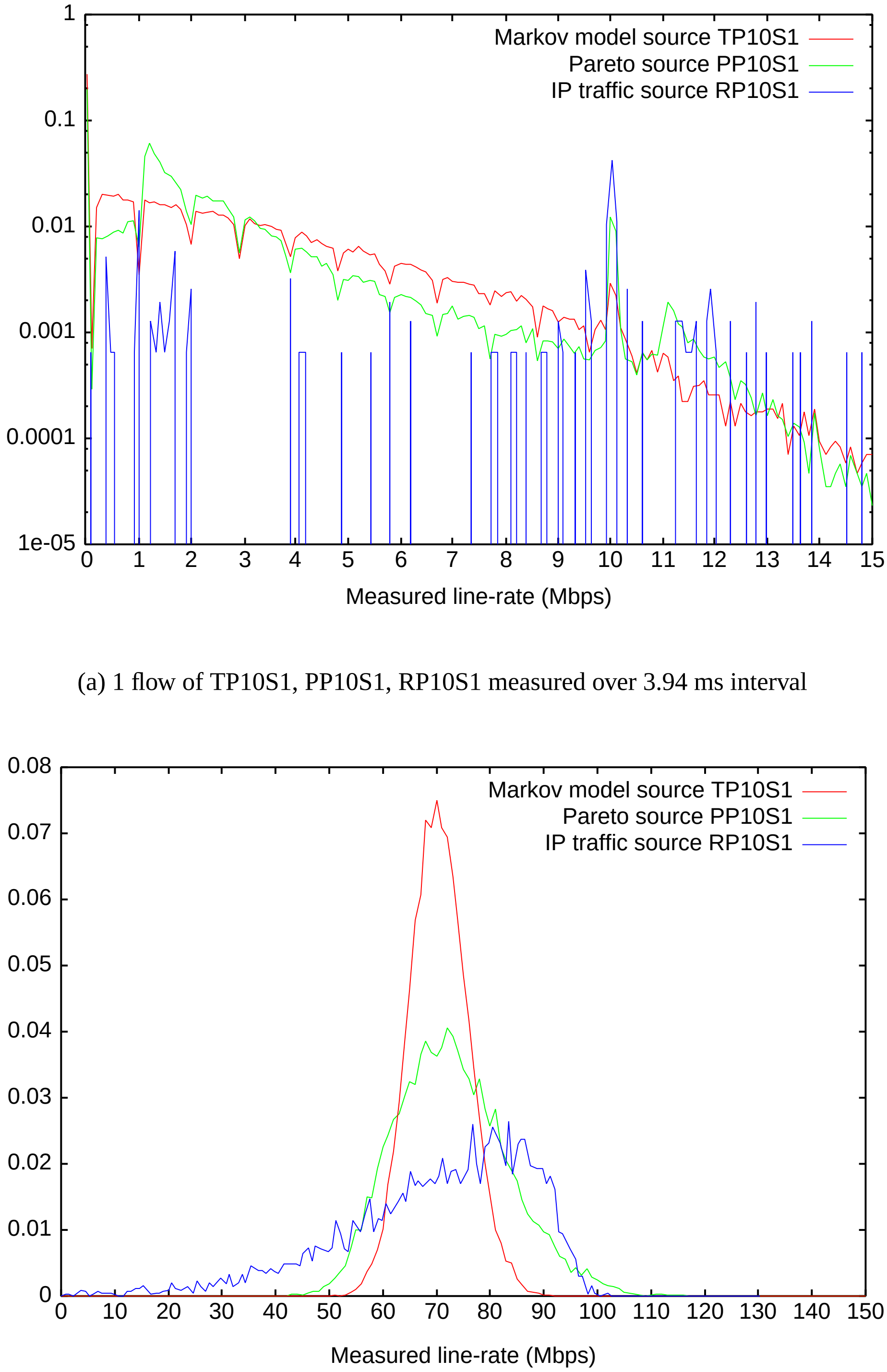


Figure 2.7: Relative frequency distribution of Markov model traffic (TP10S1).

istics, measured over the same period, can vary due to the internal statistical properties of the flows. PP10S1 is a source based upon the Pareto distribution which implies that for a multiplex of Pareto sources strong long-range dependence will be exhibited. A comparison of results from PP10S1 and TP10S1 shows that the variance in the measurement values is clearly increased in the PP10S1 source. In contrast, the multiplex of streams carrying the Internet source RP10S1, while exhibiting the same peak and mean flow rate as the multiplex of TP10S1 and PP10S1 sources, has a distribution that has significant skew. The large peak towards the line-rate implies a significant burstiness and, with a closed-loop control mechanism such as TCP, the time structures of TCP itself as well as the interaction between competing flows of traffic have noteworthy internal similarity. These properties have long been recognised in TCP (e.g. [Feldmann98a]) and when the typical MBAC algorithm was formulated, flows carrying elastic traffic such as TCP were not considered. However, the exact problems encapsulated by the measurement-based interpretation of such flows is clearly indicated in Figure 2.8(b). Aside from a significant variance, the distribution of measurements of TCP-based traffic displays a distinct skew which draws into question the representation of this traffic as a normal distribution.



(a) 1 flow of TP10S1, PP10S1, RP10S1 measured over 3.94 ms interval

(b) 70 flows of TP10S1, PP10S1, RP10S1 measured over 3.94 ms interval

Figure 2.8: Relative frequency distribution for measurement periods.

There have been various approaches to quantifying the error introduced due to the stochastic character of the measurement itself. As Section 2.3.4 notes, several MBAC algorithms document specific solutions to this measurement problem (e.g. [Gibbens95, Key95, Knightly98, Qiu98b, Grossglauser97b, Tse99]). In [Gibbens95, Key95] a solution was provided using an estimator based upon a Bayesian model. From a given initial load and a set of recursive equations, an estimate of future load is calculated from successive measurements.

The architects of these algorithms have concentrated on combining the measurement information with the *a priori* traffic characterisation. It is then assumed that the recursive-correction of the Bayesian approach would accommodate the measurement-based errors. Aside from [Gibbens95, Key95], a Bayesian approach was also adopted by [Warfield94].

In contrast, the measurement errors were handled in a significantly different way in [Grossglauser97b, Tse99]. In this work the use of a memory containing past network state has been used to improve the performance of the MBAC algorithm described. The work concentrated on the use of statistical characterisation of the traffic and measurement, particularly its variance, to give a better admission process.

Memory of measurements, in the form of second-order statistical properties, have also been used in [Knightly98, Qiu98b]. By incorporating variance into the estimation technique, the estimator is improved by accounting for errors introduced by the measurement process.

2.5 Summary

This chapter describes an overview of network traffic, noting how real traffic becomes more complex and how models fail to capture the increasing complexity of such traffic. The popular rise in sources whose complexity is not captured by traffic models leads, in part, to the core study of MBAC algorithms of this dissertation. The study uses sources with a range of characteristics from the entirely deterministic Markovian models to the traces of IP sources and the implementation of closed-loop TCP/IP-based traffic. The sources used in this dissertation were presented in this Chapter.

The idea of timescale in management was introduced, with three timescales receiving particular attention. The timescale over which scheduling of packets was covered noting the body of work that has taken place in scheduling algorithms. Following this, the burst level timescale, which incorporates buffer management techniques, was discussed. Following these two management schemes the chapter covered admission control systems operating at the session level timescale. Given that much of the measurement-based management work is derived from MBAC algorithms, this section covers admission control in general and MBAC in particular. Additionally, attention was particularly paid to the classification and comparison of AC schemes in previous work. Finally, as part of session level management schemes, the interaction between this timescale and the lower timescale was discussed.

Measurement-Based Management must incorporate unique, measurement-related issues to provide optimum performance. Such issues include delays in measurements, the length of measurements, quantity of measurements and the type of measurements: whether per-flow or aggregate. This chapter noted several potential problem areas, including the choice of measurement interval and the reliable interpretation of measurements of traffic. Clearly, there is potential for an interaction between the measurement interval and the control (management) timescale, the results of this dissertation discuss this important topic.

Chapter 3

Environment

3.1 Introduction

The previous chapter has shown that the use of the link/source approach to traffic models is inadequate for representing current or future network traffic. Building upon this premise, this chapter presents a test environment that includes the real-world aspects of implementation and system interaction, and gives the additional ability of allowing the integration of representative traffic. High fidelity results allow direct comparison between different approaches and current implementations through the incorporation of real traffic sources such as WWW derived workloads and video-sources into a test environment.

The test environment can be used several different ways. Firstly, traffic generators creating streams compliant with those generated by deterministic models[1] allow the environment to be tested for compliance with the theory as well as allowing the theoretical approaches themselves to be validated. Secondly, testing of management schemes such as bandwidth allocation or AC approaches can be done using both model-based and real traffic sources. Real traffic such as a video stream can be created using the actual video codec engine rather than a model of the traffic resulting from it, or by using appropriately configured hosts to recreate the conditions which

[1] A number of teletraffic engineering approaches are based upon deterministic models, such as the 2-state ON-OFF Markovian source. Link/source model-based admission control is an engineering approach that often assumes such Markovian sources.

could be found in a real-world deployment.

This chapter is structured into four sections. Section 3.2 presents a summary of related approaches for network investigation, noting the differences, advantages, and drawbacks of each approach. The design and construction of each component of the test-environment is outlined in Section 3.3. Following this, Section 3.4 outlines the test-environments operation. Finally, Section 3.5 presents an evaluation of the environment. This evaluation covers the reliability of the traffic generators as well as issues of finite length experiments: determining the optimum run-time and removal of initial transient periods. As part of the evaluation, this section then reports on performance limitations and error margin of the environment as predicted through experiment.

3.2 Background and Previous Work

Chapter 2 showed that teletraffic engineering has required the development of new, more complex, traffic models. These models are hard to create and validate, and are often prone to error. Improvements have taken the form of network-wide models that attempt to encapsulate the complete behaviour of protocols and, although there has been some success in using these for fixed-point approximation, these models are also prone to error or at least incompleteness.

An alternative approach to modelling is simulation. However, simulation of a complex set of protocols such as the TCP/IP suite requires a complete authoring of the entire TCP/IP protocol stack. [Baumann98] adopts this approach using an implementation of the TCP/IP protocol stack to simulate the interaction between TCP/IP and the ABR service on ATM networks. Such an approach is labour intensive and may well be unable to use any part of current or future implementations.

Improving upon this idea, [Manthorpe96] overcomes inconsistencies between simulator and an actual system by using a TCP/IP stack implementation from a common OS. His approach implies the simulator is capable of behaviour that closely duplicates the original implementation. However, the drawbacks are twofold. Firstly, such a simulator based upon actual implementation requires substantial configuration to correctly emulate the OS behaviour. Secondly, being based upon a single implementation from a specific OS, a simulator based upon this approach is limited to emulating the behaviour of only one specific TCP/IP stack. This problem is made worse as newer versions of the protocol accumulate even more significant changes.

Finally, there is the use of 'ideal' simulators, ones that make a thorough implementation of the up-to-date standards in order to encapsulate the latest developments. The commercial simulator OpNet [OPNET00] is one; while the public-domain simulator `ns` is another [Bajaj99]. The `ns` simulator is popular because it is freely available along with continual flow of fixes and improvements. `ns` is commonly used by researchers to conduct experiments to illustrate improvements and modifications to existing protocols as well as to investigate the impact new protocols may have upon current networks. Popular use in these roles has meant `ns` has become the de facto standard for simulating Internet activities. However, `ns` has several significant drawbacks. Firstly, because it cannot readily make use of the implementations of new services, some time will elapse between the completion of an implementation and the availability of a complete and fully compliant simulator. Secondly, the program has its own unique method for constructing experiments, an approach that may not suit all situations. Finally, discrete-time simualtors such as `ns` are significantly slower than an actual implementation.

The requirements for experimental speed as well as an accurate replication of the behaviour of current traffic lead to the idea of an implementation-based study. A platform suitable for such a study ought to provide a suitable level of instrumentation of the network as well as control of the network's active components: the traffic sources.

[Lazar97] describe a test environment built around the Hewlett-Packard Broadband Series Test System (HPBSTS). While it seems likely that this system could be used to evaluate AC algorithms, parts of this system are based around a limited release of proprietary information. This, combined with the high cost and low-availability of the HPBSTS itself, makes it difficult to recreate such a test environment.

[Risso99] used a test environment to assess the operation and performance of a class-based queueing router. Using a test network that included both local and wide-area components, the authors assessed the behaviour of a public-domain implementation of a Class-Based Queueing (CBQ)-style router. The tools used included two publicly available load-generators: for TCP traffic, `ttcp` [Stine90, CCCI99] and for UDP traffic, `netperf` [HP99]. The commonly available network tracing tool tcpdump [Jacobson99a] was used as the principle measurement tool. While not a comprehensive test — standard `ttcp` and `netperf` utilities only being able to create very specific traffic test-patterns — this combination of tools was used to good effect allowing the authors to locate and correct errors in the CBQ implementation.

3.3 Test environment construction

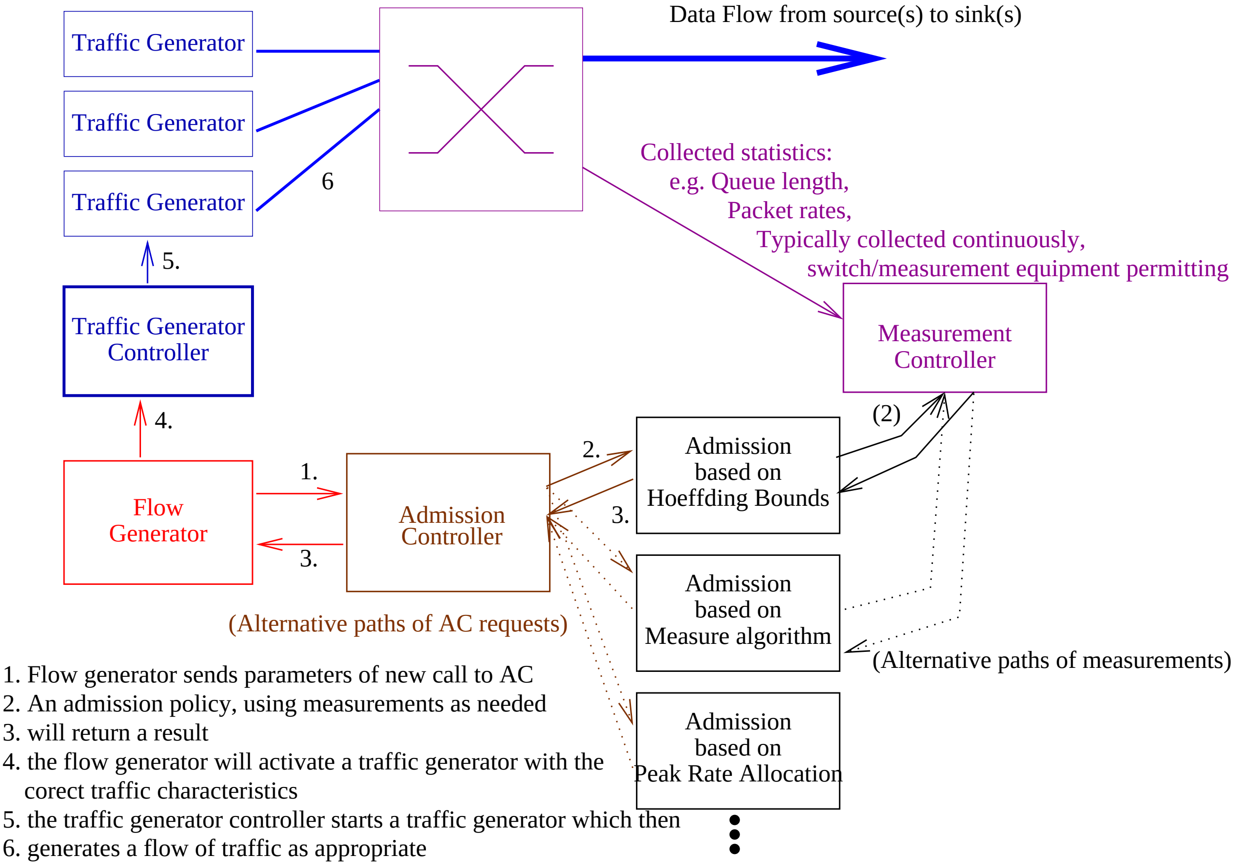
The test environment to be used in Chapter 5 is presented in this section. The presentation of this environment is ATM 'influenced' as its original purpose was to evaluate admission control algorithms designed for use under ATM Forum signalling [ATMF95]. Nonetheless, the overall structure of this system lends itself to the testing of many environments, independent of the underlying network substrate.

The AC algorithm test-environment consists of a combination of hardware, (network switch and network interface cards,) and software to generate new connections, perform AC operations, obtain measurements from the network switch, generate network traffic and control the generation of traffic sources. Figure 3.1 shows the implementation architecture adopted to evaluate AC algorithms. Specific components of the AC algorithm test-environment are discussed along with an outline of the test-environment's operation.

3.3.1 Network switch

The network switch must control where packet loss will occur and allow variables such as buffer size and buffer service rate to be controlled. In addition to being a controllable buffer, the switch also makes measurements of line utilisation, packet arrivals and packet departures. Using these measurements, the loss-ratio of the link and indeed the utilisation and loss-ratio per connection can be determined.

This implementation is based around a commercially available ATM switch: a FORE ASX-200WG. The ASX-200WG supplies CPU controls and (passive) switch-fabric along with chassis facilities such as power. The primary components of the switch were the line-cards, the SONET OC3 Revision-D network module [FORE98, FORE99]. These cards proved particularly flexible and can be arbitrarily configured to emulate a number of different capacity links. In this way


Traffic Generator
Traffic Generator
Traffic Generator
6
Data Flow from source(s) to sink(s)
Collected statistics:
e.g. Queue length,
Packet rates,
Typically collected continuously,
switch/measurement equipment permitting
5.
Traffic Generator Controller
Measurement Controller
4.
(2)
Flow Generator
1.
3.
Admission Controller
2.
3.
Admission based on Hoeffding Bounds
Admission based on Measure algorithm
Admission based on Peak Rate Allocation
(Alternative paths of AC requests)
(Alternative paths of measurements)

1. Flow generator sends parameters of new call to AC
2. An admission policy, using measurements as needed
3. will return a result
4. the flow generator will activate a traffic generator with the corect traffic characteristics
5. the traffic generator controller starts a traffic generator which then
6. generates a flow of traffic as appropriate

Figure 3.1: Architecture for the implementation of a test environment to evaluate AC mechanisms.

a 155 Mbps link could be configured to emulate a 1.5 Mbps ADSL connection or a 100 Mbps TAXI interface. The ability to select arbitrary speeds of operation was crucial for rate-scaling.

To ensure cell loss occurs in the controlled buffer, rate-scaling is used. Rate-scaling, illustrated in Figure 3.2, is where the service rate is reduced by $1/D$, where D is a chosen integer. In Figure 3.2 each link is labelled with its transmission rate relative to the full line rate of 1. The transmission rate of each traffic source is scaled by a factor of $1/D$. The rate of the interface between the input port **A** and the buffer for output port **B** is at the full capacity of the switch. The speed of the output port, **B**, is scaled by the same quantity as the traffic sources, $1/D$.

Traffic arriving at the buffer will queue in the output buffer of port **B**. The switch has a switching capacity of D times the input and output transmission rates and is, therefore, effectively non-blocking. In this way a controllable buffer can be achieved where parameters can be set (such as queue length) and about which measurements can be taken (packet loss, packet delay). Access to the switch buffer configuration, such as traffic classes, intelligent discard policies and scheduling systems allows complex environments to be constructed.

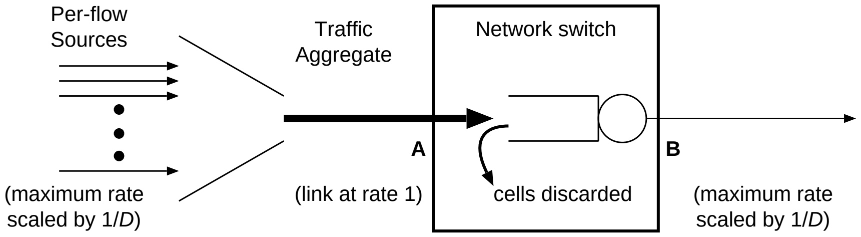


Figure 3.2: Topology of the network switch.

The network switch also makes measurements, counting packets moving into and out of the buffer in a given period of time. From these measurements the line utilisation and packet loss can be calculated; in aggregate or on a per-connection basis. These counts of packets traversing the system are periodically transmitted to an external measurement controller,[2] so as to reduce the work-load of the switch itself.

[2] Every 1.31 ms measurements are transmitted to the measurement controller.

3.3.2 Measurement controller

The measurement controller, a process running under UNIX, obtains from the switch the measurements that may be required as input into an AC algorithm. These measurements include the traffic activity and also the QoS experienced by the traffic — its loss-ratio, queue length distributions and loss events both on a per-line and per-flow basis. These measurements permit off-line comparisons between the performance of each system in operation.

Along with retrieval and storage of measurements from the switch, the measurement controller matches the asynchronous measurement requests (from each AC algorithm) to the synchronous methods by which measurements are taken. The measurement controller also interfaces between the proprietary inter-machine protocol used by the switch and a standard RPC interface that is used between the measurement server and the AC system.

3.3.3 Traffic Generator

For the work of this dissertation two different traffic generator implementations were used. The first, a generator of inelastic traffic, is described in Section 3.3.3.1. The inelastic generator is used to recreate the traffic of video streams, voice data or one of the model-based traffic definitions such as the 2-state ON-OFF Markovian source. The inelastic generator is also able to play traces of pre-recorded traffic flows.

The second generator creates elastic traffic streams and is described in Section 3.3.3.2. The elastic generator is used to create traffic of TCP/IP flows: e.g. the WWW transactions between client and server. The elastic traffic generator can emulate sets of HTTP streams with known distributions of file-size and inter-transaction time.

For the first generator a traffic source must represent traffic as carried in an ATM network. This form is needed to represent streams of traffic to be transmitted by traffic generators. If these streams are generated off-line, it involves the creation of a list of integers each of which represents a cell. In this way the traffic generation uses a simple play list of cells that counts the timing of cells, rather than the content of the cells or any other aspect. This comes about because ATM traffic can be represented as a stream of cells and the spaces between cells. Thus the traffic stream be described as a series of integers, each integer counting inclusively the number of cell-

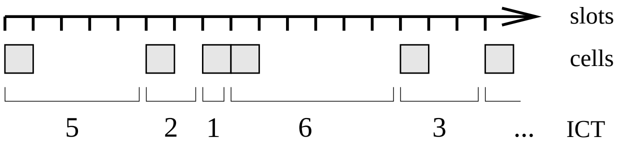


Figure 3.3: Inter-Cell Times representing a stream of traffic.

slots between one cell and the next in the stream. Figure 3.3 illustrates how a stream of cells of traffic can be characterised as a series of ICTs.

3.3.3.1 Inelastic and trace-based sources

The generator for trace-based and inelastic sources is complicated by its ability to provide a wide variety of traffic types. By describing the evolution of the traffic generator, the variety of abilities of this generator ought to be explained.

Figure 3.3 illustrates how, in an ATM network, a stream of traffic may be represented as a series of integers. A simple traffic generator may be constructed to read a series of integers in memory and transmit with a spacing described by that list of integers. Figure 3.4 illustrates such a trace-based generator: a simple generator creating the traffic for one flow.

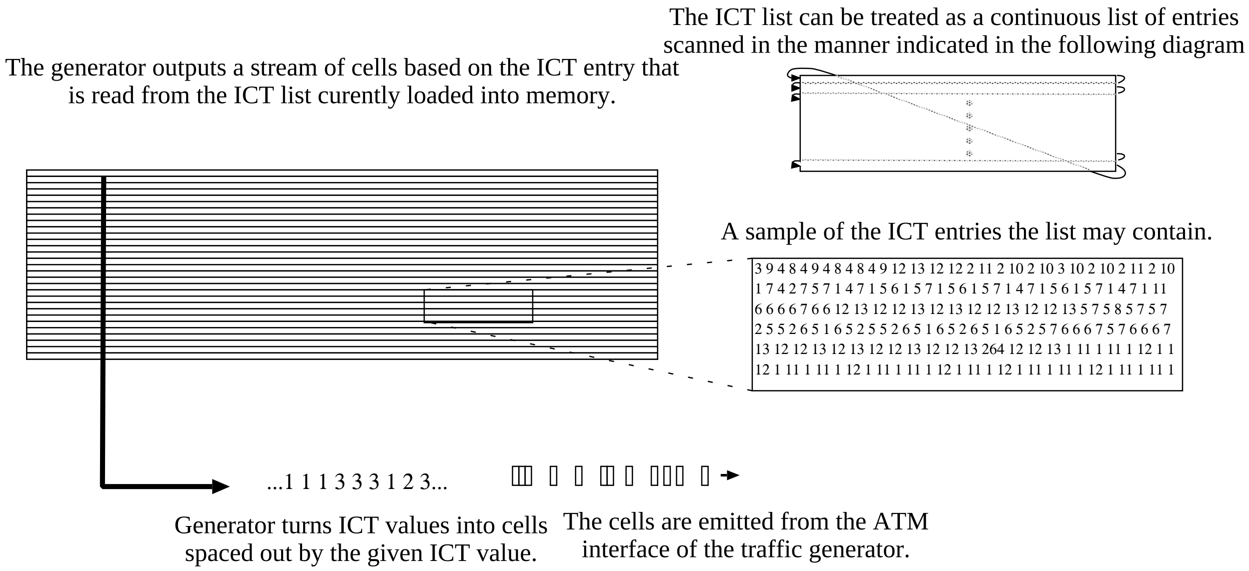


Figure 3.4: Single trace traffic generator

To create the traffic of multiple flows using traces two alternatives are available. Firstly, the multiplex of two or more different traffic sources can be used to create the desired traffic mix in the trace-file — a simple generator would carry as many flows as were multiplexed into the original trace-file. However, this approach limits the control over the number of flows of traffic, requiring a change in the trace file used each time the number of flows to be generated needs to be changed. Alternatively, the traffic generator itself could multiplex the sources from a trace-file containing the data for only one flow. Figure 3.5 illustrates four traffic streams multiplexed by the generator to provide traffic for four traffic sources.

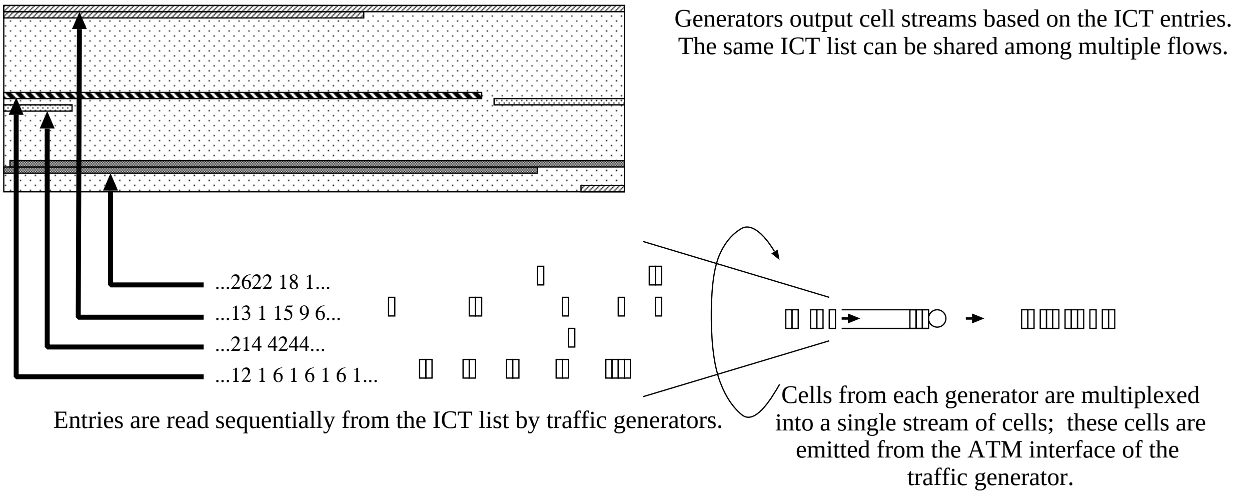
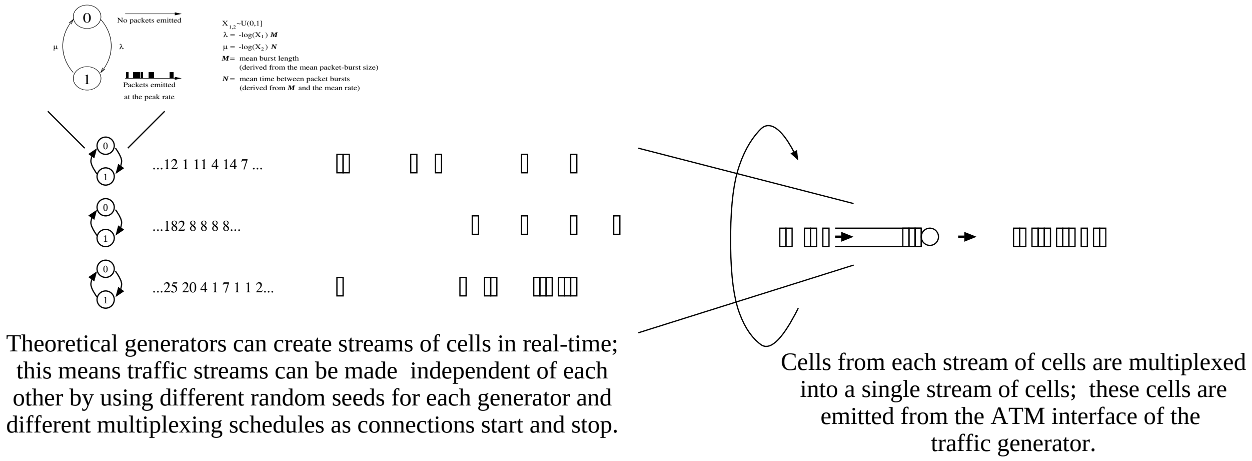

Generators output cell streams based on the ICT entries.
The same ICT list can be shared among multiple flows.
...2622 18 1...
...13 1 15 9 6...
...214 4244...
...12 1 6 1 6 1 6 1...
Entries are read sequentially from the ICT list by traffic generators.
Cells from each generator are multiplexed into a single stream of cells; these cells are emitted from the ATM interface of the traffic generator.

Figure 3.5: Single trace traffic generator generating traffic for multiple flows

The data for each single flow may be started at independent places in the trace-file. This ability, combined with a large enough amount of data in the trace-file, will ensure no unintended self-similarity arises through the use of the same trace-file data. However, a generator based upon trace files is limited in the type of traffic that may be generated: even for the generator of Figure 3.5 each flow can only carry the traffic of the trace-file.

A more flexible traffic generator is possible: one that computes the ICT of the traffic in real-time. A number of traffic types are based upon models: models such as the 2-state ON-OFF Markovian model, 2-state Poisson models, and models of video codec behaviour. Figure 3.6 shows how a 2-state ON-OFF Markovian model, (of Figure 2.4,) can be used to compute the ICT stream in real-time. The output of three such model-based sources can then be multiplexed together as shown in Figure 3.6. The traffic carried by each flow can be computed by an independent traffic-model, each using independent parameters to describe its own characteristics.


0
1
μ
λ
No packets emitted
Packets emitted at the peak rate
$X_{1,2}$~U(0,1]
λ = -log(X_1) M
μ = -log(X_2) N
M= mean burst length (derived from the mean packet-burst size)
N= mean time between packet bursts (derived from M and the mean rate)
...12 1 11 4 14 7 ...
...182 8 8 8 8...
...25 20 4 1 7 1 1 2...

Theoretical generators can create streams of cells in real-time; this means traffic streams can be made independent of each other by using different random seeds for each generator and different multiplexing schedules as connections start and stop.

Cells from each stream of cells are multiplexed into a single stream of cells; these cells are emitted from the ATM interface of the traffic generator.

Figure 3.6: Multiple-model traffic generator generating traffic for multiple flows

The traffic generator used to create inelastic traffic streams for this dissertation is a hybrid of the two techniques of trace-driven traffic and model-based traffic. The hybrid generator, illustrated in Figure 3.7, can be used to transmit a multiplexed stream of cells from generators creating cells in real-time and from generators reading from a pre-generated ATM ICT series. The generators may be based upon deterministic models, such as 2-state ON-OFF Markovian, 2-state Pareto, or upon the operation of codecs such as those that allow the transmission of video-streams at a nominated rate [NRL96]. Traces of real network traffic, such as video streams or Internet WAN or LAN traffic, can also be carried by connections. In addition to being able to deliver packet streams consisting of all required traffic types, this hybrid generator can be dynamically controlled, able to stop and start individual traffic sources using a purpose built RPC mechanism.

The hybrid generator runs on a PC that is running the Nemesis operating system [Leslie96] which allows the construction of complex, time-critical tasks (the real-time creation of traffic traces) and the timely operations of device-drivers. In addition to allowing a purpose built device driver for the network interface, using Nemesis means that guarantees of timeliness can be made to the RPC based control mechanism to ensure time-bounded actions and replies.

The hybrid generator is capable of saturating the network transmission link should this be re-

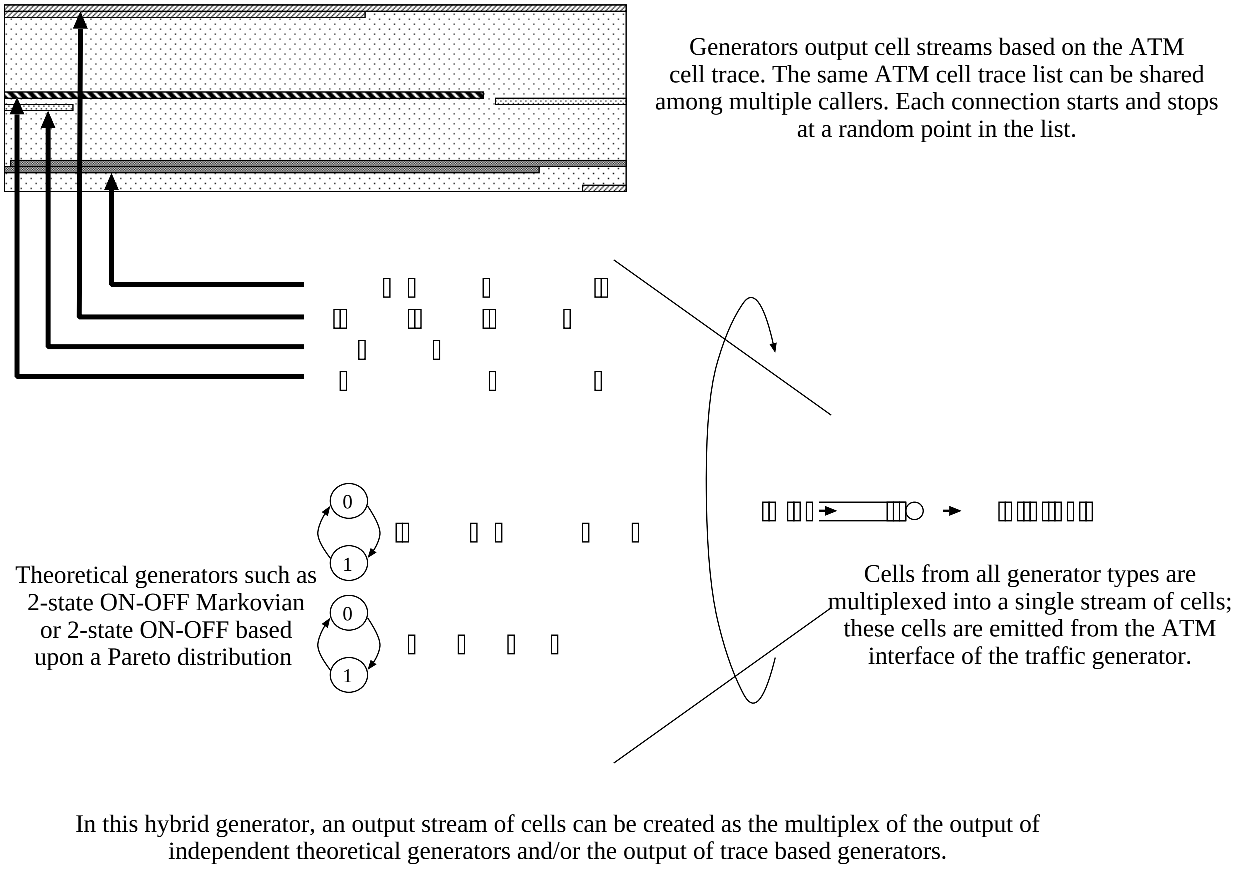

Generators output cell streams based on the ATM cell trace. The same ATM cell trace list can be shared among multiple callers. Each connection starts and stops at a random point in the list.
0
1
Theoretical generators such as 2-state ON-OFF Markovian or 2-state ON-OFF based upon a Pareto distribution
0
1
Cells from all generator types are multiplexed into a single stream of cells; these cells are emitted from the ATM interface of the traffic generator.
In this hybrid generator, an output stream of cells can be created as the multiplex of the output of independent theoretical generators and/or the output of trace based generators.

Figure 3.7: Hybrid traffic generator

quired and has the ability to combine a virtually unlimited number of traffic types derived from either deterministic models or network packet traces.

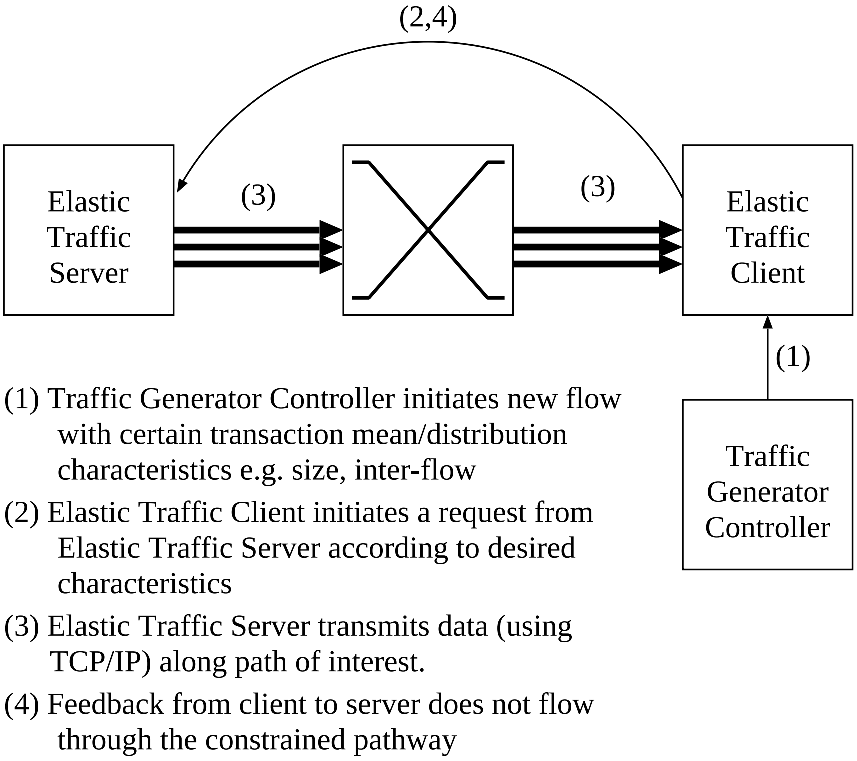
3.3.3.2 Elastic sources

Elastic sources present a very different and challenging problem in the creation of a traffic generator. Transport protocols such as TCP adapt to the behaviour of the network. As a result these protocols require a feedback path to allow this adaptation to take place. Additionally, because the traffic depends upon the behaviour of the network it does not lend itself to the creation of a source-model, as was discussed in Section 2.1.2.

The TCP traffic model used in this study was based upon the `ttcp` [Stine90] utility around which a client and server were created. Figure 3.8 illustrates how the server generated traffic after 'requests' were transmitted to it by the client. Following the recommendations of [Allman99] which documents the effective evaluation of TCP/IP, one objective was to introduce as few modifications as was possible to permit a close replication of the behaviour of current implementations of TCP/IP traffic. This included using a recent version of the TCP protocol stack that incorporated the most-recent updates to the protocol, including such facilities as slow-start congestion avoidance, fast retransmit, fast recovery, selective acknowledgement, the Nagle algorithm and the support of delayed acknowledgement in the receiver.

The client is able to hold open up to 1024 TCP flows (on a specially configured kernel) although this increase in capacity was the only major modification to an otherwise standard Linux revision 2.2.9 system. While the call generator (Section 3.3.5) is responsible for the characteristics of a total session duration, each session duration can model different traffic behaviour. A continual request of constant size files emulates the original behaviour of `ttcp` as well as providing a useful performance tool. However, the most useful ability is to recreate WWW behaviour.

The TCP/IP elastic traffic flow responds to loss at buffers using timeouts that occur within a RTT. As a result, elastic generators are sensitive to the configuration of a network: e.g. the RTT of each link, the MTU in the network and how the elastic traffic generator discovers the network MTU. Any differences in these values between experiments will change the behaviour of the generator. Thus, for valid comparisons, it is important that, along with the traffic characteristics,



(1) Traffic Generator Controller initiates new flow with certain transaction mean/distribution characteristics e.g. size, inter-flow

(2) Elastic Traffic Client initiates a request from Elastic Traffic Server according to desired characteristics

(3) Elastic Traffic Server transmits data (using TCP/IP) along path of interest.

(4) Feedback from client to server does not flow through the constrained pathway

Figure 3.8: Traffic generator for TCP/IP flows.

these parameters are also held constant across different experiments.

The precise configuration of the elastic-traffic client and generator used an MTU of 1500 bytes: a common MTU used for Ethernet networks of 10 Mbps and 100 Mbps per second speeds. A drawback of the current configuration is that, as suggested by Figure 3.8, all flows use the same set of paths which results in a common RTT across all flows. Potential mechanisms for improving this situation include the use of programmed delays into the TCP/IP stack in server and client, although such an approach is in contrast with the objective of staying as close to the original current implementation of the protocol as possible. An alternative, not implemented here but held as an improvement for the future, is the use of variable length routes for each flow. Such a scheme is made possible because the test environment uses a pre-configured virtual path for each flow and each path could be made to transit through an arbitrary pathway configured at construction to emulate the multiplexing of a range of different stacks using different RTT values.

One final drawback of the current generator is that it cannot encapsulate the heterogeneity of TCP implementations present in the Internet. [Mortier00, Figure 8], illustrates the differences between implementations: tabulating the retry timer value used in a range of computer systems. An example of many fundamental differences, implementations of the same protocol using different retry timer characteristics will lead to differing behaviour between implementations. The elastic traffic generator described here will not replicate such a wide range of implementation configurations being based upon a single version of the TCP/IP stack. To fully replicate network traffic resulting from a variety of implementations without using a large range of implementations as traffic generators, traffic derived from captured traces will continue to remain a useful alternative.

3.3.4 Traffic generator controller

The traffic generator controller, a process running under UNIX, instructs the traffic generator to start and stop individual traffic sources representing each flow as these flows are setup and pulled-down. The traffic generator controller, like the measurement controller, interfaces between the RPC interface that is used between itself and the AC system and the purpose-built inter-machine protocol used by the traffic controller.

The generator for inelastic traffic of Section 3.3.3.2 incorporates the traffic generator controller into the client side of that traffic generator. However, although the two components are merged, the overall structure and relationship between traffic generator controller and traffic generator may be considered the same in both cases.

3.3.5 Flow Generator

The flow generator, a process running under UNIX, initiates new flow attempts into the AC test-environment. Each flow-attempt will be made by passing the parameters of the new flow to the admission controller. The flow generator can create a variety of different combinations of flows, specified by the arrival rate of new flow attempts, the flow holding time and the traffic each flow attempt will carry. The inter-arrival rate of flow attempts and flow holding time can have constant values or be based upon a distribution — for example, the period over which a flow will be in progress may have an exponential distribution with a given mean. The type of traffic each flow will carry is specified only once for a set of flow attempts, but the flow generator is able to generate any number of combinations of traffic types, each with independent arrival-rate and flow-holding times.

3.3.6 Admission Controller

The admission controller forms the core of the AC test environment. The AC component has the capability to change the AC algorithm as required. Only one algorithm is in place during any experiment, although consecutive experiments can operate with only the AC algorithm itself or the control parameters of any particular algorithm being changed.

During the generation of new flows, the traffic type and the parameters that describe traffic that the flow will carry are declared to the current AC algorithm. The parameters of each new flow can be specified in any of the TM 4.0 parameter formats [ATMF95], although it would be straightforward to add RSVP-compliant [Braden97] formats. Each new flow presents its parameters to the AC system and, if accepted, requests a flow be sent across the switch.

Each AC algorithm obtains the required measurements from the measurement controller as part of that particular AC algorithm's decision process. Each algorithm can obtain the measurements of the type and format it requires. For example, in the case of a simple threshold AC algorithm the measurements are of instantaneous line utilisation while for an allocation based upon the peak rate declared by a new flow no measurements are required at all.

3.3.7 Packet time-frame scaling

Section 3.3.1 introduced the ATM switch used in the test environment. In that section it is noted that the rates of traffic sources are slowed by a factor of $1/D$, in fact, this factor is a multiplier of the time between cells. As a result, the passage of time on the network, and hence the passage of time in the experiment as a whole, has been reduced by the factor $1/D$. There is a drawback to this system — experiments run D ‘times’ longer because all operating values of time in the system have been scaled up by D. For example a connection with a holding time of 10 seconds will, from the experimenter’s perspective, have a duration of $10D$ seconds. Such a system of scaling has the immediate effect of allowing all parts of the test-environment to do more processing for the each cell, thereby giving D times the amount of time to do processing required per cell (and per connection and per experiment). Such extra time is important when the switch fabric is required to perform numerous timing and counting operations on receipt of each cell. The technique of time scaling has been used successfully in several projects, most significantly by [Crosby95].

This particular effect is important to incorporate into mechanisms that rely upon the passage of time, such as traffic generators where the video codec must also account for the passage of time being scaled by D. The TCP traffic generators, reliant upon timing of RTT as well as a number of time-out mechanisms also required scaling by D. These changes have been incorporated transparently at the generators and all other components of the test environment as required.

Throughout this document all times stated for test-environment performance, flow setup, flow holding periods, measurement period, and any other time frame in the experiment are given in unscaled time; that is time that has not been multiplied by D. Using measurements on this time scale makes reported experimental results directly comparable with measurements made on other systems.

3.4 Test environment operation

In the evaluation of AC algorithms, the system works as follows: a flow generator is responsible for 'generating' flows according to some distribution or from a previously collected trace of measured arrivals. New flows may be of various types and each flow may, according to a random distribution, determine its flow type and any set of parameters (such as sustained rate or peak rate) which it is required to present. New flows, once generated, present their parameters to the AC decision system. The currently loaded AC algorithm, using measurements from the switch, makes a decision as to whether or not to admit the flow. Only one AC algorithm operates in any one experiment.

If a flow is admitted, the AC algorithm will reply to the flow generator accepting the flow. The flow generator then instructs the traffic generator controller to set-up a new traffic source with the appropriate parameters. The traffic generator controller then starts the new flow by instructing the traffic generator to create and start a traffic source with the correct parameters. The cells of this new flow will then enter the multiplex of streams of cells that the traffic generator is transmitting into the switch. In addition to its traffic type and arrival time each new flow has associated with it a lifetime, or flow holding time. This flow lifetime, like the arrival time, can be drawn from a theoretical distribution or from a trace driven set of values. Once the flow holding time is reached the flow's traffic source is stopped and that flow is 'cleared down'.

It is important to emphasise that in this set-up there is no real network signalling, all network paths — in this case ATM VP/VC pathways — that will be required are setup as permanent circuits prior to the experiment. The processes running off-switch assume the full load of the 'signalling' and therefore it is possible to emulate the arrival of flows at rates far higher than could be sustained by any actual ATM signalling implementation.

The logging system in the AC test environment gives sufficient information that real-time graphs can be produced displaying information such as the current line utilisation, the number of flows currently in progress and statistics regarding flow acceptance or rejection. Figure 3.9 shows the plot of 100 seconds of time from the start of an experiment. The x axis in all cases is time, shown in seconds, since the start of the experiment. The top graph shows the flow arrival process. For each flow which arrives a vertical bar is drawn. In the event that an arriving flow was accepted

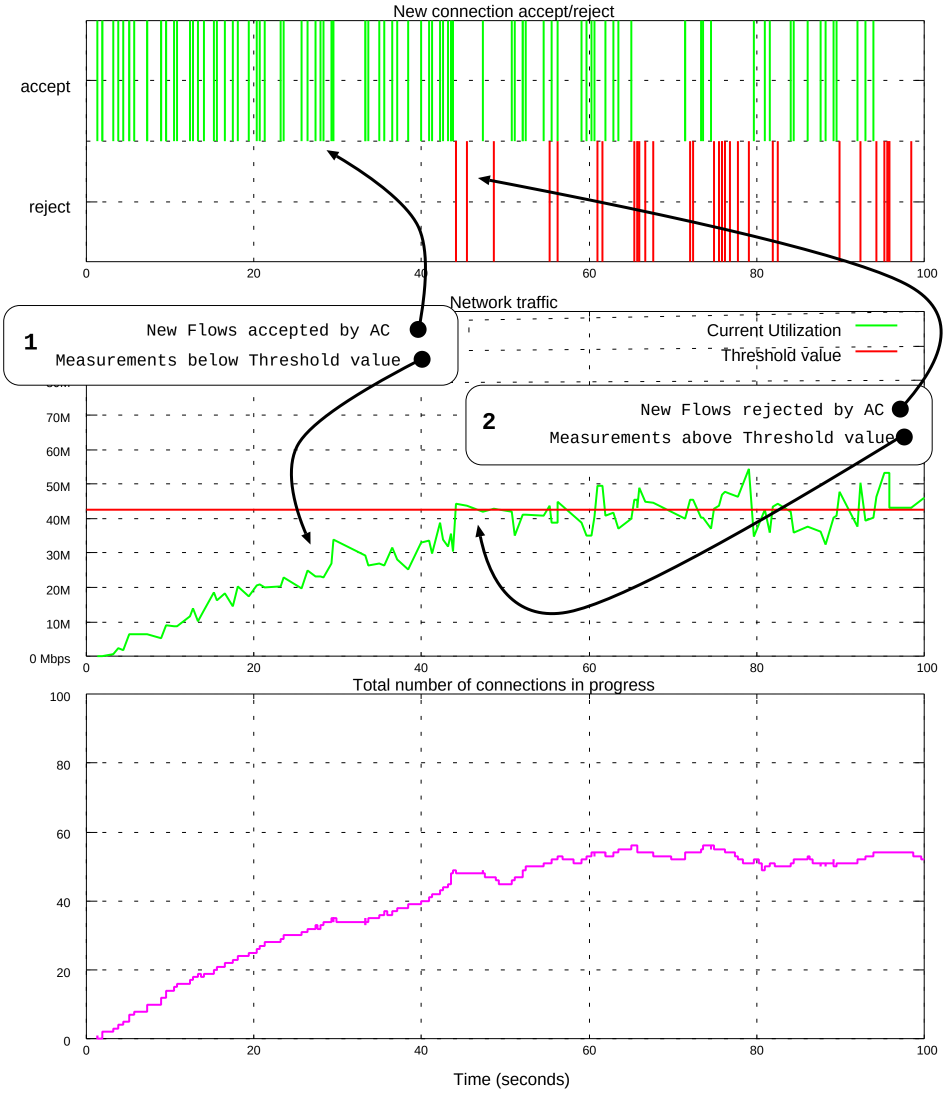


Figure 3.9: The first 100 seconds of operation

by the AC algorithm, a vertical bar is drawn extending upwards. If the flow is rejected, then the vertical bar extends downwards. In the leftmost part of Figure 3.9, where the time is less than 45 seconds, no new flows have been rejected because measurements of the instantaneous line utilisation are less than or equal to the thresholding value. However, after 45 seconds, sufficient traffic is now in the system for the instantaneous line utilisation to be above the thresholding value and as a result flows are rejected.

The second graph from the top is a display of the traffic dynamics in the switch, measured in real time. The thresholding value in use by the AC algorithm is shown along with the current measure of instantaneous line utilisation. The third graph of Figure 3.9 shows the number of flows in progress in the system, over time. This climbs rapidly, as new flows enter the system at a greater rate than they clear down. This is because in the empty system (at time zero) no flows are rejected. Once rejections occur, the number of flows in progress stabilises but displays the expected variation due to statistical fluctuations. In the experiment illustrated[3] flows carried the deterministic traffic source, TP10S1: a traffic source with a sustained rate of 1 Mbps and a peak rate of 10 Mbps and described further in Section 2.2.1. The mean flow attempt rate was 10 flows per second with a mean flow lifetime of 10 seconds. These output graphs are regularly used to check that the environment is operating correctly and to allow simple comparisons of consecutive experiments; in addition to the parameters of utilisation and flow behaviour, algorithm specific information, such as threshold values, can also be output.

3.5 Test Environment Evaluation

The evaluation reported in this section discusses four topics. Firstly, the reliability of the traffic generator and then each of the stability, performance, and repeatability of the complete test environment.

Section 3.5.1 examines the level of jitter introduced by the traffic generator. Section 3.5.2 discusses the issues of experiment run times, experiment stability and detection of the start-up transition period. Section 3.5.3 covers the flow setup performance of the test environment, and

[3] The threshold used was 42.6 Mbps, a value computed using the work of Key [Key95] to give a loss-ratio of 1×10^{-3} for the given traffic and network conditions.

Section 3.5.4 reports results conducted to assess the repeatability of experiments on the AC test environment.

3.5.1 Traffic Generator

The ability of the inelastic traffic generator to create a stream of traffic with precise characteristics was examined and the results are presented here. Analysing the generator using an isochronous stream of traffic paced, a minimum of jitter is desired. The recorded stream presented some jitter between cells. Table 3.1 compares with the theoretical cell spacing the statistics collected for a stream of traffic emitted from the traffic generator at a data rate of 1 Mbps. The jitter evident in these values is within specification for the jitter due to the SONET framing of ATM cells in the OC-3 standard.

Theoretical	Mean	Variance	Standard Deviation	95 % Confidence Interval of Mean
4.403×10^{-4}	4.403×10^{-4}	2.228×10^{-12}	1.493×10^{-6}	4.559×10^{-9}

Table 3.1: Time, in seconds, between consecutive test stream cells.

In addition to tests using single isochronous traffic streams, tests were conducted where additional traffic load was required from the traffic generator; at the same time a test stream restricted to 1 Mbps was also transmitted and compared. Apart from the SONET framing jitter, no discernible deviation was found when comparing the outcomes. These results gave confidence in the manner in which cell streams were created and transmitted by the traffic generator.

3.5.2 Run length and initial stability

For experiments made using this AC test environment, there is an initial period of instability before the system returns consistent results. Figure 3.9, (Page 88,) shows clearly how an experiment has an initial transition period before its operation has settled. In this case there is a slow ramp-up to a value of flows in progress that is then held relatively constant. Such initially unstable periods are quite common to steady-state simulation work and as a result an approach can be drawn from work in that field. [Pawlikowski90] gives a number of methods for determining at what point an experiment has become stable. A combination of two of these algorithms were used. Firstly, waiting for the longest cycle in the system to have been executed 4 times would provide an initial approximation. Secondly, the period for stabilisation of the variance of the number of flows in progress provided a second mechanism to ensure the initial period of instability had past. The point of stability was determined in off-line processing; once detected, data collected up until this point was discarded. Fortunately, the contribution of the period of instability to the the length of an experiment is small, the main factor deciding the length of experiments remained the need to collect a representative samples of events.

While QoS may describe any number of parameters such as packet delay or the acceptance ratio of new flow attempts, the QoS experienced by any particular flow or by the system overall can be expressed as the packet loss-ratio. [Lewis98] advises that other QoS values are directly derivable from the packet loss ratio experienced for a known buffer size thus it is for this measure that parameters of experiment duration are configured.

The computation of packet loss-ratio presented a special problem. Calculating the packet loss-ratio requires accurate, synchronised counts of the packets transmitted into the multiplexing buffer and those emitted by the buffer. Not only are there complications introduced by the need for synchronous measurement, but cells still in the buffer may not be accounted for in the measurements and mis-interpreted as loss. A solution for the global-counting of loss-ratios was to not tally the final loss in an experiment until a minimum period after which the traffic generators have been shutdown has passed. While this solution works well for the total packet-loss for a completed experiment, this approach is not useful to remove error in the loss-ratio's of individual flows.

The solution lay in a mechanism to combine flow-by-flow cell counting with a rotation of the current flow in progress among different network paths. Each flow is transported on its own network path (its own VP/VC in ATM parlance.) When a flow ends, that particular pathway is not available for use until a given period p has passed. Period p is selected to allow time for the traffic generator to stop and for all packets associated with that flow to pass through the incoming packet counter, the buffer and the outgoing packet counter. While such an approach limits the number of flows that may be in progress concurrently (by limiting reuse of a flow's pathway until all packets have been counted) the advantage is an accurate packet count and loss-ratio for each

Loss-ratio	99%	95%
	Confidence Interval	
1×10^{-5}	8.0%	6.2%
1×10^{-4}	2.6%	2.0%
1×10^{-3}	0.8%	0.6%

Table 3.2: Predictions of error margin for loss-ratio results

flow.

The packet-loss ratio serves as a useful QoS measure for the comparison of one AC experiment with another. Because of this the selection of the run-length period was a crucial variable impacting upon the reliability of the observed results of packet loss ratio as well as other measured properties such as the flow-acceptance rate. One alternative to ensure a constant error margin for all experiments would be to run every experiment for sufficient time to establish an accurate packet loss-ratio regardless of the loss-ratio desired. An experiment which experiences a loss ratio of 1×10^{-5} would run 100 times longer than an experiment that experiences a loss ratio of 1×10^{-3}. This would ensure that the error in measurement was maintained independently of the experiment parameters. However, such a process is quite difficult to achieve, since experiments generally will have an unknown loss-ratio when originally configured.

Instead of adapting the length of experiments for each expected outcome, a constant run-time was chosen for all experiments. However, the run-length must be sufficiently large to ensure that the error margin is small enough for the smallest value of loss-ratio. Computing the error margin for different packet loss-ratios with a constant experiment length gives the results of Table 3.2.

While the test-environment has no specific upper boundary on the number of cells to be transmitted, an upper bound of 1×10^{8} packets was used in early experiments that concentrated upon loss-ratios of 1×10^{-3}. Table 3.2 indicates that with 99% confidence this was sufficient packets to achieve an error margin of less than 1%. However, for a loss-ratio of 1×10^{-5} with the same level of confidence, (99%,) the error margin will be 8%.

In order to improve the error margin, two solutions are available. Firstly, to increase the running time of experiments: an approach not practical in the circumstances. An experiment transferring 1×10^8 cells has a running time of approximately 2 hours. A significant improvement in the error margin would require experiments with running times two magnitudes higher, 1×10^{10} cells. This would achieve an error margin of less than 1% with a confidence interval of 99% for loss-ratios down to 1×10^{-5}. Each experiment running for 200 hours was not practical, precluding any opportunity to make comparisons between many different experiments configurations.

The second solution is to accept a lower confidence interval for a constant running time. Table 3.2 illustrates that with a confidence of 95%, the error is near to 6% for all loss-ratios down to 1×10^{-5}. With increasing experiments being too time-consuming, a lower level of confidence and higher error margin were used alongside a constant experiment parameter of 1×10^8 cells transferred.

Table 3.2 also hints at another important phenomenon in the compiled loss-ratio results: if the number of events (packets) is held constant, the error-margin increases as the loss-ratio decreases. This implies that the variance among the loss-ratio values for experiments will increase as the number of loss-events is decreased for a constant number of cells in any given experiment. Such a phenomenon is important to consider when analysing any figure with the loss-ratio given as one variable, such as line utilisation versus packet loss-ratio.

3.5.3 Performance

When a flow is entered into the system an assumption is that there is a negligible amount of time between when the new flow has been generated and, assuming acceptance, when cells transmitted by the corresponding traffic generator will start entering the data stream. This assumption is not valid in anything other than a theoretical test structure. However, it is important to quantify and, where possible, overcome such a delay between a new flow entering the system and cells being produced by the system so as to minimise the impact of experimental effects being introduced into the test experiments. In this way theoretical results and experimental results can be compared more closely.

One performance goal in the construction of the AC test environment was to reduce the new flow generation, new flow test and new flow start-up delays to a minimum. In a naïve simulation, the authors may ignore such values: treating them as zero. Although, when compared to a real-world implementation, aiming for a delay of zero could be seen as unnecessary — several authors [Battou96, Niehaus97] noting that in commercial ATM switch systems the flow setup process for a new flow can take 20–200 ms. Such a quoted value for the delay does not include the additional time required for the end-system to become active. In the work to reduce the delay in the AC test environment, the delay period is measured from the generation of the new flow request to the moment cells are emitted from the interface of the traffic generators.

The delays in the pathway between the generation of a new flow request and the emission of cells into the network switch take several forms: firstly there is the time taken in the execution of code on the various machines that the AC test environment runs; secondly there are delays related to the communication between components of the AC test environment; and finally there are delays in the traffic generator that will cause a delay between the starting of traffic generators and the emission of cells from the network interface and into the network switch.

Following improvement and optimisation, final experiments on the test environment established that delays between a new flow arrival and the start of transmission of its cells from the corresponding generator has a lower bound of 8.38 ms. The statistics and distribution of this delay are shown in Table 3.3 and Figure 3.10 respectively.

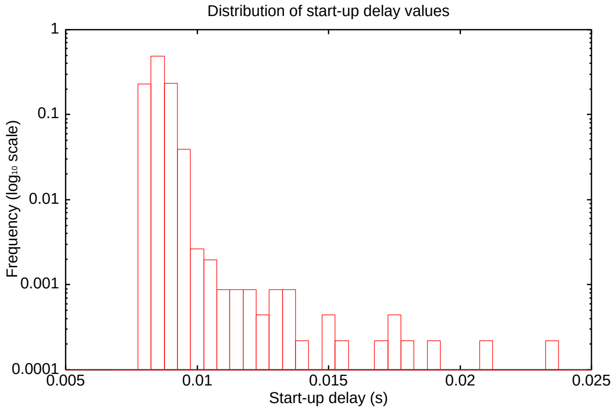


Figure 3.10: Distribution of start-up delay values.

This delay affects consecutive flow attempts: flows may be delayed if attempted within less than 8.38 ms of each other. This implies that, in the best case, flows cannot be attempted and

Mean	Variance
8.80×10^{-3}	1.67×10^{-6}

Table 3.3: Start-up delay values (seconds).

started at a rate any faster than ~ 119 flows per second. For an experiment with an exponentially distributed flow arrival rate and a mean of 10 flows per second, it can be predicted with 95% confidence that 0.3% of flows may be affected.

Figure 3.10 indicates that the startup for these results has a worst case value of nearer 24 ms. For such a delay-boundary flows would not be able to be started at a rate any faster than ~ 42 flows per second. For the same experiment as above: an exponentially distributed flow arrival rate and a mean of 10 flows per second, with 95% confidence, 1.6% of flows may be delayed for this worst case value.

For both of these extremes of start-up delay value: 8.38 ms and 24 ms, the estimates of affected flows are pessimistic. This pessimism arises because these estimates are derived for experiments where every flow attempt is started. It would be unusual for every flow to be accepted in an AC experiment; every flow that was not accepted would only incur a small part of the total delay overhead. When compared with the earlier stated values for commercial flow setup of 20–200 ms [Battou96, Niehaus97] — a mean flow setup period of less than 9 ms is excellent and even 24 ms is acceptable. In real terms this means that, in the hypothetical worst: where all flows are admitted, in an experiment of 6000 flows, with 95% certainty, up to 96 of those flows will be adversely affected by system delays.

3.5.4 Repeatability

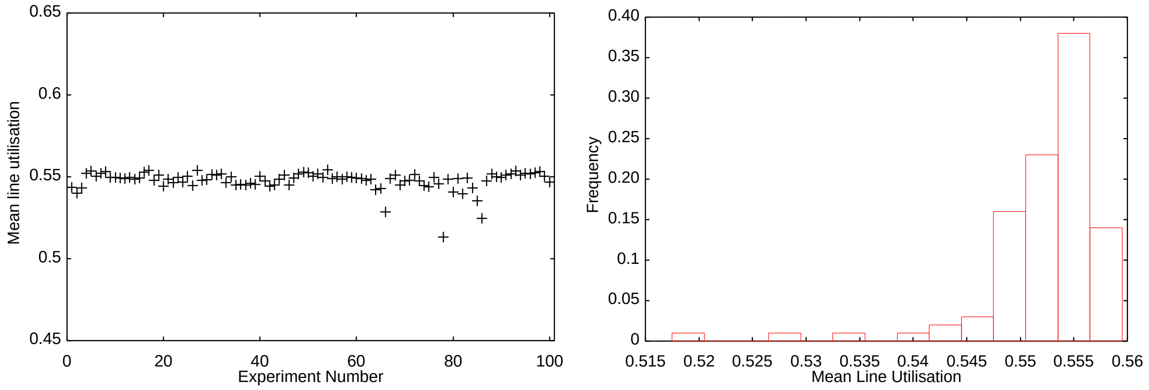
The previous section discussed how the influence that the test environment has on consecutive flows must be kept to a minimum. In order to reliably compare and contrast different AC algorithms, the impact the whole environment has upon consecutive experiments must also be kept constant. Identical experiments ought to yield identical results. However, because the behaviour and control, along with the measured criteria are statistical in nature, identical results are not realistic for the environment outlined in this chapter. Instead, a minimum level of variance between experiments is sought, a value approaching the variance induced by the finite run length upon such measurements as the packet loss-ratio.

Evaluating the repeatability error between experiments is made using a set of experiments where the input parameters were held as constants. The input traffic consists of two differing flows types: each carrying a video traffic stream of the type described in Section 2.2.5. By using such traffic the test environment was placed in conditions of actual use, using two representative, real traffic sources. One video source, restricted to a peak-rate of 10 Mbps and with a sustained-rate of 1 Mbps was carried on flows that had a mean arrival-rate of 5 flows per second and a mean holding-time of 10 seconds per flow; the other traffic type, with a peak-rate of 5 Mbps and an sustained-rate of 2 Mbps, was being carried on flows that had an mean arrival-rate of 5 flows per second and a mean holding-time of 5 seconds per flow. This experiment was representative of initial investigations and while such a configuration is not used for later comparison work the results were considered representative of the conditions under which the test environment would be placed for this dissertation.[4]

During evaluation of AC algorithms, the overall loss-ratio and mean line utilisation of an experiment are significant comparison criteria. Hence, it was these results that were commonly compared between experiment runs. Figure 3.11(a) shows the mean line utilisation values for the batch of 100 identical experiments. A statistical summary of this collection of results is in Table 3.4 and the distribution of the results is shown in Figure 3.11(b). In comparison, Figure 3.12(a) shows the loss-ratios values for the batch of 100 identical experiments. The statistics of this collection of results is in Table 3.5 and the distribution of the results is shown in Figure 3.12(b).

It is clear that even for experiments with a narrow distribution of mean line utilisation, the values for cell loss ratio have a much higher variance. This will mean that with a 95% confidence the link utilisation value will have an error of $\pm 0.21\%$. While, with a 95% confidence, the results for packet loss ratio will give experimental results with an error of $\pm 4.6\%$. Such error-margins for

[4] In order to achieve a packet loss ratio of 1×10^{-3} a threshold value used in the AC algorithm was 61 Mbps. This value was computed from a set of experiments, to determine the relationship between threshold and loss, not presented here.



(a) Mean line utilisation (MLU) values

(b) Distribution of MLU values

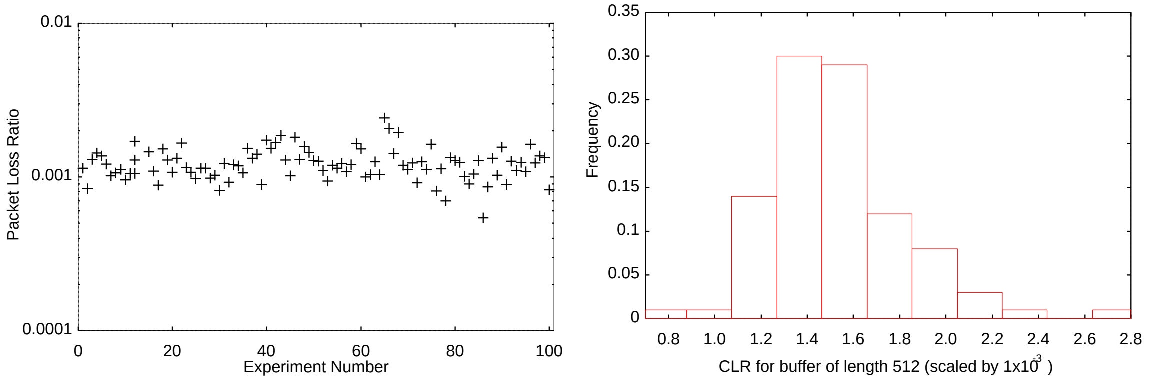
Figure 3.11: Mean line utilisation repeatability test results.

Mean	Coefficient of Variation	95% Confidence Interval
0.55	1.1	1.1×10^{-3}

Table 3.4: Statistical information on the 100 mean line utilisation results shown in Figure 3.11(a).

Mean	Coefficient of Variation	95% Confidence Interval
1.2×10^{-3}	24.3	5.9×10^{-5}

Table 3.5: Statistical information of 100 packet loss-ratio results shown in Figure 3.12(a).



(a) Loss Ratio values

(b) Distribution of Loss Ratios values

Figure 3.12: Packet loss ratio repeatability test results.

the packet loss ratio are well-within the constraints predicted for the length of the experiment, as discussed in Section 3.5.2.

A number of identical experiments were repeated to appraise the accuracy of the test environment. It is difficult to know if the results obtained suggest a sufficiently accurate test-environment as these are not common and those that do exist are not easily comparable. However, having an appreciation of the error-bounds of the measured values aids considerably in the interpretation of results presented by the test environment. Additionally, there are few avenues for improving these results without substantial changes to the experiment parameters or the architecture in use.

3.6 Summary

The environment presented in this chapter is able to evaluate an AC algorithm, allowing comparison of the same algorithm under changing conditions and allows the comparison of different AC algorithms under constant conditions. Unlike simulations, this test environment is able to use real implementations of source traffic and places the AC implementation under restrictions, e.g. finite memory and processing, found within an actual end-system, thereby giving an unparalleled fidelity with real implementations not possible with simulations. Section 3.2 presented a number of related approaches towards network investigation, and AC in particular, providing support for the approach of an implementation-based test-environment taken here.

In Section 3.3, this chapter has illustrated the modular structure of the environment, noting how each individual component such as the AC algorithm or the traffic generator may be configured specifically for each experiment. Section 3.4 shows how in operation this environment provides useful logging data allowing the off-line interpretation of results as well as real-time plotting of current experiments.

Lastly, Section 3.5 presented an evaluation which, firstly, indicated the reliable operation of the purpose-built traffic generators. Secondly, the evaluation reported the predicted error associated with the finite run lengths of experiments along with the mechanisms used to remove initial transient periods. Thirdly, overall performance limitations of the test environment were presented. Finally, the error margins from experiments conducted were measured to evaluate the environments repeatability.

While the use of a simulator offers a widely available technique for the testing of AC algorithms, the environment described here allows comparison of AC implementations constrained in the same manner that real rather than simulated implementations are constrained. By providing a faithfulness to real implementations the test environment presented here has an important role to play in the assessment of AC algorithms, not necessarily as an outright replacement for the approaches of either theoretical static-solutions or of simulators but as an equally important method of approach.

In addition to being used in the evaluation and comparison of MBAC algorithms presented in Chapter 5, this environment is able to be usefully adapted for the experiments of Chapter 6. Each of those chapters give a brief prećis of the manner in which the test environment is used.

Chapter 4

Measurement-Based Admission Control Algorithms

4.1 Introduction

Admission Control (AC) is a mechanism for traffic management, which consists of admitting a new traffic source if and only if the network can guarantee QoS to the new flow while still supporting existing QoS guarantees to sources already accepted. An AC procedure is employed to maintain a high utilisation of network resources while preserving the QoS of existing flows. It does this by balancing higher network utilisation through increased multiplexing against the satisfaction of QoS for existing clients. Such an AC scheme relies on being able to accurately establish the resource requirements of current flows, along with a prediction of the impact a new flow will have upon existing traffic sources. Commonly an AC scheme requires that a new flow declares parameters that can be used to calculate its resource requirements and, therefore, its impact on pre-existing flows.

The parameters used to describe traffic sources are typically the peak rate, the tolerance to jitter between packets, the sustained (mean) rate, and mean or maximum burst size. Using these parameters an AC algorithm may compute the impact a new flow will have on a network. However, the difficulty for users in predicting these values, particularly mean-rate parameters, often leads to overestimation of the impact and consequently to network under-utilisation. An alternative solution is to use traffic measurement as an input to the AC procedure. MBAC methods may differ in which parameters are measured, (e.g. peak rate, mean rate, or variance), how the parameters are recorded, (e.g. per link, per class or per flow), and by which parameters, if any, are required for each new flow.

MBAC algorithms have enjoyed considerable interest as an approach to overcoming some of the shortcomings of traditional AC algorithms. AC algorithms have long been of interest in the management of fixed-load and integrated services such as voice and video. In the Internet community, interest in ACs has, until recently, been limited to the controlled-load services of IP under Integrated Services (INTSERV) [Braden94, Wroclawski97]. However, more recent work on admission and control in networks based upon differentiated-services (DIFFSERV) [Nichols98, Blake98, Nichols99], at network ingress [Stoica99] and egress [Cetinkaya00, Schlembach00], has seen a resurgence of work in MBAC techniques.

MBAC algorithms are seen as an appropriate approach for the AC problem as they require minimal characterisation, and have the ability to adapt to the changing requirements of in progress flows. As indicated in Section 2.3.4, there are a significant number of MBAC algorithms; these result from work based upon a wide variety of theoretical foundations, different system requirements, differing policies controlling the admission process and thus different behaviour requirements.

This chapter presents a number of different MBAC algorithms, noting the fundamental premise upon which they are based and comparing their algorithmic structures. It examines MBAC algorithms both as a complete unit, as proposed by their authors, and as a composition of an algorithm's policy and estimation components. By adopting this approach the relationship between the policy and estimator can be shown more clearly. A useful side-effect of this approach is that it also makes clear the common elements among different MBAC algorithms.

4.1.1 Approach

It is a useful technique to group related MBAC algorithms together. The primary decomposition approach taken here separates each complete MBAC algorithm into two constituent parts: estimator and policy. In addition algorithms that are based upon the CE principle have related sets of architectural assumptions and may be grouped together. A third cataloguing, the taxonomy of [Shiomoto99] is used in the summary although, for reasons that are made clear in that section, it proves less-useful.

The decomposition approach used in this chapter has been adopted successfully by previous authors, notably [Jamin97a] and [Tse99]. Analysing the MBAC algorithm as comprising units of estimator and policy allows common as well as unique elements of each algorithm to be emphasised. Additionally, as noted by authors who have previously used this technique, differences in performance and behaviour can be tied to the estimation process, the decision process, or a combination of both.

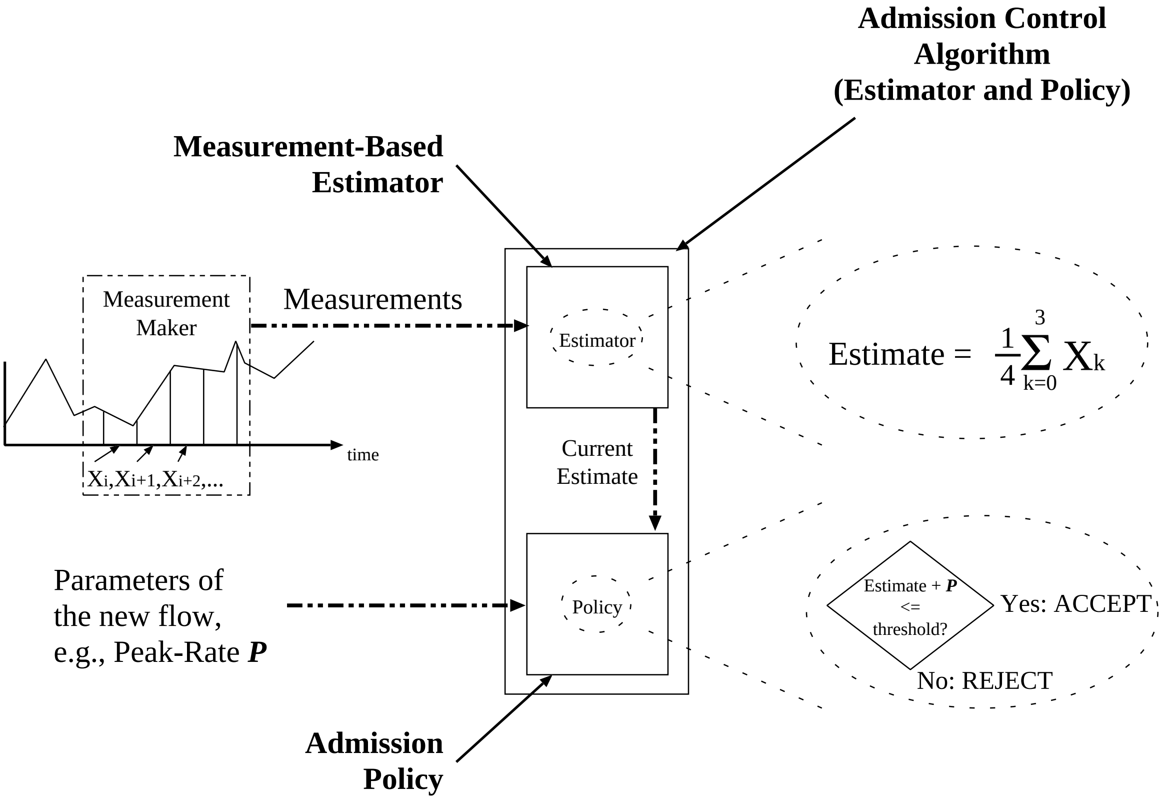

Admission Control Algorithm (Estimator and Policy)
Measurement-Based Estimator
Measurement Maker
Measurements
time
Xi,Xi+1,Xi+2,...
Estimator
Estimate = $\frac{1}{4}\sum_{k=0}^{3} X_k$
Current Estimate
Parameters of the new flow, e.g., Peak-Rate ***P***
Policy
Estimate + ***P*** <= threshold?
Yes: ACCEPT
No: REJECT
Admission Policy

Figure 4.1: Separation of AC into estimator and policy.

Figure 4.1 illustrates how an AC algorithm may be divided into an MBE and an admission policy component. Not every AC algorithm has an estimator but all AC algorithms must have a policy. The policy of an algorithm is the procedure to follow at flow admission, whereas the role of estimator is to supply information for the use of the policy (procedure.)

Figure 4.1 illustrates the combination of a pessimistic policy that uses the declared peak-rate of the new flow, combined with an estimator that computes the line utilisation from the mean of the past four samples. This MBE is a simple estimator that relies only upon the regular sampling of line utilisation. The MBE computes an estimate of the current utilisation which is then used by the admission procedure. The example illustrates a policy that combines the estimate of the line activity with a prediction of the impact of the new flow. If the combination of these values is less than a thresholding value, the flow is admitted. From this illustration it becomes clear that, in this case, the estimator and the policy components are quite separate: the estimator may be more (or less) complicated, as may the admission policy.

Following the same lines of decomposition into estimators and policy-implementations, Section 4.2 presents the approaches to estimation used by each MBAC algorithm of this study, while admission policies are examined in Section 4.3. Finally, Section 4.4 presents the complete algorithms that result from a combination of estimator and policy components.

4.2 Estimators

In order for AC algorithms to be able to maintain a level of service or a guarantee of QoS, the algorithm must have available to it an estimate of current resource requirements, typically bandwidth requirements. In an estimate of the bandwidth, demand may be based upon predictive traffic models, measurements, or a combination of both. Those based in some part upon measurements are of interest for this study, although several AC algorithms based upon predictive models are also introduced, for contrast and comparison.

[Kelly96] provides a formal definition of *effective bandwidth*. Such a definition serves to capture the subtleties of the traffic multiplex and network QoS constraints in combination with the buffer and service-capacity resources made available by the network. [Kelly96] may be interpreted to provide an *effective bandwidth* of any individual traffic source defined as the total bandwidth required to satisfy the QoS constraints of the total traffic multiplex for a given buffer resource when divided among the number of traffic sources present in the multiplex. As it applies to a single source, it is this definition of *effective bandwidth* that is used throughout this dissertation.

The original authors' notation has been translated into a common notation used throughout this dissertation. While making reference to the original papers more difficult, such a translation allows easy comparison between algorithms. The list of notation is summarised in the glossary given on Page 17.

4.2.1 E-IU — Instantaneous Utilisation

The Instantaneous Utilisation estimator bases the estimate of *effective bandwidth* upon a recent measurement of line use. While such a simple estimator presents several difficulties, (e.g. what value to make the length of the measurement period), it is robust due to its simplicity. The 'length of measurement' problem is illustrated in Figure 4.2; the period over which a measurement is taken may have a considerable effect upon the computed *effective bandwidth* estimate. The reason for this is that the measurement period directly controls the characterisation of the traffic by the estimator: measurements taken over long periods remove any short-term variance while providing insight into the long-term behaviour, while estimates taken over short periods will reflect the short-term burstiness at the expense of providing any long-term stability in the results.

If the activity on a link over the interval τ is represented as $X[s, s+\tau]$ then $\frac{X[s,s+\tau]}{\tau}$ is the rate over that particular period at time s. Thus the E-IU estimator is,

$$E = \frac{X[s, s+\tau]}{\tau}. \tag{4.1}$$

The role played by the measurement period, τ, means that this parameter is fundamental in all the MBE discussed. In the case of the Instantaneous Utilisation estimator it is the only control parameter.

4.2.2 E-CB — *Chernoff Bounds*

E-CB is an estimator based upon *Chernoff Bounds*. [Chernoff52] proposed techniques for the bounding of tail probabilities of the sums of independent random variables. These techniques may be applied to the curve of *effective bandwidth* versus mean rate for a traffic source.

[Gibbens97] present several estimators, each based upon *Chernoff Bounds* and each estimating the bound using different information about the curve of *effective bandwidth* versus mean rate. [Floyd96] discusses an MBAC algorithm based upon the Hoeffding bound, (of [Hoeffding63]).

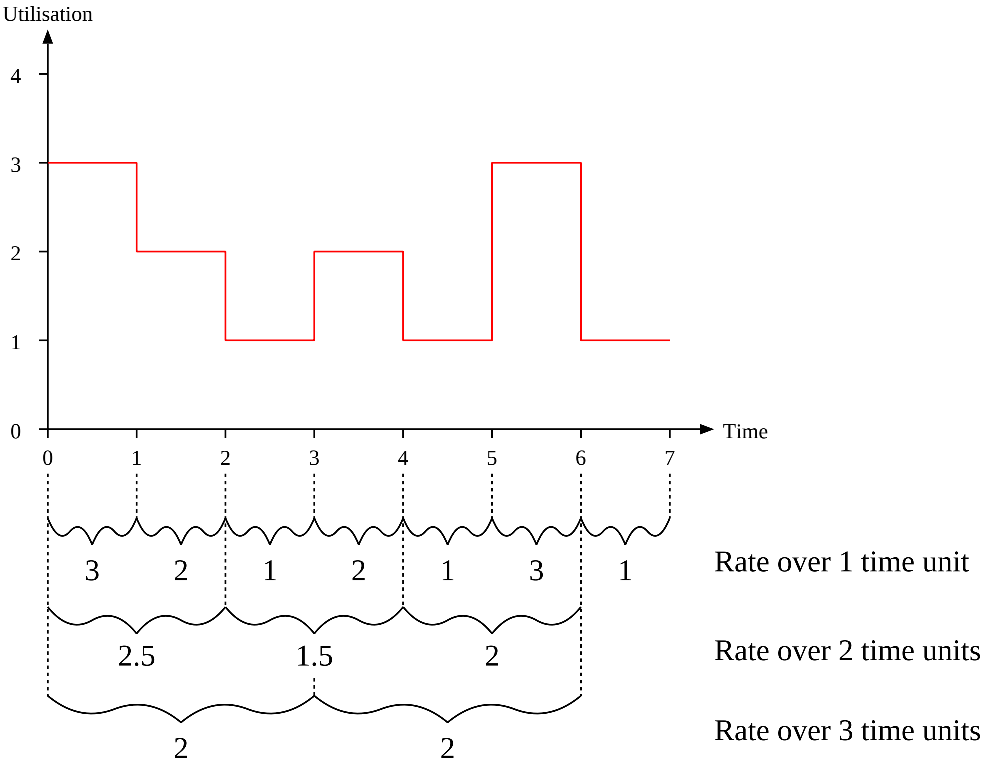


Figure 4.2: Illustration of the ‘length of measurement’ problem.

The approach of Floyd is considered in [Gibbens97] as a specific example of the class of approaches that arise from the *Chernoff Bound* techniques.

[Gibbens97] present four related techniques based upon *Chernoff Bounds*, each technique requiring different combinations of measured and declared parameters for its operation. Table 4.1 summarises the requirements of each approach, stating whether measurements are per-class or aggregate, class traffic declarations of peak or sustained rates, and the number of flows of each class.

The requirements of each *Chernoff Bound* technique make each easier or more difficult to implement. The availability and ease of extraction of measurements (e.g. per-flow versus aggregate) and the need for *a priori* traffic declarations (e.g. sustained-rate as well as peak-rate) will each affect the relative practicality of the four approaches presented by [Gibbens97].

CB technique		Requirements	
		Measurement	Per-class Declaration
I	Tangent at Peak	per-class measurements numbers of connections per class	peak rate
II	Tangent at Arbitrary Location	per-class measurements numbers of connections per class	peak rate, sustained rate
III	Tangent of Slope One	aggregate (line) measurements numbers of connections per class	peak rate
IV	Tangent at Origin	aggregate (line) measurement	peak rate

Table 4.1: Measurement and declaration requirements of *Chernoff Bound* based estimators.

As can be seen, each of the four techniques has different requirements of the measurements or the declared parameters. The 'Tangent of Slope One' technique was chosen for implementation. It requires aggregate measurements, along with peak rate declarations of incoming flows and the current number of flows in each class. The number of flows per class of traffic is easy to compute in the explicit AC environment that is assumed. Additionally, this technique had been well studied in [Gibbens97] and [Floyd96].

E-CB computes the *effective bandwidth* requirement of the aggregate of traffic (all classes added together) using Equation 4.2. In this equation E represents the estimate of the aggregate load, K the number of different types of flow, n_k the number of individual flows of a particular type, p_k the peak-rate for a particular flow-type, and X the measured aggregate utilisation. The scaling factor α may be adjusted to tune the algorithm's behaviour. The equation for *Chernoff Bound* approach III is

$$E = X + \frac{\alpha}{4} \sum_{k=0}^{K-1} p_k^2 n_k. \tag{4.2}$$

4.2.3 E-MS — Measured Sum

The Measured Sum estimator from [Jamin97c] computes an estimate of *effective bandwidth* based upon regular sampling of measured aggregate loads.

[Jamin97c] and [Jamin97b] presented a number of functions for converting regular samples into a load estimate. Their results concluded that the technique they called *time-window* performed the best for reliability and characterisation when combined with the Measured Sum estimator. For the time-window function the estimate is taken as the largest load estimate computed over a period ($T \cdot \tau$) consisting of T estimates of period τ.

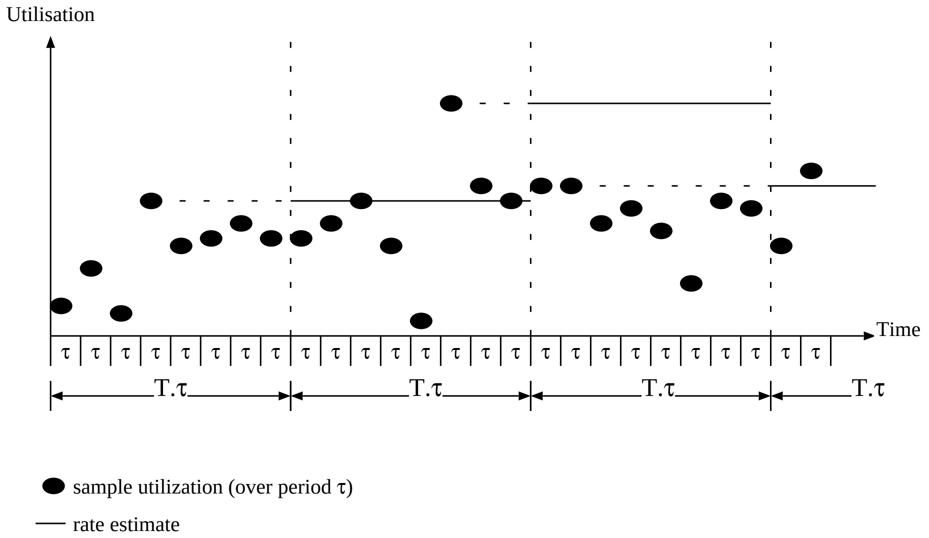


Figure 4.3: Time-window measurement of network load.

Figure 4.3, adapted from [Jamin97c], illustrates how the time-window technique works. In this figure, T is a multiple number of samples of length τ. The largest sample made in period $T \cdot \tau$ becomes the estimate of load for the next period $T \cdot \tau$. Such an approach for the computation of the current estimate is similar to the *Local Maximum Predictor* of [Duffield99b], which also uses the maximum of the rates sampled during a given measurement window as estimate of load for input into an MBE.

In the implementation a number of values for τ and T were tested; the results presented here used $T = 10$ and $\tau = 100$ ms, the same values as the comparison in [Jamin97b]. Clearly, the period τ and value of T are intended to offer a characterisation of the traffic flow: period τ, like the measurement-period of E-IU, will remove instabilities smaller than this period incorporating them into its computation of a *effective bandwidth*. The maximum of these values over the past T samples (i.e. over period $T \cdot \tau$) captures the boundary maximum conditions of the traffic over the larger time-scale.

An *effective bandwidth* is computed from the aggregate load estimate using Equation 4.3. In this equation, E represents the estimate of *effective bandwidth*, $\hat{X}$ is the current aggregate load estimate, and μ is the utilisation target (a control parameter of the algorithm that is proportional to the level of utilisation resulting from the use of this estimator),

$$E = \frac{\hat{X}}{\mu}. \tag{4.3}$$

4.2.4 E-MPFE — *Measure* Per-Flow Estimator

The *Measure* estimator, based upon the theory of large deviations, is presented in [Duffield95]. Large deviation theory allows the quantification of rare events such as packet loss in a computer network due to traffic interactions. Their approach uses the premise that the logarithm of the loss-ratio versus the buffer size may be bound by a straight line. This approximation allows description of the large deviation rate function and in turn this allows description of a Scaled-Cumulative Generating Function (SCGF). An estimate of the *effective bandwidth* becomes the slope of the SCGF for a particular set of (traffic) measurements constrained by a set of buffer characteristics (loss-ratio and buffer size).

Equation 4.4 gives the definition of the SCGF, $\hat{\lambda}(\theta)$. This function uses T periodic measurements $\hat{X}_t$, each taken over the period τ,

$$\hat{\lambda}(\theta) = \frac{1}{\tau} \log \frac{1}{T} \sum_{t=1}^{T} e^{\theta \hat{X}_t}. \tag{4.4}$$

where $\theta = \frac{-\log \epsilon}{q}$, ϵ is the desired loss-ratio, and q is the size of the buffer. The *effective bandwidth* is the the rate of change (slope) of this function for any given value of θ.

For E-MPFE, $\hat{X}_t$ is calculated using measurements of each flow. The estimation of the rate of existing flows relies upon regular measurement of the load introduced by each flow into the link. This approach differs from a simple aggregate measurement by computing the value of $\hat{X}_t$ from measurements of current flows only. If a flow is removed from the system any contribution it made to the values of $\hat{X}_t$ in the past are then removed. In this way, any computation of the SCGF is made using only the current contributions of current flows to values of $\hat{X}_t$.

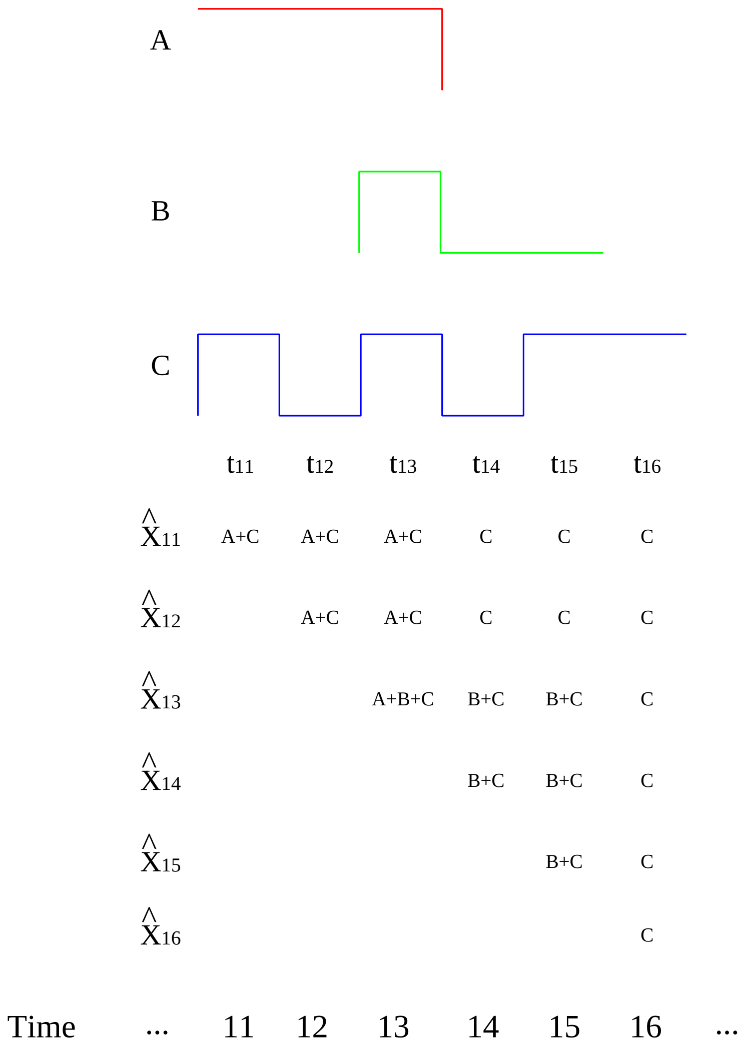
The measurements are not used to deduce the current queue length, which might easily be done by reading the physical queue length. The point is to discover the distribution of the queue length and the long-term probability of overflow for the current mix of traffic (not including the flow or flows just departed). The answer depends not upon the current, instantaneous queue length but upon the statistics of the arrivals.

Nevertheless, in this approach there is an intrinsic assumption that the queue is running with a high quality of service and that the queue is emptied very often — possibly empty more often than it is non-empty. Thus the residual effects of any dead flow are assumed to be completely wiped when the queue empties.

The manner in which the composition of each $\hat{X}_t$ sample changes over time is illustrated in Figure 4.4. As each flow leaves the system, the contribution it has made to previously-captured samples is removed. The reason for this is that the contributions of such a flow, no longer in the multiplex, has no relevance to the computation of the SCGF (and thus the loss-events) of the current (or future) flows.

A potential problem with this approach is that in the computation of θ it is assumed that the loss properties are asymptotically bound by $\epsilon = e^{-\theta q}$.

[Choudhury96b] and [Botvich95] reported that *effective bandwidth* estimates computed using



	t_{11}	t_{12}	t_{13}	t_{14}	t_{15}	t_{16}
$\hat{X}_{11}$	A+C	A+C	A+C	C	C	C
$\hat{X}_{12}$		A+C	A+C	C	C	C
$\hat{X}_{13}$			A+B+C	B+C	B+C	C
$\hat{X}_{14}$				B+C	B+C	C
$\hat{X}_{15}$					B+C	C
$\hat{X}_{16}$						C

Figure 4.4: Composition of $\hat{X}_t$ from per-flow measurements for E-MPFE.

such asymptotic bounds may be either optimistic or conservative depending on the nature of the arrival streams. However, the *Measure* estimator is based upon the relationship, $\theta = \frac{-\log \epsilon}{q}$. Errors both optimistic and pessimistic may occur that can be attributed to this assumption. An example is that approximation of the decay curve with a function that is continuous unto infinity does not capture the reality that decay curve is finite in length, such a shortcoming of the estimate having particular impact when large numbers of sources are present in the multiplex.

4.2.5 E-MAE — *Measure* Aggregate Estimator

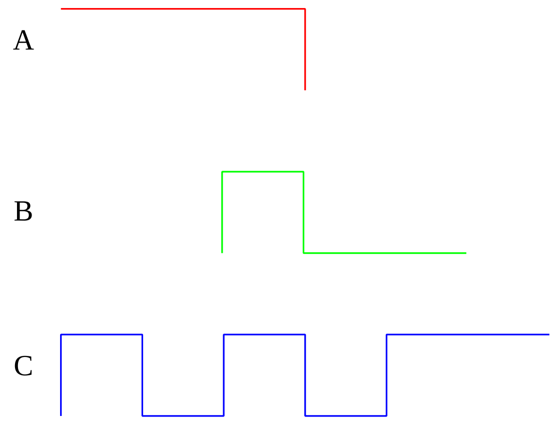
This MBE differs from the previous one by its calculation of the utilisation samples used as input into the equation to derive the estimate of *effective bandwidth*. By computing the *effective bandwidth* using aggregate measurements this approach avoids the management and computational overheads required for computing utilisation samples from per-flow measurements.

Figure 4.5 illustrates how each utilisation sample, $\hat{X}_t$, is computed using aggregate measurement: measurements of all traffic at that time. Contrasting this with Figure 4.4 reveals how an estimator based upon aggregate utilisation measurements differs from one based upon per-flow measurements.

The use of aggregate measurements implies that the content of each block, $\hat{X}_t$, may contain measurement data for flows that have since been removed from the network. In the worst-case, a block does not contain measurement data for any active flows. In this case its contribution to the computation of an *effective bandwidth* estimate will be unpredictable at best, and unhelpful at worst. In E-MPFE, only the measurements of active flows are used to calculate the value of each utilisation value $\hat{X}_t$, even though this means recalculating the values of each $\hat{X}_t$ whenever a flow is removed from the system.

The authors of the *Measure* algorithm had considered only per-flow utilisation as input to the estimator but E-MAE was worth pursuing for more practical reasons. Aggregate measurements are more commonly available, require a lower overhead, and have smaller management requirements than per-flow measures. Early success with E-MAE, combined with performance problems with the E-MPFE, encouraged this approach.

Aggregate measurements still presented several problems. Firstly, some blocks used in the com-



	t_{11}	t_{12}	t_{13}	t_{14}	t_{15}	t_{16}
$\hat{X}_{11}$	A+C	A+C	A+C	A+C	A+C	A+C
$\hat{X}_{12}$		A+C	A+C	A+C	A+C	A+C
$\hat{X}_{13}$			A+B+C	A+B+C	A+B+C	A+B+C
$\hat{X}_{14}$				B+C	B+C	B+C
$\hat{X}_{15}$					B+C	B+C
$\hat{X}_{16}$						C

Time ... 11 12 13 14 15 16 ...

Figure 4.5: Composition of $\hat{X}_t$ from aggregate measurements for E-MAE.

putation of the SCGF contained measurements of flows that no longer existed. A solution may be to place an upper limit on the lifetime of any particular block: once a block becomes older than a certain period, it is removed from the computation of the SCGF. The impact of this approach is that if the history length is shorter than the average length of a flow some blocks with valid data will be discarded. On the other hand, if the history length is longer than the average length of a flow, some blocks will be included in the computation of the SCGF that contain no valid data at that time.

Thus, for the estimator based upon aggregate measurements, the selection of an optimum 'maximum history length' becomes a problem. While an obvious length may be the mean flow lifetime, it is important to consider the statement made above: the point of measurements is not to determine the queue length but the probability of queue overflow in the future. As a result, the relevance of previous measurements of aggregate utilisation will share a relationship with the overall size of the system and the overall size of the system is given by the number of flows multiplexed at the buffer.

[Grossglauser97b] noted that the period required for an accurate measurement of the variance in a system shared a relationship with the size of that system. They postulated that only events that occurred over a period inversely proportional to the square-root of the number of flows were of relevance to the variance. This led to the idea that a critical time-period existed beyond which previous events were not relevant to the current or (immediate) future system. [Grossglauser97b] concluded that this critical time period is $\frac{T_h}{\sqrt{n}}$, where T_h is the mean lifetime of flows and n is the number of flows in progress.

The concept of a critical time-period had a great appeal for application in an estimator based upon aggregate measurement. In estimator E-MPFE all flows whose traffic would have direct impact upon the current estimate are present in the computations — although this is restricted to currently active flows only. In such a system where a flow was very long lived, it is easy to imagine significant memory (and computation) resource being required to compute the *effective bandwidth* estimate.

A second problem remains for the E-MAE estimator of what the upper boundary, beyond which blocks are discarded, should be. Choosing this involves balancing the requirements of keeping blocks containing data attributable to current flows while discarding blocks for which the majority of data is not relevant. The use of the critical time-period as a time limit was a logical approach.

This approach cannot be rigorously justified but early success with E-MAE means that it is included in the comparison results presented in Chapter 5.

4.2.6 E-MVE — Mean Variance Estimator

E-MVE is an estimator based upon a combination of measurement mean and variance:

$$E = \bar{x} + \alpha\sigma. \tag{4.5}$$

Equation 4.5 gives of an estimator based upon the mean and variance measurements. In this equation, E is the estimate of *effective bandwidth* of the measured traffic, $\bar{x}$ is the mean and σ is the standard deviation, each computed from a series of measurements of the aggregate line utilisation. The mean of the flow captures the long-term changes in the multiplex of traffic, while the variance characterises the variability of the traffic mix within the time-scale of the measurements. A pre-multiplier of α allows the estimator to accommodate a range of variability in the traffic measurements made.

Apart from the selection of a value of α, such an estimator requires appropriate selection of the measurement period and the number of samples used to compute mean and variance of the measurements.

The version of this estimator from [Duffield99b, § 3.1 Local Gaussian Predictor] incorporates a correction to α. The correction is used to account for the burstiness of traffic such as Internet sources and video-streams observed at multiple time-scales (as discussed in Section 2.1). In such traffic there will be an increase in variability of traffic averaged over decreasing window sizes. In addition to burstiness-averaging, sampling error also plays an important role.

Sampling error will arise in the estimation of mean and variance because the samples from which the estimates are computed are in themselves random variables. In order to avoid violation of QoS guarantees due to this, [Duffield99b] applied the explicit correction of Equation 4.6, where

T is the number of samples:

$$\alpha' = \max\{\alpha, \sqrt{(T+1)(e^{\frac{\alpha^2}{T}} - 1)}\}. \tag{4.6}$$

The value of α' can then be substituted so that Equation 4.5 becomes $E = \bar{x} + \alpha'\sigma$. For such a correction there is an assumption that the sampling error possess Gaussian properties.

[Duffield99b] noted that a predictor based upon samples of any particular width may underestimate the bandwidth requirement needed to satisfy a particular QoS constraint specified over a shorter time-scale, although no specific solution was suggested or applied. This issue is discussed at length in the descriptions of the E-GT and E-KQ estimators below.

4.2.7 E-KQ — Traffic Envelope

The *traffic envelope* approach embraces the central issue that to characterise the rate of a particular traffic flow a period must be specified over which that characterisation is conducted. As a result this MBE, proposed by Knightly in [Knightly96] and further explored in [Knightly98] and [Qiu98b] is able to characterise traffic over a series of time periods. The intention of this multi-period characterisation is to represent the short term burstiness of traffic as well as that of the longer term variation of the aggregate due to measurement error and longer time-scale fluctuations.

Firstly it is assumed that there exists a basic measurement period, τ. Measurements may be taken over a multiple of this period and thus $I_{1,2,\ldots} = 1, 2, \ldots \times \tau$. Thus, if the activity on a link over the interval $[s, s + I_\eta]$ is represented as $X[s, s + I_\eta]$ then $\frac{X[s,s+I_\eta]}{I_\eta}$ is the rate over that particular period. [Knightly98] noted that the peak rate over any interval of length I_η can be given by $R_\eta = \max_s X[s, s + I_\eta]$. This allows the specification of the *maximal rate envelope*: a set of rates R_η that represent the maximum rate of the flow for each of the intervals I_η.

The activity in time slot t is represented as x_t such that $x_t = X[t\tau, (t + 1)\tau]$. This allows a definition of the maximal rate envelope for the past T time slots from the current time t as

$$R_\eta^1 = \frac{1}{\eta\tau} \max_{t-T+\eta \leq s \leq t} \sum_{u=s-\eta+1}^{s} x_u \tag{4.7}$$

for $\eta = 1, \cdots, T$. The envelope R^1_η, $\eta = 1, \cdots, T$ describes the aggregate maximal rate envelope over intervals of length $I_\eta = \eta\tau$ in the most recent $T \cdot \tau$ seconds. [Knightly98] assert that this will describe short time-scale burstiness along with autocorrelation structure present in the flow.

If every $T \cdot \tau$ periods the current envelope is updated $R^n_\eta \leftarrow R^{(n-1)}_\eta$ for $\eta = 1, \ldots, T$ and $n = 2, \ldots, N$, then a new envelope R^1_η is computed using Equation 4.7. This allows the empirical mean $\bar{R}_\eta$ of the R^m_η's to be computed as $\Sigma^M_{m=1} \frac{R^m_\eta}{M}$. In turn this allows the variance between envelopes for the past M windows of time $T \cdot \tau$ to be computed using

$$\sigma^2_\eta = \frac{1}{M-1} \sum_{m=1}^{M} (R^m_\eta - \bar{R}_\eta)^2. \tag{4.8}$$

Taking the mean and variance of M consecutive traffic envelopes allows the variability of the traffic envelope itself to be characterised at longer time-scales.

From the traffic envelope, the E-KQ approach computes two estimates of *effective bandwidth*, one for each of the two time-scales: short-term burstiness and long-term variance. For the long-term time-scale resulting from variance between traffic envelopes, the mean and standard deviation of the maximal traffic envelopes (those measured over $T \cdot \tau$) provide one estimate of the *effective bandwidth*,

$$E_{\text{long}} = \bar{R}_T + \alpha_{\text{long}} \sigma_T. \tag{4.9}$$

The value of α_{long} will determine how the estimator behaves in response to variability in the measured flow. It is possible to formulate α_{long} to dictate a specific confidence interval for these constraints. [Knightly98] considered a large variety of distributions on which to based α_{long} — settling upon a Gumbel distribution for its ability to describe the asymptotes of the extremes for a large range of other distributions (e.g. Gaussian, exponential, log-normal, Gamma, Raleigh). However other work — [Qiu98a] and [Tse99] — indicated that a Gaussian distribution is adequate, as well as allowing a more tractable computation. Thus in each case the computation of α_{long} is based upon computing the inverse of a complementary CDF of an N(0,1) Gaussian distribution ($Q^{-1}(\cdot)$) based upon the maximum packet loss (ϵ) and the traffic envelope:

$$\alpha_{\text{long}} = Q^{-1}(\frac{\epsilon \bar{R}_T}{\sigma_T}). \tag{4.10}$$

For the shorter burstiness time-scale, a different estimator is is used. The *effective bandwidth* requirement of the burst time-scale relates to the size of the buffer, q. The estimate of *effective bandwidth* requirement is computed from the maximum of the traffic envelope mean and standard deviation. In the following equation, C is required to compute the rate at which the buffer can be drained:

$$E_{\text{short}} = \max_{\eta=1,2,\cdots,T}\{\frac{(\bar{R}_\eta + \alpha_{\text{short}}\sigma_\eta)\eta\tau}{\eta\tau - \frac{q}{C}}\}. \tag{4.11}$$

Unlike E_{long}, E_{short} is computed using every value of η in the traffic envelope. Once again, the standard deviation pre-multiplier will determine the response to variability in the measured flow. The derivation of α_{short} from the user supplied packet-loss, ϵ, and traffic envelope is

$$\alpha_{\text{short}} = Q^{-1}(\frac{\epsilon\bar{R}_T}{\sigma_\eta}). \tag{4.12}$$

The maximum of the two equations 4.9 and 4.11 can be considered the worst-case *effective bandwidth* estimate of the traffic flow described by the traffic envelope. This is given by

$$E = \max\{E_{\text{long}}, E_{\text{short}}\}. \tag{4.13}$$

[Knightly98] documents the importance of the value of T, the maximum number of samples for a traffic envelope. An ideal value of T will provide the optimum use of resources, while too small a value of T causes the variation over σ_T to be large, so that the capacity-based estimate of Equation 4.9 will be pessimistic. Alternatively, if T is too big the estimate derived for buffer occupancy will be too large causing the buffer based estimate, Equation 4.11, to be pessimistic. In [Knightly98] a discussion is given over to locating the optimum value of T, a value typically on the order of a few seconds.

4.2.8 E-GT — Time-scale Decomposition

First introduced by [Grossglauser97b] and extended to heterogeneous flow environment in [Tse99] and [Grossglauser00], this algorithm also uses characterisation of traffic flows by mean and variance. This algorithm is different to all the other MBAC algorithms presented in this chapter as it is based upon a *bufferless* model. As a result direct comparisons of performance with other algorithms is not as relevant. However, given that this algorithm adopts a unique and oft-cited approach to problems of time-scale, an implementation was seen as crucial to an objective of providing a broad coverage of MBE and MBAC algorithms.

Several assumptions are made explicitly in the design of this algorithm. Firstly the traffic is assumed to have a uniformly distributed variance, be independent, and stationary. Additionally, it is assumed that the mean and variance of traffic remain fixed once the flow is in progress. Finally, it is assumed there is a large link capacity with no single flow dominating the multiplex. Recent traffic characterisation studies may draw these assumptions into question, notably the assumptions about sources (see Section 2.1).

The design of the MBAC algorithm of [Tse99, Grossglauser00] is based on the assumption that fluctuations slower than the time-scale $\widetilde{T_h}$, (the mean flow life-time,) are able to be absorbed by changes in the number of flows in progress, while the high-speed fluctuations (those on a time-scale faster than $\widetilde{T_h}$) can be absorbed by over-booking the capacity required. The result is the decomposition of the bandwidth measurements into high and low frequency components using a cut-off frequency of $\frac{1}{\widetilde{T_h}}$. Figure 4.6, adapted from [Grossglauser00], illustrates how the stream of samples is filtered into high and low-frequency components using this cut-off frequency. In this figure, δ represents the unit impulse function.

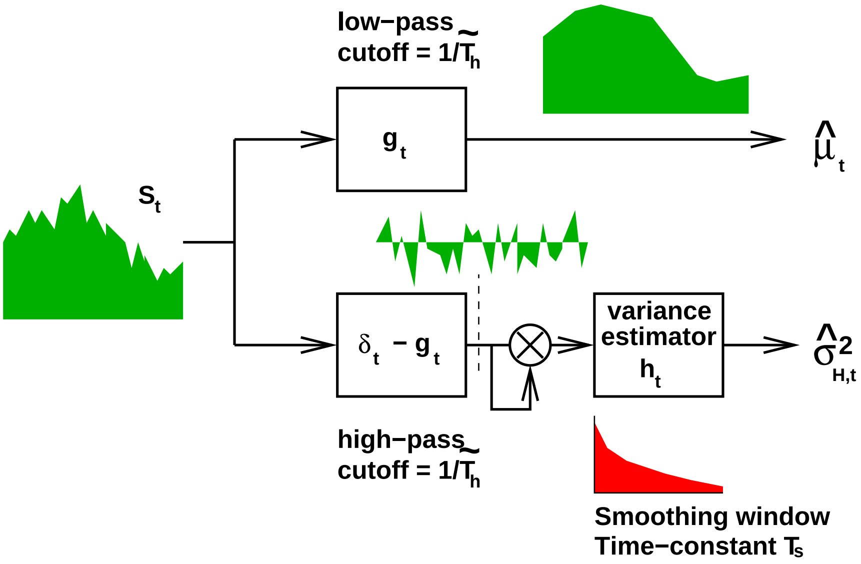


Figure 4.6: Decomposition of measurement into mean and variance.

The requirements created by fast time-scale fluctuations and provided-for through over-booking are computed using the high-frequency component S_t^H. The over-booking required is computed from the variance of S_t^H: ${\sigma^2}_{H,t}$. The mean requirements of the flows, $\hat{\mu}_t$, are estimated using the low-frequency components S_t^L. Dominated by slow time-scale flow admissions and departures, the mean requirements of the flows can be used to estimate the number of flows that can be accommodated.

The low pass filter is defined as

$$g_t = \frac{1}{\widetilde{T_h}} e^{(-\frac{t}{\widetilde{T_h}})} u_t, \tag{4.14}$$

where u_t is the unit-step (heavy-side) function. The filter used in the estimation of variance is

$$h_t = \frac{1}{T_s} e^{(-\frac{t}{T_s})}. \tag{4.15}$$

Equation 4.14 and Equation 4.15 use a related time-scale: $T_s = M\widetilde{T_h}$. In earlier work [Grossglauser97b] used a measurement-per-flow mechanism. However, because of the overheads of such a measurement scheme, the use of aggregate utilisation measurements were proposed in the more recent paper (e.g. [Grossglauser00]). In the transformation from per-flow to aggregate measurements, [Grossglauser00] noted that the estimation of variance was poor due to the reduced number of samples over which it is made. Thus, to ensure a robust estimate, more measurements are required over time to make up for the lack of individual measurements over flows. This leads to the need for $M \gg 1$ to provide a sufficiently robust estimate of $\hat{\sigma}_t^H$.

Using Equations 4.14 and 4.15, the estimated mean of a single flow, where S_t is the aggregate load at time t and N_t is the number of flows present in the system at time t, is given by:

$$\hat{\mu}_t = \int_0^\infty \frac{S_{t-\tau}}{N_{t-\tau}} \, g_\tau \, d\tau. \tag{4.16}$$

The high-pass component is given as

$$S_t^H = S_t - N_t \hat{\mu}_t, \tag{4.17}$$

and from this, the estimate of high-pass variance of a single flow may be estimated by

$$\hat{\sigma}_t^H = \int_0^\infty \left[\frac{S_{t-\tau}^H}{N_{t-\tau}} - \int_0^\infty \frac{S_{t-u}^H}{N_{t-u}} \, h_u \, du \right]^2 h_\tau \, d\tau. \tag{4.18}$$

Using $Q(\cdot)$, the complementary CDF of a N(0,1) Gaussian distribution, Grossglauser showed that the loss probability (ϵ) for a system of given capacity C, subject to a number of flows N_t, with mean and standard-deviation properties of $\hat{\mu}_t$ and $\hat{\sigma}_t^H$ is given as

$$\epsilon = Q\left[\frac{C - N_t\mu}{\sigma\sqrt{N_t}}\right]. \tag{4.19}$$

Thus for a known loss probability, ϵ, the appropriate value of N_t ought be computable for any given mean and standard deviation. Rearranging Equation 4.19 gives

$$C = Q^{-1}(\epsilon)\sqrt{N_t}\hat{\sigma}_t^H + N_t\hat{\mu}_t. \tag{4.20}$$

The *r.h.s.* of this equation can then be used to compute an estimate of *effective bandwidth*. In order to make clear the similarities this estimator shares with E-KQ, a pre-multiplier of the variance, α, based upon the complementary CDF of an N(0,1) Gaussian distribution let

$$\alpha = Q^{-1}(\epsilon). \tag{4.21}$$

This value can then be substituted into Equation 4.20. Thus for a value of α which characterises the tolerance to loss ϵ, along with N_t the number of flows in progress, and $\hat{\sigma}_t^H$ and $\hat{\mu}_t$ which together characterise the current traffic flow, provides the *effective bandwidth* estimate

$$E = \alpha\sqrt{N_t}\hat{\sigma}_t^H + N_t\hat{\mu}_t. \tag{4.22}$$

4.2.9 E-LBE — Loss-Based Estimator

While the majority of MBAC algorithms, including the majority presented here, use an estimation of *effective bandwidth* as central to the admission decision, a number of authors have presented approaches that base their decisions on loss-ratios. As an example, [Saito91] presents a mechanism that measures the marginal distribution of cell arrivals and applies the loss-ratio upper-bound formula from [Saito92]. The computation of a loss-ratio upper-bound allows the construction of an estimator, and subsequently an admission algorithm.

The loss-based estimator presented here makes a prediction of long-term loss based upon measurements over the short-term, although it does not use the marginal distribution approach in [Saito92]. Instead, the current measured loss-ratio is used directly to predict the future loss-ratio.

A problem exists with this approach: large numbers of observation events are required for reliably predicting low loss-ratios. This problem could be overcome in one of two ways: ensure a significantly high quantity of traffic such that the number of observation events is sufficient or increase the period over which the events are counted. The solution adopted in E-LBE is to use an exponentially weighted moving average. This approach gives a long 'effective' observation period thereby aiding the system stability, while being able to adjust the relative impact the prior history and the most recent measurement will have upon the current estimate.

The current estimate of loss, $\hat{\epsilon}_t$ can be computed from l_t (the actual number of loss events for a given period τ) and x_t (the total number of events for the same period) using:

$$\hat{\epsilon_t} = (1-\alpha)\hat{\epsilon}_{t-\tau} + \alpha\frac{l_t}{x_t}. \tag{4.23}$$

Such an approach has the effect of damping out wild oscillations and thus giving long-term stability. However, selection of a low value for α will cause the system to respond slowly, triggering the same behaviour as if the measurement periods were long.

The approach of using loss feedback is not new and has been used in a number of MBAC algorithms, such as that of [Saito91]. Ideas incorporating delay as feedback in addition to loss were proposed in [Jamin92] and [Tedijanto93]. In the realm of transfer protocols examples are TCP, which will modify its transmission characteristics in response to loss, or the explicit transfer of loss information which is central to the Real-time Transport Protocol (RTP) [AVTWG96].

The technique presented here makes several sweeping assumptions. Firstly, and most importantly, flow-arrivals are assumed to be distributed in a positive Gaussian fashion while loss events will have a fully Gaussian error. This assumption may be poor for traffic sources that have correlated structure. Such correlated traffic will over-emphasise any burstiness present, causing loss-events to be clustered. Selection of the time period over which loss-events are measured also presents a problem. The time period before which the thresholding loss measurement is updated will need to be related tightly to the flow arrival-rate. Too large a period and many flows, each with unknown potential to introduce loss into the network, may be admitted.

4.2.10 E-GAN — Equivalent Capacity

A number of estimators have been proposed that, provided with line speed, buffer capacity and a target loss probability along with *a priori* declarations about the traffic are able to compute an equivalent capacity for each source or multiplex of sources. Such a computation can be configured to calculate the maximum number of flows admissible to a multiplexer while satisfying the user-provided loss-constraint. [Guérin91] reports a system based upon fluid-flow approximations of the traffic multiplex.

Assuming each source can be described as an Interrupted Fluid Process (IFP), of which the 2-state ON-OFF Markov source such as TP10S1 (Section 2.2.1) is an example, then the source may be characterised by the vector (p, r, b), where the peak-rate is p, r is the fraction of time the source is active (peak rate divided by sustained rate) and b is the mean duration of the active period. For a system where the queue is of capacity q and for a desired loss-ratio of ϵ, the *effective bandwidth* estimate E_F, is given as

$$E_F = \frac{a - q + \sqrt{(a-q)^2 + 4qar}}{2a} p, \tag{4.24}$$

where,

$$a = b(1-r)p\log(\frac{1}{\epsilon}). \tag{4.25}$$

For N sources, the AC of [Guérin91] proposed a combination of fluid-flow approximation and stationary approximation. Using a stationary approximation, where E_S is the stationary approximation based upon $\bar{s}_i$ (the mean bit-rate of the ith source) and σ_i (the variance in bit-rate of the ith source) gives:

$$E_S = \sum_{i=1}^{N} \bar{s}_i + a' \sum_{i=1}^{N} \sigma_i \tag{4.26}$$

where,

$$a' = \sqrt{-2\log\epsilon - \log 2\pi}. \tag{4.27}$$

Now the estimate of equivalent capacity for N sources is given by

$$E = \min\left(E_S, \sum_{i=1}^{N} E_{Fi}\right), \tag{4.28}$$

where E_{Fi} is the equivalent capacity of an individual source.

Thus, using *a priori* descriptions of mean rate, peak rate and mean burst length along with the buffer characteristics, it is possible to compute an *effective bandwidth* estimation of a collection of traffic sources. Importantly, the sources must be able to be modelled as constructions of Markov chains (and thus be IFP). An important point for this model-based estimator is that the characterisation of traffic contains the mean burst size, not the maximum burst size as is declared in the ATM Forum TM4 [ATMF95] signalling proposal or INTSERV's RSVP [Braden97] signalling system.

The authors of [Elsayed99] made a note of this difference but did not reflect on the intrinsic incompatibility that exists between the parameters required by such model-based systems and those supplied by the signalling system. As was noted earlier in this chapter and in Chapter 2, sustained values such as sustained rate and mean burst size are difficult to predict for any particular traffic source. This is because the computation of such values require that the traffic be either analysed prior to use, regulated via a scheme such as *leaky-bucket*, or a combination of both. Such a requirement (mean burst size) is common to any scheme which is based upon the Markovian traffic models, thus this requirement (and intrinsic drawback) is shared in common with the following model-based estimator.

4.2.11 E-BD — Exponential Upper-Bounds

The E-GAN algorithm proposed by [Guérin91] provides an approximation of per-source *effective bandwidth*. This estimate does not take account the additional gain available when the *effective bandwidth* estimate incorporates the buffer-space available at the point of multiplexing. Working with large deviation based estimators [Buffet92] introduced a factor to account for the additional gain that may be computed with prior knowledge of the buffer size. This allows for a refined computation of available capacity and thus a refined computation of the *effective bandwidth* per-source.

If a buffer of size q is being serviced at rate s, for a fixed size buffer, a higher the service rate would result in a lower number of packets lost from the buffer.

Similarly, increases in the service rate of a fixed length buffer would also reduce the number of packets lost. It has been shown (e.g.[Glynn94]) that if the arrivals at the buffer are approximately stationary, and are not long range dependent, then the loss ratio decays exponentially for large buffer sizes. Thus the decay can be expressed as

$$\lim_{q\to\infty} \frac{1}{q} \log F(q,s) \approx -\delta(s), \tag{4.29}$$

where $F(q,s)$ is the loss ratio function, and $\delta(s)$ represents the decay function.

Figure 4.7 (adapted from [Lewis98]), shows the empirical overflow probability for a number of multiplexed JPEG video sources. In this figure a straight-line through the origin suggested by Equation 4.29 offers a functional approximation.

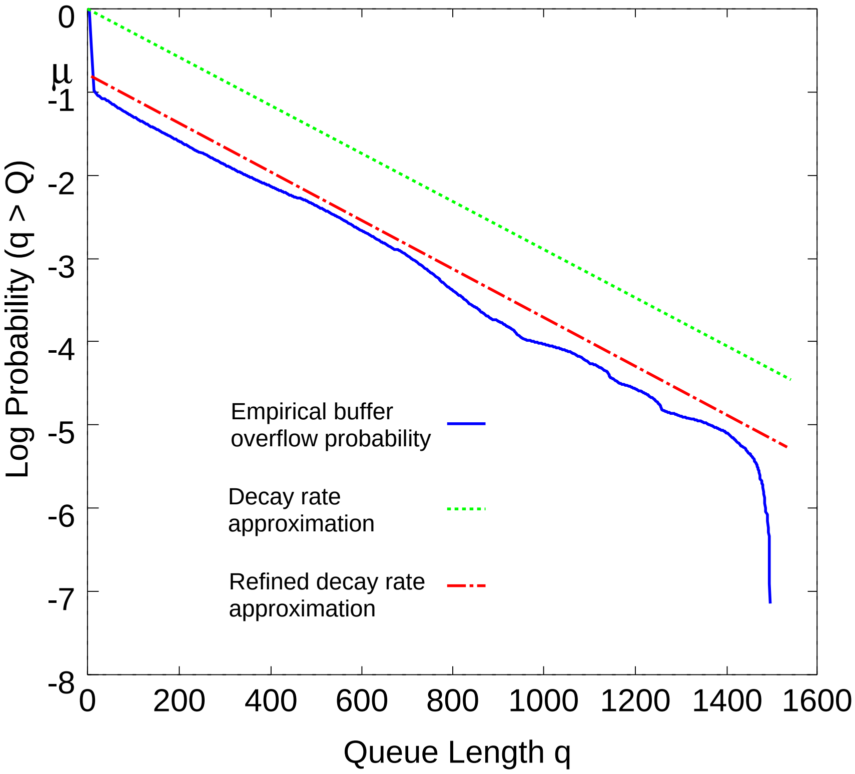


Figure 4.7: Empirical loss probability for 18 streams of JPEG coded video.

In Equation 4.29 the decay rate $\delta(s)$, a function of the service rate (s), was introduced as an approximation to the decaying loss-probability curve. Thus the probability of a buffer of length q overflowing, is given as

$$\mathbb{P}(Q > q) \asymp e^{(-q\delta)}. \tag{4.30}$$

Figure 4.7 illustrates a refinement to the decay rate approximation. The refinement is a straight-line that intercepts the vertical axis (rather than the origin). The figure illustrates the refined estimate suggested in [Buffet92].

The refined approximation introduces the pre-factor, φ, into Equation 4.30 thus becoming

$$\mathbb{P}\left(Q > q\right) \geq \varphi e^{(-q\delta)}. \tag{4.31}$$

[Buffet92] noted that where N is the number of independent identical 2-state Markov sources and $e^{-\mu}$ is the pre-factor for a single source served at a rate $\frac{C}{N}$ (C being the service rate of the buffer,) φ can be expressed as

$$\varphi = e^{-\mu N}. \tag{4.32}$$

Armed with an enhanced ability to approximate the buffer overflow probability, [Buffet92] presents an explicit bound for the tail of the queue length distribution. Adapted from [Buffet92, Eq. 3.1], Equation 4.33, is the bound for the probability of queue overflow. In this equation, mean time between bursts is given as $1/\alpha$, the mean period of a burst is $1/d$, and q is the size of the buffer. The service-rate of the queue is C, N represents the number of flows in the system and σ is the proportion of the service-rate each flow has ($\sigma = C/N$).

$$\begin{aligned}\mathbb{P}\left(Q \geq q\right) \quad \leq \quad & \frac{a\left(1-\left(a+\sigma\left(1-a-d\right)\right)\right)}{d\left(a+\sigma\left(1-a-d\right)\right)}\left[\frac{\left(1-\sigma\right)\left(a+\sigma\left(1-a-d\right)\right)}{\sigma\left(1-\left(a+\sigma\left(1-a-d\right)\right)\right)}\right]^{q} \times \\ & \left[\frac{1}{a+d}\left(\frac{d}{1-\sigma}\right)^{1-\sigma}\left(\frac{a}{\sigma}\right)^{\sigma}\right]^{N}\end{aligned} \tag{4.33}$$

An estimator can be constructed provided, like E-EMW before, the traffic characteristics of peak rate, mean rate, and burst size are declared in advance. Once these values are known, a value of N can be computed that satisfies the desired loss bound. Using this limit on the number of flows a simple estimator can be constructed where E is the *effective bandwidth* estimate, N is the maximum number of flows in progress (as computed using the previous formula), and n is the actual number of flows in progress

$$E = \frac{C}{N} \cdot n. \tag{4.34}$$

In addition to the need for full declaration of the traffic characteristics (peak rate, mean rate, mean burst size) an implicit assumption exists that the traffic exhibit an exponential arrival characteristic, and is i.i.d.

4.2.12 E-EMW — Effective Bandwidth Model

[Elwalid95] proposed a resource allocation scheme that is modelled by a shared buffer multiplexor fed by ON-OFF, periodic arrival processes. Resource allocation at a network buffer may be considered a two-resource allocation problem. As was seen for the buffer overflow in E-BD, the loss probability is related to both buffer size and to the service rate of the buffer. To provide a desired loss-ratio the estimator must deal with two parameters: service-rate and buffer-size. The approach of E-BD was to use a *correction* performing additional estimation of the multiplexing gain. The approach of E-EMW is to reduce the two-resource allocation problem (buffer and bandwidth) to a single-resource allocation problem by estimating the loss-probability of a buffer-less multiplexor. The idea of a 'virtual buffer' and 'virtual trunk' (with a virtual service-rate) allows the interchange of each resource. [Elwalid95] then described the application of this method to admission control.

The method for estimating an *effective bandwidth* proposed by [Elwalid95] assumed that all traffic to be multiplexed has been subject to policing by traffic regulators such as *leaky-bucket*. The resulting estimation is computed as a boundary case, based upon a *leaky-bucket* regulator figures of burst tolerance, peak rate and sustained rate.

Estimates of the packet-loss boundary are drawn from *Chernoff Bounds*, further refining that estimate through the use of a large deviations approach. [Elwalid95] computed, using a fluid-flow approximation, the loss estimate for a particular number of flows, and then a root-solver can be used to locate the number of flows for a desired loss-ratio.

The flows are defined by the parameters of peak-rate, p, sustained-rate r, and maximum burst-size, B_T, while the system is defined with a buffer size of B and a link capacity of C.

In a system based upon the equivalence of buffer and transmission capacity, [Elwalid95] defined a value called the *effective capacity* for lossless performance, ϵ_0, as

$$\epsilon_0 = \begin{cases} \frac{p}{1+\frac{B/C}{B_T}(p-r)} & \text{if } r \leq \frac{B_T}{B/C}; \\ r & \text{if } \frac{B_T}{B/C} \leq r < p \,. \end{cases} \tag{4.35}$$

The flow characteristics are defined as T_{on} and T_{off} which, using the maximum burst size B_T, peak-rate p and mean-rate r gives: $T_{\text{on}} = \frac{B_T}{p-r}$ and $T_{\text{off}} = \frac{B_T}{r}$. [Elwalid95] noted that the formula for ϵ_0 is more easily understood when expressed in terms of these variables:

$$\epsilon_0 = \begin{cases} \frac{p}{1+\frac{T_{\text{buf}}}{T_{\text{on}}}} & \text{if } T_{\text{buf}} \leq T_{\text{off}}; \\ r & \text{if } T_{\text{off}} \leq T_{\text{buf}} \,, \end{cases} \tag{4.36}$$

where T_{buf} is the maximum delay time in the buffer and may be defined as $T_{\text{buf}} = B/C$.

In [Elwalid95] the loss-ratio is computed using

$$\epsilon = \frac{e^{-F_K(s^*)}}{s^*\sigma(s^*)\sqrt{2\pi}}, \tag{4.37}$$

where

$$s^* = \frac{1}{\epsilon_0} \log \left[\frac{a}{1-a} \cdot \frac{1-w}{w} \right] \tag{4.38}$$

k is the number of flows, $a = \frac{C}{k \cdot \epsilon_0}$, and $w = \frac{r}{\epsilon_0}$.

The function $F_K(s^*)$, derived using the *Chernoff Bound*, is defined as

$$F_K(s^*) = K \left[a \log\left(\frac{a}{w}\right) + (1-a) \log\left(\frac{1-a}{1-w}\right) \right], \tag{4.39}$$

while the expression for $\sigma(s^*)$ is

$$\sigma(s^*) = \epsilon_0 \cdot \frac{\sqrt{k(1-w)we^{\epsilon_0 s^*}}}{1-w+we^{\epsilon_0 s^*}}. \tag{4.40}$$

In order to compute the effective capacity, Equation 4.37 is solved for the target loss-ratio. This will derive the maximum number of flows and from this the *effective bandwidth* for each flow can be computed. Thus, the traffic descriptors of peak rate, mean rate and maximum burst size, along with the buffer space, queue service rate and the maximum number of flows (k) can be determined.

Using this information an estimator can be constructed where E is the *effective bandwidth* estimate, k is the maximum number of flows in progress (as computed using the previous formula), and n is the actual number of flows in progress

$$E = \frac{k}{C} \cdot n. \tag{4.41}$$

4.2.13 Estimator Summary

In the preceding sections estimators based solely upon flow parameters as well as those using measurements have been described. A number of these estimators have their basis in the solution or approximation of the *Chernoff Bounds*, while others approach the estimation problem from different theoretical backgrounds such as large-deviation theory or statistical analysis.

Firstly, a simple estimator was introduced: E-IU, based upon a single instantaneous utilisation measurement E-IU is a useful template estimator that, while not performing particularly well, forms a simple base-reference against which to illustrate the importance of measurement-period (Section 2.4), and compare the effect and differences between admission policy (Section 4.3).

First of the estimators based upon the *Chernoff Bound*, E-CB is the instantiation of one of the family of estimators proposed by [Gibbens97]. In contrast, the measured sum estimator, E-MS, takes a much simpler approach with little theoretical foundation combining a local-maximum prediction with a control over the level of line utilisation.

From the theory of large-deviations, E-MPFE and E-MAE are derived. These algorithms differ in their measurement requirements, but each has the potential to provide estimated bandwidth requirements based directly upon available buffer-space and desired packet loss-ratio.

A family of estimators based upon statistical information derived directly from the measurement of line utilisation is introduced next. E-MVE is a simple estimator using a combination of mean and variance over one time-scale. E-KQ introduces traffic envelopes (descriptions of the mean and variance of traffic over multiple time-scales) and a loss-boundary mechanism that allows computation of the *effective bandwidth* from the traffic envelope. Finally E-GT, a bufferless algorithm, introduces interesting ideas on the separation of time-scales and in particular the relevance of events to the computation of mean and variance.

The last measurement-based estimator, E-LBE, takes a naïve approach based upon the current measurement of loss. While this mechanism does not offer a computation of the *effective bandwidth* requirements, it does provide a prediction of the current loss-ratio.

Finally, three algorithms are described that use only the declared parameters of current flows to compute an *effective bandwidth* estimate. E-GAN, an estimator proposed by [Guérin91], computes the equivalent capacity for traffic with a given set of flow descriptors. E-BD, from [Buffet92], refines the E-GAN estimator by explicitly accounting for the multiplexing that will occur among flows at the buffer — both the E-GAN and E-BD algorithms assume Markovian traffic models with mean burst sizes. The final algorithm is E-EMW, proposed by [Elwalid95]. This algorithm differs significantly from the two previous algorithms because, while it assumes ON-OFF periodic sources, these sources are assumed to be regulated through *leaky-buckets*, so the source descriptor is of a maximum burst size not a mean.

The estimators E-CB, E-BD and E-EMW each specifically reference the *Chernoff Bounds*. While the *Measure* estimators E-MPFE and E-MAE use an exponential approximation of queue service derived from this work. The validity of this assumption (the exponential approximation of queue service) has been called into question: [Choudhury96b] and [Botvich95] have noted both optimistic and pessimistic errors may result from this approximation. Additionally, [Knightly99] notes that algorithms that perform well for Markovian ON-OFF sources can suffer considerably from the inaccuracies introduced when such techniques are applied to multiple-time-scale sources such as VBR video.

The failure of such an approximation can draw into doubt the particular approaches taken in many estimators of *effective bandwidth*. Many of the estimators presented here may also contain the assumption of logarithmic decay of loss-probability as related to buffer size at their root. However, authors such as [Choudhury96b] and [Botvich95] have made suggestions for approaches that improve upon this approximation. Additionally, these approaches may each be expected to fail in a similar fashion (although the results of Chapter 5 indicate a broader range of behaviour than might be expected.)

Table 4.2 lists each MBE studied as part of this work. The table provides an estimate of the overheads for memory, measurement and computation. While only O-notation is given, the table quickly reveals potential problems such as the potential complexity in the E-MPFE versus E-MAE estimators.

MBE	Measurement	Memory	Computation Complexity
E-IU	aggregate load, per-new-flow	None	
	O(1)		O(1)
E-CB	aggregate load, per-new-flow	Traffic Parameters	
	O(1)	O(C)	O(1)
E-MS	aggregate load, continuous	Measurement	
	O(M)	O(M)	O(M)
E-MPFE	per-flow load, continuous	Measurement	
	O(N)	O(N·M)	O(N·M)
E-MAE	aggregate load, continuous	Measurement	
	O(1)	O(M)	O(M)
E-LBE	aggregate loss, continuous	None	
	O(1)		O(1)
E-MVE	aggregate load, continuous	Measurement	
	O(1)	O(M)	O(M)
E-KQ	aggregate load, continuous	Measurement	
	O(1)	O(M)	O(M)
E-GT	aggregate load, continuous	Measurement	
	O(1)	O(M)	O(M $\log$ M)
M = number of bins in distribution/sampling function			
C = number of types of traffic present in system			
N = number of connections			

Table 4.2: Measurement, memory, and computation requirements of Measurement-Based Estimators.

4.3 Policies

The Admission Control Policy, as described in Section 4.1.1, is the procedure an AC algorithm will follow when a new flow is admitted. Such policies may incorporate the results of previous admissions or admission attempts as part of the flow-regulation process. Admission policy does not imply an MBAC algorithm; e.g. the standard Internet AC has a policy of open-admission called best-effort, while at the other extreme the telephony AC will only allow admission if all capacity requirements can be met. However, a number of the complete MBAC algorithm policies share common policy and as a result only five approaches to admission are presented here.

Alongside its description, each policy is illustrated as used in combination with the instantaneous utilisation estimator (E-IU). Each of Figures 4.9 through 4.12 graphically represent the arrival of flow attempts showing their acceptance or rejection and the total number of flows in progress. In the top graph of each figure, a vertical stroke represents a new flow-attempt: if the stroke is down (red), the attempt was rejected; if the stroke is up (green), the attempt is accepted. The lower graph of each figure plots the number of flows in progress versus time. The center graph of each figure illustrates the current line utilisation, and the thresholding value being used where relevant.

For this illustration flows arrive at a mean rate of 100 flows per second (exponentially distributed). Each flow carries a TP10S1 source: a 2-state ON-OFF Markovian source with a peak rate of 10 Mbps and a sustained rate of 1 Mbps (described further in (Section 2.2.1).

4.3.1 P-T — Target

Target refers to a simple admission policy that uses no estimator. It allows a nominated number of flows of a particular type into the multiplex. Target requires a control value of the total number of flows to be admitted will admit flows until that number are accepted. This algorithm is combined with estimators that compute the target value, called the *acceptance region*, and is used here to allow validation of AC results against optimal results. This policy allows construction of an AC similar to the *Quota* algorithm used by [Breslau00] to derive the *Performance Frontier* values for MBAC behaviour.

Figure 4.8 illustrates the Target policy in operation. In this figure AC-T is configured to allow up to 70 flows to be admitted. The variance in the measured line utilisation is noteworthy particularly for this traffic type, TP10S1 with a mean rate of 1 Mbps clearly shows a mean utilisation for 70 flows that hovers at 70 Mbps. However, the peak rate of 10 Mbps is manifest in the significant

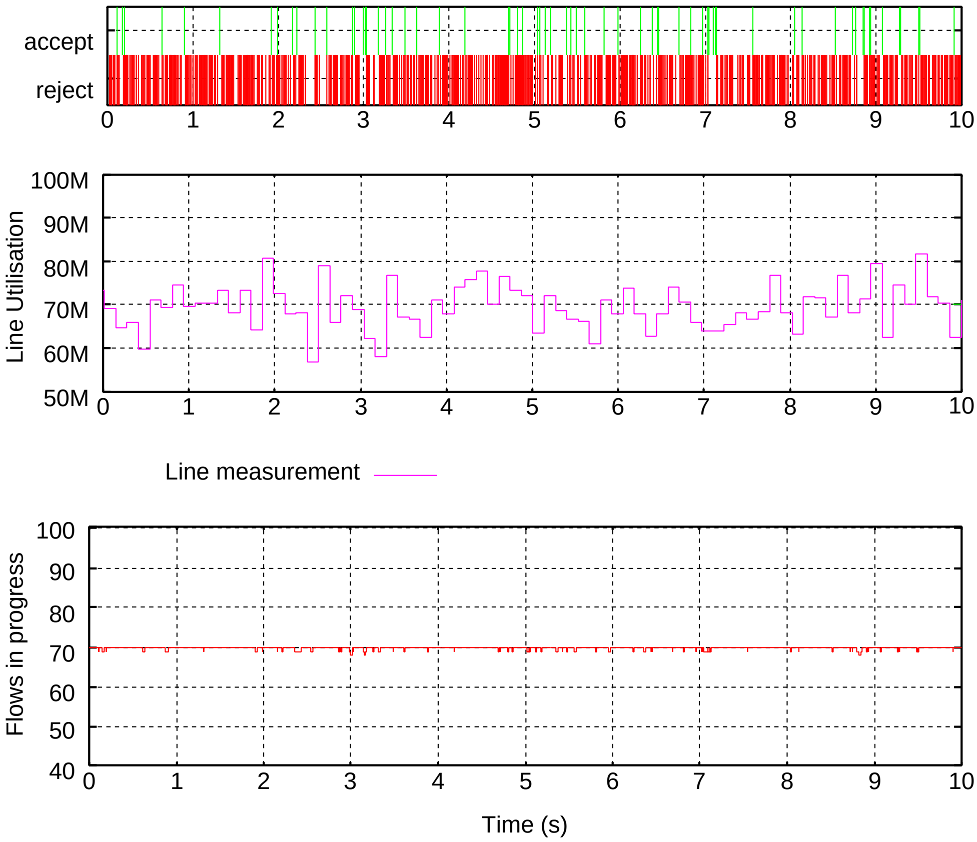


Figure 4.8: Target algorithm (AC-T) using Target policy (P-T.)

variance illustrated for the mean utilisation figure. Figure 4.8 indicates that successfully admitted flow arrivals occur without obvious periodic function, as to be expected of a system that will rely largely upon the (Markovian) departure process. This implies that the variance in line utilisation is due largely to the variance present in the near constant number of sources: either to the variance of the sources themselves or of the measurements illustrated in this figure.

4.3.2 P-TO — Threshold Only

P-TO is a policy that will allow new admissions if a current utilisation value is below a particular threshold. Commonly the value is provided by an estimator such as those described in Section 4.2. An example MBAC algorithm might be the E-IU estimator combined with the P-TO policy. For this combination a new flow is accepted or rejected based solely upon whether the instantaneous measurement is within an acceptable threshold or not.

Figure 4.9 illustrates the combination of P-TO and E-IU. If the current measurement is above the threshold value (70 Mbps) new flows are admitted. The line on the figure *Decision* is intended to indicate the value used by the admission process. In this figure, that line is superimposed upon the line representing current measurements. The figure shows that if the decision (and current measurement) is below the threshold the flow attempt will be admitted. The result is that admission bursts occur over the duration of the current measurement period. Because the measurement process is periodic in nature it is only after a new measurement (above the thresholding value) is available that the admission of new flows will stop. One characteristic of this MBAC algorithm is that for the high rate of admissions of this example, it has significant variation in the number of flows in progress at any time and a corresponding large variation in the measured line utilisation.

4.3.3 P-BP — Back-off Period

The back-off period policy is based upon work of Bean [Bean93, Bean94] who noted that if an AC is under conditions of high load, correct decisions by the AC algorithm alone may not be enough. If the AC algorithm is making admission decisions at a high rate, even a low possibility of an admission made in error may be too high — erroneous admissions will still occur at an unacceptable rate. In this policy if a flow is rejected, the algorithm does not admit another flow of the rejected type until an existing flow of that same type leaves the system. Such a scheme

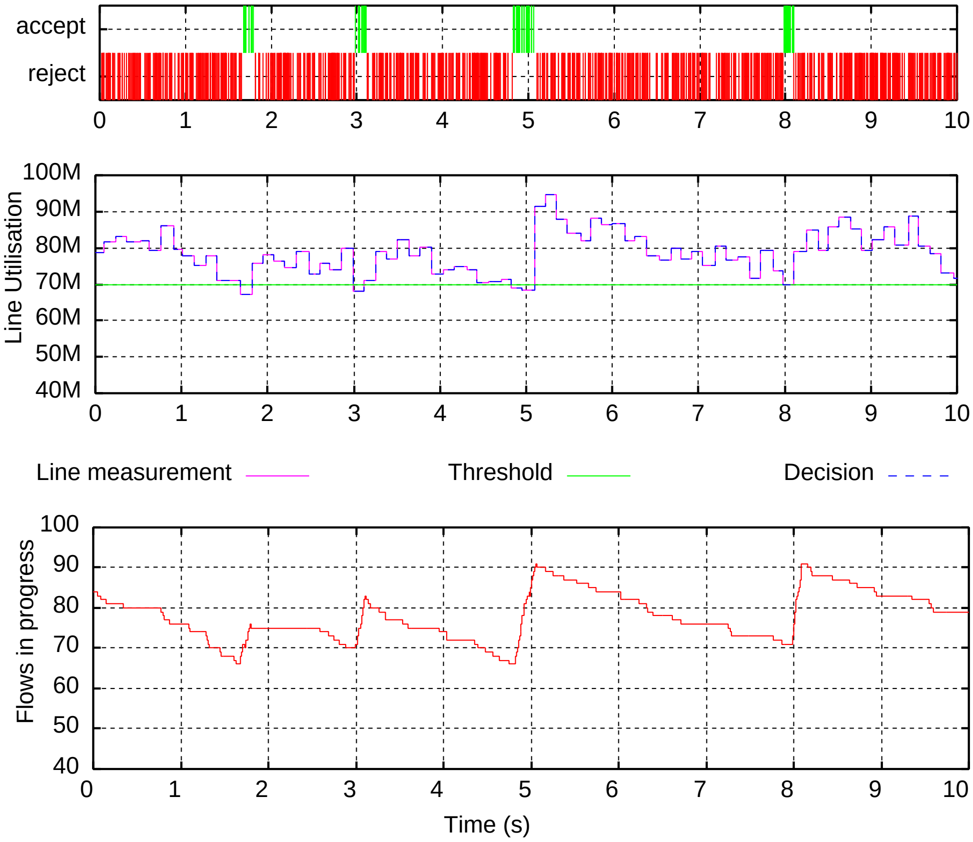


Figure 4.9: Instantaneous utilisation estimator (E-IU) in combination with threshold-only policy (P-TO.)

changes the time scale of the damping characteristics of the admission process, from that of measurements to the time scale of flow lifetimes. Subsequently, this technique proves useful for improving AC behaviour at the scale of flow lifetimes.

When the E-IU estimator is combined with the Back-off Period policy, P-BP, the results illustrated in Figure 4.10 are gathered. The line on this figure marked *decision* also indicates the value used by the policy to compare against admission at the time of a new flow attempt. In contrast to the previous policy, (P-TO), the *decision* line is not superimposed upon the current measurement. Instead, the *decision* line will not be reset to the measurement until a flow has exited the system. As a result the *decision* line has long periods where it is above the threshold despite of the current measurement being below the threshold value.

The P-BP policy will not allow admission of any new flows of the rejected type until an existing flow of that type leaves the system. This means that the stability of the AC algorithm is altered, towards the time-scale of flow-lifetimes. The threshold-only policy also causes a reduction in the size of the admissions bursts because, as compared with the (no) policy illustrated in Figure 4.9, admissions are admitted only once the measurement is below-threshold and a flow has left the system. The effect on admissions can be seen in the top graph of Figure 4.10: the burst of admissions may include between one flow and as many flows can be admitted in one measurement period.

[Bean93, Bean94] noted that the back-off period policy will only have an impact on the behaviour of the admission algorithm if there is significant separation between time scales of new flow-arrival and flow holding times. Section 2.3.1 notes that such a separation of time scales is an assumption central to many AC algorithms. A comprehensive discussion of the relationship between the separation of the time scales of flow-arrival and flow-lifetime and the stabilisation periods of flows and traffic as relates to P-BP has been made in [Bean93, § 4.3]. Bean noted that the period between the arrival of flows, important to allow adequate observation of the current flow-mix, must also be short enough to minimise waste of resources but not so long as to cause the probability of a flow ending in this time to be too high.

Illustrating its adaptability, [Gibbens97] presented a version of this policy that is able to perform active discrimination between particular classes of flows. The mechanism works by preserving

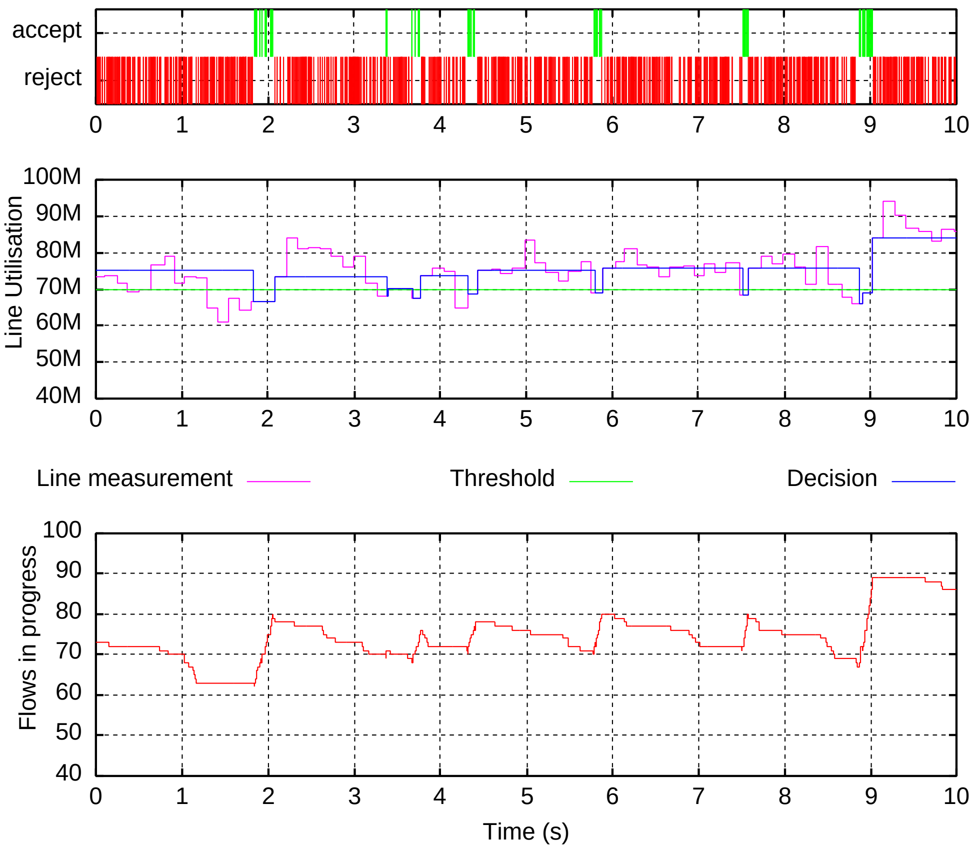


Figure 4.10: Instantaneous utilisation estimator (E-IU) in conjunction with back-off period policy (P-BP).

an absolute priority between different flow types. If a high priority flow departs the system only new high-priority flow attempts will be admitted, while if a low priority flow departs the system both low and high priority flows will be admitted.

4.3.4 P-PA — Pessimistic Admission

Under this policy, new flow requests are (pessimistically) assumed to be transmitting at that traffic's worst-case transmission rate. Once the algorithm can ascertain the actual contribution of the new flow (for an MBAC this commonly follows one measurement-period) the P-PA policy assumes the contribution of the new flow is included in the measurement-based estimate and no longer assumes a peak-rate contribution.

Thus when combined with the E-IU estimator the admission decision is based upon adding the aggregate line utilisation to the peak-rate declared by the new flow, p. If the resulting value is less than or equal to the acceptance threshold then the flow is admitted. Until the estimate has been upated,[1] the algorithm assumes a worst case contribution from all recently admitted flows.

If two flows have been successfully admitted since the last update of the estimate and a third flow attempts admission, the decision would compare $p_1 + p_2 + p_3 + E \leq C$. Where $p_{1,2,3}$ are the peak-rates of the new flows, E is the most recent estimate of line utilisation and C is the capacity of the link.

In Figure 4.11 the result of combining the instantaneous utilisation estimator, E-IU, with the pessimistic policy, P-PA, is shown. For this policy we see that the MBAC *decision* value is based upon the peak-rate of the most recent admissions added to the current measurement value. The most recent admissions are those made after the last measurement was taken. It can be seen in this experiment, where the rate of flow-attempts is very high, that the admissions become periodic, structured around the regular measurement period. This may be compared with P-TO (Figure 4.9) which shows no periodicity at the measurement time scale.

[1] For the E-IU, the estimate of utilisation is updated each measurement period.

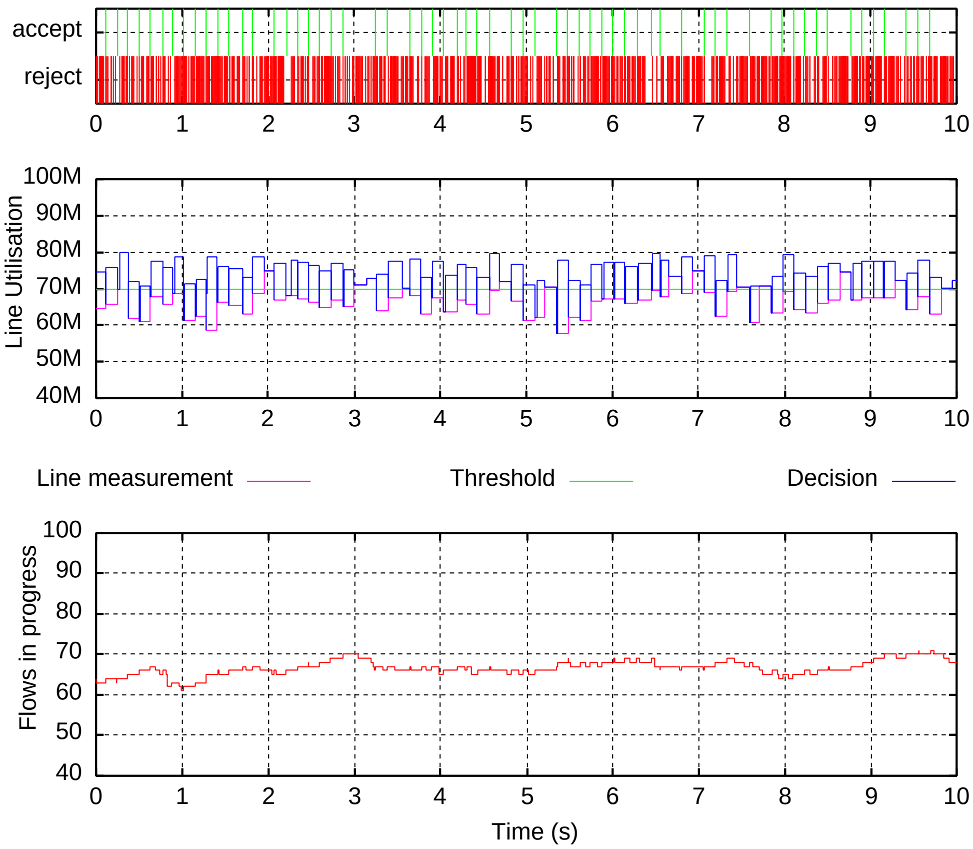


Figure 4.11: Instantaneous utilisation estimator (E-IU) in conjunction with Pessimistic Admission policy (P-PA.)

4.3.5 P-AR — Policy of AC-AR

The policy approach used by the AC-AR of Section 4.4.3, is unique to this AC. It uses the P-BP policy described above but also incorporates the peak-rate declared by the new flow into the admission decision. The admission decision is made based upon whether the current utilisation plus the peak rate declared by the new flow is less than or equal to the line capacity. However, in contrast to the P-PA policy, no memory exists from admission to admission, so the success or failure of a previous flow admission will have no impact upon the behaviour of the policy for future admissions.

For the P-AR policy, this technique of adding the declared peak rate of the new flow is combined with using the back-off period policy described above in Section 4.3.3. Figure 4.12 graphs results when using the hybrid policy of the P-AR algorithm in combination with the E-IU estimator. A threshold of 70 Mbps is used for the admission decision.

The results of Figure 4.12 appear similar to that of Figure 4.10, although as if operating using the P-BP with a lower threshold value. Admissions still occur in bursts, however, once rejecting new flows the system must get both a measurement below the thresholding value and have had a current flow leave the system. Such a combination would partly improve the stability in exchange for lower line utilisation.

Importantly, the authors of this scheme readily admit that its performance appears poorer for the same threshold value (when compared with P-TO, as in Figures 4.12 and 4.9). However, this situation arises because the comparison is made using the same thresholding value, not the same performance outcome. While an approach of combing P-BP and P-PA may be worth investigating, the technique of not artificially increasing the current measurement for subsequent admissions (as per the P-PA policy) is the same approach used by other implementations of this algorithm, notably that reported in [Jamin97c].

4.3.6 Policy Summary

In the previous sections five admission policies for AC algorithms have been described. The policy is the process an AC algorithm follows as part of the admission decision and may incorporate memory from past admission attempts as well as a record of the declared descriptors of flows.

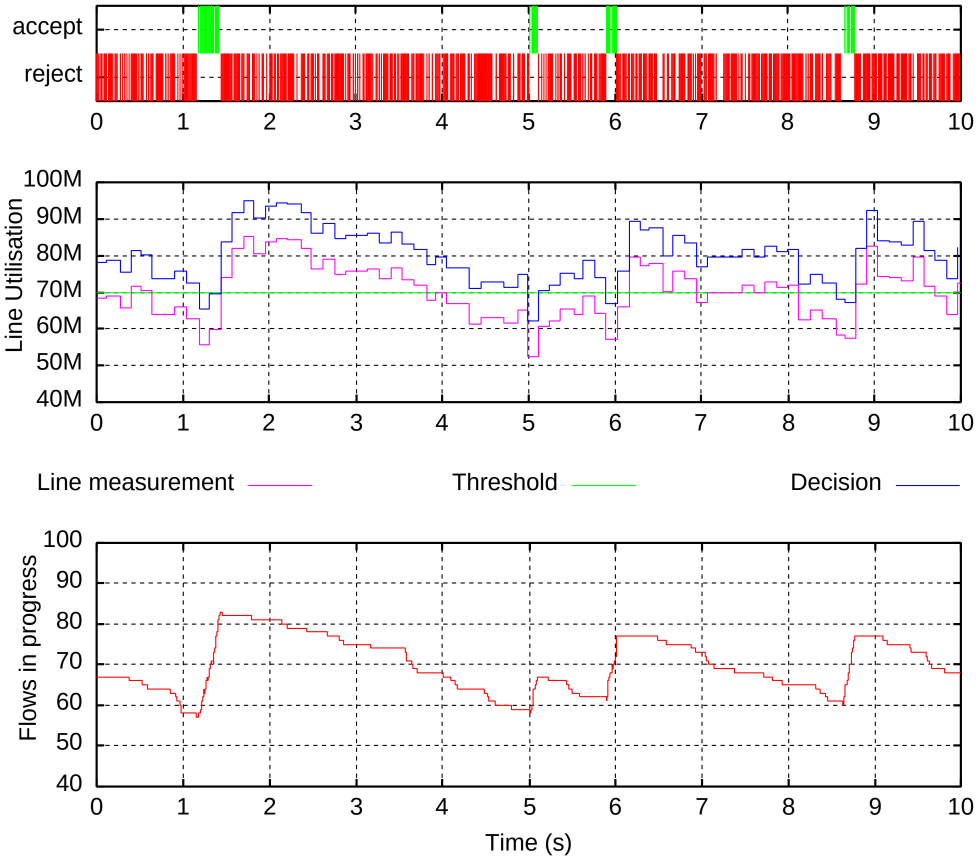


Figure 4.12: P-AR policy, a peak rate addition combined with the back-off period policy (P-BP.)

The precise information maintained by each is policy dependent.

Policy	Quantity	Period	Type
P-T	None		
P-TO	None		
P-BP	O(N)	flow holding-time	traffic descriptor
P-PA	O(N)	period between estimates	traffic descriptor
P-AR	O(N)	flow holding-time	traffic descriptor
N = number of connections			

Table 4.3: Memory and time-scale of admission policies

Table 4.3 lists the policies shared among the AC algorithms presented in Section 4.4. Alongside each policy is indicated the quantity of information each policy would need to maintain, the mean period over which this information would need to be held and the type of information to be stored.

4.4 AC Algorithms

This section outlines a number of AC algorithms including a number of MBAC algorithms along with several other AC algorithms. Table 4.4 lists each admission control algorithm described in this section along with the key idea of that approach and the estimator and admission policy used for its implementation.

In the final column ‘Descriptors’, the per-flow-admission requirements are given. While only the P-PA policy explicitly requires peak-rate, AC-KQ and AC-CB also require the peak-rate of flows as part of the admission process. AC-AR requires *a priori* knowledge of the mean and peak rate traffic descriptors in order to compute the admission surface. The estimators E-GAN and E-BD each require descriptions of the Markovian characteristics of the traffic: the peak rate, sustained rate and mean burst size. In contrast E-EMW requires the parameters used to describe a traffic regulator: peak rate, sustained rate and maximum burst size. It is clear that algorithms that do not require any declarations at time of admission will be simpler to implement.

In this section details of the implementation of each AC algorithm are given along with a rationale

AC algorithm	Key Idea	Policy	Estimator	Descriptors
AC-PRA	Declared Peak	P-PA	-	peak rate
AC-ST	Simple Threshold	P-TO	E-IU	—
AC-AR	Acceptance Region	P-AR	E-IU	peak rate, mean rate
AC-CB	*Chernoff Bounds*	P-BP	E-CB	peak rate
AC-MS	Measured Sum	P-PA	E-MS	peak rate
AC-MPFE	Large-Deviation Theory	P-PA	E-MPFE	peak rate
AC-MAE	Large-Deviation Theory	P-PA	E-MAE	peak rate
AC-MVE	Mean-Variance Estimator	P-TO	E-MVE	—
AC-GT	Time-scale Decomposition	P-TO	E-GT	—
AC-KQ	traffic envelope	P-TO	E-KQ	peak rate
AC-LBE	Loss-ratio	P-TO	E-LBE	—
AC-GAN	Equivalent Capacity	P-T	E-GAN	peak rate, mean rate, mean burst size
AC-BD	Exponential Upper Bounds	P-T	E-BD	peak rate, mean rate, mean burst size
AC-EMW	Effective Bandwidth Model	P-T	E-EMW	peak rate, mean rate, max. burst size
AC-T	Target	P-T	—	—

Table 4.4: Admission Control algorithms as combinations of policy and estimator.

for the approach. Many of the algorithms share a common implementation approach. However, as the performance comparison of Section 5.3.4 intends to illustrate, this does not imply that the computational overheads of each implementation will be the same.

Following the description of each AC, Section 4.4.16 summarises the AC presented along with a comparison of each algorithm based upon a number of criteria including the limitations of its approach and the requirements of the policy and-or estimator.

4.4.1 AC-PRA — Peak-rate Allocation

This algorithm is implemented as a useful comparison point: based upon the peak-rate declarations of flow-attempts, it will admit flows if the declared peak-rate of the new attempt can be admitted in the current allocation. Results gained using this mechanism can be considered as the lower utilisation-bound of any AC algorithm — achieving the best preservation of QoS through worst-case assumptions of the traffic flows.

This algorithm can be considered a special case of the *leaky-bucket* based characterisation mandated for ATM in [ATMF95] and proposed for the Internet INTSERV [Braden94, Wroclawski97]. The expression *leaky-bucket* refers not to an admission algorithm but to a mechanism for characterising traffic. However, it is possible to construct an admission policy and, thus, an AC algorithm based upon the default traffic declarations. Such a policy is better able to characterise the traffic as more complete descriptors are available. This means that if a traffic source is able to describe many aspects such as its sustained rate and its peak rate along with the burstiness characteristics, the algorithm will compute a bandwidth requirement for that source that is more accurate and perhaps smaller than if the traffic source is only able to characterise itself using a peak rate declaration.

An assumption made throughout this work is that incoming sources will only provide the bare minimum of characterisation — declaring a peak rate only. This assumption is based upon the difficulty inherent in a source making a more informative declaration. For some variable bit-rate streams (video, or elastic streams such as those based upon TCP,) computation of sustained (mean) rates and sustained jitter requirements are difficult or impossible to declare. Because of this intrinsic assumption that sources can only provide minimum characterisation, new traffic sources are assumed to be able to provide only the most basic declaration of peak-rate. Additionally, because even a fully specified *leaky-bucket* describes the boundary case for acceptable performance, this may still result in pessimistic admission and subsequent poor resource utilisation.

4.4.2 AC-ST — Simple Threshold

Perhaps the simplest MBAC algorithm is one based upon a thresholding policy such as P-TO, where the user sets an arbitrary admission threshold, combined with the simplest MBE, E-IU.

While the inflexibility of such an approach, using a fixed threshold value, would not see substantial use, this algorithm is useful to demonstrate the principle of MBAC, the behaviour of different policy systems and several problems inherent in the use of measurements. The computation of the appropriate thresholding values can be considered the major work of any MBAC algorithm and [Gibbens95] and [Key95] have attempted to tackle this issue directly using the approach of AC-AR.

Implementation

The simple threshold implementation requests a measurement of current (aggregate) line utilisation synchronously with each new flow arrival. The measurement sub-system (described in Section 3.3.2) has available the latest value measured at the switch. The only overheads are the request and reply of measurements.

4.4.3 AC-AR — Acceptance Region

As noted in Section 2.3.4, the concept of an acceptance region as applied to the number of traffic-flows in progress has existed for some time. The computation of an acceptance region that defines the maximum number of flows of a particular type as permitted is a framework in which AC-GAN, AC-BD and AC-EMW may be placed. An acceptance region as originally proposed in [Hyman91] need not restrict itself to the computation of the maximum number of flows permitted. Rather the acceptance region may be based on the current level of line utilisation using a recent measurement as input.

An acceptance region driven by a measurement of current line utilisation is the approach taken in [Gibbens95] and [Key95]. The acceptance region is computed to maximise line utilisation for a nominated packet loss, given a set of flows with a known declaration of peak and mean rates. One aspect of the approach proposed by [Gibbens95, Key95] is that the system is robust to mis-specification of the mean and peak rates as well as being robust to the errors in measurement.

The scheme itself is a memory-less MBAC, the outcome of previous events (e.g. admission attempts, or measurements) having no direct influence over the current admission decision. A perfect time-scale separation is assumed, with the network states seen by successive flow-arrivals considered independent. As a result such a system may suffer when placed in conditions where correlation exists across several time-scales.

The combination of a Bayesian prior describing the flow statistics and the P-BP flow rejection mechanism serve to improve the performance of this algorithm. The Bayesian prior will smooth-out the fluctuation in successive estimates. The P-BP flow-rejection policy (whereby following a rejection a new flow is not admitted into the network until one has left the system) acts to counter measurement variance in the face of very high arrival rates. Rather than using a sequential Bayesian approach that updates the posterior distribution with each successive observation of load, both [Gibbens95] and [Key95] provide a decision theoretic framework that gives the approach its name. This algorithm assumes that the burstiness of a traffic source is relatively stable without a significant loss in the robustness of the algorithm, resulting in a system requiring minimum characterisation by either the sources or the flow-arrival process.

Implementation

Like AC-ST, in operation the Admission Region algorithm has few operating overheads requiring only current (aggregate) utilisation measurement. The most significant effort is the computation of the acceptance region, although following discussion with the authors of [Gibbens95] the analysis of this AC algorithm did not need the explicit computation of the Admission Region. Instead, the admission region could be inferred from several consecutive experiments run using different admission thresholds. This same inference approach was also adopted in a comparison made using this algorithm by [Jamin97c].

4.4.4 AC-CB — *Chernoff Bounds*

The MBAC algorithm implementing the *Chernoff Bounds* approach uses the *Tangent at Slope One* estimator described in Section 4.2.2. The admission algorithm proposed by [Gibbens97] consists of the back-off period policy, P-BP combined with an admission decision.

For the admission decision, X represents the current aggregate-load measurement, α is the algorithm control parameter, p_k is the peak-rate of class k and n_k is the number of flows present in class k. This expression is compared against the line capacity C. If the estimate is less than or equal to the line capacity, the new flow is admitted. The formula itself is given as:

$$X + \frac{\alpha}{4} \sum_{k}^{K} p_k^2 n_k \leq C. \tag{4.42}$$

Importantly as part of the new admission attempt the appropriate class counter n_k is (artificially) incremented for this test. If the test succeeds, sufficient capacity exists to allow the new admission to be made. If the test is unsuccessful the modified class counter is returned to its previous value.

Two parameters control this algorithm: the control parameter α, which is the algorithm-supplied parameter to control the quantity of resource over-commitment, and the length over which the measurement X is made. The importance of the selection of the measurement-period is mitigated by the use of the P-BP: if the period does not allow good characterisation of the aggregate flow-mix, the stability of the admission process is still strongly influenced by flow-departures. Because P-BP will not allow new flows to enter the system until a similar-class flow has departed the system is expected to reach and maintain stable operation effectively.

Implementation

AC-CB is implemented as a single component system, the estimation of bandwidth requirements is computed using the current utilisation measurement and the declared peak-rates of current flows for each new flow admission.

4.4.5 AC-MS — Measured Sum

The Measured Sum algorithm is based upon a simple admission formula, Equation 4.43. Referring to the explanation of this estimator in Section 4.2.3, part of the complexity of this method lies in the computation of the line-load estimate.

Given that $\hat{X}$ is the measured load of existing traffic, p is the peak-rate of the new flow seeking admission, C is the link capacity and μ is the user-defined utilisation control, the admission decision can be based upon:

$$\hat{X} < \mu C - p. \tag{4.43}$$

The Measured Sum estimator of Section 4.2.3 is combined with the pessimistic admission policy, P-PA. As a result following admission, $\hat{X}$ is (temporarily) increased by the value of p. This temporary increase is removed once a new estimate of existing traffic load is available.

As an MBAC algorithm Measured Sum algorithm uses a modified version of the load-estimator; recall that a procedure the algorithm authors call *time-window* is used to capture the maximum value during a characterisation period. Alongside the explanation of the estimator in Section 4.2.3, Figure 4.3 illustrated how the time-window mechanism worked. Augmenting this approach for admission-control, the modified time-window mechanism (adapted from [Jamin97c]) is illustrated in Section 4.13.

Note how the new flow causes an immediate reset of computation period $T \cdot \tau$ as well as temporarily increasing the working utilisation value by the peak-rate of the new flow. The peak-rate of the new flow is assumed to be incorporated into the estimate once the characterisation period $T \cdot \tau$ has passed.

Implementation

The implementation of the Measured Sum MBAC algorithm uses a back-end and front-end.[2] Such an approach was necessary as the estimate of current use based upon a local maximum predictor requires a continuous supply of utilisation measurements. A suitable approach was

[2] Front-end and back-end refer to the separation of the algorithm into two different processes. In one process the front-end tasks are performed, e.g. admission policy, while tasks such as continuously requesting measurements and computing estimates are performed in the second back-end process.

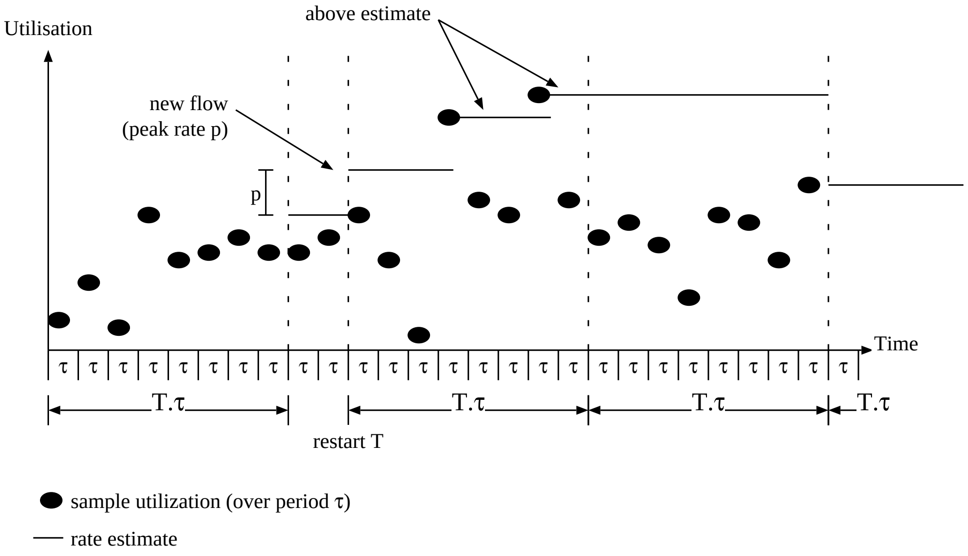


Figure 4.13: Impact of new flow on utilisation measurement.

to decouple the computation of the local-maxima from the estimation of *effective bandwidth*, Equation 4.3. The current value of local-maxima is communicated between front and back-end via shared memory.

4.4.6 AC-MPFE — *Measure* Per-Flow Estimator

For the *Measure* per-flow MBAC algorithm the combination of the MPFE-E and P-PA give rise to an implementation where E is the current estimate arrived at using the mechanism described in Section 4.2.4, p is the peak-rate declaration of the flow attempt and C is the total line capacity.

$$E + p \leq C. \tag{4.44}$$

In this equation the P-PA policy is implemented by increasing the comparison value by the peak-rate of the new flow. Once the new flow is admitted, the estimate, E, is also artificially increased by p until an up-to-date version of the current estimate is available.

Implementation

This algorithm requires continuous access to measurements of current utilisation. In contrast with other estimators, this algorithm requires that the measurements be made per-flow. The front-end process for AC-MPFE is a simple decision while the back-end process retrieves and manipulates the required measurements continuously computing a current value for the total *effective bandwidth* requirements of the system.

As is the case for all implementations separated into back-end and front-end components, communication is performed via common data structures held in a shared memory segment.

4.4.7 AC-MAE — *Measure* Aggregate Estimator

While the estimators differ, the *Measure* aggregate-based MBAC algorithm shares much in common with the *Measure* per-flow-based MBAC algorithm. Like the *Measure* per-flow-based MBAC algorithm, the combination of the E-MAE and P-PA uses an admission decision based upon Equation 4.44.

Also like the AC-MPFE, in this equation the P-PA policy is implemented by increasing the comparison value by the peak-rate of the new flow. Once the new flow is admitted the estimate, E, is artificially increased by p until an up-to-date version of the current estimate is available.

Implementation

The AC-MAE implementation is identical to the AC-MPFE implementation except in one important detail. The traffic samples used in the computation of the *effective bandwidth* estimate of Equation 4.4 are based on aggregate line utilisation rather than the sum of the current flow contributions. This reduces the complexity of the algorithm by simplifying the requirements of data management. However, the issues of the upper bounding of the history period are required as part of the back-end process. Like AC-MPFE, the front-end implementation is a simple decision process incorporating P-PA, along with some record-keeping required to compute the mean holding time of flows as well as the number of flows in progress. These figures are required by the back-end process to compute the boundary on the total block history.

4.4.8 AC-MVE — Mean-Variance Estimator

Section 4.2.6 described how an estimator may be constructed that used measurements of mean utilisation and variance. The E-MVE estimator is combined with the simple P-TO, threshold-only policy to create an admission algorithm.

Thus for a given mean $\bar{x}$, standard deviation σ, and capacity C,

$$\bar{x} + \alpha' \sigma \leq C. \tag{4.45}$$

The value of α' derived from α is scaled to allow for sampling errors and errors that arise because the duration of characterisation is too small. α' is given by

$$\alpha' = \max\{\alpha, (\sqrt{(1+T)(e^{\frac{\alpha^2}{T}} - 1)}\}, \tag{4.46}$$

where T is the number of samples taken. The value of α allows the algorithm to be adjusted to accommodate more or less variance in the estimate, and thus admit fewer or greater flows in the admission algorithm.

Implementation

The Mean Variance Estimator uses a back-end process to continuously collect measurements as these become available, thereby maintaining an up-to-date estimate of the mean and variance. The front-end component is invoked only for each flow-admission, and at this time the mean, variance and a pre-specified value of α are combined to compute a current estimate of utilisation for use by the admission decision.

4.4.9 AC-KQ — Traffic Envelope

For the AC implementation of AC-KQ using the estimator described in Section 4.2.7, it is useful to construct a traffic envelope of flow to be admitted. This worst-case description of the new flow, R_k^β, is computed from the peak rate of the new flow (p) and the minimum measurement interval of the envelope, τ, using

$$R_k^\beta = \frac{p}{k \cdot \tau}, \tag{4.47}$$

where $k = 1, 2, \ldots, T$. T describes the number of intervals over which the envelope is computed.

In the MBAC algorithm proposed in [Knightly96] the traffic envelope of the new flow can also incorporate other declared parameters (such as mean rate). However, for the comparison of this dissertation, MBAC algorithms only have access to the peak rate of new flow attempts.

From the E-KQ estimator of Section 4.2.7 an admission algorithm is constructed with a new flow described by R_k^β requesting admission to a system with capacity C and buffer size of B. The use of a traffic envelope describing the new flow allows the new flow properties to be directly incorporated into the admission equations.

The traffic envelope approach separates the admission process into two parts. Firstly for the short time-scale events consider the buffer dynamics of the multiplexer. The following equation (from Equation 4.11) ensures the buffer resource is not over committed. In this equation the traffic envelope derived from measurement is represented by $\bar{R}_k$ and σ_k, C is the capacity of the link and B is the buffer capacity of the multiplexer giving the expression

$$\max_{k=1,2,\cdots,T}\{k\tau(\bar{R}_k + R_k^{\beta} + \alpha_{\mathrm{short}}\sigma_k - C)\} \leq B. \tag{4.48}$$

The α_{short} in this expression is the upper bound on the probability of queue over-flow and was given in Equation 4.12.

The second part of this algorithm is the long time-scale, and a comparison is made between the mean resource requirements of the new flow and current flows. The current requirements are represented by $\bar{R}_T$ and σ_T. Importantly, the time-scale of interest is that of $T \cdot \tau$ the maximum characterisation period. The following equation allows an admission decision to be constructed to admit flows if sufficient long-term resource is available,

$$\bar{R}_T + R_T^{\beta} + \alpha_{\mathrm{long}}\sigma_T \leq C. \tag{4.49}$$

The value of α_{long}, is given earlier in Equation 4.10.

Such an MBAC algorithm incorporates an admission policy equivalent to P-TO: given the traffic envelope of the new flow (R_k^{β} and R_T^{β}) is explicitly incorporated into the admission process no explicit adjustment is made to values following admission nor is any history of admission (success or failure) kept.

Implementation

Like the AC-MVE algorithm, this system continuously collects measurements, these are used by the back-end to maintain up-to-date values for R_k^1, σ_k^2, $\bar{R}_T$ and σ_T. While the traffic envelope for the current flows can be maintained continuously, the complete computations can only be performed at the time of each admission. This is because the values of R_T^{β} and R_k^{β} are derived from the declared peak rate of the new flow attempt. This also gives an order of the front-end computation the same as the order of the back-end computation, the order being derived from the size of the samples array used to contain the traffic envelope.

Following [Qiu98a], a value of $T \cdot \tau$ that limits the total traffic envelope to 2 seconds is used throughout the work of this dissertation.

4.4.10 AC-GT — Time-scale Decomposition-based Estimator

The E-GT estimator, detailed in Section 4.2.8, provides an estimate of the variance and mean of a single (theoretical) flow from the current aggregates. Equation 4.50 illustrates how this value can be used to compute an admission decision. In Equation 4.50, $\hat{\sigma}_t^H$ is the standard deviation of the high-pass measured components of a single flow in the measured aggregate (Eq. 4.18,) while $\hat{\mu}_t$ is the estimated slow time-scale mean (Eq. 4.16.) The line capacity is C, while N_t is the number of flows currently present in the multiplex — the admission decision is based upon a test that uses $N_t + 1$:

$$\alpha\sqrt{(N_t + 1)}\hat{\sigma}_t^H + (N_t + 1)\hat{\mu}_t \leq C. \tag{4.50}$$

The value of α, like the AC-KQ and AC-MVE estimators, controls the tolerance to variance of the admission algorithm. Like AC-KQ, α is computed from the overflow probability via an inverse complementary CDF of an N(0,1) Gaussian distribution (e.g. Equation 4.21).

Implementation

The AC-GT, is separated into a back-end and a front-end component. In the implementation made for this work, the front-end is a simple decision process using a P-TO threshold-only policy.

While in most respects the computations of AC-GT are near-identical to those of AC-KQ, an asymptotic order of complexity is as a result of the convolutions of Equations 4.16 and 4.18 performed using a Fast-Fourier Transform implementation.

4.4.11 AC-LBE — Loss-Based Estimator

A simple admission algorithm can be constructed from the loss-based estimator E-LBE of Section 4.2.9. Consisting of a test against a user-specified level of loss, ϵ, the admission algorithm given in

$$\epsilon \leq \hat{\epsilon} \tag{4.51}$$

relies only upon the current loss-estimate, $\hat{\epsilon}$.

Implementation

To compute a continuously available exponentially weighted moving average (EWMA) as required by this estimator, measurements are regularly collected and the new estimate computed by a back-end process. The front-end process performs a simple admission test using the most recently computed loss estimate.

4.4.12 AC-GAN — Equivalent Capacity

The estimator proposed in [Guérin91] and outlined in Section 4.2.10 can be converted into an admission algorithm. Given the parameters defining the configuration (buffer space, line speed), the QoS provision the user desires (the target loss-ratio), and needing the parameters describing the traffic in the multiplex, the maximum number of flows can be pre-computed. The AC process becomes a simple admission equation

$$n + 1 \leq N_{\max}, \tag{4.52}$$

where n is the current number of flows in progress and $N_{\max}$ is the pre-computed maximum.

If Equation 4.52 is satisfied for the current number of flows in progress (n), a new flow will be admitted. Aside from this admission decision the admission policy does not perform any other tasks.

Implementation

The P-TO target policy performs the admission policy of Equation 4.52, for a user-supplied value of $N_{\max}$. As noted above, this value is computed off-line with an *a priori* knowledge of traffic parameters. The computation of the $N_{\max}$ value is via the equations of Section 4.2.10.

4.4.13 AC-BD — Exponential Upper-Bounds

The AC-BD, like the AC-GAN, uses the buffer-size and line capacity, along with *a priori* characterisation of traffic and a user specified loss-ratio to compute a maximum number of flows that may be admitted. This pre-computed value for the maximum number of flows may then be used in an admission process based upon Equation 4.52.

Implementation

The implementation is similar to that of AC-GAN, for a user-supplied value of $N_{\max}$ the admission algorithm described by Equation 4.52 is performed.

4.4.14 AC-EMW — Effective Bandwidth Model

Like both AC-BD and AC-GAN, given values of buffer-size and line capacity, along with *a priori* characterisation of traffic and a user specified loss-ratio, the maximum number of flows that may be admitted can be pre-computed. The estimate of the number of flows is computed off-line for the desired loss-ratio using the estimation procedure outlined in Section 4.2.12. The maximum number of flows may then be used in an admission algorithm described by Equation 4.52.

Implementation

The operating AC algorithm is similar to that of AC-GAN: for a user-supplied value of $N_{\max}$ the admission algorithm described in Equation 4.52 is performed.

4.4.15 AC-T — Target

Policy ‘Target’ refers to a simple admission algorithm that will allow a nominated total number of flows in the multiplex. The Target AC allows the user to nominate the maximum number of flows to be admitted.

Implementation

The Target policy is configured to admit a maximum number of flows, $N_{\max}$. Using Equation 4.52, a new flow is admitted if the current value of n is below the target value. For the use of AC-T in this dissertation the AC is configured to admit a maximum number of total flows; no bias is configured towards one flow type in favour of another.

4.4.16 AC Summary

The above AC algorithms are summarised in Table 4.5 along with the key idea behind each algorithm and several comparison criteria.

Whether the gains made though buffering are taken into account by an AC is indicated by the column ‘Buffer Effect’ of Table 4.5. The Rate Envelope Multiplexing (REM) approach describes where the effect of buffering is not taken into account, while Rate-Sharing Multiplexing (RSM) takes into account gains made through buffering. For RSM the combined rate at which data enters the buffer may exceed the buffer service-rate for small periods before packet-loss will occur. Because of this, the burst rate and burst duration of sources play an important contribution in the computation of the *effective bandwidth* of sources. For REM the burst duration and burst rate need not be considered. As a result, the *effective bandwidth* of the loss is computed from only the sustained rate and the peak rate. For algorithms that are measurement-based the incorporation of an RSM approach may be implicit, such as that of AC-MS or explicit such as AC-KQ or AC-MPFE.

Following [Shiomoto99], the second criteria of Table 4.5 is whether the AC algorithm evaluates admission based upon a loss-ratio or upon an *effective bandwidth*. This describes the criteria of the decision process: if the characteristic is met then a new flow is admitted, otherwise it is rejected. Only one MBAC algorithm (AC-LBE) explicitly uses the loss-ratio as its admission criteria. However, the AC-GAN and AC-EMW algorithms each use a root-solver based upon finding the maximum number of flows admissible for a particular loss-ratio, and in light of this may both be considered as loss-ratio based.

The ‘Measure or Describe’ criterion indicates which algorithms are based upon an MBE or use declared traffic descriptors alone. AC-GAN, AC-BD and AC-EMW each use the full set of traffic descriptors available, while AC-PRA uses the peak-rate declaration alone. The approach of AC-AR deserves special mention as the computation of the acceptance region requires the *a priori* declaration of the descriptors of traffic, while the admission process itself is based upon a measurement of current load.

Finally, Table 4.5 lists which of the MBAC algorithm are based upon Certainty Equivalence. The concept of Certainty Equivalence in AC algorithms was introduced in [Tse99]. These methods use a static AC algorithm but insert measurement derived estimations rather than those computed from *a priori* traffic descriptors. The attraction in this approach is the ability to reuse existing ACs. However, as [Grossglauser97b] point out, CE methods may give too optimistic results due to the reliance upon measured quantities. The random nature of measured quantities must be incorporated into the algorithm. As noted in Section 2.4, several MBAC algorithms document specific solutions to this measurement problem [Gibbens95, Key95, Knightly98, Grossglauser97b]. The approaches of [Grossglauser97b] and [Knightly98] incorporate computation of the measurement variance directly while in [Gibbens95, Key95] a solution is provided using an estimator based upon a Bayesian model that incorporates the error as the prior is developed.

AC algorithm	Key Idea	Buffering Effect	Control	Measure or Describe	Certainty Equivalent?
AC-PRA	Declared Peak	REM	Bandwidth	Describe	—
AC-ST	Simple Threshold	RSM	Bandwidth	Measure	Yes
AC-AR	Acceptance Region	REM	Bandwidth	Both	No
AC-CB	*Chernoff Bounds*	REM	Bandwidth	Measure	Yes
AC-MS	Measured Sum	RSM	Bandwidth	Measure	Yes
AC-MPFE	Large-Deviation Theory	RSM	Bandwidth	Measure	Yes
AC-MAE	Large-Deviation Theory	RSM	Bandwidth	Measure	Yes
AC-MVE	Mean-Variance Estimator	RSM	Bandwidth	Measure	No
AC-GT	Time-scale Decomposition	REM	Bandwidth	Measure	No
AC-KQ	Traffic Envelope	RSM	Bandwidth	Measure	No
AC-LBE	Loss-ratio	RSM	Loss-Ratio	Measure	Yes
AC-GAN	Equivalent Capacity	REM	Bandwidth	Describe	—
AC-BD	Exponential Upper Bounds	RSM	Loss-Ratio	Describe	—
AC-EMW	Effective Bandwidth Model	RSM	Loss-Ratio	Describe	—
AC-T	Target	—	—	—	—

Table 4.5: Admission Control algorithms as combinations of policy and estimator.

Comparisons among MBAC algorithm are conducted in the next Chapter on the basis of experimental results.

Chapter 5

Comparing MBAC algorithms

5.1 Introduction

In the previous chapter, a number of AC algorithms were presented. The theoretical foundations along with an analysis through decomposition was presented for ten MBAC algorithms along with several other AC approaches. This chapter presents the results of comparisons made between the MBAC algorithms presented in Chapter 4. The comparison is implementation based, offering fidelity with real applications as well as an insight into aspects such as the algorithmic complexity, overheads, and performance in an actual AC system.

5.1.1 Objectives

The aims of this chapter are to compare the approaches taken by the MBAC algorithms of Chapter 4, contrasting the different MBAC algorithms as well as other approaches to AC presented in that chapter. The objective of this comparison is to identify the ideal MBAC algorithm, an algorithm that is simple to implement, simple to operate, requires the minimal overheads in memory, CPU or results, and gives the greatest flexibility in the situation it is able to manage.

5.1.2 Structure

The remainder of this chapter is structured as follows. Firstly, Section 5.2 discusses the comparison criteria, summarises the evaluation environment and outlines the experiments used. Along with an outline of each experiment, is a justification for the particular selection of traffic, flow arrival and flow lifetime characteristics. Section 5.3 presents the results of the MBAC algorithm comparisons.

Following the presentation of results, Section 5.4 discusses those results and draws conclusions from the MBAC algorithm comparison work. Alongside this discussion, the interaction between MBAC algorithm and timescale is studied further. Finally, Section 5.5 summarises the findings of this work, addressing the objective of an ideal MBAC algorithm.

5.2 Method

This section outlines the method adopted in the examination and comparison of AC algorithms presented in this chapter. Throughout this study it is assumed that network users will make requests for new flows be admitted using a protocol such as RSVP [Braden97] or ATM Forum's signaling specification version 4.0 [ATMF95]. Under these protocols, each service request contains a traffic descriptor of the worst-case behaviour of the traffic requesting admission.

The unique characteristic of this investigation has been that the results are gained using implementations of the algorithms not in a simulator but in an experimental network. An implementation of each algorithm has given access to performance and behaviour aspects of each algorithm not available to a simulation. Relevant aspects of the implementation environment, presented in Chapter 3, along with the experimental approach are discussed in this section.

This section also outlines the evaluation environment in Section 5.2.2, including broad assumptions common to each experiment scenario, then each experiment configuration is outlined in Section 5.2.3 through 5.2.10.

5.2.1 Criteria

A variety of criteria for identifying the *best* MBAC algorithm have been put forward by previous authors. Suggestions for comparison criteria have included packet-loss versus utilisation, flow-acceptance rates versus utilisation and packet-delay versus utilisation (e.g. [Jamin97b, Jamin97c, Knightly98, Qiu98b, Breslau99, Breslau00]). This chapter presents results of comparisons made using packet-loss versus utilisation, a common method for illustrating AC algorithm performance.

Packet-loss versus utilisation results are presented for several experiments, each of which uses different combinations of flow characteristics (arrivals and holding times) carrying a variety of traffic types. The experiment configurations are outlined in Sections 5.2.3 through 5.2.10.

Each MBAC algorithm studied here incorporates parameters to control the performance of the algorithm. The exact control exercised both for algorithms that purport to have a relationship between control parameter and performance, and for those algorithms that do not have such a relationship provides a useful comparison. The performance profile of each MBAC algorithm is studied for a range of control-values using each of the experiment configurations of Sections 5.2.3 through 5.2.10 to ensure coverage across a range of different flow and traffic conditions. As these control parameters affect the desired performance criteria, comparing the performance allows an insight into algorithm behaviour.

In operation, an MBAC algorithm must be able to detect changes in resources or requirements. While such detection is intrinsic to an MBAC algorithm, it may be speculated that precisely how each algorithm responds to such change will differ. An investigation of results for a number of stability experiments is presented in this chapter and the details of each method are discussed alongside the results.

Closely associated with the stability of a system is the repeatability of results. With an automated test-environment, it is possible to rerun an identical experiment many times over; allowing the repeatability of an outcome to be assessed statistically. While the error-margin of the test environment itself will affect the results, a comparison of the repeated results reveal MBAC algorithm-dependent characteristics.

The final comparative analysis of the MBAC algorithm contrasts the resource requirements of each algorithm. Instrumentation internal to the test-environment is able to reveal the resource consumption of each MBAC algorithm component. Alongside such resource comparisons is a table presenting an overview of each MBAC algorithm's complexity. A comparison of such data allows a useful insight into the actual overheads required for implementation of what, in some cases, have heretofore been only theoretical MBAC proposals. Tabling the order of measurement requirements and memory overheads allows further comparison to be made between algorithms. Finally, taking further the issue of CPU overhead, the impacts of overheads for one particular MBAC algorithm are examined in detail allowing insight into assumptions about MBAC algorithm operation and performance.

5.2.2 Environment

The experimental environment used in the evaluations of this Chapter is described in Chapter 3. However, an outline of the relevant aspects of the configuration are summarised in this section. The test-environment allows for the implementation of an AC algorithm in a pre-existing framework of flow generation, traffic generation, network and measurement systems. This modularised environment allows a direct comparison of one AC against another when each is placed under identical connection loads and traffic types. Additionally, a comparison of consecutive experiments is made possible using one AC algorithm but adjusting tuning parameters for that algorithm over consecutive runs.

A test-environment allows a faithful comparison with the real implementation — the ability to assess real implementation issues of memory capacity and algorithm speed. Additionally, common modelling techniques use simulated sources of traffic — both well understood and easily generated. However, due to the variety of sources and the continual development of new network users, such simulated sources of traffic are not adequately representative. Testing of the algorithm may be considered more realistic using a test environment with real traffic as well as that based upon models.

For each experiment in this chapter the buffer offers an undifferentiated FIFO tail-drop service; the packet-loss due to buffer overflow will be born by one or more flows currently in progress without any differentiation in the buffer between the flows themselves.

Aside from providing a realistic work load, the mean rate of flow-admission attempts is selected to ensure the AC algorithm under test is placed under a high load. This is needed to ensure that each test contains enough attempts for a meaningful comparison to be made.

5.2.3 Exp1 — TP10S1 — 2-state Markov ON-OFF

Using the TP10S1 traffic type (described in Section 2.2.1), each experiment consists of a Markovian process initiating new flow attempts with a particular mean attempt rate. Each flow attempt is independent of every other flow attempt. The lifetime of a flow is based upon a negative exponential distribution with a mean of 10 seconds unless otherwise noted.

The purpose of this experiment was to validate the behaviour of algorithms under the ideal circumstances of Poisson traffic and to contrast results with those gained using static-solution AC algorithms. The buffer at the bottleneck has a capacity of 512 packets (27136 octets) and a service-rate of 100 Mbps. 27136 octets is a common buffer size found in the line-card interfaces of commercial switch equipment used in the test-environment [FORE99].

Exp1 sees common use throughout this chapter not only as one of the set of different experiments, but also in individual tests where such parameters as measurement-period or the impact of flow arrivals require individual attention. This configuration is useful because it consists of Markovian flows carrying a Markovian source, thereby providing the simplest system for each AC algorithm to manage.

5.2.4 Exp2 — PP10S1 — 2-state Pareto ON-OFF

The network configuration used in Exp2 is a duplicate of that in Exp1. The experiments differ in the traffic source used. Rather than the TP10S1 source, Exp2 uses the Pareto PP10S1 traffic source described in Section 2.2.2. With this experiment, the intention was to allow a direct comparison with Exp1, changing only the traffic type from that of a predictable Poisson model to traffic that exhibits long-range dependence through the multiplex of traffic with heavy-tailed distributions.

5.2.5 Exp3 — RP10S1 — LAN

The RP10S1 Internet traffic source detailed in Section 2.2.6 is used in a network with a configuration (buffer size, link bandwidth) identical to that of Exp1 and Exp2. However, for this experiment the time between flow arrivals is dictated by a Pareto distribution with a mean of 12.5 flows per second, and a shape of 1.2. These parameters are selected to maintain a sufficiently high rate of arrivals and provide the arrivals process with LRD properties. The holding time of flows had a mean 160 seconds and was based upon a log-normal distribution, following [Bolotin94].

It is unclear how appropriate it is to use the flow-lifetime characteristics from ISP dial-ups as representative of the flow-lifetimes for an internal IP network because most experiments concentrate upon the lifetime of the single flow of a TCP connection. However, given a lack of more relevant experimental results, the ISP lifetime characteristics appeared a useful approximation.

5.2.6 Exp4 — VP25S4 — Video

Using the video traffic source VP25S4 (Section 2.2.5) this experiment presents AC algorithms with flows at a mean rate of 12.5 flows per second following a Pareto distribution with a mean of 1.2 seconds. The shape of the distribution was 1.2, selected to present an arrival process with LRD properties. The holding time of the flows was based upon a log-normal distribution with a mean 300 seconds, also following [Bolotin94].

The rationale for this experiment was that while this distribution had a low flow lifetime in comparison to a typical movie-length, the use of the log-normal distribution encapsulated the heavy-tail properties illustrated in [Bolotin94]. High flow arrival rate is required to ensure the AC algorithm will face significant numbers of flow attempts during the lifetime of an experiment. Thus, while having a small mean flow lifetime, these video streams may be envisaged as music-video clips, video phone calls, trailers for movies, or as a result of ‘channel-flipping’ behaviour where a user is locating a desired video programme.

5.2.7 Exp5 — RP10S1/VP25S4 — LAN with Video (long flows)

Using a mixture of the conditions of Exp3 and Exp4, this experiment was conducted using the RP10S1 Internet traffic source and the VP25S4 video traffic source, the experiment was a combination of Exp3 and Exp4 emulating a local-area network used for both computer intranet communications and for the dissemination of video material.

The configuration was intended to emulate the situation faced at a border switch, which must multiplex a number of heterogeneous lower rate sources onto a higher capacity link. This situation could reasonably have been expected to exist when common desktop bandwidth (e.g. 10 Mbps Ethernet) was substantially lower than backbone or intra-office capacity (e.g. 100 Mbps Ethernet or 155 Mbps OC-3 Sonet).

5.2.8 Exp6 — RP10S1/VP25S4 — LAN with Video (short flows)

Using a network configuration similar to the previous Exp5 experiment, this experiment also uses a combination of the VP25S4 video traffic and the Internet traffic source RP10S1.

However, in this study the flow-holding time and the time between flow attempts is based upon an exponential distribution with a 10 second mean. The rate of flow attempts are distributed evenly between each of the traffic types using a mean of five flow-attempts per second of either traffic type. The arrival process of each traffic type is independent.

5.2.9 Exp7 — EP6S480k/VP64S64/VP64S23 — LAN with Voice

The design of Exp7 and Exp8 was to emulate issues faced in admission control in an ADSL facility. The configuration consisted of a single, uni-directional, bottleneck. The buffer at the bottleneck has a capacity of 512 packets (27136 octets) and a service-rate of 6 Mbps.

ADSL [ITU-T99] allows for several base line rates. The figure of 6 Mbps line capacity was selected to allow emulation of the upper limit on this technology as would be offered by telecommunications companies in the United Kingdom [Enrico00].

The traffic loads consist of three independent arrivals processes. Firstly, the EP6S480k traffic, based upon external IP traffic and described in Section 2.2.7. Flows carrying this traffic type made five flow attempts per second, with values based upon a Poisson distribution. Such a distribution is found to properly describe human-related activities, such as browser-induced user-sessions [Karlsson97, Paxson95].

The other two traffic sources were VP64S64, emulating a 64 kbps voice channel (described in Section 2.2.3), and VP64S23 emulating a compressed voice channel (described in Section 2.2.4). A new flow-attempt for flows carrying these sources was made using an exponential distribution with a 2.5 second mean for each type. This made five flow attempts of either uncompressed or compressed voice with half the flow attempts split to either type. For all flow types the lifetimes were based upon a log-normal distribution with a 300 second mean — this follows experience with voice phone calls documented previously [Molina27, Bolotin94, Duffy94].

The arrivals process and the process that determines the flow lifetime is independent for all three classes.

5.2.10 Exp8 — VP64S64/VP64S23/IPbg — Voice with LAN background

Using the topology of Exp7, this experiment tests the AC algorithms in the admission of voice flows against a constant background of IP traffic. Like Exp7, flows carrying traffic source VP64S64 and VP64S23 are attempting to enter the system. Flow attempts are exponentially distributed with a mean of 5 attempts made per second, for each of the compressed and uncompressed voice traffic. Like Exp7, the mean holding time of these flows is 300 seconds on a log-normal distribution.

However, the difference in this study was the configuration of a background traffic flow. Unlike the previous experiment, the number of IP flows is held constant; five flows carrying EP10S800k traffic are running continuously throughout the experiment. This permanent background traffic has a nominal mean of 4 Mbps at all times, although this traffic is bursty over a wide range of time scales.

5.3 Results

The results of this section present comparison of MBAC algorithm using a number of different criteria. In Section 5.3.1, in addition to using comparison criterion that has been commonly used by previous authors (e.g. line utilisation versus packet loss, packet loss versus flow acceptance rate) as well as several new criteria. Section 5.3.2 investigates the mechanism by which a user specifies the performance-objective and how this varies among different MBAC algorithms. By contrasting MBAC algorithm behaviour, this approach allows insight into the independence of such controls for different traffic types.

An MBAC algorithm, using control of flow-arrivals combined with a dependence upon flow-departure, is intended to provide a stable mechanism for managing the network resources. The behaviour of complete MBAC algorithms as well as the component pieces of policy and estimator allow an insight into the ability for MBAC algorithm to maintain stability in a system. Section 5.3.3 presents results from an investigation of the bounds for detecting changes in system conditions as well as examining the stabilisation period required by MBAC algorithms.

Finally, alongside comparisons of the performance results for MBAC algorithms, Section 5.3.4 presents results indicating the ease of implementation, and operational overheads. With appropriate timer instrumentation, the implementation-based approach is able to provide directly comparable information on the CPU overheads for each MBAC algorithm. Alongside this comparison is an analysis of the impact that implementation has upon an MBAC algorithm's performance.

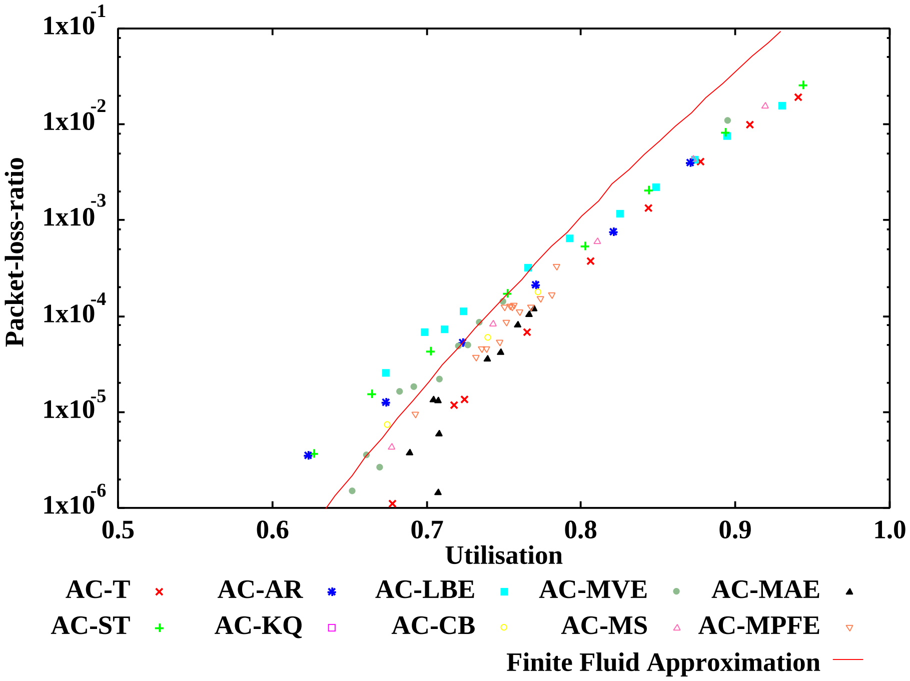
5.3.1 Traditional performance criteria

The relationship between the data-loss and line utilisation, herein referred to as the loss-load curve, has been used by a number of previous papers (e.g. [Gibbens97, Jamin97b, Jamin97c]).

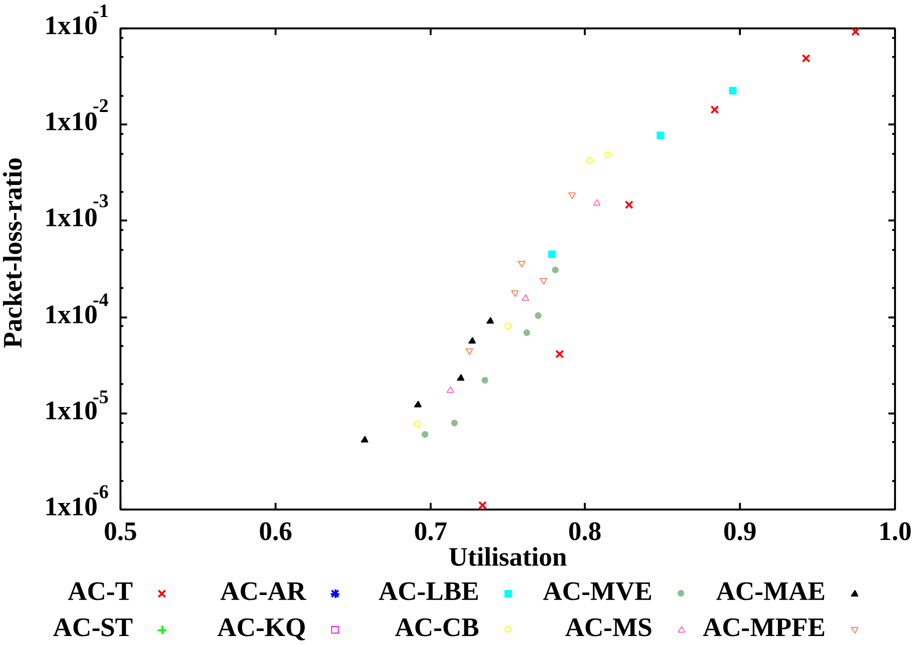
Comparisons may show how well each MBAC algorithm performs relative to each other or to an 'optimum' loss-load curve. However, such results are shown to give similar relationships between loss and load for a particular configuration of traffic and buffer characteristics regardless of the MBAC in use.

While one conclusion of this work is that such comparison is flawed and reveals little useful information indicates limited usefulness in this particular approach, the omission of such a commonly used comparison would be difficult to justify. In [Breslau00] an algorithm, (referred to therein as *Quota*), was used to derive the *performance frontier* values for MBAC algorithm behaviour when faced with particular traffic. The performance frontier may be considered the best possible performance for any given criteria. Such an algorithm allows the computation of optimal results for a given level of flow-arrival activity and traffic by allowing a fixed number of flows to be active at any time. As noted in Section 4.4.15, the AC-T algorithm implemented in this work serves an identical purpose.

Figure 5.1 presents the loss-load results of a number of AC algorithms conducted against a representative sample of the experiments of Section 5.2. The trend towards a utilisation boundary for a given loss ratio is evident. The variance will be higher as the loss ratio is decreased as a consequence of the experiment run-length discussed in Section 3.5.2. For each graph in Figure 5.1,



(a) Exp1: 2-state ON-OFF Markovian



(b) Exp3: Internet traffic

Figure 5.1: Packet-loss-ratio versus line utilisation.

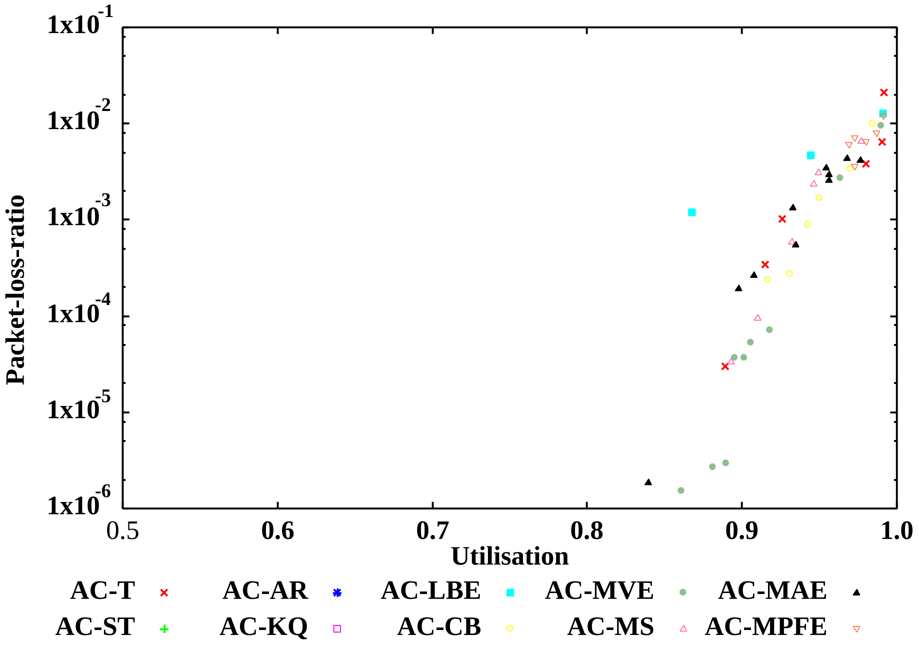


Figure 5.2: Exp8: Voice/Background IP Packet loss-ratio versus line utilisation

this trend for the increasing variance as the loss ratio is decreased is clear.

Figure 5.1 illustrates how graphs of data-loss versus line utilisation return a similar function regardless of the AC in use. As reported in [Breslau00], the small deviations from the loss-load achieved by the target algorithm (AC-T) may be because of MBAC algorithm behaviour. The error in the results in [Breslau00] resulting from sampling error are undocumented. It appears that an implicit assumption of correct or near-correct results is made. However, by examining the results gained here it is clear that part of the behaviour may be apportioned to the behaviour of the algorithm but part will also be due to the error arising from the environment.

Section 3.5.2 notes an error margin due to sampling error of $\approx 0.6\%$ with 95% confidence for a loss ratio of 1×10^{-3}. However, when testing MBAC algorithms repeatability, Section 3.5.4 noted an error margin of $\pm 5\%$ with 95% confidence for the same loss-rate. This implies, with 95% confidence that an error margin of up to 5% is present in the results for 1×10^{-3}. For smaller loss-ratio values this error will be increased: with 95% confidence, an error margin of 12% is present in the results for 1×10^{-5}, and the margin due to sampling-error will have increased to $\approx 6\%$ resulting in a total potential error-margin in results of 18%. Such a large error-margin is clear in the results of Figure 5.1(a) so its contribution should not be ignored.

Alongside the experimental data, Figure 5.1 illustrates the results of a fluid flow approximation [Anick82] for the TP10S1 traffic source. While the experimental and theoretical results have similar slope, differences are clear. The cause for this may be speculated as a result of the fluid flow approximation and perhaps further serve to justify experimental evaluations alongside theoretical or simulation studies.

Each AC algorithm was operated over a reasonable range of parameters. However in several cases (e.g. AC-KQ for Figure 5.1(a)) the reasonable range of operating parameters, e.g. a target loss rate between 1×10^{-1} and 1×10^{-6}, did not generate results that fell within the graphed area. Generally, if an algorithm did not create results to be plotted, the experiment did not generate a sufficient level of loss.

While at first this may seem to be a sure indication of the limitations and the effectiveness of an algorithm, it results from a quite different cause. Algorithms such as AC-KQ maintain an upper boundary on the loss-ratio, the algorithm is based upon worst-case behaviour of the measured traffic. Thus a lack of loss is an indication that to get the algorithm to achieve arbitrary levels of loss would require operation outside what was a reasonable range for the parameters (e.g. an artificially high loss boundary).

In addition to considering that the relationship between buffer and traffic is the sole influence on the ideal loss-load relationship, Grossglauser in [Grossglauser97b, Tse99] noted that MBAC algorithms commonly compensate for errors introduced as part of the measurement process by using a conservative handling of the measurements themselves. A common tuning variable for an MBAC algorithm is how conservatively the measurements will be handled. Subsequently, an MBAC algorithm can achieve a given loss-load curve target. This has been done, however, by removing any safety margin to account for poor measurements and it increases an algorithm's reliance on the measurement characteristics — such as variance due to measurement period.

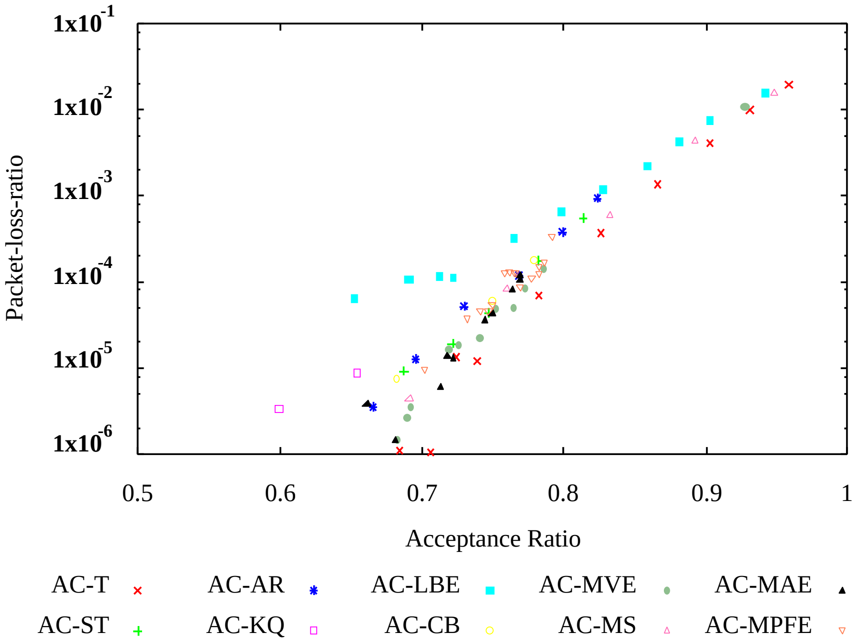
Therefore, algorithms able to achieve a particular point on the load-loss curve have done so by sacrificing any safety-margin (maintaining the QoS guarantee) for all flows in exchange for higher utilisation. If a QoS, such as a loss ratio, was a bounded agreement across all flows, then such algorithms operating outside the safety margin could not make such a QoS guarantee to all flows in the system.

Upon inspection it is clear that a comparison of measurement based algorithms based upon the loss-load relationship is well intentioned but misleading. Each loss-load relationship is dependent solely upon the traffic and buffer characteristics. An MBAC algorithm's ability to achieve a given loss-load curve depends as much upon the measurement characteristics used as input for the estimator as upon the estimators themselves. Additionally, as illustrated further in this section, an MBAC algorithm may specifically achieve different points on the loss-load curve by changing the current mixture of flows present. While not available for a system with heterogenous flows, such changing the mix of flows is further-investigated later in this section.

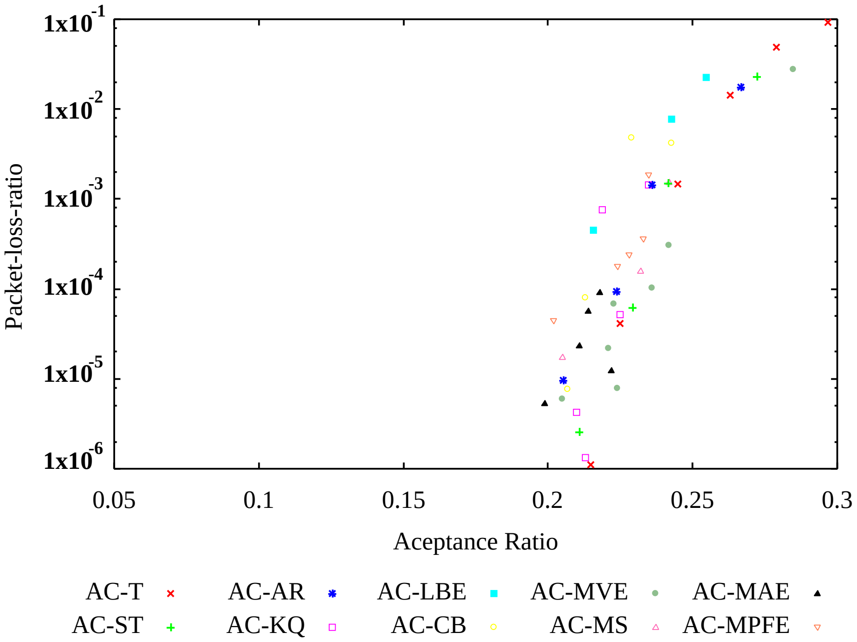
Differences between MBAC algorithms lie in the manner in which each approach the desired point on the utilisation curve. Thus, a comparison of MBAC algorithms must encompass predictability (the ability to achieve a given point on the loss-load curve), stability (the speed at which recovery from changing circumstances can be effected), and fairness (how an MBAC treats flows of different characteristics). These issues are in addition to those of measurement, computation and memory overhead, the relationship between the MBAC policy and flow characteristics, or the association between estimator and measurement characteristics.

An alternative comparison criterion was to consider the curve relating flow-blocking and packet-loss, proposed in [Jamin97b]. If the packet-loss requirements are strict, a high flow-blocking probability will occur; similarly high packet loss will go with a low blocking probability. In [Jamin97b] the authors stated that every well-tuned MBAC has (for a given set of input traffic characteristics) the same loss-load curve.

The results presented in Figure 5.3 and Figure 5.4 allow this idea to be explored a little further. In the experiments conducted for these results, the relationship between acceptance rate and utilisation is near identical for each MBAC; additionally, as would be predicted by these results, the acceptance rate and packet-loss-ratio also express a clear relationship. This can be seen in Figure 5.3(a) and 5.3(b), particularly when compared with the counterparts expressing the relationship between loss and utilisation: Figure 5.1(a) and 5.1(b). However, in each case the flows carry traffic that is homogeneous and the processes describing flow-arrivals and flow-lifetimes

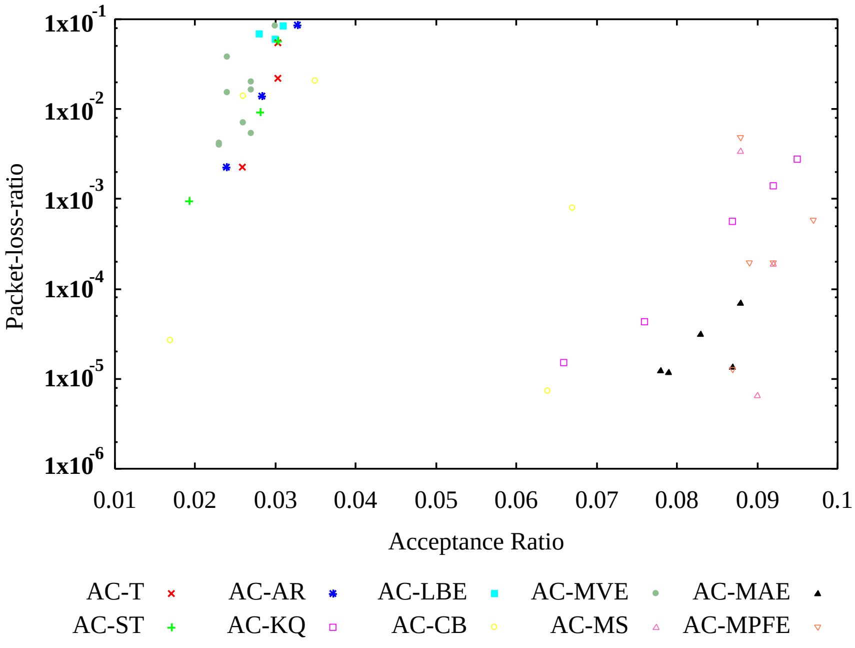


(a) Exp1: 2-state ON-OFF Markovian

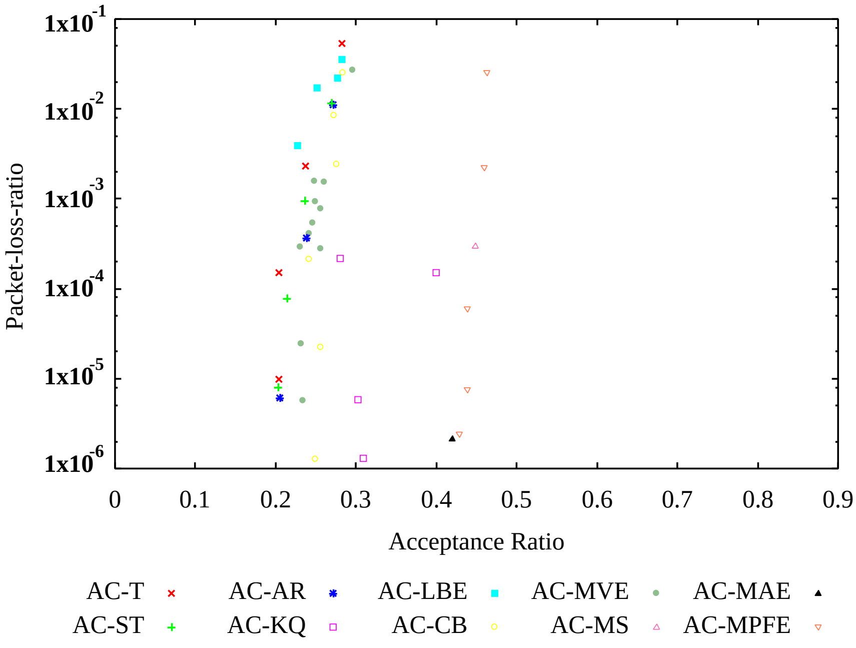


(b) Exp3: Internet traffic

Figure 5.3: Packet-loss-ratio versus flow acceptance ratio.



(a) Exp5: Internet traffic/Video Streams



(b) Exp6: Internet traffic/Video Streams

Figure 5.4: Packet-loss-ratio versus flow acceptance ratio.

are statistically stable: each using fixed mean values with a distribution based upon an exponential decay. This is of greater interest when the process controlling flow-attempts are varied, and the behaviour when the traffic itself is heterogeneous.

For Figures 5.4(a) the results of a heterogeneous experiment are illustrated: unlike the results of earlier figures (5.3(a) and 5.3(b),) different flow-acceptance behaviour is occurring dependent upon the particular AC algorithm. Algorithms such as AC-AR, AC-ST and AC-MVE that do not implement an admission policy dependent upon the declared parameters of new flows have results that are clustered in the top-left of the figure. In contrast, algorithms that use a pessimistic admission process, (e.g. P-PA,) such as AC-MPFE, AC-MS, or AC-MAE give results that are clustered in the lower right of the figure.

The clustering is related to the policy used by each particular AC algorithm. Those algorithms that use a pessimistic admission policy will be biased towards flows with low declared peak-rate values. As a result, a global acceptance ratio may serve little value, giving information about an AC algorithm only when it is under one particular flow-load. Emphasising this point, Figures 5.4(a) and 5.4(b) illustrate results of Exp5 and Exp6. Both experiments use the same heterogeneous traffic mix made up of flows with either a 10 Mbps peak-rate or a 25 Mbps peak-rate. However, the flow arrival and departure characteristics are different between the two experiments.

Acceptance ratios, intrinsically tied as they are to the utilisation process, may reveal little that allows comparison of MBAC algorithms. However, an area in which MBAC algorithms do differ from one another is the difference between acceptance rates for a mixture of flows arriving with different traffic types. Figure 5.5 plots the acceptance ratio for the two different traffic types used in Exp6: a mix of traffic with peak-rates of 10 Mbps and 25 Mbps.

The AC-MS algorithm implements a peak-rate policy and it is characterised by a discrimination towards the class of flows that declare a lower peak-rate. Interesting effects of this policy are evident as the acceptance rate drops. For AC-MS, as the acceptance rate of all flows falls below 0.45, (marked **A**), the flows declaring a large peak-rate are accepted at ever-lower rates while the flows declaring a small peak-rate maintain a constant level of acceptance. This characteristic is caused by the discrimination of an MBAC algorithm against larger flows, smaller flows are able to be accepted where larger flows are not.

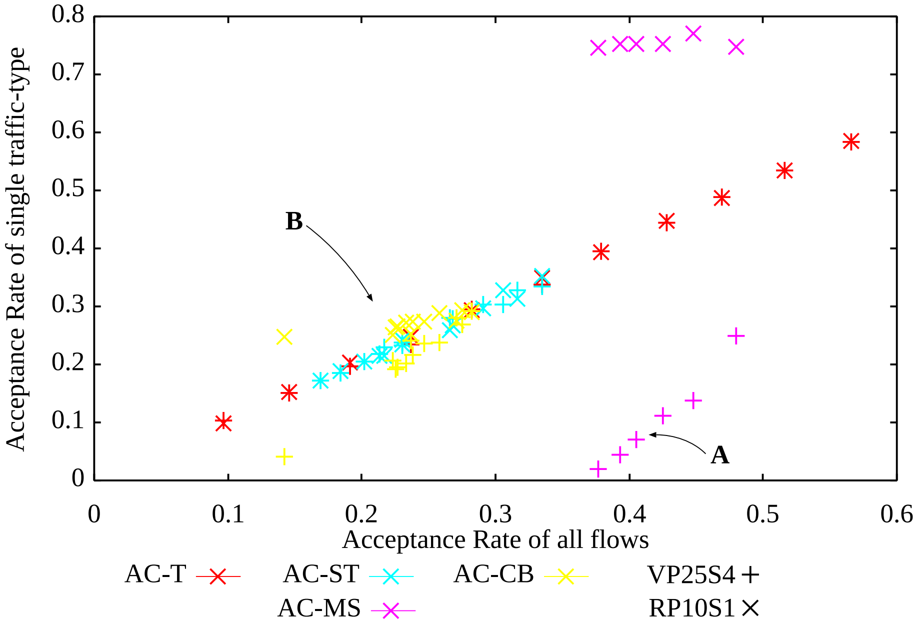


Figure 5.5: Flow admittance ratio for traffic type

An anomaly remains for AC-CB, examining the curves near **B** reveals some discrimination for AC-CB. However, the AC-CB algorithm uses the policy P-None which has no specific discrimination. However, the decision process for AC-CB given in Equation 4.42, incorporates the peak-rate of each class in the computation of an estimate. The estimate is compared against the link-capacity before an admission can be made. Because of this a bias against flows declaring large peak-rate values will occur. The reason the discrimination against flow-classes varies as the total acceptance rate changes is that the impact the peak-rate has in Equation 4.42 changes as the control parameter is altered. The control variable α is also the mechanism for changing the total number of flows admitted by the algorithm.

This set of results, along with similar results from each AC algorithm, illustrate that the admission policy will affect the discrimination of the flows admitted by the algorithm. The underlying MBE algorithm affects this discrimination only if the declared parameters of new flows are directly incorporated into the estimation (e.g. AC-CB).

This leads to the conclusion that policy alone will dictate the differences between the admission characteristics of many AC algorithms. Thus, while these conclusions partially support the proposal from [Jamin97b], significantly, they have only identified a case true for one class of MBAC algorithms. Algorithms that differ in policy, particularly those using no admission policy (P-None) versus a pessimistic admission policy (P-PA) will give significantly differing results in the admission for each traffic class. Additionally, algorithms such as AC-CB that incorporate the declared parameters of classes directly into the MBAC algorithm will also generate results different from the generic MBAC algorithm.

A comparison of results emphasising the effect of delay is used by [Knightly98, Qiu98b]. Using an MBAC to control flows of MPEG traffic through a fixed-length buffer, the overflow-probability and the probability of exceeding a maximum delay bound were configured as the same value (thus fixing the buffer length as the maximum delay length). Using results gained from simulation, this comparison is not an assessment of the measurement of delay but rather a treatment of the upper-delay bound as fixed by the buffer length. The results presented are a comparison of a several algorithms giving utilisation versus delay for a fixed loss ratio — a loss ratio fixed to be the same as the probability of exceeding the upper-delay bound. In addition to showing the inefficiencies of using a bufferless model of traffic when a buffer (delay) is permitted, the comparison is used to illustrate the drawbacks of the alternatives given. Firstly, for low delay values, using the algorithm in [Jamin97a] causes the utilisation to fall because the admission algorithm will become overly pessimistic, not admitting calls due to the delay test [Jamin97a, Eq. 7]. While [Floyd96] stayed at a fixed utilisation level for a given loss-ratio independently of any flexibility in the buffering and thus the delay bounds; thus leading to inefficiency as the delay-bound increases. In comparison to both of these algorithms, the technique using aggregate traffic-envelopes presented in [Knightly98] offered a better utilisation and was for this reason considered more effective.

The comparison results from [Knightly98, Qiu98b] must be considered carefully as for each of the comparison algorithms. [Jamin97a] and [Floyd96], implied that the control parameters were not adjusted as the delay-bound was increased. While the control parameter takes the form of a target utilisation, it is a significant mistake to assume that that the adjustment of this value cannot be used to change the behaviour of an algorithm in the face of changing buffer size. The results of this section illustrate that while conducted using a fixed buffer size, the adjustment of utilisation target (such as in AC-MS) could cause the algorithm to achieve any desired loss ratio.

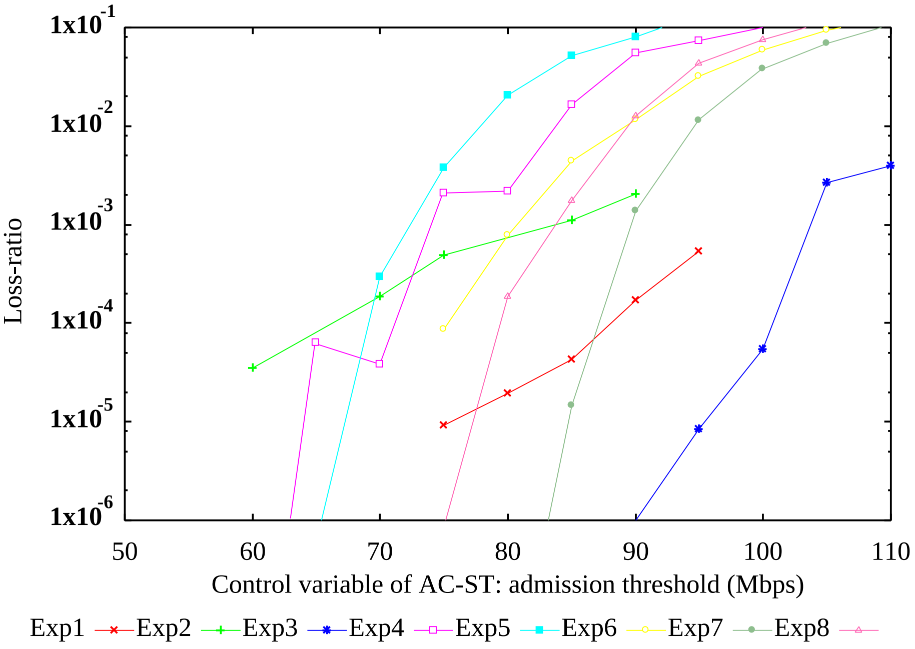
Of additional importance, [Knightly98] noted that of the three algorithms they tested: the aggregate envelope techniques along with those of [Floyd96] and [Jamin97a], only the aggregate envelopes technique offered the user to specify loss and delay figures directly, rather than via a less specific, 'target utilisation' parameter. This implies that, of the three algorithms compared, the aggregate envelopes technique offers the user a mechanism for specifying performance criteria that may most directly relate to system configuration and behaviour.

5.3.2 MBAC algorithm Parameters

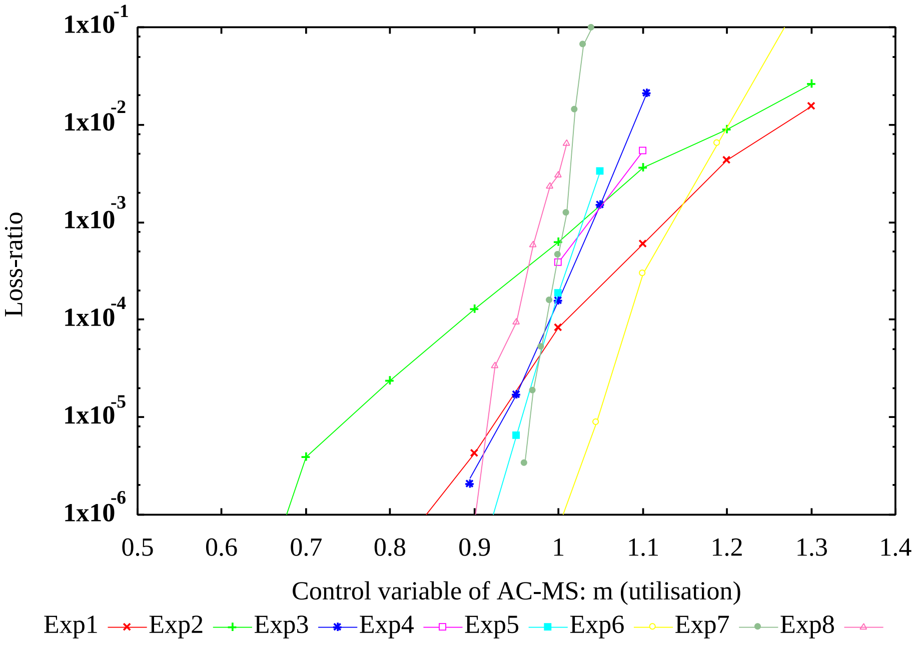
The previous section revealed how criteria for performance comparison among AC algorithms commonly used by previous authors (loss-load, loss-acceptance, and loss-delay results), do not present a complete picture of an AC algorithms performance. While still using the technique to draw conclusions, [Breslau00] noted many of the limitations of the approach. The previous section concludes that the operational control, reliability and stability of algorithms are not included in such a performance-frontier comparison. This section presents results illustrating the control of algorithm by their respective parameters.

Figures 5.6, 5.7, 5.8 and 5.9 each give results in which the measured loss ratio is plotted against a control parameter of each AC algorithm. In each of these figures a data-point represents a single experiment, additionally, lines had been added to assist visualising the relationship between the control parameter and loss ratio. In each set of results it is clear that no universal algorithm exists that is able to control the variety of traffic presented by the experiments of Section 5.2. Of particular interest in Figures 5.6, 5.7, 5.8 and 5.9 is the independence the algorithm has from the traffic type. Such independence will relate to the desired outcome (the desired outcome for these examples is a QoS guarantee of packet-loss).

While every algorithm has at least one specific control parameter, many such as AC-MAE or AC-KQ have a variety of other parameters. Additionally, the measurement period is a basic parameter to all MBAC algorithms, although other parameters such as the number of samples also become parameters. This section restricts itself to a study of the principle control parameter as proposed with each algorithm. Readers should be mindful, however, of the existence of these additional parameters along with their defaults (detailed in the implementation information of Section 4.4).

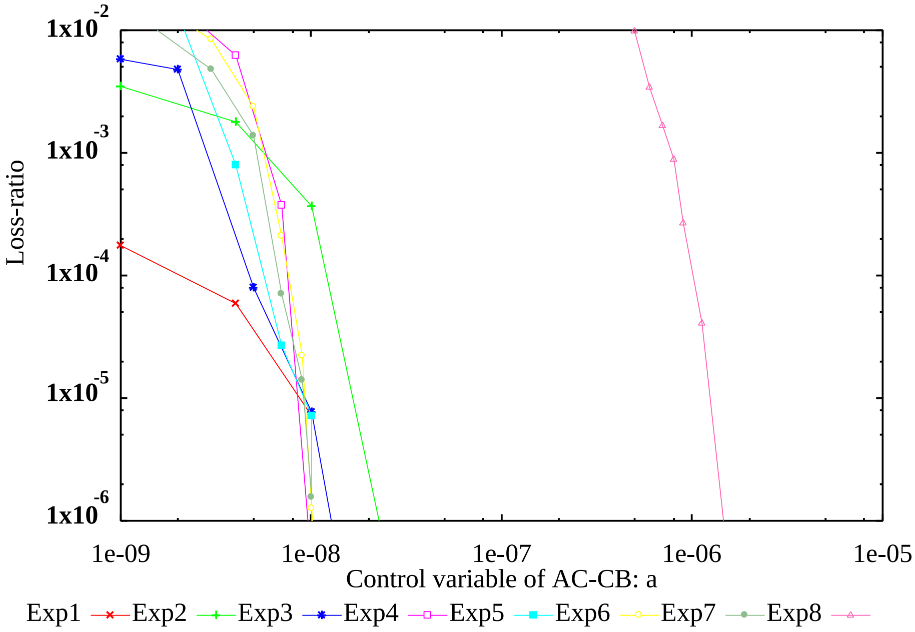


(a) AC-ST

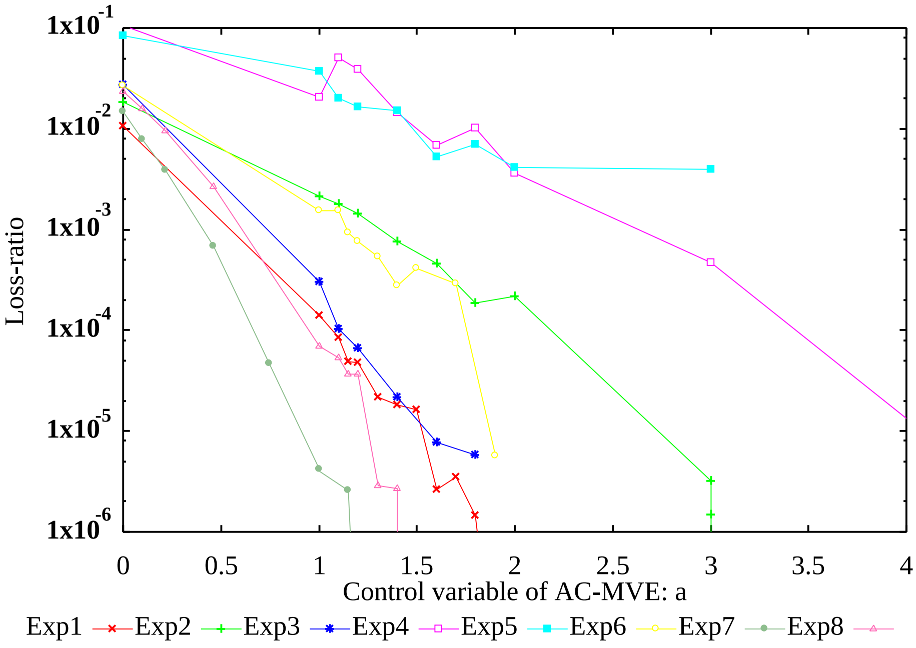


(b) AC-MS

Figure 5.6: Packet-loss-ratio versus control parameter — I.



(a) AC-CB



(b) AC-MVE

Figure 5.7: Packet-loss-ratio versus control parameter — II.

Figure 5.6(a) presents results for experiments using the AC-ST algorithm. All experiments for these results use a common measurement interval 10 ms. Figure 5.12 in Section 5.4.1 presents results where the measurement interval is varied while the threshold is held constant. As may be expected, considerable variation between experiments exists across the same range of threshold values. In contrast, AC-AR, uses a similar threshold-based technique but proposes an integrated adaptation to the acceptance region (from which the threshold value is derived). This implies that while the threshold versus loss-ratio curves would resemble those of AC-ST, the algorithm would adapt the threshold to the conditions of traffic, the QoS guarantee to be met, as well as buffer size and link capacity.

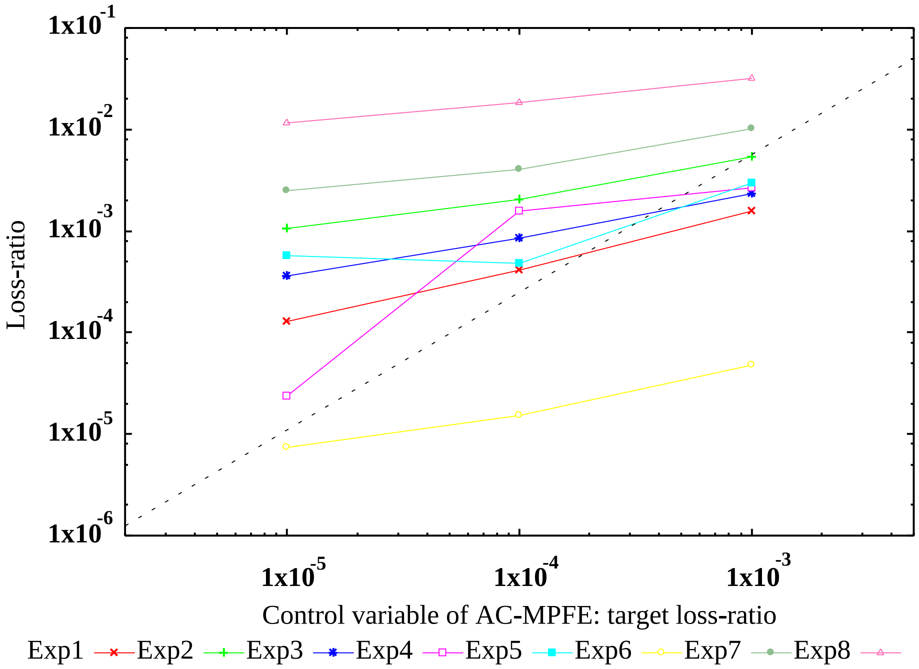
Using a control that stipulates a level of utilisation, AC-MS gives the results of Figure 5.6(b). Specifying the utilisation results in different traffic types being treated in a uniform manner. Utilisation shown in Figure 5.1 does not express a clear, traffic-independent relation to the loss ratio. However, if the QoS were based upon a guaranteed level of link utilisation, this mechanism may be appropriate.

The AC-CB algorithm (Figure 5.7(a)) uses an un-calibrated control: α as a pre-multiplier on a factor derived from the flow's peak-rate. Thus, α controls the degree of robustness the algorithm will have to flows. A small scalar allows more admissions and, thus, more (potential) packet-loss by reducing the impact the declared peak-rate has upon the system, while a large scalar value has the opposite effect of reducing potential for packet-loss. Clearly, this value is of limited use when selecting an objective as a particular loss ratio.

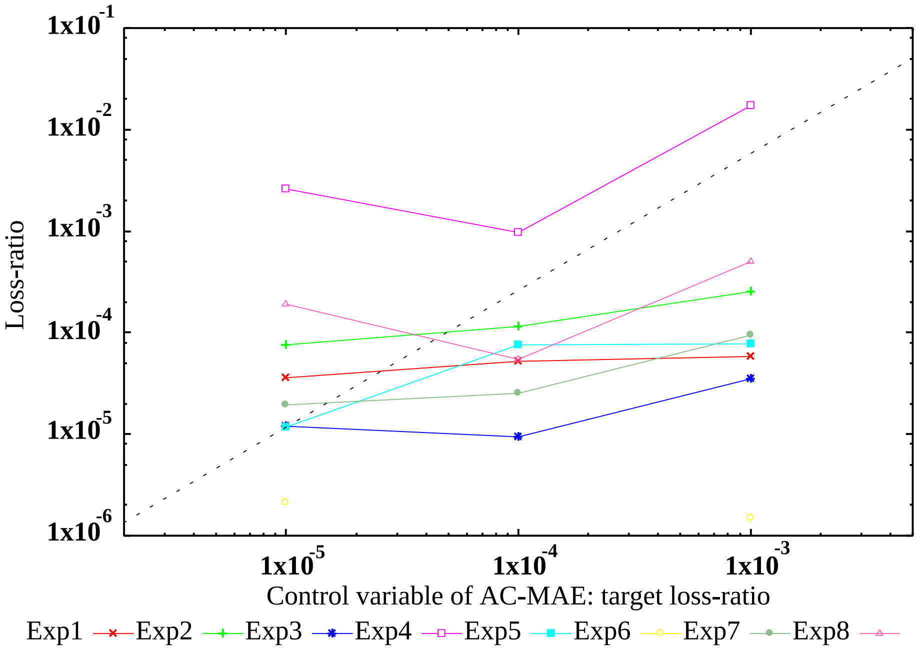
Using AC-MVE, an algorithm based upon mean and variance measurements, returns the results shown in Figure 5.7(b). These results illustrate no clear relationship between the variance of measurements made over one timescale and the measured packet-loss when this estimator is incorporated into an AC algorithm. The control parameter, α controls the contribution the measured variance will make to the estimate.

Figure 5.8 and 5.9 plots results of algorithms that allow the nomination of a target loss ratio. In each of these figures, a dotted line is added to aid in comparing target with measured loss ratio.

Both the AC-MPFE and AC-MAE algorithms, (Figures 5.8(a) and 5.8(b),) have multiple tuning

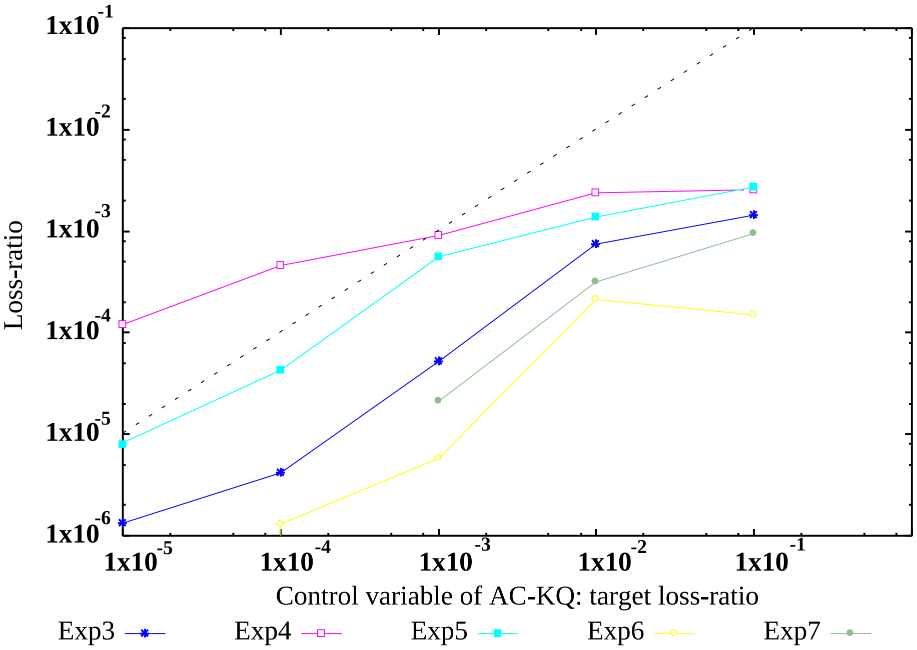


(a) AC-MPFE

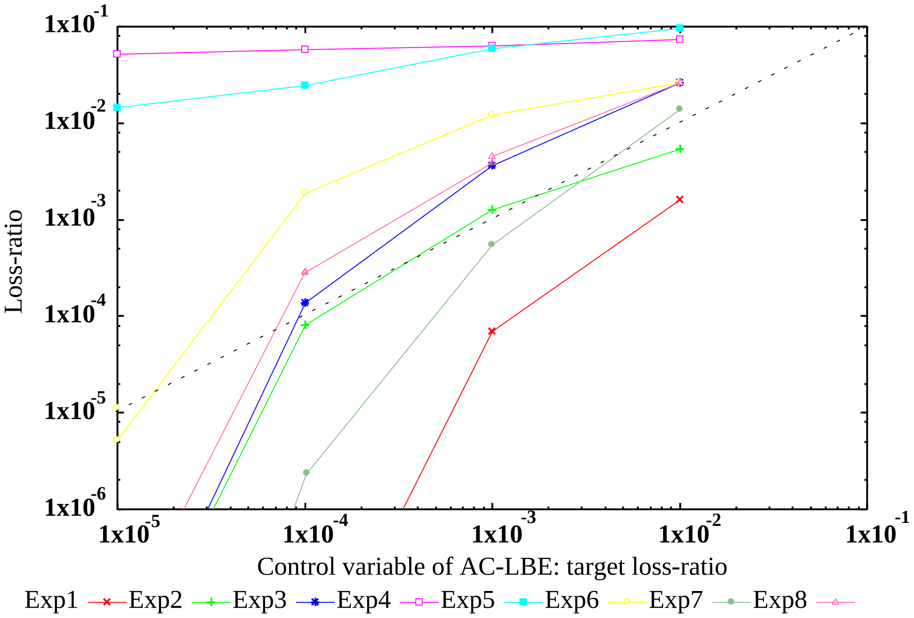


(b) AC-MAE

Figure 5.8: Packet-loss-ratio versus control parameter — III.



(a) AC-KQ



(b) AC-LBE

Figure 5.9: Packet-loss-ratio versus control parameter — IV.

factors. Aside from a target loss ratio similar to all other measurement-based estimators, these MBE will characterise traffic over one period. The disadvantage in using only one characterisation period, (a common block length was used for all the results of Figures 5.8(a) and 5.8(b),) is that only one period may not characterise traffic correctly for all desired loss-ratios.

Figure 5.10 illustrates how the measured loss ratio will change for a constant target loss ratio when the characterisation period (the block-length) is varied. The differences each traffic type may have as well as the different behaviour that may arise for measurements made for the same traffic at different measurement-lengths leads to the need to characterise traffic over several measurement periods, such is the approach of AC-KQ.

The results of AC-KQ also indicate considerable variation for different traffic types, Exp1, Exp2 and Exp8 are not shown, as no significant loss ratio was recorded in these experiments. The objective of the E-KQ estimator is to limit the loss of flows. Of all the algorithms that allow specification of target loss ratio AC-KQ is the only one where the majority of loss results are maintained below the target loss ratio.

The results of the loss-based estimator AC-LBE, presented in Figure 5.9(b) were disappointing, although this algorithm also depended upon a critical measurement period. However, as outlined alongside discussion of this algorithm in Section 4.4.11, the implementation was immature and the results reinforce this conclusion.

The control parameters of a range of AC algorithms have been compared; several algorithms use parameters related to utilisation, others use parameters intended to control the loss-ratio while those such as AC-CB use a control parameter that may best be described as uncalibrated. Each type of control has particular application: utilisation controls are necessary should it be important to elect the level of network utilisation, while bounding the loss-ratio is critical for a system that attempts to maintain such a QoS guarantee to flows in the network. In contrast, the uncalibrated control of AC-CB may prove useful for traffic that adapts its packet-loss to current conditions, e.g. elastic traffic, although AC-CB is still measurement based and may serve as a unique approach in this respect.

5.3.3 Stability and Repeatability

Previous sections have illustrated that an MBAC algorithm will operate at a particular point on the loss-load curve for a given combination of traffic and resources and that the control over the algorithm may specify measures such as utilisation or loss-ratio targets. In this section, the stability and repeatability of MBAC algorithms is examined. Stability is used to describe the ability for an AC algorithm to recover from a change of circumstance, e.g. a change in available resource or a change in traffic behaviour. Repeatability examines the capacity of an AC algorithm to attain a particular point on the loss-load curve.

The stability of a system under control of an AC algorithm depends upon the interaction of flow arrivals and flow lifetimes with the traffic characteristics, and the characteristics of the system such as link and buffer capacity. A number of stability issues are not under the complete control of the AC algorithm — an algorithm may not retrospectively evict flows, even if the resource is not adequate or the flows require more resource than predicted. In such situations an AC algorithm would change the number of flows admitted to the system, but the AC algorithm depends on the finite lifetime of flows to ensure the release of resource.

Estimator Stability

An AC algorithm has limited control over the system: depending upon the release of resources through the regular departure of flows. For situations where flow arrival and departure is a regularly occurring process, the AC algorithm will admit new flows as it is able. In this sense the stability relates to the speed an AC algorithm will locate its ‘operating point.’ Stable operation requires that an AC algorithm detects changes in circumstances, such as changes in resource requirements. For an MBAC algorithm the MBE must detect such changes.

Table 5.1 lists the minimum time taken before each MBE will detect a change in the resource requirements. Each MBAC algorithm will only detect a transition at the end of its respective measurement period. These periods can be considered a ‘worst-case’ value, after which each estimator will have detected a transition. Interestingly, when reduced to a common nomenclature, it is clear that all the MBAC algorithms examined will have the same minimum period of measurement and, thus, the same minimum period before detecting a change.

MBAC	Detection of transition	History
E-IU	τ	τ
E-CB	τ	τ
E-MS	τ	T
E-MPFE	τ	$\max T_h$
E-MAE	τ	$\frac{\overline{T_h}}{\sqrt{n}}$
E-LBE	τ	system lifetime
E-MVE	τ	$Tn \cdot \tau$
E-KQ	τ	$M \cdot T \cdot \tau$
E-GT	τ	$T \cdot \tau$

Table 5.1: MBE detection periods.

The periods given in Table 5.1 are cumulative with the processing time each estimator will incorporate; thus detecting changes in available resource or resource requirements requires both measurement and processing.

Table 5.1 also lists the historical information of prior traffic conditions each algorithm incorporates into its estimation algorithm. Holding historical information can permit an algorithm to better tune its performance, and algorithms such as E-MPFE and E-MAE rely upon complete historical information to accurately calculate loss events. Each algorithm incorporates the historical information either implicitly such as E-MAE or explicitly in the case of E-MS and E-CB where the estimator's configuration incorporates settings reliant on the estimated lifetime of flows and flow arrival rate as well as the longer term trends of traffic.

The E-MPFE algorithm differs in an important aspect from the others because it maintains complete historical information for each of the current flows. Additionally, unlike each of the other MBAC estimators examined, E-MPFE only incorporates historical information for current flows — once a flow leaves the system, any contributions it made to the data flow statistics are removed. While, conceptually, a justifiable solution to this problem — Section 5.3.4 discusses problems that such an approach may introduce in implementation.

Incorporating a period of history of past measurements into the computation of an estimate can make the algorithm perform in a more stable fashion. The selection of history periods is noted in the presentation of many algorithms as a critical component in traffic and flow characterisation [Lewis98, Jamin97b, Grossglauser97b]. Table 5.1 reveals several different history values being used in the collection of algorithms implemented here. For an algorithm such as AC-AR, the historical data is intangible as the byproduct of previous calculations the historical information may be negligible if the priming values of the Bayesian estimator are set correctly. Estimators such as E-ST do not incorporate any history from previous measurements and by definition, the combination of threshold and measurement period must incorporate a characterisation of both the traffic to be carried and characteristics of the flow-arrival and flow-departure processes.

Of the analysis put forward as part of papers presenting AC-GT, [Grossglauser97b] and later [Grossglauser00] noted that historical information intrinsic in flows being present in the system can be used to smooth out variations in traffic due to flow arrival and departure. Variation in traffic occurring on a considerably shorter timescale (those due to burst and packet-level effects) are accommodated with an estimate of excess bandwidth requirement made using shorter timescale information from the estimate history. Following a presentation of this decomposition approach to traffic timescales [Grossglauser97b] presents an AC algorithm that incorporates filtering processes to extract measurements of the appropriate long or short timescale. In this way the AC algorithm uses historical information recorded over an arbitrary (long) period. However, this historical information is separated based upon relevancy for either shorter or longer timescales. Section 4.4.10 gives details of this critical timescale in the context of the AC-GT algorithm and Section 2.3.1 discusses this particular issue within the context of timescale separation.

Of the other algorithms based upon mean-variance estimators; AC-KQ and AC-MVE each use a lower bound on the number of samples thereby ensuring the variance figure returned by the estimator will provide an accurate estimate. Interestingly, in the AC-KQ algorithm, the tests upon which admission are based will tend towards pessimism if the total time sampled is either too long or too short; Section 4.4.9 details precisely why this situation arises.

Finally, the algorithm based upon ongoing measurement of the system loss ratio is examined. AC-LBE using an EWMA function incorporates the history of loss over the total life of the system. Adjusting the weighting function appropriately will bias the estimate towards the longer timescale of the total system or towards the shorter timescale of the most immediate loss measurement.

Algorithm Stability

In this section, the period in which the complete AC algorithms of Section 4.4 regain stability is investigated through a comparison of experimental results. The stability of algorithms is tested by a comparison of the mean time taken to stabilise following a change in circumstances. An AC algorithm must contend with decreases in available bandwidth or increases in the estimate of required bandwidth.[1] The stability tests in this section time the period before which an AC algorithm will have stabilised under changes in available bandwidth.

Each of Exp1 through Exp8 was used to examine the stabilisation characteristics of each AC algorithm. In each experiment, a stable system is presented with a significant change in available bandwidth. The number of flows in progress is taken as the indicator of system stability — the measure of stability becomes the time taken before the system establishes a constant value for the number of flows in progress. While packet loss has been used as a metric for past comparisons, it is subject to a significant error margin that varies with the number of loss events. The number of flows in progress will not vary with the number of observations, as the value results from the combination of the flow attempt process and the MBAC algorithm.

As mentioned in Section 3.5.2, in the creation of the test environment it is important to eliminate start-up (transient) periods. The same techniques used for detecting such initial transition can be used to detect a transition period. [Pawlikowski90], summarising the work of a number of previous authors in the field of steady-state simulation work, presented several methods for the detection of initial transient periods of experiments. In order to establish the period of initial transition *R11* from that work is used. It establishes the duration of the transient period as the point at which a given sequence of observations of the number of flows in progress behaves in a way consistent with a standard stochastic. This technique is only made possible with the prior knowledge of the variance once the system is in steady state. The transient period is computed based upon the convergence of the variance from the change in conditions towards the steady-state value. The time taken for the system to stabilise is the period taken for the variance to converge to within 1% of the steady-state variance.

[1] The cases where resource becomes over-committed is not examined here. This is because an AC algorithm, unable to revoke over-committed resource, relies upon the regular departure of flows to restore resource.

AC Algorithm	Exp1	Exp1a	Exp2	Exp3	Exp4	Exp5	Exp6	Exp7	Exp8
AC-ST	26.1	6.7	21.3	39.5	17.8	8.7	24.6	64.0	52.6
AC-AR	27.4	15.2	24.4	49.9	12.4	160.8	8.8	56.8	50.0
AC-CB	26.5	15.1	25.9	56.3	64.5	215.1	6.5	52.5	45.5
AC-MS	29.1	6.3	24.4	36.0	52.3	205.7	22.1	65.6	59.9
AC-MPFE	34.9	17.0	31.8	36.9	72.1	181.4	10.1	74.4	58.2
AC-MAE	35.5	9.9	35.7	21.1	15.1	191.8	10.6	75.8	51.8
AC-MVE	35.0	12.4	30.6	32.3	33.6	199.6	13.4	73.9	59.6
AC-KQ	34.5	11.8	32.0	39.2	33.8	222.4	11.9	72.6	54.7
AC-GT	21.2	7.3	28.9	11.6	10.2	120.0	11.8	61.6	38.9
AC-LBE	41.3	18.6	35.6	45.6	25.0	14.0	16.7	49.8	74.7
AC-T	43.4	17.1	42.8	21.6	5.3	30.6	15.7	51.9	52.6
AC-PRA	27.6	4.8	27.1	9.9	5.9	8.4	71.4	187.4	32.9

Table 5.2: Stabilisation period (seconds).

Experiment	Sources	Summary
Exp1	TP10S1	2-state Markov ON-OFF
Exp1a	TP10S1	2-state Markov ON-OFF (more flow attempts)
Exp2	PP10S1	2-state Pareto ON-OFF
Exp3	RP10S1	LAN
Exp4	VP25S4	Video
Exp5	RP10S1/VP25S4	LAN with Video (long flows)
Exp6	RP10S1/VP25S4	LAN with Video (short flows)
Exp7	EP6S480k/VP64S64/VP64S23	LAN with Voice
Exp8	VP64S64/VP64S23/IPbg	Voice with LAN background

Table 5.3: Summary of experiments

For each experiment, the MBAC algorithms were configured to return a similar loss ratio around 1×10^{-3}. The appropriate parameters to use for each MBAC algorithm had already been established through the experiments of Section 5.3.2.

Table 5.2 gives the time before a stable number of flows-in-progress is settled on. Results for AC-PRA are added for information; while this algorithm achieves a far lower number of flows in progress and no loss-ratio, this set of values give an insight into the process of flow-attempts. Table 5.3 provides a summary of experiments for the convenience of the reader.

Clearly, some of the experiments have consistent results that appear to depend solely on the arrival process of the flow attempts. This is particularly noticeable when the stabilisation periods for AC-T is similar to that gained for the MBAC algorithms. Notable exceptions are: AC-CB, which consistently has a smaller stabilisation period than any other algorithm for Exp6 (LAN and video traffic over short flows). Aside from a different estimator, a major difference in the AC-CB is the use of the back-off policy, P-BP. This policy is used in combination with an estimator that incorporates the characterisation of the flow to be admitted. It appears that this policy interacts well with the combination of traffic for Exp6. For Exp7 (LAN and video traffic over long flows) all algorithms performed better than the AC-PRA. This is because of the significant difference in bandwidth requirements of the two flows being multiplexed. The AC-PRA can only admit a few large flows and then must restrict itself to the admission of the lower-bandwidth voice calls to make-up the remaining bandwidth. By restricting itself to fewer of the flow attempts, the stabilisation is slower to achieve, dependent as it is on the arrival process of the smaller flows.

The two algorithms: AC-MAE and AC-MPFE have a significantly faster stabilisation period for the Exp6, yet for other experiments carrying the less-predictable IP traffic (Exp7 and Exp8) they offer no advantage.

On the basis that the admission rate was having a significant effect on the stabilisation period and may be providing an upper bound this period, a different flow-rate was attempted for some experiments. Exp1a was a re-run with a mean rate of 100 flow arrivals per second. In all other ways the experiment was the same as for Exp1.

While the stabilisation period for some algorithms is highly influenced by this change in the rate of flow arrivals, the reason is not clear. AC-ST, AC-MAE and AC-MS stabilise faster but do not share a common admission policy. Additionally, the effect of the measurement period on both estimator and policy, as noted in Section 4.3, should not be neglected. When experimenting with the high-attempt rate (Exp1a) it was recorded that the AC-ST algorithm stability is achieved in 5 seconds when the measurements are taken over 131 ms but 6.7 seconds when the measurement was taken over a 3.9 ms period. These results illustrate the important relationship of factors such as flow-rates on both estimators and policy and thus overall stability. Due to the complexity of the relationship between the tuning parameters for each AC and the flow-arrival and flow-lifetime processes, simple numerical comparisons of stabilisation time may return little useful comparative data without an exhaustive multidimensional study.

Policy Stability

AC algorithm results of the previous section hint at the importance policy plays in the stable behaviour of algorithms. Results in Table 5.4, compare the mean flows in progress achieved by each admission policy. Measured continuously throughout the experiment, the resulting mean flows-in-progress, along with figures for the coefficient of variation, and the 95% confidence interval are contained in this table. Such figures allow a numerical illustration of the admission characteristics of the algorithms.

The results of Table 5.4 were obtained using Exp1 with two different periods. Firstly a measurement was made every 131.2 ms, then the experiments were repeated with a measurement being taken each 3.94 ms. The use of two periods further reinforces differences between the various policies when faced with differing offered loads. Note how the variance differs between most measurement-based algorithms when smaller measurement periods are used, the exception being the experiment using the pessimistic P-PA policy. It may be recalled from Section 4.3.4, this policy displays strong cyclical properties, dependent upon the measurement period. Clearly, the smaller measurement period and thus smaller admission cycle causes a significant increase in the variance of the number of flows in progress.

For the combination of the E-IU and P-TO, Table 5.4 illustrates the dramatic effect the selection of the measurement interval has upon this AC. The variance of the number of flows-in-progress is very significant when the measurement period is long. However, referring to Figure 4.9 it can

	Period	Mean flows	Coefficient of Variation	95 % Confidence Interval
E-IU with P-TO	131.2 ms	78.2	7.2	4.2×10^{-2}
	3.94 ms	92.7	3.4	2.5×10^{-2}
E-IU with P-PA	131.2 ms	66.4	3.4	1.6×10^{-2}
	3.94 ms	74.1	4.9	2.6×10^{-2}
E-IU with P-BP	131.2 ms	76.2	7.6	4.3×10^{-2}
	3.94 ms	72.0	4.7	2.4×10^{-2}
AC-AR (incorporates P-AR)	131.2 ms	66.3	8.1	3.8×10^{-2}
	3.94 ms	61.6	5.5	2.3×10^{-2}

Table 5.4: Mean flows-in-progress statistics for algorithm/policy combinations.

be seen that the variance is caused by a large number of admissions within one measurement period, and then a number of admission periods where the damage caused by this over-admission is gradually eliminated as flows end and leave the system during a period with no new admissions. Following this, one would expect a smaller measurement period, as per the second example of Table 5.4, to give a mean number of flows in progress tending towards the threshold value — instead, what is seen is a significant over-admission of flows. The reason for this over admission is due to the significant variance in measurements made over such a small period. Section 2.4 discusses the issues of measurements as inputs into admission control and the subject of variance is discussed at length. Examining the log files for this admission process for a mean utilisation measurement of 92.89 Mbps, and a standard deviation of 17.90 Mbps, (assuming a normal distribution,) implies that at as many as 1 in 10 measurements were at or below the threshold of 70 Mbps. Even at the higher measurement level, such a significant number of results falling below the threshold would account for the large number of admissions occurring in the case where the measurement period is small.

From these results, properties of the measurement such as the length of the measurement period and variance in the measurements themselves have an important role in MBAC algorithms. Analysing the results for the combination of E-IU and P-PA reveals the relationship the measurement period has with the P-PA policy.

For the large measurement period, both Figure 4.11 and Table 5.4 illustrate that pessimistic admission policy forces the MBAC to have a much-reduced variation in the number of new flow attempts that succeed. However, the reduction in variation is in exchange for a lower mean number of flows in progress, indicating that in order to achieve a similar level of utilisation a higher threshold value would be required.

Using a smaller measurement period means that the pessimism period of the P-PA policy will be substantially shorter. When the mean number of flows in progress increases, the variance in this figure is also significantly increased. The effectiveness of the P-PA policy is impaired when the measurement period is, as in this case, substantially smaller than the rate at which new flows are attempted; such a situation eliminates the advantage of pessimism gained from the P-PA policy.

For the P-BP policy, Table 5.4 reveals that a change in the measurement period does not make a significant change in the mean and variance. The small change in variance is possibly due to a speedier reaction of the algorithm because of the smaller measurement period. This speedier reaction is possible because the algorithm compares the threshold against a more recently updated value of the current utilisation.

Finally, Table 5.4 illustrates how the timescale of measurement affects the AC-AR algorithm. For this algorithm the mean number of flows would be expected to settle close to 60 flows per second (70 Mbps is the threshold and 10 Mbps is the peak-rate for each flow giving a pseudo-threshold of 60 Mbps. Given that each flow has a mean rate of 1 Mbps, approximately 60 flows are expected. Note that for the experiments conducted at a high rate of measurement, the number of flows more closely approaches this anticipated value of 60 Mbps. The reduction in variance (and subsequent stability of the overall system) is also a result of the smaller measurement period. The reason is that this algorithm does not account for the effect multiple flows have when admitted in one measurement period. The only test performed is that flows will be admitted if the current measurement plus the peak flow rate are less than the threshold. This means that if several new flows attempt admission during the same period each will be admitted. When the measurement period is reduced, approaching the mean interval between flow attempts, the potential for over commitment due to multiple flow admissions in one measurement period is reduced. For the policy of the AC-AR algorithm, measurements over smaller periods are better; whereas for the

P-PA policy measurements made over a period less than the mean flow-arrival rate will impair performance.

Repeatability

Previously the stability of a system has been shown as a function of the policy and the estimator and dependent upon the interaction between them. Stable behaviour leads to repeatable behaviour and the repeatability, (the capacity with which an AC will provide the same results given similar conditions) is examined in this section.

A simple test of repeatability is performed: each AC algorithm is run 100 times against a simple experiment (Exp1) and the variance of recorded loss-ratio, mean flows in progress and mean line utilisation is reported. Table 5.5 records the resulting mean and coefficient of variation for each of the loss ratio, mean flow-acceptance and mean line-utilisation.

To aid comparison and reduce errors representing such numbers, this table records the statistics of the negative $\log$ of the loss ratio.

AC Algorithm	$-\log($ Loss-Ratio $)$		Flow Acceptance		Mean Line Utilisation	
	Mean	Coefficient of Variation	Mean	Coefficient of Variation	Mean	Coefficient of Variation
AC-ST	9.577	10.1	0.742	3.2	0.708	3.3
AC-AR	9.362	9.1	0.743	2.6	0.710	1.8
AC-CB	9.513	9.5	0.745	3.0	0.709	1.6
AC-MS	9.929	7.2	0.750	3.2	0.745	1.6
AC-MPFE	8.653	4.0	0.786	3.2	0.782	1.4
AC-MAE	9.294	4.1	0.766	3.0	0.735	2.2
AC-MVE	9.329	4.7	0.764	3.2	0.759	3.2
AC-KQ	8.588	9.6	0.751	4.5	0.743	3.4
AC-LBE	9.246	7.5	0.704	4.1	0.697	3.2
AC-T	9.665	5.3	0.786	7.2	0.748	6.2

Table 5.5: Repeatability Results.

The results of Table 5.5 do not contain any significant surprises except perhaps for the degree of uniformity across the range of AC algorithms. While direct numerical comparison is less informative, the coefficient of variation highlights several areas of interest. The uniformity of results supports the conclusion that that the issue of repeatability lies as much with the flow arrival and flow departure characteristics as with the AC algorithm in use.

The AC algorithms that exhibit large variance in the loss-ratio, (AC-ST, AC-AR, AC-CB, AC-KQ, and AC-LBE) will also admit flows in bursts. This is because none of these algorithms implement any sort of pessimistic admission policy leading to (potentially) many flows being accepted during the course of one measurement period. However, while only the loss may vary, the AC-LBE algorithm shows significant variation in the flow acceptance ratio. This implies that in any one experiment only AC-LBE may exhibit significant differences in the resulting loss-ratio. For AC-LBE, this may be because once the estimate of the system loss has been made, changes to its value will take successively longer and longer periods of time to impact the estimate. The AC-LBE is acknowledged as being a simple implementation. This may indicate it requires further refinement in the form of an upper limit on the amount of system history maintained.

These results indicate an equivalence in the AC algorithms when it comes to repeatability while results from the stability section show selection of the the admission policy is the overriding issue. The results reinforce the importance of the other differences between AC algorithms, such as the control over the algorithm and the resource overheads of each algorithm.

5.3.4 Resource Overheads

In this section the resource overheads and implementation complexity of each MBAC algorithm is examined. Following this, the impact of implementation-requirements of one particular estimator (E-MPFE) is examined in depth.

The computational overhead of each admission control algorithm forms an important point of comparison; each algorithm will require time to perform the policy and to calculate the estimation of traffic characteristics. In Table 5.6, these time periods are given for each algorithm using a sample run against Exp1. The results for Exp1 are presented as representative of runs against each of the experimental configurations of Section 5.2.

As noted in Section 4.1.1, AC algorithms are constructed from combinations of policy and estimator. Section 4.4 notes in the discussion for each AC algorithm how each implementation may take the form of a front-end only system, or a system based upon a combination of front-end and back-end processes. Table 5.6 reflects whether the timing is for the front or back-end as appropriate.

Algorithm		Mean Deviation	95 % Confidence Interval
		(milliseconds)	
AC-PRA		0.3	0.132
AC-ST		30.6	1.804
AC-AR		31.0	1.811
AC-CB		37.8	1.546
AC-MS	(front-end)	3.0	0.634
AC-MS	(back-end)	87.4	1.010
AC-MPFE	(front-end)	11.0	1.203
AC-MPFE	(back-end)	131.2	0.046
AC-MAE	(front-end)	0.7	0.308
AC-MAE	(back-end)	75.4	1.111
AC-MVE	(front-end)	5.7	0.899
AC-MVE	(back-end)	0.2	0.002
AC-KQ	(front-end)	12.6	1.155
AC-KQ	(back-end)	80.0	0.032
AC-LBE	(front-end)	0.8	0.313
AC-LBE	(back-end)	36.7	6.340
AC-GT	(back-end)	27.8	0.097
AC-T		0.2	0.181

Table 5.6: Time taken in AC components

Several algorithms do not lend themselves to a strict separation of policy and estimation. AC-CB and AC-ST use a current measurement of the line utilisation. However, this does not imply an estimator solely working to extract measurements of the line utilisation as the use of these measurements is a function of the flow-attempt arrival process. It is because of this that several algorithms in Table 5.6 have front and back-end components while many do not.

The results obtained, shown in Table 5.6 are interesting to contrast with the theoretical estimations of the overheads of algorithms presented in Table 4.2. The overheads of estimator E-MPFE: O(N M) for memory and computation become clearly illustrated in Table 5.6. However, the computational requirements should be noted in any of the computationally expensive algorithms. In AC-KQ, there are significant requirements in both the front and back-ends of the implementation — while for most AC algorithms the front-end commonly only implements the admission policy, for this algorithm a significant part of the total estimator is placed into the front-end as well.

Clearly, the computational limits of an algorithm will have a serious impact on the nature of results. While the resolution of measurements may need to be very high, if a computation will take many times the measurement period, it may be necessary to only supply the computation at a lower rate with batches of measurements rather than a continuous stream of measurements at the rate at which they are made.

Overheads for AC-ST, AC-CB, and each of the back-end AC implementations is the time taken to receive a measurement. While a possible contributor to the variance of each implementation, in most cases the time taken to retrieve a measurement may be considered a flat rate overhead. However, because most components of the test environment run as processes on quiescent UNIX machines there is potential for interference and thus a variation in the time taken to extract measurements from the switch even for these, simpler, algorithms. However, an estimate of the contribution made by the measurement system can be made from a minimal MBAC algorithm such as AC-ST. Aside from a decision (to admit or not), the time taken for this algorithm solely consists of that taken to extract a measurement.

The difference between E-MAE and E-MPFE are well emphasised by this table; the availability of estimates in the per-flow estimator, as used in AC-MPFE (back-end), is very low at less than 6 estimates per second. This would imply that an AC based upon this estimator and using the pessimistic policy P-PA would severely impede the admittance of high rates of admission. Additionally with nearly 175 ms between estimates, there is ample potential for admission decisions to be made based upon incorrect characterisation of the current traffic flows. In the *Measure* estimators the block length τ is the tuning variable used to characterise the measured traffic. The role of the block length in the behaviour of the AC policy is being usurped by the time taken to perform an estimate of the current line utilisation. The performance of no other estimator emphasises the importance of not a high but an appropriate level of performance in the estimator. Should this estimator be relied on for traffic that varies primarily over much longer timescales then such a long computation period would be less important. But, because the computation period is significantly slower than the mean period between new flow arrivals (e.g. Exp1), this is serious cause for concern.

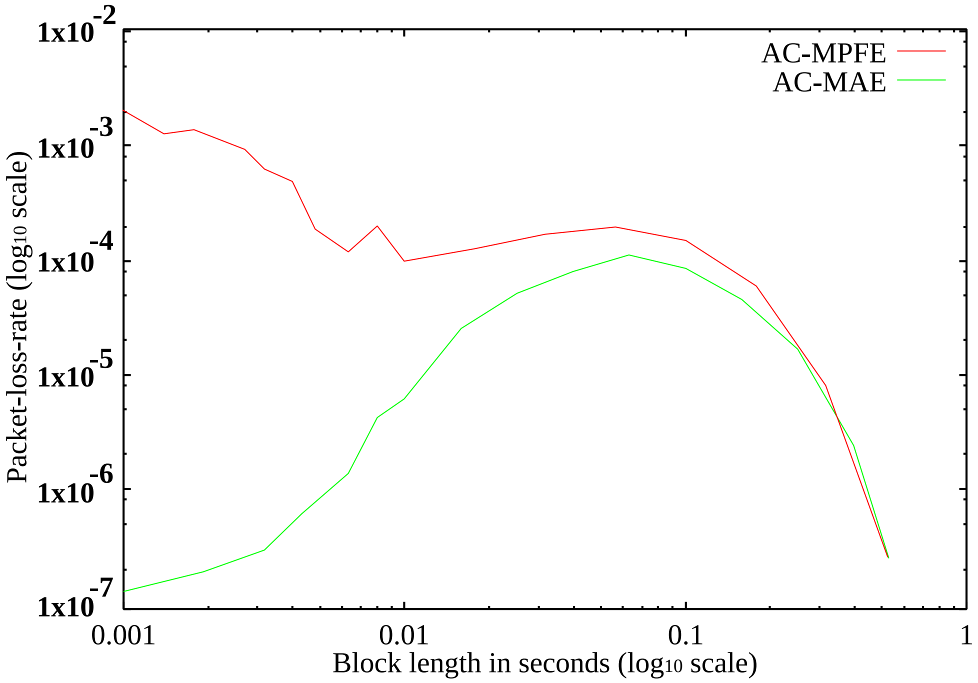


Figure 5.10: Loss ratio versus block length for two implementations of the *Measure* MBAC algorithm.

Using the AC-MAE and AC-MPFE algorithms as examples — Figure 5.10 shows the loss-ratio plotted against a range of values of the block factor, τ, of the estimator for that algorithm. The reason for the ‘maxima’ features may not seem clear — why should the measured loss-ratio decline as the block length increased beyond a certain value? The reason was the use of pessimistic admission policies, P-PA.

Both AC-MAE and AC-MPFE, incorporate pessimistic admission policy P-PA that assumes an incoming flow is transmitting data at the peak-rate declared when the flow was attempted. The algorithm will continue to assume the new flow is transmitting data at the peak-rate until the next estimate is available. Before the availability of a new estimate, the MBAC will add to the currently available estimate the peak-rate of the newly established flows. This peak-rate component is removed once the new estimate is available; since the new flow will now have data as part of the most recent block length τ. If the block length τ is significantly longer than the period between which new flow attempts are being made a pessimistic algorithm will turn away a significant number of these new flow attempts. This effect accounts for part of what we see in Figure 5.10.

Once the values of the block length τ are greater than 100 ms, (the mean rate at which flows were arriving for the experiments of Figure 5.10), the algorithm becomes increasingly pessimistic. As the algorithm becomes increasingly pessimistic, the loss ratio drops as fewer flows are admitted by the MBAC algorithm.

As the block length τ decreases to < 100 ms, the number of flows can be increased without the pessimism policy overly-affecting its behaviour; this can clearly be seen in the results for AC-MAE of Figure 5.10. In contrast to this behaviour, AC-MPFE in Figure 5.10 displays quite a different effect. When the block length is greater than 100 ms, the algorithm becomes pessimistic and the response curve reflects fewer concurrent flows with a lower loss ratio. In contrast to AC-MAE, the loss ratio to block length relationship displays a larger loss ratio across all block values and as the block-length becomes smaller, the loss-ratio relationship diverges from that of the AC-MAE curve. As the number of flows increases and then as the block-size decreases the performance of the algorithm deteriorates. The reasons for this are in the nature of the E-MPFE estimator and in its implementation: this topic will receive specific attention following a comparison of the performance of algorithms.

As displayed in Figure 5.10: the difference in behaviour of E-MAE and E-MPFE significantly alters the behaviour of the AC algorithm. The per-flow estimator implementation has a slower calculation period than the estimator based upon aggregate measurements; this can introduce significant errors into the estimation of effective bandwidth. Principally these errors occur because the calculated estimate is working with data collected, on average, 160 ms before the estimate is finally available. It was noted in Section 2.4 how the random-variable nature of measurements can cause certainty-equivalent algorithms to give poor performance. What is seen here is an expansion of this effect into the calculated estimations: not only do the measurements encapsulate some figure of deviation but the delayed delivery of these figures to the estimator and variable time for estimation causes a significant deviation in the estimate itself. As a result, the traffic of new flows may not form part of the mix measured and an under-estimate will result. Incorrect (under) estimation of the effective bandwidth requirements would be a direct cause of the results seen in Figure 5.10. The reason that this system does not show an underestimation is that the MBAC algorithm will greedily admit flows if the current calculation of effective bandwidth will allow it — in this way the 'greed' of the admission process will maximise the impact of significant variance in the measurements, or estimate.

The performance of an implementation does not simply depend on the speed with which Equation 4.4 can be calculated. For the implementation based upon per-flow measurements Table 5.7 reveals a great deal of information about how the time is used in the estimation of an effective bandwidth. Data management involves the summing of $\hat{X}_t$ for each flow into the single $\hat{X}_t$, the input of Equation 4.4. This task is slow to perform because it requires a large amount of information to be manipulated with each set of per-flow measurements. This information indicates which flows are active at the time of the measurement. Checking the validity of each block for each flow accounts for a significant portion of the 'data management' entry in this table. The two items of this table account for 93.5% of the time required to calculate the effective bandwidth estimate. No other single component of the per-flow estimator accounts for more than 0.1% of the total time. Any reduction of these components stands to return significant improvements in the speed at which estimates can be made available.

Table 5.7 shows that, for a E-MPFE estimator based upon per-flow measurements, a significant time is spent managing the per-flow data including validating measurements and summing this data into $\hat{X}_s$ terms.

Module	% of time
Calculating the $\sum_{t=1}^{T} e^{\theta \hat{X}_t}$ term of Equation 4.4	19.3
Management of per-flow data: the conversion of observations into $\hat{X}_t$	74.2

Table 5.7: Top two *Measure* estimator modules listed by the percentage time used in the recalculation of an estimate.

The speed of the per-flow estimator is influenced by the number of active flows at any time. The greater the number of current flows then the greater the number of samples to be summed to create each $\hat{X}_s$. As a result, the greater the number of concurrent flows, whether due to flows with particularly small traffic requirements, or due to the efficiency of an algorithm producing a smaller per-flow estimate of utilisation, the slower the effective bandwidth estimate will be calculated.

In addition to the number of flows, the time taken by both per-flow and aggregate algorithms will be directly proportional to the number of blocks, T. The longer the history, the more $\hat{X}_t$ terms and thus the longer it will take to calculate an effective bandwidth estimate. The increase in the calculation of the $\sum_{t=1}^{T} e^{\theta \hat{X}_t}$ term is because compared with the aggregate algorithm, in the per-flow algorithm there are more $\hat{X}_t$ terms. The number of $\hat{X}_t$ terms will relate directly to the longest flow in progress.

Testing the premise that the estimates may be out-of-date or not contain measurements of traffic from the most recently admitted flows, a comparison is conducted of the same traffic type under two different flow-attempt conditions. It is postulated that fewer flow attempts spaced less frequently would lead to better results, this would be because the algorithm had, on average, more time (between flow attempts) to obtain a sample of the current flows of traffic and thus would be able to calculate a more accurate estimate of the current utilisation.

In contrast with the previous experiments where the mean rate of flow attempts is 10 per second, the mean rate is set to one attempt every 2 seconds. In order to maintain the same high traffic load on the system the mean flow lifetime of a successful flow is 200 seconds. The AC algorithm will have access to an effective bandwidth estimate created using a larger sample of the current traffic mixture, and as a result, it is expected that the measured loss ratio will be lower due to a more pessimistic (and more accurate) behaviour of the AC.

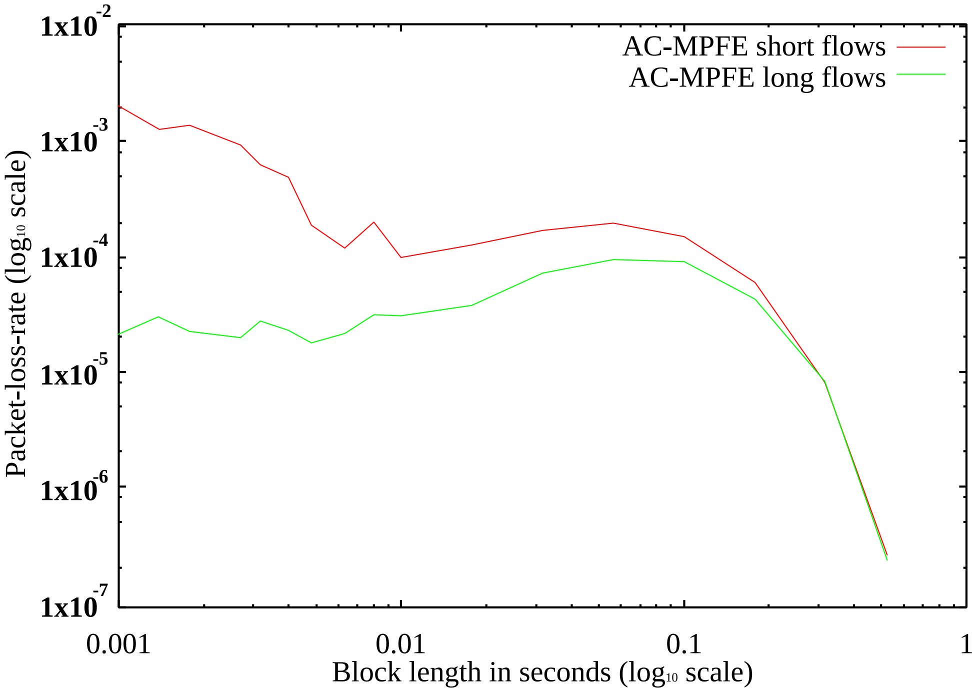


Figure 5.11: Measured loss ratio versus Block length using traffic type TP10S1 — a contrast in flows-per second.

Figure 5.11, made using AC-MPFE, shows that the form of each curve is similar, but that in the case where the number of flows attempted per second is lower, the loss-ratio results are smaller as the block length is reduced. A conclusion from this is that with fewer flow attempts being made, there is an improvement in the quality of effective bandwidth estimates available — as a result, the CAC algorithm decisions give an improved measured loss ratio. As was discussed above, the improvements seen here are a combination, both of improved estimates with additional time to collect more traffic samples from which the effective bandwidth estimate is to be calculated, and more time to allow for the improved estimates to be calculated and made available for the next admission decision.

Alongside these algorithmic contributions to the inefficiencies of the E-MPFE estimator is the realistic proposition that this algorithm is not implemented in the most efficient manner possible. Considerable time was invested improving the performance of both the basic estimator and the management of the per-flow data through code restructuring and the use of profiling tools. Figures given in this section use the optimised estimator.

Algorithm		Lines of Code	Difficulty to Implement 1 (easiest) — 5 (hardest)
AC-PRA		200	1
AC-ST		210	2
AC-AR		250	2
AC-CB		300	3
AC-MS	(front-end)	120	2
AC-MS	(back-end)	250	3
AC-MPFE	(front-end)	200	1
AC-MPFE	(back-end)	2050	5
AC-MAE	(front-end)	200	1
AC-MAE	(back-end)	1500	4
AC-MVE	(front-end)	200	1
AC-MVE	(back-end)	300	1
AC-KQ	(front-end)	200	4
AC-KQ	(back-end)	400	4
AC-LBE	(front-end)	200	1
AC-LBE	(back-end)	200	2
AC-GT	(front-end)	150	1
AC-GT	(back-end)	900	5
AC-T		150	1

Table 5.8: Implementation complexity given by the amount of code in each AC algorithm and a subjective assessment of the difficulty of making the implementation.

Any MBAC algorithm has a perceived complexity — the implementation of any complex algorithm may have a significant code investment. Table 5.8 gives an assessment of the complexity of the code for each MBAC algorithm implemented here. These results are based on the size of the code base and a subjective assessment of the difficulty making the implementation. The values of code size given for policies and admission algorithms encompasses routines that would accept (and reject) new flows and handle the departure of flows, but this code does not perform any actual measurement function. The layout of this table allows easy comparison with the entries in previous Table 5.6 giving times of execution.

While it is important not to over-interpret these values, a conclusion from the results of Table 5.8 is that the implementation of even the most complicated algorithm could be considered relatively straightforward. In the context of switch/router software, which can run to many hundreds of thousands of lines, the code investment in these algorithms is not trivial but certainly minimal. However, as found for the E-MAE and E-MPFE estimators, care and attention needs to be applied to the implementation and a substantial investment in optimisation can be anticipated.

5.4 Discussion

In the presentation of the theoretical foundations of MBAC algorithm and MBE, Table 4.2 on Page 131 gave a summary of the measurement and long-term memory requirements of the estimators.

Table 4.2 summarises an important finding in this work: while an algorithm based upon these estimators may be better in theory — physical limitations may make it impractical. In contrast, the simpler algorithms of AC-ST and AC-MS have openly left the complexity of adjusting the algorithm up to the user. Algorithms such as AC-MPFE with the advantage of being able to nominate buffer characteristics but requiring per-flow measurements, may require too much information measurement, too much computation and have too high a storage overhead to make the current implementation effective for the tests it was placed under. Figure 5.11 reinforces the possibility that the nature of the experiments in this evaluation may not be best suited to showing the true potential of the *Measure* per-flow estimators. Due to the slow implementations, these estimators may be better suited to utilisation estimation with a control over flows that is taken over hours and days rather than minutes, seconds and tenths-of-seconds.

Algorithms of simple implementation complexity such as AC-CB, while relying upon some *a priori* flow information to supplement its estimation, combine this with simple implementation requirements to give a potentially powerful solution to the MBAC algorithm problem. In this way, such algorithms cover a middle ground that may provide some use in circumstances where adapting the algorithm to changes in new flows is, at least in part, performed by the algorithm itself.

While the computational requirements of any algorithm will place an upper limit on the number of estimates that can be computed in any period; it was clearly illustrated that the management of measurements can have a greater effect on the computation period than the estimation algorithm itself. As general purpose computer systems see use as the central processors of routers and switches there is a lessening importance of computational overheads. However, these requirements can not be considered insignificant and memory issues, particular memory interconnection with the measurement point is still an important resource. Thus, from Table 4.2, the drawbacks of a simpler algorithm may be overcome by its lower memory and CPU overheads.

Table 4.2 illustrates the advantages of the aggregate-measurement approach of E-MAE over the original per-flow implementation, E-MPFE. In addition to the issues of measurement management, there is the significant quantity of measurements that need to be moved from the statistics collector to the component calculating the estimate. For a port carrying 256 flows and a block length τ of 440 μs in length, 32 bit measurement made every 440 μs will require over 2 Mbytes per second be moved into the estimator for each port of the switch. For aggregate measurements, this figure would be reduced to 9 Kbytes per second for the same measurement period.

Measurements made using a 32 bit quantity are common in SNMP implementations [Case88], although a greater quantity of data could be transferred using fewer bits per measurement. Selecting the correct measurement width is related to the maximum number of events that would occur within a particular measurement period before a counter-wrap occurred. For example, in a SONET OC-3 system, over 300,000 cell (packet) events may occur in one second, so for a measurement period of 100 ms 16 bits would not adequately represent the potential throughput in that period. However, if the measurement period was lower, e.g. 10 ms, 16 bits would be ample. Naturally if the measurement period is 10 ms then it is assumed these results would require transmission 10 times more often than those made over 100 ms; the precise selection of parameters would need to be a balance between the bandwidth available to transport measurements and the resolution of the measurements themselves.

5.4.1 Timescale

In this section the issues of timescale as related to MBAC algorithms is examined. In particular, the effect that the measurement period can have on admission behavior is demonstrated, and the interaction between the policy of the MBAC algorithms and the period over which measurements are made is discussed.

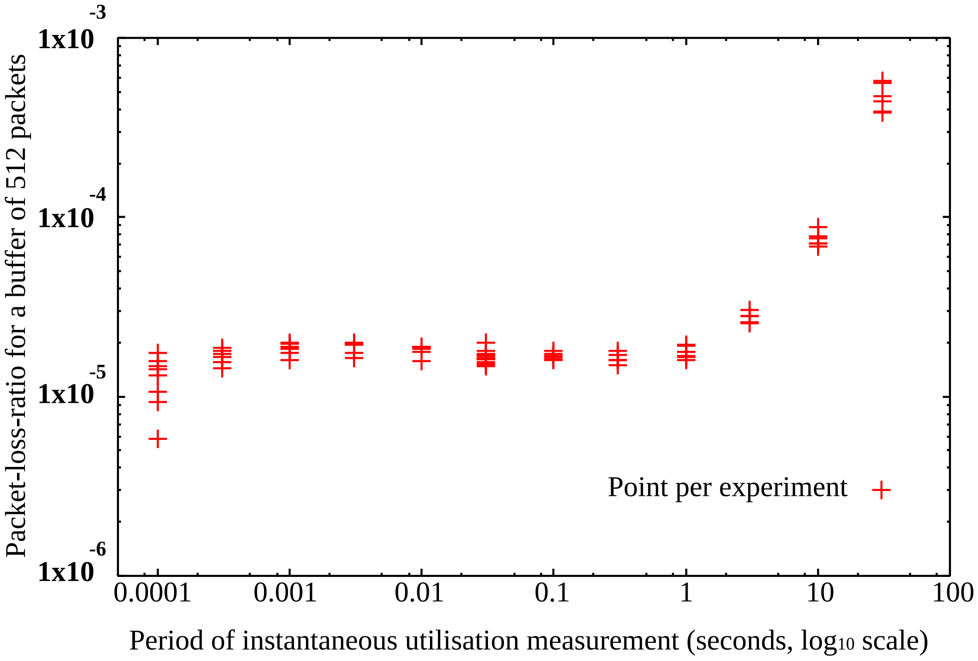


Figure 5.12: Packet-loss-ratio versus period of instantaneous utilisation measurement

The E-IU algorithm uses a single measurement as input; this measurement is combined with a threshold value. Because of its simplicity, the E-IU algorithm makes useful illustration of aspects of both MBE and AC policy. The E-IU algorithm, implemented with policy P-TO, is used to gather the results of Figure 5.12.

Figure 5.12 shows the achieved loss-ratio for a range of experiments where the period over which this single measurement is varied. Using Exp1, each of the experiments in this figure was repeated using different measurement intervals. In this figure, each point represents one experimental result, several experiments were run for each value of the measurement period.

From this graph, it is clear that as the period over which the instantaneous measurement is taken approaches 10 seconds, the mean holding time of the incoming flow requests, the loss ratio deteriorates. Such a result can be predicted by the knowledge that as the measurement period is increased, sharp short-term transient periods in the line utilisation will not be reflected in the line utilisation measurement. Therefore, flows will be more likely admitted when the traffic of those flows cannot be supported at the desired loss-ratio. As the measurement period is reduced below 2.25 ms, the buffer length multiplied by the transmission speed, the measurement is sufficiently small that the algorithm may over-react to traffic fluctuations normally absorbed by the buffer. Such a conclusion is neither clearly supported nor clearly rejected by Figure 5.12. Measurement below this period is unrealistic, overly interacting with the physical limit of the test hardware, particularly its minimum measurement period, the speed of access to the measurements. As a result, any further work to investigate below this value, would require a new test environment or different approach.

5.5 Summary

The intention here has been to discuss a comparison of a subset of MBAC algorithms. However, unlike the comparisons of [Knightly98], [Breslau00] or [Jamin97b], the approach here has been to implement the MBAC algorithms in a purpose-built test environment that allows a modular substitution of one MBAC for another between consecutive test runs. The result has allowed a high-fidelity comparison of MBAC algorithms, testing each against real-world traffic sources. In addition to comparisons of algorithm performance (as measured by line utilisation and loss-ratio), a comparison of aspects of the implementation such as algorithmic complexity and overheads has been made that an implementation can only simplify or ignore.

The comparison methodology was given in Section 5.2: Section 5.2.1 proposed a set of criteria intended to duplicate common comparison practice as well as contrasting this with several new approaches to MBAC algorithm comparison. The environment based upon that presented in Chapter 3 was summarised and the eight different experiments were detailed in Sections 5.2.2 to 5.2.10. These experiments covered a variety of situations both with full control of the network and control in the face of background traffic as well as using a wide variety of traffic types such as voice, video and IP traffic.

Section 5.3 presented the comparison results. In Section 5.3.1 traditional technique for comparison, system-wide loss versus system-wide utilisation has been a common technique for comparing AC algorithms. However, this chapter identifies that such a loss-load curve is not exclusively the result of an MBAC algorithm but rather is a characteristic determined by the traffic in combination with the network resources of buffer and link capacity. This is an important issue as it may invalidate conclusions drawn by previous authors after examination of such comparison results. Further to this, conclusions from the loss-load results indicate that any AC algorithm can be configured to achieve any particular loss-load results at the risk of minimising any safety margin required to overcome poor measurement assumptions (such as measurement variance.)

Alongside loss-load results, figures for loss-acceptance were shown, in itself the acceptance process was shown to be a function of the AC policies and the flow-arrival process. While the use of loss-acceptance curves as a mechanism for differentiating AC algorithms was doubtful, arrival figures gave an interesting insight into the behaviour of AC algorithms, acceptance policy in particular for heterogeneous flow arrivals.

Section 5.3.2 illustrated the difficulties that exist developing a relationship between the AC algorithm parameters and the resulting level of utilisation or loss ratio. The wide range of parameters being dependent upon desired outcome, traffic characteristics and upon flow characteristics, was illustrated for each MBAC algorithm. The importance of particular parameter controls was noted — in particular how uncalibrated parameters do not provide users with any form of control while values such as target loss or utilisation allow selection of a target criteria that is more tangible to a user. Many algorithms posses additional tuning parameters alongside the primary control parameters, the difficulties introduced by such parameters, particularly in undesirable operation were also discussed. A example common to all MBAC algorithm is the minimum width of measurement, a parameter about which the authors of MBAC algorithms will make assumptions but the selection of which is largely left as an exercise for the implementer.

The stability or response to change along with an examination of the repeatability of outcome of any particular AC algorithm was examined in Section 5.3.3. In stability experiments the policy in use was identified as a critical component in an AC algorithm's response to changing circumstances, while little was revealed in repeatability experiments leading to a conclusion that the repeatability of results for AC algorithms depend more upon measurement and traffic characteristics, elements common to all experiments, than to the AC algorithm used.

Finally, Section 5.3.4 presents results on the resource overheads and implementation complexity of the AC algorithms. This section showed that, independently of the elegance of an algorithm's theoretical structure, the difficulty in implementation might reduce its operational efficiency. A simple comparison of implementation code demonstrated that any AC algorithm is only within one order of magnitude of another in code complexity and a numerical comparison of computational duration gives insight into the finite nature of computation and the relative complexity each implementation faces.

Following the results, Section 5.4 provides a discussion of how the experiments may be biased in favour of or against a particular MBAC algorithm. The physical reality of an MBE, requiring measurements that have finite properties such as the transmission time between measurement-agent and processing agent was highlighted. It became clear that an MBAC proposal that may appear fantastic on paper could, due to implementation constraints, perform much more poorly than a simpler approach. Lastly, the discussion centered upon the issue of timescale. Section 5.4.1 uses a simple estimator to reinforce how an AC algorithm must cover a spectrum of timescales from cell-multiplexing events through to flow-lifetime events.

In the original intentions of this chapter, the ideal MBAC algorithm was sought. Through this chapter it was illustrated that the wide range of MBAC algorithm address a variety of needs covering outcome and accuracy as well as computational overheads and implementation complexity. As might be predicted, no perfect MBAC algorithm exists — with a requirement to control events at multiple timescales using only the ability to accept or reject new admission requests, it became clear that for such a difficult task each of the approaches investigated had unique advantages and disadvantages. The very nature of the task set for AC algorithms will result in any particular offering occupying only a small part of the complete solution-space.

Chapter 6

Dynamic Allocation of Resource

6.1 Introduction

Rather than addressing the issues of resource control through flow admission discussed in Chapters 4 and 5, this chapter identifies the resource-management contribution that measurement-based estimation can make through control of flow scheduling. This chapter proposes using a measurement-based estimator to provide differentiated service by adjusting the service-weighting on a queue scheduler.

Routers and switches that employed resource partitioning, such as the division of link bandwidth between different classes of traffic, have used a scheduler that fixed, as part of its algorithm, the amount of the resource (outgoing bandwidth) to be allocated to each class, e.g. [Clark92, Floyd95, Stoica97].

As a result, traffic in queues that were not serviced could be either delayed or lost as the queue filled and overflowed. In such a scheme the resources of buffer-space and the service-weights of the scheduler were allocated according to policy e.g. based on a simple priority scheme or with an assigned weighting based on the value of each traffic class.

In the current networking environment, changes in traffic requirements have forced fixed resource-allocation systems to require updating or to waste the resources of link bandwidth or buffer capacity. Section 2.1.1 noted how the correct commitment of resources can be unknown or difficult to compute. It can require a configuration for either the worst-case conditions or for a first-come first-served scenario. The use of MBE, the solution embraced by the AC community, was investigated in Chapters 4 and 5. Enhancing the performance of resource scheduling through the use of an MBE is proposed in this chapter.

The remainder of this section discusses the rôle of a switch that offers service differentiation and the objective of the approach proposed. Section 6.2 describes scheduler requirements. Finally, this initial section justifies the choice of MBE used in the implementation of this chapter. The implementation of this technique is presented in Section 6.3. Section 6.4 discusses the approach taken to produce the results of this section, identifying relevant parts of the environment presented in Chapter 3. The outcomes reported in Section 6.5 indicate the success of this approach as well as identifying areas for improvement. Section 6.6 discusses several matters that arose during trials of the dynamic-allocation mechanism. The first of these is a discussion of the interaction of elastic traffic sources with the proposed mechanism followed by a discussion about the relationship that exists between the dynamic allocation approach, the MBE, and the session traffic. Lastly, Section 6.7 summarises the approach taken and identifies some of the problems with this technique as well as noting potential approaches to solving those problems and directions future work could take on this approach to adaptive scheduling systems.

6.1.1 Background

This section establishes the environment in which a network switch offering service differentiation would operate. It is proposed not as a catch-all answer to current networking problems but as a novel and unique solution to problems arising from the desire to offer diverse and sometimes orthogonal service facilities to a wide variety of traffic types.

The objective of any network supplying varied services over a single network infrastructure is the ability to guarantee differing grades of quality of service to the network customers. The dual approach of admission control and an appropriate scheduling algorithm has long been considered central to supplying QoS in an integrated services network (e.g. [Hyman91]). Such approaches are suitable for INTSERV's RSVP [Braden97] or ATM Forum's signalling specification version 4.0 [ATMF95].

However, an alternative method is sought that will allow the supply of QoS to traffic that does not possess explicit flow admission and, thus, for which admission control is not available. Networks wishing to provide QoS but without explicit admission control are a central idea of the approach of the differentiated-services network architecture, DIFFSERV [Blake98, Nichols98].

Each flow admitted using an AC algorithm (such as those described in Chapter 4) is guaranteed to receive the resources it requires. In contrast, flows carried in a differentiated services system such as DIFFSERV do not receive an individual guarantee of resources. Instead, a guarantee is made to the class of traffic to which that flow belongs. The class of traffic will receive all the resources it requires but individual flow properties and flow interaction will mean that the per-flow resourcing will be only statistical in nature. At any time a flow may not receive the resources it requires although the aggregate of flows in the class source receive the necessary resources.

The example used throughout this chapter is a single *core* router which is assumed to handle several classes of traffic. The classes themselves could be classifications of service, as the DIFFSERV model would propose — perhaps a combination of Olympic service which consists of three classes: bronze, silver, and gold. Data using this model are assigned to these three classes so that the gold class experiences lighter load than the silver and the silver experiences lighter load than the bronze [Heinanen99]. An alternative set of offerings may be the combination of a high-throughput service, a low-delay (or delay-variation) service, and a best-effort service. This set of traffic classes is a combination of the assured and expedited forwarding classes [Heinanen99, Jacobson99b] with the current default Internet behaviour.

Figure 6.1 illustrates the hierarchy of flows, sessions, classes and links used within this chapter. This simple structure allows flexibility in the grouping of flows within sessions and of sessions within classes of traffic. A link carries all data between switches consisting of one or more classes of traffic. A class describes traffic with a set of properties such as a loss-rate or a boundary on delay. A session carries a particular type of traffic. For any link there are one or more sessions, each session carrying one or more flows of traffic. A flow of traffic represents the data transmitted by one source along one link. Clearly, each session may carry many flows and each link may carry many sessions.

Justification for the common carriage of several different traffic types has been seen with the

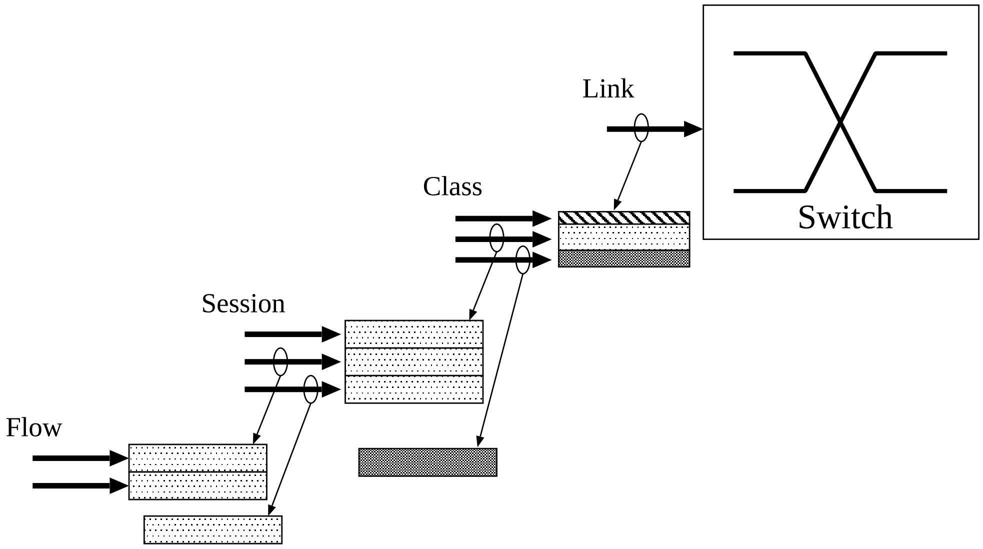


Figure 6.1: Flow hierarchy

increasing popularity of the VPN. A VPN is the emulation of a private data network overlaid onto a public network infrastructure. For example, a company could contract with an ISP to set up a VPN using the Internet to connect two geographically-separated sites rather than set up a dedicated WAN or use a leased line. A VPN can give the company the same capabilities at lower cost by sharing its infrastructure between a number of customers. A VPN is commonly defined as a mesh of endpoints with a given level of expected service. The Service Level Specification (SLS) is a technical specification[1] of the requirements of the network user. VPN services have been available in various forms for an extended period of time and have recently received considerable attention within the ATM, frame-relay, Multi-Protocol Label Switching (MPLS), and IP networking communities [Fotedar95, Rosen99, Gleeson00, Rooney98].

In previous work the use of measurement-based feedback to assist in the provision of VPNs has been examined in [Duffield99b]. Concentrating upon bandwidth-only provision of service, this work showed that measurement-based estimates were a useful contribution as part of an improved VPN provisioning mechanism. While the model presented in this paper allowed the identification of required resources, this information was used only when a new resource provi-

[1] More properly, an SLS is the technical specification part of the contractual Service Level Agreement (SLA) made between a network provider and its customer; in addition to the SLS, the SLA also specifies details of cost, renumeration in the event of failure and related material not relevant to the configuration of the VPN of this example.

sion was required, although this work did prove that an MBE would prove a useful asset in the provisioning of resources.

In the technique documented in this chapter, both bandwidth and delay constraints are considered and the (re)allocation of resource is on a sufficiently-small timescale to allow either allocation of unused resource or the appropriate withdrawal of resource in the case of over-commitment. Provisioning in this way would allow services sharing the network, but relying on uncommitted network capacity, to be used by traffic that only has access when all other users are serviced, such as the not-quite Best Effort traffic proposed in [Carlberg01].

If the scheduler is to offer differentiation between traffic it must differentiate between each traffic-type in the multiplex. An assumption throughout this work is that the traffic is already classified into its traffic-class. Such classification may occur intrinsically in the request for a particular grade of service for a new VPN or classification may be made packet-by-packet at entry to the network. A quantity of literature has arisen on the classification of traffic and the techniques to be used to ensure a router has enough information to assist in the best possible scheduling decision, (e.g. [Nichols99, Feng99a, Cao00]), although it is also apparent that there is still scope for future research in this area.

In the proposed dynamic-allocator mechanism there is less interaction between the classification and differentiation performed by the scheduler. While the scheduling requirements may be seen to change, the classification of traffic is considered to be a task performed outside the control of the scheduler. Feedback from the classification, while not explicitly dealt with, is assumed to only take the form of configuring the maximum resources any particular class of traffic is permitted to consume.

6.1.2 Objective

The objective of this chapter is to show that differentiated service guarantees may be made for combinations of bandwidth and the orthogonal guarantees of loss-rate and delay. While previous work has shown this to be possible for strict AC (e.g. [Elwalid95]), little work has shown that such a system can be realised on current hardware while being able to adapt to changes in load requirements. Combining an MBE with both an appropriate scheduler implementation and control of buffer sharing gives a realisable system that is able to provide the required differentiation in services.

The solution proposed here of a dynamic allocator shares some common objectives with proposals to perform QoS-renegotiation in connection-based systems. QoS-renegotiation schemes adapt the QoS of a connection as required and release unused resources or request additional resources on demand [Zhang95, Grossglauser97a]. The system proposed here would also reallocate unused or over-committed requirements as dictated by the MBE. However, instead of renegotiating the resources of an individual contract, the dynamic allocator renegotiates resource of all the users of a particular traffic class.

6.2 Theory

The packet scheduler required for this implementation is discussed in Section 6.2.1 and pays particular attention to constraints of delay and throughput. The topic of buffer management has been an active research area, Section 6.2.2 highlights some of the directions development has taken related to the dynamic allocator. Finally, Section 6.2.3, building upon the work of Chapter 4, outlines the choice of estimator for the implementation described in this chapter.

6.2.1 Scheduler

In order to provide agreed support of multiple sessions of traffic over a common transmission link, the bandwidth of the link must be divided between the different sessions. Section 2.3.2 presents a number of schedulers and discusses how the simplest scheduler to divide the bandwidth in this way is the WRR scheduler. For the implementation in this chapter, the scheduler must able to bound the delay any particular session incurs in addition to simply dividing up the bandwidth resource. However, Section 2.3.2 notes that WRR can not honour any delay bound. This makes it unsuitable because the desired scheduler must be able to make delay-bounded guarantees while still offering service guarantees. The ideal scheduler is one able to emulate GPS scheduling. The GPS scheduler does not suffer unbounded delay constraints and is work-conserving, minimising the waste of unused link capacity. A close emulation of GPS is limited to packet networks that use a fixed cell length.

Fortunately the test-environment around which the work of this dissertation was done is based upon an ATM network. ATM networks use a fixed packet length so the use of the GPS emulating algorithm is allowed. A suitable GPS emulating algorithm is WF^{2}Q+, proposed in [Stephens99]. The WF^{2}Q+ scheduler is implemented in the switch of the test-environment (described in Section 3.3.1) providing an environment on which the dynamic allocator can be constructed.

6.2.2 Buffer Management

A scheduler will allow a network node to allocate link bandwidth to each session. However, for services such as voice, which is delay sensitive, bandwidth control is not enough. Buffering is a mechanism that improves the loss-rate of both packet and burst multiplexing. Therefore, buffer management provides the controls over loss while also controlling packet delay.

Control over the buffer (size) available to each session is required if the implementation to provide resources for loss or delay constraint as well as link bandwidth, The hardware ([FORE98, FORE99]) used in the test-environment performs VP-VC queueing using two levels of loss priority. This allows a flexibility in the buffering of sessions.

It is possible to specify the depth of the buffer available to each session. For delay-bound sessions, packets that exceed a buffer threshold are discarded but for loss-bound or throughput-guaranteed services the arriving packets are marked as eligible to be discarded if no further capacity remains in the total buffer pool shared among sessions. This technique results in loss-sensitive or high-throughput flows receiving additional buffering for as long as the additional packet arrivals do not impact on any other session. These loss-sensitive or high-throughput flows may have an additional delay constraint imposed, which may be implemented with an upper limit on the buffering of both marked and unmarked packets.

It is clear that the implementation relies heavily upon the facilities of the underlying hardware. However, given the ever increasing complexities of scheduling and queueing disciplines available in switches, such a reliance does not seem unreasonable.

6.2.3 Measurement-Based Estimator

The potential result of combining an adjustable scheduler with a form of buffer management is to simultaneously offer delay-bound services, and loss-bound services, as well as those with a throughput guarantee. However, the task of computing the appropriate size of buffers and scheduler bandwidth falls to the MBE.

An ideal MBE would allow three critical resource computations: firstly, a computation of the capacity required to maintain a given delay-bound with a given probability; secondly, a computation of the capacity required to maintain a loss-rate given a particular buffer size, and lastly, the buffer size required to maintain a loss-rate for a given rate of service which would be required for a service with throughput guarantee. Additionally, the estimator must be adaptive so as to avoid the need for a number of traffic-dependent configurations.

Estimators such as *Measure* (Section 4.2.4) initially seem ideal for the task because they are able to combine a series of measurements with any two of the input parameters of buffer size, loss rate, or *effective bandwidth* and compute an estimate of the third parameter. However, as shown in Section 5.3.2, this estimator depends critically upon a traffic-dependent tuning value τ: the period over which measurements are taken. Unlike other self-tuning estimators such as E-TG (Section 4.2.8) no robust mechanism currently exists for computing this value.

Estimators such as E-MS, E-CB, E-ST, or any estimator based upon a bufferless model of the network requires the computation of a complicated surface relating the desired outcome to the tuning parameter for each traffic-type. Additionally, this surface would need to exist in multiple dimensions in order to account for changes in each of link bandwidth, loss rate and queue size. Currently E-TG (Section 4.2.8) is based upon a bufferless model but it shows the greatest promise because it is self-tuning and robust in the face of a large variety of traffic types.

The E-KQ estimator of Section 4.2.7 is used as the MBE of this chapter. This algorithm relies on little prior characterisation of traffic flows or critical timescales and is able to characterise traffic over a range of time scales. In addition to requiring little per-source tuning, this algorithm is able to compute the minimum resource requirements of buffer size or link capacity for each of the three situations outlined above. It is not ideal because it tends towards overestimation of the upper bound based upon its measurement characterisation. However, as is demonstrated in the results section of this chapter, using a pessimistic computed upper-boundary allows a substantial improvement over techniques based upon the peak-rate allocation of resources.

Section 4.2.7 outlined the E-KQ estimator, and this estimator, as represented in Equations 4.9 through 4.13, allows computation of an estimate of *effective bandwidth* for a nominated loss rate and queue size.

In order to compute a queue size for a given loss rate and link capacity combination, as would be the case for a traffic-class with a guaranteed throughput, Equation 4.48 — the admission test of short-term traffic effects against the queue size — can be used to estimate the queue capacity q from the measured traffic envelope, R_k and the capacity, C estimated using an equivalent transform of Equation 4.49. The computation of the capacity, transformed from Equation 4.49, is given as

$$\bar{R}_T + \alpha_{\text{long}} \sigma_T = C, \tag{6.1}$$

where α_{long} is given in Equation 4.10. A transform of Equation 4.48 gives an estimate of the queue size q by:

$$\max_{k=1,2,\cdots,T} \{k\tau(\bar{R}_k + \alpha_{\text{short}}\sigma_k - C)\} = q. \tag{6.2}$$

where Equation 4.12 defines the value of α_{short}.

By using the ability to nominate queue size and overflow probability, service allocations can be computed for certain queue sizes. The ability to compute maximum buffer sizes from delay constraints[2] allows the computation of service allocations treating the overflow probability as the same probability that packets will be delayed beyond the delay-bound.

For this implementation, previous experiences with active buffer management in partially-shared buffers (e.g. [Kroner91, Matsufuru00]) indicated that to form an adequate differentiation between traffic, such systems are sensitive to the load of each traffic type in the buffer. [Dovrolis00] concludes that such systems are hard to tune, being highly sensitive to the actual threshold value used.

[2] The boundary on the delay experienced through the buffering of any packet in a flow may be considered as the transmission time per packet multiplied by the capacity of the queue.

As a result, the approach taken here is different. The buffer sizing is not used as a principle mechanism to differentiate one session from another. Instead, buffer sizing is used principally as an upper-bound on the delay properties of traffic where appropriate. If traffic is delay sensitive it stands to reason that traffic delayed by more than a nominated amount has no value and that traffic exceeding this delay ought to be discarded. In contrast, the traffic may not be discarded if it exceeds the buffer thresholding values for flows that do not have an explicit delay constraint. This approach makes available transmission capacity that would have otherwise been wasted on traffic that was outside its delay constraint.

This scheme may be thought of as a form of work-conservation for the flows that have no delay-constraint but non-work-conservation for flows that do have a delay-constraint. The link-capacity that may be wasted on the delay-constrained traffic with packets now too delayed to be of use are used by the traffic that has no such delay-constraint.

6.3 Implementation

In Section 6.3.1 the representation of policy is discussed. In addition to the data-structures holding policy, the handling of connection data is reported.

In section 6.3.2 the implementation of the dynamic allocator is described. Using the WF^{2}Q+ scheduler (from [Stephens99, FORE98]) combined with an estimator derived from the AC-KQ MBAC algorithm ([Qiu98b, Knightly98]) and summarised in Section 6.2.3, the implementation is able to establish the requirements of current flow-aggregates and adapt the resource allocations made to provide for those flow-aggregates as required.

6.3.1 Policy

A policy will arise to divide the resources whenever there is a set of finite resources to be divided among competing interests. This Section discusses aspects of the policy implementation as well as noting several areas where the current implementation may be extended with further work.

By default, the mechanism of the dynamic allocator groups flow-aggregates into classes and a set of default class-specific system-wide policies are implemented to provide resources for each class. In addition to a set of system wide resources, specific flow-aggregates may also elect

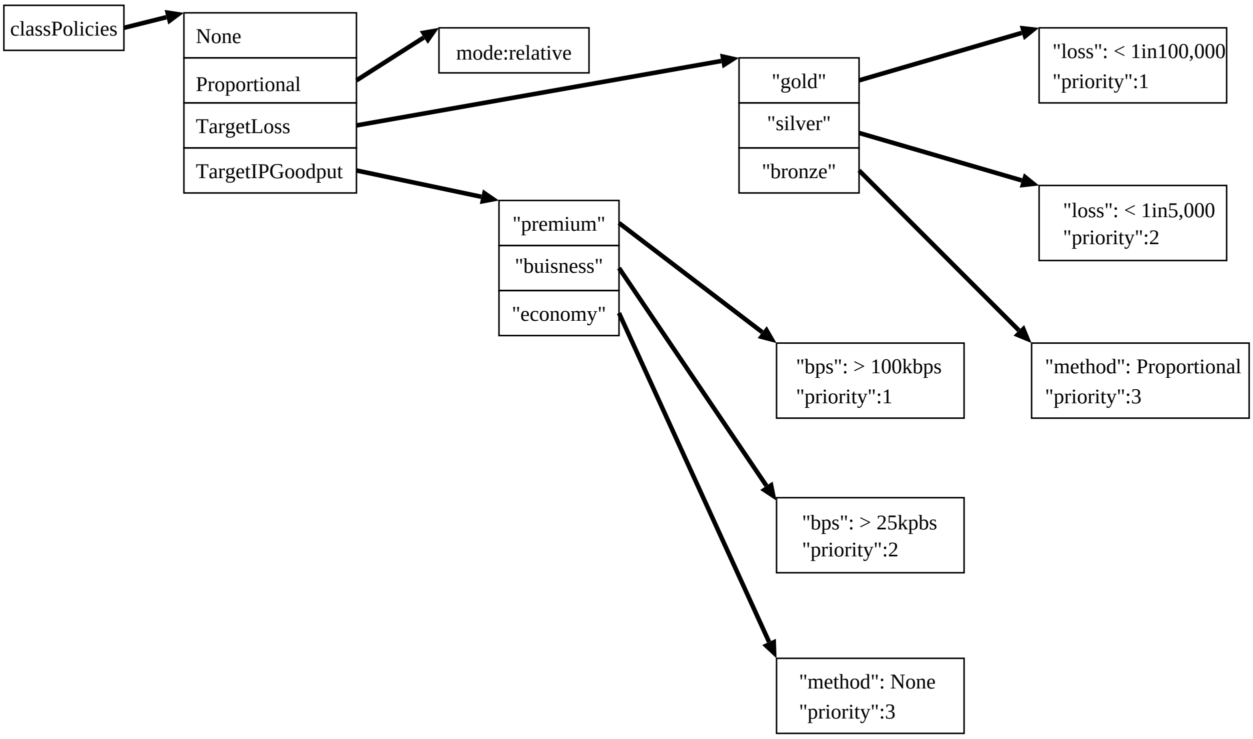


Figure 6.2: Policy structure defining default actions for a switch.

special handling with a specific policy or set of policy-settings. Such a flexible approach allows the management of flow-aggregates with resource requirements that do not correspond to the system-wide class defaults.

A sample of default policy actions are represented in Figure 6.2. In this structure the default actions for four policies are defined: *None*, *Proportional*, *TargetLoss* and *TargetIPGoodput*. With the exception of *None* each policy has the default settings and actions defined in this structure. This structure contains the initial behaviour when the system is processing traffic of a particular class. In this example *TargetLoss* sub-class “gold” gets a default target loss ratio of better than 1 cell in 100,000.

Policies are cascadable, in the example any connections with *TargetLoss* sub-class “bronze” use the method *Proportional*. Such connections would include the defaults for the *Proportional* method, e.g. option “mode” being set to “relative”. For the proportional method, absolute mode would allow the specification of a dedicated quantity of link bandwidth while the relative mode

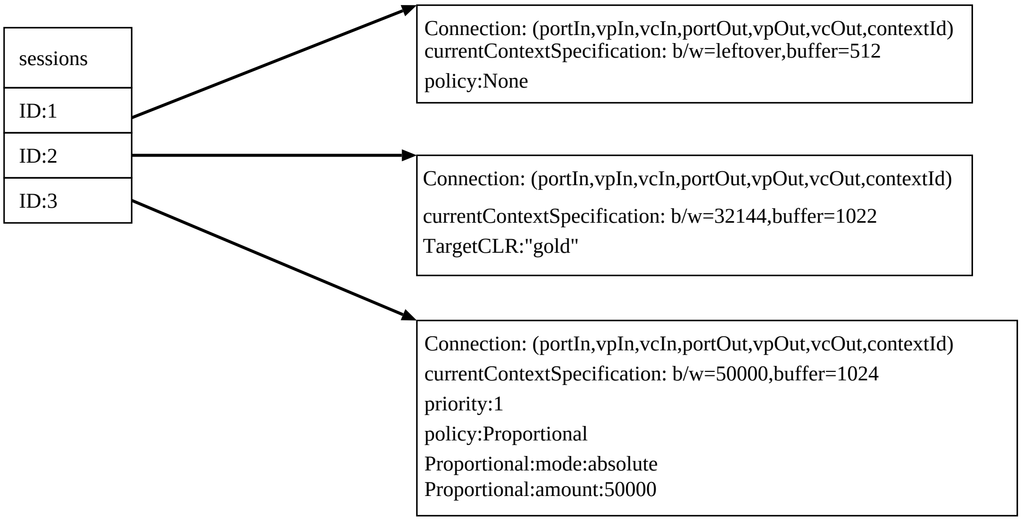

sessions
ID:1
ID:2
ID:3
Connection: (portIn,vpIn,vcIn,portOut,vpOut,vcOut,contextId)
currentContextSpecification: b/w=leftover,buffer=512
policy:None
Connection: (portIn,vpIn,vcIn,portOut,vpOut,vcOut,contextId)
currentContextSpecification: b/w=32144,buffer=1022
TargetCLR:"gold"
Connection: (portIn,vpIn,vcIn,portOut,vpOut,vcOut,contextId)
currentContextSpecification: b/w=50000,buffer=1024
priority:1
policy:Proportional
Proportional:mode:absolute
Proportional:amount:50000

Figure 6.3: Structures held for each connection

would indicate access to an equal proportion of the remaining bandwidth. The absolute mode may easily allow an over-commitment of resources and is provided only as an example of the flexibility available.

The use of default-policy structures means little information need be carried for each set of flows. The default policy will be the switch default and if the set of flows requires special treatment, perhaps “gold” service, it need only to be tagged as such. Such tags could be derived from the traffic type in the case of multiplexed multi-media traffic or from the SLA with the customer from whom the traffic arrived.

Information on flows is held in a structure such as Figure 6.3, this structure contains the physical information for each connection (or group of connections in the case of aggregates), along with the current policy and the connection defaults for policy.

In this example three connections are given, each specified slightly differently. The first *ID:1* specifically uses the *None* (best-effort) policy while *ID:2* specifies itself as “gold”. *ID:3* indicates that its allocation is to be specifically made using the *Proportional* method in “absolute” mode and that, using this method, a rate of exactly 50,000 packets per second will be allocated.

In a hierarchical classing structure (gold, silver and bronze) traffic may need classification. For example, a customer has an SLA that allows a certain amount of each particular class of traffic

— the traffic from this customer may require classification into each class as its SLA dictates. Such classification of traffic is outside the scope of this paper although classification schemes are being explored within the context of DIFFSERV [Blake98, Nichols98] as well as outside the DIFFSERV system [Clark98, Feng99b, Cao00].

Policy systems have existed for a long time and a more-recent example for the Internet is the Common Open Policy Service (COPS) [Boyle00b, Boyle00a] system written to describe the client and server service models in the INTSERV environment. A similar mechanism is expected to arise for DIFFSERV [Bernet99]. Such systems may specify the structure of policy in a mechanism that allows complex policy specification but using a formal language to allow the system to ensure internal consistency.

The prototype implementation here is sufficiently functional to demonstrate the dynamic allocator based upon an MBE. Currently, policy instantiations along with experiment configurations are done using a general-purpose programming language interfaced directly with the network components. This has been done to make experiment and policy construction as simple a process as possible. Such a system is not suitable for a final implementation — extending and formalising the policy language is an acknowledged area of future research.

On an issue related to the formal policy processing mechanism, the nature of the dynamic allocator may mean that at any time the current resource requirements are beyond the system's ability to honour them (perhaps beyond the ability to honour these commitments continuously or on a statistical basis — the precise requirements may themselves be part of the SLS agreement). Most likely, over-commitment events would need to be analysed statistically and if the occurrence of these events is greater than a given threshold (e.g. any violation in a rigorous configuration or a number of violations in a configuration that is tolerable of such events) it would alert the network provider over potentially chronic resource requirements.

6.3.2 Allocation mechanics

The scheduler implements a guaranteed fair-service queueing algorithm so the scheduler service delay constraints are bounded and, being a work conserving scheduler, unused link capacity will not be wasted as they might in the non-work-conserving WRR scheduler. The WF^{2}Q+ supplies

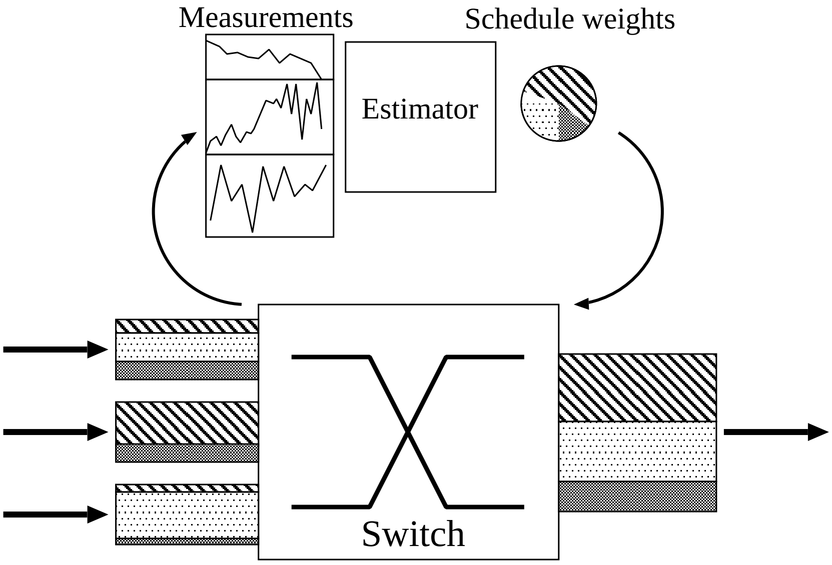


Figure 6.4: Overview of automated weight setting

a weighted service for queued traffic with weights corresponding to the amount of service (link bandwidth) each aggregate-flow of traffic may use. The facilities of this scheduling algorithm mean that, for the implementation, no regard need be given to the potential delay of large weight values. Additionally, because the scheduler is work conserving, there is no wasted resource: the scheduler will divide any unused resource among waiting queues. The only requirement of the system is to install suitable weights to allow the scheduler to allocate the link bandwidth in the first instance.

Figure 6.4 shows an overview of the implementation. Traffic flowing through the switch is measured as inputs for an MBE. Using allocation-policy nominated control parameters (e.g. target loss-ratio, delay-bounds), the MBE computes resource requirements for each set of traffic. The available resource is then divided up using a weighted value derived from these estimates and each appropriate weighted value is then installed into the switch. This process is continuously repeated, updating the weights of the WF^{2}Q+ values dynamically, when the traffic characteristics change.

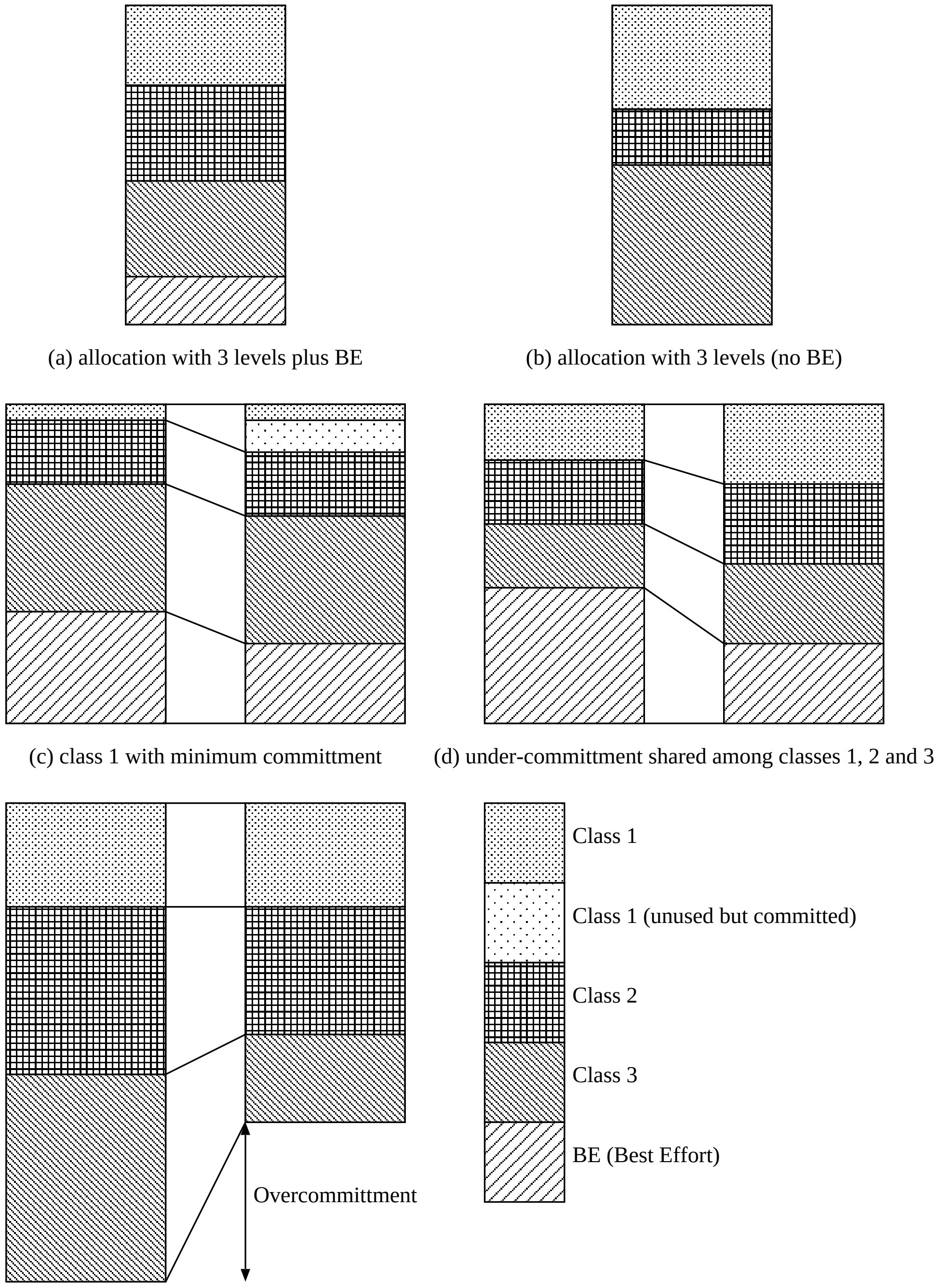
Two assumptions have been at the base of the implementation presented here: firstly, for traffic with a delay-bound, packets that fall outside the delay-bound have no value and may be discarded, and secondly, traffic with an assured throughput or a loss-boundary have either a nominated delay-bound or minimal delay-bounds. Such simplifications mean that in parallel with the adjusting of weights for the scheduler, a buffer management scheme is used allowing some control of buffer requirements for traffic with guaranteed loss or guaranteed throughput.

This presumes that the delay-bound resources receive guaranteed access to the appropriate buffer size (as limited by the delay bound) and are served at a rate (set by the appropriate weight in the scheduler) to ensure that the delay criteria can be met. Such a class of traffic would not have access to any additional resources as packets delayed by longer than the maximum time spent in the buffer would be outside the delay-boundary.

For the classes of traffic that require assured throughput, the buffer can be sized to ensure that the minimum loss guarantee can be met with the guaranteed service weight. In a similar fashion, flows with a loss-guarantee combined with the delay-bound (translating to a buffer size boundary) will require that they are served at the appropriate service rate. All that remains is the process whereby over or under-commitment is reconciled.

Figure 6.5 illustrates examples allocating link capacity. The situation is assumed to consist of three classes of traffic for which guarantee is made (**1**, **2** and **3**) and a fourth class of traffic that has access to resource only when it is unused by the other classes, the best effort (**BE**) in the diagram. Figure 6.5(a) illustrates an example where allocations have been made to the three classes and the left-over link capacity is made available to the 4th, BE class. In contrast, Figure 6.5(b) illustrates an example where allocations have been made to the three classes but all the resources have been required and as a result no link capacity is made available to the 4th, BE class.

In Figure 6.5(c), the allocations are computed but Class 1 has a minimum commitment and this requirement is served (in this example) from the left-over allocation that would have been made available to the BE class. While such a minimum commitment may be seen to potentially waste resource, such a situation may arise if Class 1 has a delay-bound: a minimum service rate may be needed to satisfy the delay constraints. A work-conserving scheduler ensures that while Class 1 receives its service guarantee, unused link capacity will not be wasted but will be shared among queues with packets still remaining, such as the 4th class. In this way the minimum allocation ensures the guarantee while the scheduler implementation ensures a minimum of resource is wasted.



(a) allocation with 3 levels plus BE

(b) allocation with 3 levels (no BE)

(c) class 1 with minimum committment

(d) under-committment shared among classes 1, 2 and 3

(e) overcommittment shared among class 2 and 3

Figure 6.5: Example (re)allocation of link capacity

For the example shown in Figure 6.5(d), resource requirements are computed with left-over resource shared among the traffic of classes 1, 2, and 3 as well as the BE traffic class. In an example where each class has throughput guarantees, the policy of the implementation is to share part of the uncommitted resource among those classes with throughput guarantees. The precise control, along with all other aspects of the allocation, is controlled by the current policy. Resource reallocations may be a system-wide default (e.g. equally-proportioned, or proportioned-relative to an income-scale) or divided-relative to the current requirements of each class.

Finally, Figure 6.5(e) illustrates how a policy will allow the reallocation of resource in the event of over-commitment. In this example the computed requirements for Class 1, 2 and 3 exceed the available link-capacity. As a result the amount of resource over-commitment will need to be shared among the traffic classes. Policy, either on the basis of class-by-class or system-wide configurations, can describe the mechanism used to share this over-commitment. In the figure, Class 1 does not relinquish any of its resource requirements, while Classes 2 and 3 share the withdrawal of resources from their original allocations.

The withdrawal of over-committed resources may, like the sharing of uncommited resources, be based upon proportion relative to the income of each class or the current resource requirements of each class. The precise approach for under or over-allocation procedure does not need to be permanently enshrined within the implementation; the use of a set of policy instructions laying out per-class and system-wide procedure allows flexibility in implementation.

6.4 Method

In this section the details of experiments, the results of which are presented in Section 6.5, are discussed. Using the test environment modules outlined in Chapter 3, the system of Figure 6.6 is used for the evaluation of the MBE-based dynamic allocation scheme.

Most notably, the dynamic allocator has no control over the flow-generation process. Figure 6.6 illustrates that the dynamic allocator MBE, using measurements of current utilisation, regularly recomputes and updates the configuration of the switch installing the latest configuration for scheduler weights and buffer limits.

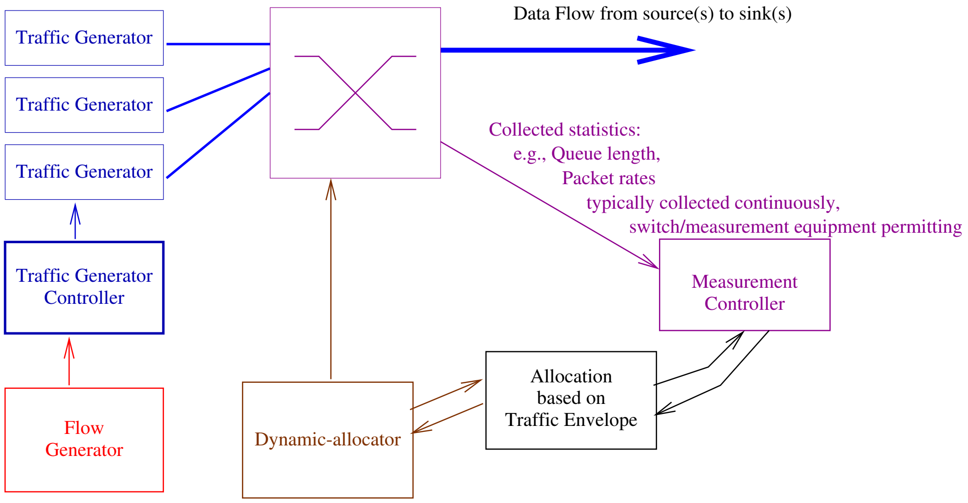
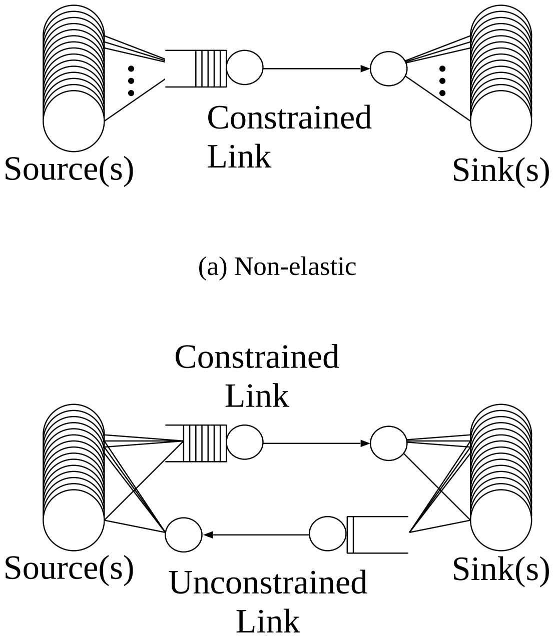

Traffic Generator
Traffic Generator
Traffic Generator
Traffic Generator Controller
Flow Generator
Data Flow from source(s) to sink(s)
Collected statistics:
e.g., Queue length,
Packet rates
typically collected continuously,
switch/measurement equipment permitting
Measurement Controller
Allocation based on Traffic Envelope
Dynamic-allocator

Figure 6.6: Test environment

In this framework, it is possible to start flows of traffic consisting of any of the traffic types including model sources, video stream sources, pre-recorded traffic flows and actual elastic traffic such as TCP/IP. Such flows are started and stopped without any direct interaction with the dynamic allocator. Such a circumstance is a duplicate of the environment in which this system may be deployed.

Two configurations are used to illustrate the behaviour of the MBE-based dynamic allocator. These include, firstly, one based upon Olympic differentiated service using three classes each receiving a proportion of the available link capacity and, secondly, one based upon absolute differentiated service where three different classes (one delay-bound, one loss-bound and one best-effort) share available resource. The precise configuration of the policy is given with each set of results. The network is similar to that of Section 5.2.2, a dumbbell configuration with a single constriction point at the switch. The link capacity is configured for 100 Mbps.

The network configuration is illustrated in Figure 6.7, as noted below experiments using non-elastic traffic are conducted on the configuration of Figure 6.7(a), while those using elastic sources are conducted on the configuration of Figure 6.7(b). Figure 6.7(b) illustrates how the configuration reticulates feedback for elastic sources; in the dynamic allocation example each class has access to its own buffer area in the constrained direction — in the unconstrained direction shared buffering is not an issue.



(a) Non-elastic

(b) Elastic

Figure 6.7: Dumb-bell network experiment configurations

6.5 Results

The results of this section illustrate that the dynamic allocator is able to provide a differentiated service with a range of guarantees without the need for fixed allocation policy. This system is able to use the resources of link-capacity and buffer-space to provide a service to all competing quality assurances with reduced resource waste. Importantly, this system performs better than best-effort by supplying differentiation, better than fixed resource policy by adapting to changing requirements, and (in adapting to changing demands) dynamic allocation does not waste resources in the manner that fixed resourcing policy does.

Section 6.5.1 reports results illustrating the operation of resource allocation policy. Section 6.5.2 presents results for a set of experiments providing quantitative assessment of the performance of the dynamic allocation mechanism.

6.5.1 Policy Operation

In this section, figures illustrating the operation of the dynamic allocator are shown along with the operation of various of the policy resolution examples discussed in Section 6.3.1. Using TP10S1 traffic, an example of the dynamic allocator in operation is given in Figure 6.8. The bottom graph shows the current resource demand of the three provisioned services. The top graph shows the allocation of the scheduler to each traffic class. At each time the current configuration of the scheduler is illustrated — the subdivisions represent the separation of scheduling resource among the four competing traffic classes.

In Figure 6.8 it is clear that at 200 seconds, an increase in the requirements for the Gold service have (virtually) eliminated any left-over resource. At the 300 second mark, the resource requirements of the Silver class have increased resulting in the Bronze service being penalised. Following a restoration in requirements of both Gold and Silver services to their former levels, resource is automatically made available to the Bronze service and remainder is available for a fourth service. It is quite apparent that any commitments made to the Bronze service were not sustained between 300 and 400 seconds although such drop-outs in service may be part of the SLS agreed to by the parties. The alternative for the provider is, apart from the provisioning of a greater amount of resource over-all, to implement a restriction on the impact each service may

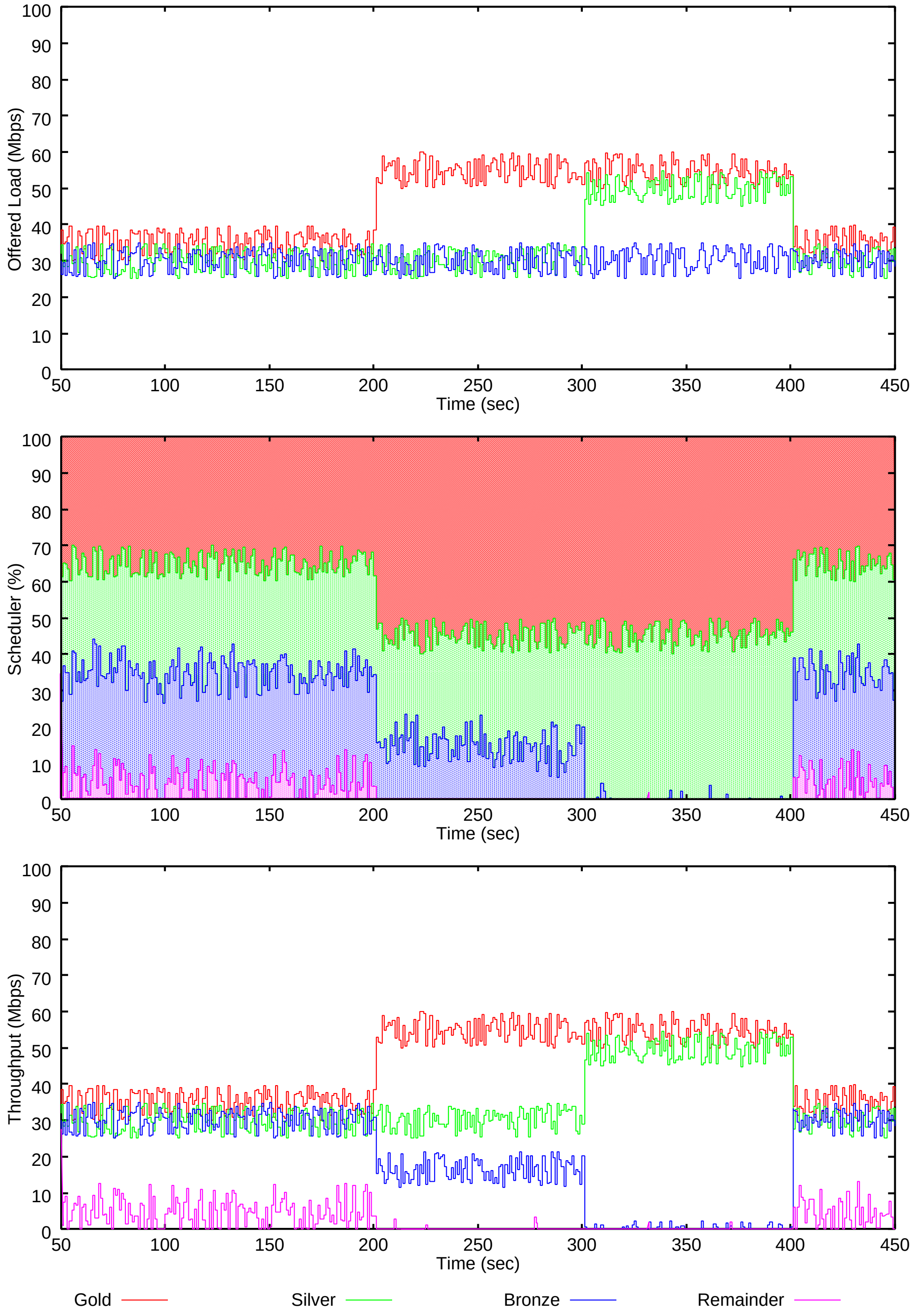


Figure 6.8: Olympic service

have upon another. Because the allocation system is programmable, the process may incorporate any procedure the policy dictates. It is easy to place an upper limit on the allocation to any particular resource class and it is possible to change from a prioritised allocation as indicated here to an allocation that follows the proportional schemes of Figure 6.5(d) and (e) as appropriate.

As is illustrated by this example the scheme is able to operate as required. The next section details the performance gained for experiments run over longer periods of time. These results are compared with the performance gained using non-dynamic allocator such as best-effort and fixed-allocation resourcing.

6.5.2 Performance

In this section an example is given of the performance achieved by the sample implementation. The dynamic allocator is configured with three classes of traffic, one each of delay-bound and loss-bound along with a class intended to use the remaining, left-over capacity. The dynamic allocator was configured to reassess the current allocation every 100 ms. The configuration of the estimator had measurements made every 1.3 ms, with E-KQ configured so that the measurements covered a period sufficiently large to sample beyond the reallocation period,[3] therefore providing maximum protection from traffic fluctuations between consecutive allocations.

Table 6.1 indicates the set of traffic parameters for the three classes of traffic used alongside the parameters of each traffic class. A combination of a low-delay voice aggregate with a high-demand video aggregate consume the majority of the available capacity. The voice traffic operates as a continual arrival and departure process affording the test traffic the full dynamics of a multiplex of voice data. The video data consists of four permanent streams of VP25S4 (Section 2.2.5). Starting at random (uncorrelated) locations in the video-stream this multiplex of four streams of traffic provides the characteristics of high-capacity, high-throughput users combined with the statistical effects evident both in individual traffic streams and evident in a multiplex of strongly structured data. WP10S1, traffic representing WWW transactions, is transmitted as the 3rd class. This class is elastic using the remaining, unused capacity and as a result is affected by the ongoing availability of capacity. For this traffic type, performance of the available capacity

[3] For the E-KQ of this algorithm, $\tau = 1.3$ ms, $T = 200$, and $M = 4$.

can be measured by the rate at which bytes of data are able to be transferred between the elastic-generator's server and client; this performance figure is given as the *goodput* in the results of Table 6.2.

Traffic	Flow Parameters	Policy
VP64S23	300 seconds mean hold time, log-normal distribution, 2 flows per second mean, exponentially distributed	Delay Sensitive: 1×10^{-5} ratio for packets-delayed by $>$ 500 μs (1st priority)
VP25S4	4 flows continuous	Loss Sensitive Target loss ratio 1×10^{-4} (2nd priority)
WP10S1	10 flows continuous	Remainder

Table 6.1: Parameters for traffic and policies of long duration dynamic allocator experiments

Aside from the goodput, Table 6.2 presents the achieved loss ratio for the video stream traffic and the ratio of voice packets delayed beyond the nominated delay boundary. This table presents results gained using the approaches of best-effort, fixed allocation, and the dynamic allocator. The best-effort results were for a system that offered no service-differentiation between the three different classes. Clearly the WWW traffic WP10S1 gained excellent goodput at the expense of the loss and delay of the video and audio traffic.

The fixed service allocation results gave the voice (in this example the highest priority) all the bandwidth required. The immediate result for the video traffic was that there was not sufficient bandwidth for an allocation that would satisfy the requirements outlined in Table 6.1. Instead the results show that the packet loss was substantially higher than the desired target of 1×10^{-4}. Finally, with allocations made for voice and video, there was no bandwidth available for the WWW traffic source and as a result no goodput either.

Finally, the the dynamic allocator results indicate that it was able to honour the policy agreements used in this system. However, the delay/loss ratios for both the VP64S23 and VP25S4 are indicative long-term results and results on a smaller timescale may indicate lower performance. Additionally, for the dynamic allocator results, the error margin on the packet-delay figures as they were gathered is still quite high at $\pm 12\%$ with a 95% confidence interval for

Traffic	Results		
	Mean Utilisation	Mean Allocation	Performance
Best Effort (no service differentiation)			
VP64S23	13.8 Mbps	—	9.4×10^{-4} packets-delayed
VP25S4	17.6 Mbps	—	2.7×10^{-3} packets-lost
WP10S1	10. Mbps	—	4.4 Mbps goodput
Fixed Service Allocation			
VP64S23	13.7 Mbps	39.3 Mbps	0 packets-delayed
VP25S4	17.7 Mbps	60.7 Mbps	6.5×10^{-4} packets-lost
WP10S1	0 Mbps	0 Mbps	0 kbps goodput
Dynamic Allocator			
VP64S23	13.8 Mbps	27.6 Mbps	2.2×10^{-5} packets-delayed
VP25S4	17.6 Mbps	71.2 Mbps	1.7×10^{-5} packets-lost
WP10S1	0.8 Mbps	1.2 Mbps	314 kbps goodput

Table 6.2: Results of long duration dynamic allocator experiments

1×10^{-5} and $\pm 5\%$ with a 95% confidence interval for the packets-loss figure.[4] However, taking into account the precision of the results gained, it may be contended that the dynamic allocator prototype worked successfully.

From these results another important aspect of the dynamic allocator approach is made clear. The dynamic allocator is able to provide a service that is better than the fixed allocation approach, providing both voice and video data with the desired conditions of loss and delay. Additionally, by using the dynamic allocator in place of a fixed allocation, provision was available for a third, best-effort service that used the left over bandwidth, a service not provided for at all in the fixed allocation approach.

[4] In accordance with the conclusions of Section 3.5.2, experiments for these results were run for a sufficient time to reduce the error due to sampling to less than $\pm 1\%$ with a 95% confidence interval. The majority of the error is assumed to be systematic in nature, following Section 3.5.4.

6.6 Discussion

Several matters that arise from the use of the dynamic allocator are noted here. Section 6.6.1 discusses elastic-traffic, traffic that is considered not to respond as well to an MBE based upon utilisation measurement. Section 6.6.2 reports on issues of the timing of resource updates and potential unwanted interaction between the MBE-based mechanism and traffic for which resource is being controlled.

6.6.1 Elastic Traffic

Elastic traffic has the potential to adapt to available resources. One result is such traffic will attempt to use as much resource as is available but it may not be compatible with the dynamic allocation mechanism. From the perspective of a dynamic allocator attempting to provide for a gold or silver service carrying TCP, undesirable effects may occur because TCP depends upon packet loss for feedback about congestion. The TCP protocol will cause congestion at bottlenecks in the network (such as any node using the dynamic allocator) with the result that for a mechanism computing a desired allocation based upon a loss-boundary, the MBE of the dynamic allocator would continue to increase the allocation made available to the traffic class carrying TCP. Any active TCP flows would then increase its utilisation to match the (newly) available allocation. This process would continue until all available resources for (re)allocation to the TCP class were consumed, with the result that the TCP class would still have a high level of loss. Clearly this is an example where loss results not from a poor MBE algorithm but from an inappropriate use.

An alternative approach to the problem would be to make a fixed allocation of resource to the class of traffic carrying TCP flows. This idea is both simple and practical. Additionally, any unused resource can be reallocated for use by the TCP flows as and when it's required. An alternative suggestion, particularly when it is desirable to attempt differentiated services on a per-flow basis, would be to allocate resources to the class carrying TCP flows based upon a basic rate per flow multiplied by the number of flows.

Assuming fair TCP behaviour, such an approach would allow TCP flows to share the available resource among themselves as well as providing an inherent mechanism for concurrent flows in the traffic class to consume unused bandwidth of that class. However, such an approach has two drawbacks.

Before covering these two drawbacks it is worth noting that fair behaviour of TCP cannot always be assumed. TCP flows are documented as exhibiting a phenomenon called source synchronisation — a situation where the efficiency of an aggregate of flows is reduced due to a combination of synchronised loss events and timer values common to many implementations. While proposals have been made to attend this problem (e.g. RED proposed in [Floyd93]) such mechanisms are not universally available or appropriate. The source synchronisation phenomena, which may cause inefficiencies between TCP flows in an aggregate, is not something the dynamic allocators mechanisms can attend to because the mechanism works at the level of class-aggregate rather than the individual (TCP) flow.

The first drawback of the static allocation approach towards TCP would be that changes in the number of flows would not be reflected in changes in the allocation. A solution may be to use the number of flows present to compute an appropriate allocation. A simple mechanism, such as that proposed in [Morris00], caches TCP header information on source and destination to compute an approximation of the number of active flows.

Measurements of utilisation of a limited number of TCP/IP flows will produce informative results on the activity of those sessions, such as the activity cycle a single web browser or an end-user's machine goes through. However, measurements of utilisation is of limited value when the flow-aggregate being measured is made up of a number of TCP/IP flows: an aggregate measurement is unable to discern if ten TCP/IP flows are consuming a single shared resource or if one flow is consuming ten times the resource. This is a result of the TCP/IP mechanism that uses feedback to regulate the utilisation of each flow. In the dynamic allocator, instead of measuring the utilisation (alone) it may be more informative to also measure the number of flows present in an aggregate. The TCP/IP flow regulation mechanism is unable to differentiate between packets being delayed due to communications delay and packets being delayed due to potentially unnecessary buffering. As a result it is desirable, by knowing the number of flows present in an aggregate, to compute an appropriate (minimum) level of buffering that will preserve the flow throughput while minimising unnecessary delay.

If it is assumed that a flow-aggregate consists of many simultaneously-active flows: the dynamic allocator presented here has the capacity to place a delay-bound on buffers used by any particular flow-aggregate — knowing an appropriate delay value enables an appropriate buffer sizing.

If the dynamic allocator had access to additional TCP flow information such as the Round-Trip-Time, the models of [Padhye98] or [Cardwell00] may be used to compute a more appropriate value of link-capacity and buffer, however such a scheme would require significant information be extracted from the TCP flows themselves.

Finally, it is important to note that TCP does not respond well to leaky-bucket regulation [Sahu99] because the achieved rate is not proportional to the assured rate. Additionally, in some circumstances, the token bucket parameters have no effect on the assured rate or the target rate is not possible to achieve. Rudimentary experiments with an aggregate of TCP flows using a fixed allocation of link capacity show that, while a strict formula can not be derived, TCP is sufficiently responsive to changes in the available resource allocation that the dynamic allocator would work sufficiently well. However, the work in [Sahu99] reinforces the difficulties of constructing a model upon which TCP goodput could be computed from TCP throughput.

TCP in particular and adaptive traffic sources in general present an interesting challenge to the ability to apply dynamic allocator techniques. However, at several levels — from a strict partitioning through to a system incorporating flow-information — such traffic is able to be incorporated into the dynamic allocator approach.

6.6.2 Timescales

The dynamic allocator incorporates a critical timescale: the time between reallocation events. There is a potential for elastic traffic sources to interact with this time-frame but by identifying TCP flows as the most common elastic source, and identifying the need of special treatment of these flows, the issues of timescale would not present the problem that might be assumed at first sight.

A link becomes clear between the characterisation of traffic and the reallocation period. The critical timescale of the dynamic allocator (the time between reallocation events) is also the critical timescale for traffic characterisation. While not a difficult conclusion, it is nonetheless important to note that the measurement-based estimator must characterise traffic over timescales that include the time between reallocation events. In the dynamic allocator proposed here, this is easily done by appropriate configuration of the MBE parameters and using an MBE that characterises traffic at timescales both greater and less than the reallocation period.

6.7 Conclusion

In this chapter a dynamic allocator was presented that offered differentiated service support to traffic aggregates sharing a common switch and transmission route. Section 6.1 notes how the approach presented here, sharing concepts in common with the idea of contract-renegotiation during a flow lifetime, adapts the current allocation of resources: buffer size limits and scheduler service allocations through the use of MBE of resource requirements. Section 6.1.1 notes that while ideas of resource allocation and control through admission-control have seen little impact on popular networks such as the Internet, the idea of offering differentiated services to users sharing common resource is still prevalent. The Internet DIFFSERV system is an architectural proposal for providing QoS guarantees to aggregate traffic flows. In parallel with DIFFSERV, VPN proposals are intended to offer users network resources with particular characteristics (e.g. throughput, loss or delay) available on demand. The dynamic allocator presented here may be considered as a proposed switch implementation of such architectural proposals. Section 6.1.2 notes that the objective of the dynamic allocator was to provision guaranteed QoS to services rather than a simple differentiated service.

Section 6.2 presented the background issues, including the decisions affecting the choice of scheduler in Section 6.2.1. Section 6.2.2 discusses the complications arising in active buffer management and the need for buffer management to ensure queueing-delay constraints. Then, the selection of MBE algorithm is outlined in Section 6.2.3

The implementation of the dynamic allocator is discussed in Section 6.3 with the flexibility of a programmable policy discussed in Section 6.3.1. The mechanics of the implementation were covered in Section 6.3.2.

The environment in which the implementation was made was discussed in Section 6.4. This implementation, superficially similar to that used for the previous chapter, was used in the generation of the test results presented in Section 6.5. The process by which the dynamic allocator transfers resource from one traffic-class to another is illustrated in Section 6.5.1, and Section 6.5.2 presents a set of performance figures for an experiment conducted over a longer lifetime. The results are encouraging, illustrating that the dynamic allocator provides a practical approach.

Section 6.6 gives a brief discussion of both elastic traffic sources (Section 6.6.1) and specific timescale issues (Section 6.6.2) as these things relate to the dynamic allocator.

6.7.1 Future work

Problems still remain with over-allocation due to delay constraints, however without the support of a more complex scheduling algorithm [Sariowan95, Stoica97] the approach taken — to allow over allocation but enforce some sanity checking on the allocation value — gave acceptable results. Future work also remains in the policy implementation, the next natural course of work being the use of a formal policy description language. Finally, the elastic nature of TCP flows still requires further work, although Section 6.6.1 outlines an approach whereby these flows can be accommodated in the dynamic allocator as presented.

Chapter 7

Conclusion

This dissertation has presented a unique evaluation of MBAC algorithms using a purpose built test environment. It further presented a measurement-based approach supporting differentiated services in a network switch. This chapter summarises the work, suggests areas for further study and offers concluding remarks drawn from the work presented in this dissertation.

7.1 Summary

Chapter 1 identified the importance of the Measurement-Based Management of network resources. Measurements provide input to allow management to both characterise unknown traffic and to adapt to changes in resource requirements while continuing to provide QoS commitments. Embracing Measurement-Based Management in Admission Control (AC), MBAC algorithms have allowed minimal characterisation of new traffic flows, and adaptation to changing flow requirements. However, there have been many MBAC algorithms proposals, often with no clear differentiation between them. This has motivated the need for a realistic, implementation based comparison in order to identify an ideal MBAC algorithm. Further, identifying that the Measurement-Based Estimators as used in MBAC algorithms need not be tied to the AC problem alone, one of these Measurement-Based Estimators is used in an implementation of a network switch that provides differentiated service while adapting to each traffic class.

Chapter 2 provided fundamental information on the context of the two approaches of this dissertation. The evolution of network traffic and its characterisation is discussed. This gives insight into the problem that has developed in the provisioning of network resources. A wide variety of network traffic sources, used throughout the experiments of this dissertation, were detailed. Background was provided for the management techniques of AC, packet scheduling, and active buffer management, as these network control methods were augmented using Measurement-Based Management techniques described in this dissertation. The nature and the use of measurements as input to a management and control process were also covered in this chapter.

An evaluation environment was described in Chapter 3. This environment allowed the unique approach of the implementation-based comparison of MBAC algorithms to be conducted under controlled, repeatable conditions. Aside from a description of the structure and components of the environment, an evaluation of the error margins and limits of the environment were also reported. The environment provided an approach to the evaluation of MBAC algorithms that is able to contrast repeated runs of the same MBAC algorithm under varying conditions as well as allowing comparison of different MBAC algorithms under identical conditions. An implementation based evaluation allowed comparisons to be conducted under more realistic conditions than those offered by simulation or an evaluation based upon link/source model solutions, this included real traffic loads, allowing insight into the operational constraints and requirements of each MBAC algorithm.

Ten MBAC algorithms along with five other AC algorithms were presented in Chapter 4. Each AC algorithm was decomposed into constituent policy and estimator components. This approach allowed common and unique elements of each algorithm and policy to be identified and highlighted the existence of only five distinct policies shared among the AC algorithms discussed. The different requirements of Measurement-Based Estimators were illustrated with a table of estimated measurement, computation and memory requirements. Further comparisons highlighted those MBAC algorithms whose design makes allowance for the random nature of measurements.

Chapter 5 presented results for an implementation-based comparison of MBAC algorithms. Comparison results were reported for the traditional criterion of system loss versus system load. These supported previous studies that suggested such results are a function of traffic and network resources and are independent of the AC algorithm. The loss-acceptance function was shown to be useful to characterise AC policy illustrating behaviour for heterogeneous flow arrivals. A unique insight into the relationship between control parameters and each system was also reported. It was concluded that while such control parameters are important, they rarely provide an AC algorithm user with any calibration of the control. The use by architects of MBAC algorithms of poorly defined parameter values was also noted.

This chapter went further into a stability study of AC components and algorithms although the stability was illustrated to be a function of traffic load and measurement period rather than AC algorithm. Repeatability studies of the outcome, through repeated execution of the same experiment, were inconclusive but indicated great importance in the admission policy employed by each algorithm. Finally, this chapter gave a unique insight into the impact implementation may have upon an AC algorithm as well as documenting the relative complexity and computational overheads of each MBAC algorithm. A discussion in Chapter 5 noted the bias in experiments for or against a particular MBAC algorithm as well as noting that no clear *ideal* MBAC algorithm existed. However, it was clear that the task demanded of the AC approach has meant that any algorithm may only occupy a small part in a total solution space that includes: ease of operation, simplicity, negligible overheads of computation or memory, while still able to maximise utilisation and maintain previously committed QoS.

Using an appropriate Measurement-Based Estimator tested as part of the previous chapter, Chapter 6 illustrated how this estimator combined with a sophisticated scheduling algorithm and rudimentary buffer control can implement a network switch able to offer differentiated network services. The Measurement-Based Estimator is able to provide estimates of resource (buffer and bandwidth) requirements for each current traffic class. Two experiments were used to illustrate the dynamic allocator in action. Firstly, support for an Olympic service, distributing link-capacity based upon class category, and secondly, support for the demands of three services with the different requirements of minimum delay, minimum loss and a best effort service using the remaining network resource. Such a dynamic allocator is an important approach towards offering support for differentiated services in a network and the approach here illustrates that such a mechanism can adapt to changes in resource requirements while offering useful service support.

The work presented in Chapters 3 though 5 successfully addressed the first aim of the dissertation identified in Chapter 1, namely the evaluation-based comparison of a sample of MBAC algorithms. To this end the evaluation has been conducted with a wide variety of traffic under the conditions of a real network. By being implementation-based, each MBAC algorithm was subject to the same constraints of memory and computation limits as well as realistic access to measurements. The resulting comparison highlighted the importance of the timescales of a network to the behaviour of the AC and indicated that many MBAC algorithms rely upon timescale separation that may not exist in modern network traffic. Using actual implementation has allowed insight into the impact performance has upon behaviour — algorithms that make unrealistic assumptions about measurement, memory or computing constraints give a deteriorated performance. It is common for this performance edge-case to be defined by the admission policy in use although this assumption can not always be relied upon.

MBAC algorithms are not, however, seen as the only approach to the management of network resource, and more recently differentiated services based upon the DIFFSERV model have risen in popularity. The second aim raised in Chapter 1 was addressed in Chapter 6 which adapted the MBAC technology to an estimator suitable for computing the dynamic resource requirements of orthogonal service demands. This estimator is able to provide a prediction of resource requirements of either buffer space or bandwidth.

7.2 Future Work

The comparison reported in Chapters 4 and 5 as well as the implementation of Chapter 6 give significant scope for future work. Alongside such work this section considers other approaches of Measurement-Based Management arising from this dissertation that are considered relevant.

Ideally, any future comparison work will broaden and update the limited number of MBAC algorithms studied. The comparison criteria and environment used in this dissertation have attempted to encapsulate a current (and predicted) *snap-shot* of network conditions and of the expectations of AC algorithms. However, in the period over which this dissertation was conducted, network evolution has seen the AC approach become less appealing, being replaced by the service differentiation ideas addressed by the implementation of Chapter 6.

Any successful AC algorithm must account for the variation in timescales present in traffic and in the multiple timescales present in control over which such management techniques must work. Thus the AC-KQ and AC-GT algorithms, both incorporating measurements made at many timescales, had significant advantage. Future work for AC algorithms may concentrate on enhancing the AC-GT to incorporate handling of buffers, perhaps through application of large deviation theory. An alternative is that of AC-KQ, while requiring some parameter configuration, offers another approach, providing a promising potential in creating a hybrid of the two techniques of AC-KQ and AC-GT. This would give an MBAC algorithm robust across many timescales and appropriate for networks with multiplexor buffering.

The Measurement-Based Management scheme of Chapter 6 is of necessity a prototype system. While sufficient to prove the power and potential of this system, a number of open issues remains before such a dynamic allocation approach can be engineered for deployment into a production network.

Due to the prevalence of TCP/IP network traffic, support for this traffic using a method of flow-detection is of greatest immediate interest. This would allow support through the dynamic allocator of fine-grained differentiation between TCP traffic classes, thereby affording the direct support to TCP/IP flows already afforded to other non-elastic traffic types. The Measurement-Based Estimator used in the current dynamic allocator may also benefit from the AC revisions proposed above.

A number of further research topics have also been raised as a result of this dissertation. These topics are considered now.

Another application of MBE is the area of QoS routing. In [Isaacs00] it was noted that while QoS routing is commonly acknowledged as important, few authors have addressed the problem of how to perform it directly. QoS routing is complicated, not only by the (potentially) diverse range of requirements of each flow, but also the importance of providing a tractable, timely solution to the routing problem. Noting the difficulty introduced by using dynamic computation of traffic requirements, MBE may allow QoS routing to be performed more simply.

A routing strategy presented in [Kodialam00] places paths on the basis of minimum interference with future routing requests. [Kodialam00] note the difficulty such a system may incur with differing, perhaps orthogonal requirements of flows such as bandwidth and delay — it is in this area that an appropriate MBE may make a significant contribution. The approach of the dynamic allocator of Chapter 6 may offer a suitable technique providing the information required by this routing algorithm. Indeed, should the routing algorithm operate upon a network of nodes of the type of Chapter 6, the requirements of each flow is available to the routing algorithm in a form (a buffer-size and bandwidth minimum) computationally compatible with available network resources.

Redundant routing could be provided by an approach that computes not only the primary paths but redundant paths using the same estimates of flow-requirement and the minimum interference routing algorithm. Alternative routes may be computed by the routing agent using a modified network. Network paths may be considered unavailable and alternative routing configurations may be precomputed on this basis. Using regularly updated flow-requirements, based upon the MBE values, flows may be migrated in order to maintain a balance between minimum interference and a maximum redundancy. In order to ensure computational tractability, network sub-netting may be required to limit the number of flows, paths, and nodes for which the routing computations need be performed.

Finally, for management at long timescales, network planning has made some use of MBE, in the form of traffic-traces providing characteristics of traffic flows (e.g. [Paxson95, Crovella97, Feldmann98b]). [Gibbens00] showed how combinations of information derived from measurements: flow arrival rates and flow holding times, can be combined with traffic models to perform a fixed-point analysis of a network. In the face of missing information and points at which measurement is not possible, road-traffic engineers have developed techniques to estimate this missing information. Most notable is that utilisation and trajectory information has commonly been derived for the road-traffic fraternity using estimation techniques such as maximum entropy methods [Ortúzar94]. Such techniques, using partial measurements but supplying an estimate of a complete (road) network's operation, have the potential to provide network planners with information about the utilisation of paths by estimating the origin-destination matrix of the nodes of interest. These techniques have a great potential contribution to make to network management.

In addition to assisting the planning process, such estimates of origin-destination and hence trajectory in the network forms a complement to the work reported in [Duffield00], computing the trajectory of data in a network for tasks such as fault location. Unlike [Duffield00], the maximum-entropy approach uses the minimum of information to estimate flow trajectory. As was noted in [Tutschku98], such spatial trajectory information can be of critical importance in planning and characterising mobile communications. Thus, measurement based estimation of this nature has great potential in the field of network planning.

Chapter 6 illustrated how an MBE may be used as input to enable a network node to offer differentiated service with a minimum of wasted resource. Chapters 4 and 5 illustrated part of the broad range of work on MBAC algorithms, using MBE to both characterise new flows and existing flow requirements. In both these approaches MBE has been shown as a powerful tool for the characterisation of network traffic, able to characterise flows recognised as incompatible with current traffic models as well as able to adapt to changes in traffic. These abilities of MBE will ensure the continued incorporation of this powerful tool into network management over a wide range of timescales.

7.3 Conclusions

A key property of measurement-based management is the use of estimators that rely on little or no *a priori* traffic characterisation. Practical measurement-based estimators can be implemented within the current bounds on memory, processor and measurement. An effective measurement-based estimator must characterise the sources across all timescales of relevance to the management scheme — this means an estimator will commonly require both a significant history of previous events and incorporate a statistical prediction of future events.

Measurement-based estimation is both an adjunct to the modelling of traffic and a replacement for it. Measurement-based estimation is able to provide a continuously up-to-date estimate of resource requirements. In doing so, measurement-based management can perform tasks not possible with resource allocation based upon static *a priori* characterisation — as was shown by the dynamic resource allocator of Chapter 6 in this dissertation.

Summarising the finding of Chapter 5, conclusions based upon the studies of this dissertation for the selection of an appropriate MBAC algorithm are that:

- To be useful, an MBAC algorithm must offer some relationship between calibrated controls and the traffic behaviour, only a small number of the tested MBAC algorithms were able to provide this.
- An MBE must incorporate allowance for the statistical nature of measurements.
- The MBE of an MBAC algorithm must be matched to the task required. Common MBE offer high-precision at the expense of overheads such as computational speed, memory requirements, or measurement requirements.
- Further to this, an MBAC algorithm, both policy and estimator, must be implementable with realistic demands on memory and processing along with realistic demands of the measurements required as input.
- Finally, for the particular case of MBAC algorithms, the policy performed by the admission algorithm plays a critical role in the behaviour and performance of the algorithm overall. The policy of an MBAC algorithm must not be neglected as its behaviour often leads to the difference in performance between MBAC algorithms.

The implementation described in Chapter 6, allows a set of conclusions for measurement-based allocation of differentiated services to be formulated:

- Like the estimators of the MBAC algorithms, the MBE of the dynamic allocator must offer some relationship between calibrated controls and the traffic behaviour to be of use.
- The inherent reaction between closed-loop (elastic) traffic such as TCP/IP flows and measurement-based estimators make this traffic unsuitable for control through any sort of dynamic allocation system. However, a solution lies in the guaranteeing of partial bandwidth to such elastic flows based upon an externally agreed hard committment.
- The dynamic allocator works with a primitive configuration of the MBE. This implies that a faster, less accurate, MBE is able to provide a workable solution and also implies that

for overall system performance, the ability to correctly maintain current provision values is more important than the ability to accurately characterize short term traffic events.

Further work with dynamic allocators on the relationship between the size of the system and the timescale of the allocation would better address this final point, although the implementation reported in Chapter 6 illustrates a working system.

It is the thesis of this dissertation that measurement-based estimation provides improved performance when realistic implementations are incorporated into current management schemes and that measurement-based estimation makes possible new, adaptive, management schemes that are able to operate independently of traffic type.

Bibliography

[ACM99] ACM SIGCOMM. *The Internet Traffic Archive*. Technical Report, June 1999. `http://www.acm.org/sigcomm/ITA`. (p 46)

[Allman99] M. Allman and A. Falk. *On the Effective Evaluation of TCP*. ACM Computer Communication Review, 29(5):59–70, October 1999. (p 82)

[Anick82] D. Anick, D. Mitra, and M. M. Sondhi. *Stochastic theory of a data-handling system with multiple sources*. Bell System Technical Journal, 61(8):1871–1894, October 1982. (p 170)

[Arpaci00] M. Arpaci and J. A. Copeland. *Buffer Management for Shared-Memory ATM Switches*. IEEE Communications Surveys & Tutorials, 3(1), First Quarter 2000. (p 55)

[Arvidsson99] Å. Arvidsson and P. Karlsson. *On Traffic Models for TCP/IP*. In Proceedings of 16th International Teletraffic Congress (ITC16), Edinburgh, UK, June 1999. (pp 37, 38)

[ATMF95] ATM Forum. *Traffic Management Specification, Version 4*. Technical Report, October 1995. 95-0013R8. (pp 50, 74, 85, 124, 144, 160, 210)

[AVTWG96] Audio-Video Transport Working Group, H. Schulzrinne, S. Casner, R. Frederick, and V. Jacobson. *RTP: A Transport Protocol for Real-Time Applications*. RFC 1889, January 1996. (p 122)

[Bajaj99] S. Bajaj, L. Breslau, D. Estrin, K. Fall, S. Floyd, P. Haldar, M. Handley, A. Helmy, J. Heidemann, P. Huang, S. Kumar, S. McCanne, R. Rejaie,

P. Sharma, K. Varadhan, Y. Xu, H. Yu, and D. Zappala. *Improving Simulation for Network Research*. Technical Report 99-702, University of Southern California, March 1999. (p 73)

[Barford98] P. Barford and M. Crovella. *Generating Representative Web Workloads for Network and Server Perform ance Evaluation*. In Proceedings of ACM SIGMETRICS'98, Madison, WI, 1998. (p 47)

[Battou96] A. Battou. *Connections Establishment Latency: Measured Results*. ATM-Forum T1A1.3/96-071, October 1996. (pp 94, 95)

[Baumann98] M. Baumann. *YATS — Yet Another Tiny Simulator (ATM Simulation)*. `http://www.ifn.et.tu-dresden.de/TK/yats/yats.html`, January 1998. (p 72)

[Bean93] N. G. Bean. *Statistical Multiplexing in Broadband Communication Net works*. PhD thesis, Statistical Laboratory, University of Cambridge, June 1993. (pp 134, 136)

[Bean94] N. G. Bean. *Robust connection acceptance control for ATM networks with incomplete source information*. Annals of Operations Research, 48:357–379, 1994. (pp 134, 136)

[Bennett96a] J. C. R. Bennett and H. Zhang. *Hierarchical packet fair queueing algorithms*. In Proceedings of ACM SIGCOMM'96, Stanford, CA, August 1996. (pp 51, 52)

[Bennett96b] J. C. R. Bennett and H. Zhang. *WF^2Q: Worst-case Fair Weighted Fair Queueing*. In Proceedings of IEEE INFOCOM'96, San Francisco, CA, March 1996. (pp 51, 52)

[Bennett97] J. C. R. Bennett, D. Stephens, and H. Zhang. *High speed, scalable, and accurate implementation of fair queueing algorithms in ATM networks*. In Proceedings of International Conference on Network Protocols (ICNP'97), Atlanta, GA, 1997. (p 51)

[Bernet98] Y. Bernet, R. Yavatkar, P. Ford, F. Baker, and L. Zhang. *A Framework for End-to-End QoS Combining RSVP/Intserv and Differentiated Services*, March 1998. INTERNET DRAFT: draft-bernet-intdiff-00.txt. (p 25)

[Bernet99] Y. Bernet, J. Binder, S. Blake, M. Carlson, B. E. Carpenter, S. Keshav, E. Davies, B. Ohlman, D. Verma, Z. Wang, and W. Weiss. *A Framework for Differentiated Services*, February 1999. INTERNET DRAFT: draft-ietf-diffserv-framework-02.txt. (p 221)

[Blake98] S. Blake, D. Black, M. Carlson, E. Davies, Z. Wang, and W. Weiss. *An Architecture for Differentiated Service*. RFC 2475, IETF, December 1998. (pp 102, 211, 221)

[Bolotin94] V. A. Bolotin. *Modeling Call Holding Time Distributions for CCS Network Design and Performance Analysis*. IEEE Journal on Selected Areas in Communications, 12(3):433–438, April 1994. (pp 33, 164, 166)

[Bolotin97] V. A. Bolotin. *New subscriber traffic variability patterns for network traffic engineering*. In Proceedings of the 15th International Teletraffic Congress (ITC15), June 1997. (p 33)

[Botvich95] D. Botvich and N. Duffield. *Large deviations, the shape of the loss curve, and economies of scale in large multiplexers*. Queueing Systems, 20:293–320, 1995. (pp 110, 130)

[Boyle00a] J. Boyle, R. Cohen, D. Durham, S. Herzog, R. Rajan, and A. Sastry. *COPS usage for RSVP*. RFC 2749, IETF, January 2000. (p 221)

[Boyle00b] J. Boyle, R. Cohen, D. Durham, S. Herzog, R. Rajan, and A. Sastry. *The COPS (Common Open Policy Service) Protocol*. RFC 2748, IETF, January 2000. (p 221)

[Braden94] R. Braden, D. Clark, and S. Shenker. *Integrated Services in the Internet Architecture: an Overview*. RFC 1633, IETF, June 1994. (pp 102, 144)

[Braden97] R. Braden, L. Zhang, S. Berson, S. Herzog, and S. Jamin. *Resource ReSerVation Protocol (RSVP) — Version 1 Functional Specification*. RFC 2205, IETF, September 1997. (pp 23, 24, 85, 124, 160, 210)

[Brady68] P. T. Brady. *A Statistical Analysis of On-Off Patterns in 16 Conversations*. Bell System Technical Journal, 47(1):73–91, January 1968. (p 30)

[Brady69] P. T. Brady. *A model for generating ON-OFF speech patterns in two-way conversations*. The Bell System Technical Journal, 48(9):2445–2472, September 1969. (pp 30, 42)

[Breslau00] L. Breslau, S. Jamin, and S. Shenker. *Comments on the Performance of Measurement-Based Admission Control Algorithms*. In Proceedings of IEEE INFOCOM 2000, Tel Aviv, Israel, March 2000. (pp 59, 132, 160, 167, 169, 177, 206)

[Breslau99] L. Breslau, S. Jamin, and S. Shenker. *Measurement-Based Admission Control: What is the Research Agenda?* In Proceedings of ACM SIGCOMM ’99, Cambridge, MA, August 1999. (p 160)

[Brichet97] F. Brichet and A. Simonian. *Measurement-based CAC for video applications using SBR service*. In Proceedings of Performance and Management of Complex Communications Networks, IFIP 6.3 and 7.3, Tsukuba, Japan, 1997. (p 60)

[Buffet92] E. Buffet and N. G. Duffield. *Exponential upper bounds via martingales for multiplexers with Markovian arrivals*. Journal of Applied Probability (Special Edition), 31:1049–1061, 1992. (pp 32, 124, 126, 130)

[Caceres00] R. Caceres, N. Duffield, A. Feldmann, J. Friedmann, A. Greenberg, R. Greer, T. Johnson, C. Kalmanek, B. Krishnamurthy, D. Lavelle, P. Mishra, K. K. Ramakrishnan, J. Rexford, F. True, and J. van der Merwe. *Measurement and analysis of IP network usage and behavior*. IEEE Communications Magazine, May 2000. (p 61)

[Cao00] Z. Cao, Z. Wang, and E. Zegura. *Rainbow Fair Queueing: Fair Bandwidth Sharing Without Per-Flow State*. In Proceedings of IEEE INFOCOM 2000, Tel Aviv, Israel, March 2000. (pp 213, 221)

[Cardwell00] N. Cardwell, S. Savage, and T. Anderson. *Modeling TCP Latency*. In Proceedings of IEEE INFOCOMM 2000, Tel Aviv, Israel, March 2000. (pp 38, 235)

[Carlberg01] K. Carlberg, P. Gevros, and J. Crowcroft. *Lower than Best Effort: a design and implementation*. In Proceedings of ACM SIGCOMM Workshop on Data Communications in Latin America and the Caribbean, San Jose, Costa Rica, 2001. (p 213)

[Case88] J. D. Case, M. Fedor, M. L. Schoffstall, and J. Davin. *Simple Network Management Protocol*. RFC 1067, IETF, August 1988. (p 204)

[Casetti96] C. Casetti, J. Kurose, and D. Towsley. *An adaptive algorithm for measurement-based admission control in integrated services packet networks*. In Proceedings of the International Workshop on Protocols for High-Speed Networks, Sophia Antipolis, France, October 1996. (p 58)

[CCCI99] Chesapeake Computer Consultants, Inc. *ttcp*. Technical Report, 1999. `http://www.ccci.com/tools/ttcp/`. (p 73)

[CCITT90a] CCITT SG XVIII, Geneva. *Recommendation I.362 — BISDN ATM Adaptation Layer Functional Description*. Technical Report, January 1990. (p 44)

[CCITT90b] CCITT SG XVIII, Geneva. *Recommendation I.363 — BISDN ATM Adaptation Layer Specification*. Technical Report, January 1990. (p 44)

[Cetinkaya00] C. Cetinkaya and E. W. Knightly. *Egress Admission Control*. In Proceedings of IEEE INFOCOM 2000, Tel Aviv, Israel, March 2000. (p 102)

[Chao97] H. J. Chao, H. Cheng, Y-R. Jeng, and D. Jeong. *Design of a generalized priority queue manager for ATM switches*. IEEE Journal on Selected Areas in Communications, 15(5):830–843, July 1997. (p 54)

[Chernoff52] H. Chernoff. *A Measure of Asymptotic Efficiency for Tests of a Hypothesis Based on the Sum of Observations*. Annals of Mathematical Statistics, 23:493–507, 1952. (p 105)

[Choudhury96a] A. Choudhury and E. Hahne. *Dynamic queue length threshold in a shared memory ATM switch*. In Proceedings of IEEE INFOCOM'96, San Francisco, CA, March 1996. (p 55)

[Choudhury96b] G. Choudhury, D. Lucantoni, and W. Whitt. *Squeezing the most out of ATM*. IEEE Transactions on Communications, 44(2):203–217, February 1996. (pp 110, 130)

[Claffy93] K. C. Claffy, G. C. Polyzos, and H-W Braun. *Application of Sampling Methodologies to Network Traffic Characterisation*. In Proceedings of ACM SIGCOMM'93, San Francisco, CA, September 1993. (p 65)

[Clark92] D. D. Clark, S. Shenker, and L. Zhang. *Supporting Real-Time Applications in an Integrated Services Packet Network: Architecture and Mechanism*. ACM Computer Communication Review, 22(4), October 1992. (pp 50, 209)

[Clark98] D. Clark and W. Fang. *Explicit Allocation of Best Effort Packet Delivery Service*. IEEE/ACM Transactions on Networking, 6(4):362–373, August 1998. (pp 54, 221)

[Courcoubetis95] C. Courcoubetis, G. Kesidis, A. Ridder, J. Walrand, and R. Weber. *Admission Control and Routing in ATM Networks using Inferences from Measured Buffer Occupancy*. IEEE Transactions on Communications, 43(2/3/4):1778–1784, April 1995. (pp 58, 59)

[Courcoubetis97] C. Courcoubetis, V. A. Siris, and R. Weber. *Investigation of cell scale and burst scale effects on the cell loss probability using large deviations*. In Proceedings of the 15th UK Workshop on Performance Engineering of Computer and Communication Systems (UKPEW'97), Ilkley, UK, July 1997. (p 53)

[Crosby95] Simon Crosby. *Performance Management in ATM Networks*. Technical Report 393, Cambridge University Computer Laboratory, May 1995. Ph.D. dissertation. (pp 36, 86)

[Crovella97] M. E. Crovella and A. Bestavros. *Self-similarity in World Wide Web traffic: Evidence and possible causes*. IEEE/ACM Transactions on Networking, 5(6):835–846, 1997. (pp 21, 33, 35, 42, 46, 244)

[Cruz92] R. Cruz. *Service burstiness and dynamic burstiness measures: A framework* . Journal of High Speed Networks, 1(2):105–127, 1992. (p 52)

[Cruz95] R. Cruz. *Quality of service guaranteed in virtual circuit switched network*. IEEE Journal on Selected Areas in Communications, 13(6):1048–1056, 1995. (p 52)

[Demers89] A. Demers, S. Keshav, and S. Shenker. *Analysis and simulation of a fair queueing algorithm*. In Proceedings of ACM SIGCOMM'89, Austin, TX, 1989. (p 51)

[Dovrolis00] C. Dovrolis and P. Ramanathan. *Proportional Differentiated Services, Part II: Loss Rate Differentiation and Packet Dropping*. In Proceedings of 8th IEEE/IFIP Workshop on Quality of Service IWQoS, Pittsburgh, PA, 2000. (pp 54, 55, 217)

[Droz97] P. Droz. *Wavelet-Based Resource Allocation in ATM Networks*. In IEEE Global Telecommunication Conference, Phoenix, AZ, November 1997. (p 58)

[Duffield00] N. G. Duffield and M. Grossglauser. *Trajectory Sampling for Direct Traffic Observation*. In Proceedings of ACM SIGCOMM 2000, Stockholm, Sweden, August 2000. (p 245)

[Duffield95] N. G. Duffield, J. T. Lewis, N. O'Connell, R. Russell, and F. Toomey. *Entropy of ATM Traffic Streams*. IEEE Journal on Selected Areas in Communications, 13(6):981–990, August 1995. (p 109)

[Duffield99a] N. Duffield. *Asymptotic sampling properties of effective bandwidth estimation for admission control*. In Proceedings of IEEE INFOCOM'99, New York, NY, March 1999. (p 58)

[Duffield99b] N. G. Duffield, P. Goyal, A. Greenberg, P. Mishra, K. K. Ramakrishnan, and J. E. van der Merwe. *A Flexible Model for Resource Management in Virtual Private Networks*. In Proceedings of ACM SIGCOMM'99, Cambridge, MA, August 1999. (pp 109, 115, 116, 212)

[Duffy94] D. E. Duffy, A. A. McIntosh, M. Rosenstein, and W. Willinger. *Statistical Analysis of CCSN/SS7 Traffic Data from Working CCS Subnetworks*. IEEE Journal on Selected Areas in Communications, 12(3):544–551, April 1994. (p 166)

[Dziong97] Z. Dziong, M. Juda, and L. G. Mason. *A Framework for Bandwidth Management in ATM Networks — Aggregate Equivalent Bandwidth Estimation Approach*. IEEE/ACM Transactions on Networking, 5(1):134–147, February 1997. (p 60)

[Elsayed99] K. M. F. Elsayed and H. G. Perros. *A Comparative Performance Analysis of Call Admission Control Schemes in ATM Networks*, 1999. To appear in the Journal of Computer Networks and ISDN Systems. (pp 56, 124)

[Elwalid95] A. Elwalid, D. Mitra, and R. H. Wentworth. *A New Approach for Allocating Buffers and Bandwidth to Heterogeneous Regulated Traffic in an ATM Node*. IEEE Journal on Selected Areas in Communications, 13(6):1115–1127, August 1995. (pp 60, 127, 128, 130, 213)

[Enrico00] M Enrico, N. Billington, J. Kelly, and G. Young. *Delivery of IP over broadband acess technologies*. BT Technology Journal, 18(3), July 2000. (p 165)

[Erlang17] A. K. Erlang. *Solution of some problems in the theory of probabilities of significance in automatic telephone exchanges*. Post Office Electrical Engineers Journal, 10:189–197, 1917. Translated from the Danish 1917 article in Elektroteknikeren, Volume 13. (p 32)

[Erramilli00] A. Erramilli, O. Narayan, A. Neidhardt, and I. Saniee. *Performance Impacts of Multi-Scaling in Wide Area TCP/IP Traffic*. In Proceedings of IEEE INFOCOM 2000, Tel Aviv, Israel, March 2000. (p 40)

[Feldmann98a] A. Feldmann, A. C. Gilbert, and W. Willinger. *Data networks as cascades: Investigating the multifractal nature of Internet WAN traffic*. ACM Computer Communication Review, 28(4):42–55, September 1998. (pp 35, 66)

[Feldmann98b] A. Feldmann, A. C. Gilbert, W. Willinger, and T. G. Kurtz. *The Changing Nature of Network Traffic: Scaling Phenomena*. ACM Computer Communication Review, 28(5):5–29, April 1998. (pp 21, 33, 35, 244)

[Feldmann99] A. Feldmann, A. C. Gilbert, P. Huang, and W. Willinger. *Dynamics of IP traffic: A study of the role of variability an d the impact of control*. In Proceedings of ACM SIGCOMM'99, Cambridge, MA, August 1999. (p 47)

[Feng99a] W. Feng. *Improving Internet congestion control and queue management algorithms*. PhD thesis, The University of Michigan, 1999. (p 213)

[Feng99b] W. Feng, D. Kandlur, D. Saha, and K. Shin. BLUE*: A new class of active queue management algorithms*. Technical Report CSE-TR-387-99, EECS Department, University of Michigan, May 1999. (p 221)

[Floyd93] S. Floyd and V. Jacobson. *Random early detection for congestion avoidance*. IEEE/ACM Transactions on Networking, 1(4):397–413, August 1993. (pp 54, 234)

[Floyd94] S. Floyd. *TCP and Explicit Congestion Notification*. ACM Computer Communication Review, 24(5):10–23, October 1994. (p 49)

[Floyd95] S. Floyd and V. Jacobson. *Link-sharing and Resource Management Models for Packet Networks*. IEEE/ACM Transactions on Networking, 3(4):365–386, August 1995. (pp 50, 209)

[Floyd96] S. Floyd. *Comments on Measurement-based Admissions Control for Controlled-Load Services*. Technical Report, Lawrence Berkeley Laboratory, July 1996. (pp 58, 59, 105, 107, 176, 177)

[Floyd99] S. Floyd and K. Fall. *Promoting the use of end-to-end congestion control in the internet*. IEEE/ACM Transactions on Networking, 7(4):458–472, August 1999. (p 38)

[FORE98] FORE Systems Inc. *Eagle: Rate-based Flow Control ASIC*. Technical Report, January 1998. (pp 74, 215, 218)

[FORE99] FORE Systems Inc. *Fore — ATM Switch Network Modules Product Overview*. Technical Report, October 1999. (pp 74, 163, 215)

[Fotedar95] S. Fotedar, M. Gerla, P. Crocetti, and L. Fratta. *ATM Virtual Private Networks*. Communications of the ACM, 38(2):101–109, February 1995. (p 212)

[Garrett93] M. W. Garrett. *Statistical Analysis of a Long Trace of Variable Bit Rate Video Traffic*. PhD thesis, Columbia University, 1993. (p 44)

[Garrett94] M.E. Garrett and W. Willinger. *Analysis, Modeling and Generation of Self-Similar VBR Video Traffic*. In Proceedings of ACM SIGCOMM'94, London, UK, August 1994. (pp 33, 36, 44)

[Gibbens00] R. J. Gibbens, S. K. Sargood, C. Van Eijl, F. P. Kelly, H. Azmoodeh, R. N. Macfadyen, and N. W. Macfadyen. *Fixed-Point Models for the End-to-End Performance Analysis of IP Networks*. In Proceedings of the ITC Specialist Seminar on IP Traffic Modeling, Measurement and Management, Monterey, CA, September 2000. (pp 39, 40, 50, 244)

[Gibbens95] R. J. Gibbens, F. P. Kelly, and P. B. Key. *A Decision-Theoretic Approach to Call Admission Control in ATM Networks*. IEEE Journal on Selected Areas in Communications, 13(6):1101–1114, 1995. (pp 58, 59, 60, 68, 145, 146, 157)

[Gibbens97] R. J. Gibbens and F. P. Kelly. *Measurement-based connection admission control*. In Proceedings of 15th International Teletraffic Congress (ITC15), June 1997. (pp 58, 59, 105, 107, 129, 136, 147, 167)

[Gibbens99] R. J. Gibbens and F. P. Kelly. *Distributed connection acceptance control for a connectionless network*. In Proceedings of 16th International Teletraffic Congress (ITC16), Edinburgh, UK, June 1999. (p 49)

[Gleeson00] B. Gleeson, A. Lin, J. Heinanen, G. Armitage, and A. Malis. *A Framework for IP Based Virtual Private Networks*. RFC 2764, IETF, February 2000. (p 212)

[Glynn94] P. W. Glynn and W. Whitt. *Logarithmic asymptotics for steady-state tail probabilities in a single-server queue.* Journal of Applied Probability (Special Edition), 31A:131–156, 1994. (p 125)

[Goyal96] P. Goyal, H. Vin, and H. Cheng. *Start-time Fair Queuing: A Scheduling Algorithm for Integrated Services Packet Switching Networks*. In Proceedings of ACM SIGCOMM'96, Stanford, CA, August 1996. (p 51)

[Grossglauser00] M. Grossglauser and D. Tse. *A Time-Scale Decomposition Approach to Measurement-Based Admission Control*. IEEE/ACM Transactions on Networking, December 2000. Submitted for publication. (pp 118, 119, 120, 186)

[Grossglauser96] M. Grossglauser and J.-C. Bolot. *On the relevance of long-range dependence in network traffic*. In Proceedings of ACM SIGCOMM'96, Stanford, CA, August 1996. (pp 36, 53)

[Grossglauser97a] M. Grossglauser, S. Keshav, and D. Tse. *RCBR: A simple and efficient service for multiple time-scale traffic*. IEEE/ACM Transactions on Networking, 5(6):741–755, 1997. (p 214)

[Grossglauser97b] M. Grossglauser and D. Tse. *A Framework for Robust Measurement-Based Admission Control*. ACM Computer Communication Review, 27(4):237–248, Oct 1997. (pp 50, 58, 68, 114, 118, 120, 157, 170, 186)

[Guérin91] R. Guérin, H. Ahmadi, and M. Naghshineh. *Equivalent Capacity and its application to bandwidth allocation in high speed networks*. IEEE Journal on Selected Areas in Communications, 9(7):968–981, September 1991. (pp 32, 123, 124, 130, 154)

[Guérin98] R. Guérin, S. Kamat, V. Peris, and R. Rajan. *Scalable QoS provision through buffer management*. In Proceedings of ACM SIGCOMM'98, Vancouver, British Columbia, September 1998. (p 55)

[Habib92] I. W. Habib and T. N. Saadawi. *Multimedia traffic characteristics in broadband networks*. IEEE Communications Magazine, 30(7):48–54, July 1992. (pp 30, 42)

[Hallsall96] F. Hallsall. *Data Communications, Computer Networks and Open Systems*. Addison-Wesley Publishing Company, Harlow, England, 4th edition, 1996. (p 30)

[Heinanen99] J. Heinanen, F. Baker, W. Weiss, and J. Wroclawski. *Assured Forwarding PHB Group*. RFC 2597, IETF, June 1999. (p 211)

[Hoeffding63] W. Hoeffding. *Probability Inequalities for Sums of Bounded Random Variables*. American Statistical Association Journal, 58:13–30, March 1963. (p 105)

[Hori98] Y. Hori, H. Sawashima, H. Sunahara, and Y. Oie. *Performance evaluation of UDP traffic affected by TCP flows*. IEICE Transactions on Communications, E81-B(8):1616–1623, August 1998. (pp 37, 38)

[HP99] Hewlett Packard Ltd. Information Networks Division — Networking Performance Team. *Netperf: Network Performance Measurement Tool*. Technical Report, 1999. `http://www.netperf.org/`. (p 73)

[Hui88] J. Y. Hui. *Resource Allocation for Broadband Networks*. IEEE Journal on Selected Areas in Communications, 6(9):1598–1608, December 1988. (pp 48, 49)

[Hurst51] H. Hurst. *Long-Term Storage Capacity of Reservoirs*. Transactions of the American Society of Civil Engineers, 116:770–799, 1951. (p 35)

[Hyman91] J. M. Hyman, A. A. Lazar, and G. Pacifici. *Real-Time Scheduling with Quality of Service Constraints*. IEEE Journal on Selected Areas in Communications, 9(7):1052–1063, September 1991. (pp 61, 145, 210)

[Hyman93] J. M. Hyman, A. A. Lazar, and G. Pacifici. *A Separation Principle between Scheduling and Admission Control for Broadband Switching*. IEEE Journal on Selected Areas in Communications, 11(4):605–616, May 1993. (p 61)

[Isaacs00] R. Isaacs. *Dynamic Provisioning of Resource-Assured and Programmable Virtual Private Networks*. PhD thesis, University of Cambridge Computer Laboratory, December 2000. Submitted for examination. (p 243)

[ITU-T99] ITU-T. *Recommendation Q.992.2, Asymmetric Digital Subscriber Line (ADSL) Transceivers*. ITU, Geneva, Switzerland, June 1999. (p 165)

[Jacobson99a] V. Jacobson, C. Leres, and S. McCanne. *tcpdump*. `ftp://ftp.ee.lbl.gov/tcpdump.tar.Z`, 1999. (p 74)

[Jacobson99b] V. Jacobson, K. Nichols, and K. Poduri. *An Expedited Forwarding PHB*. RFC 2598, IETF, June 1999. (p 211)

[Jaffe64] J. Jaffe, L. Cassotta, and S. Feldstein. *Markovian model of the time patterns of speech*. Science, 144:884–886, 1964. (p 30)

[Jamin92] S. Jamin, S. Shenker, L. Zhang, and D. D. Clark. *An Admission Control Algorithm for Predictive Real-Time Service*. In Proceedings of the 3rd International Workshop on Network and Operating Systems Support for Digital Audio and Video (NOSSDAV'92), La Jolla, CA, 1992. (p 122)

[Jamin97a] S. Jamin, P. B. Danzig, S. J. Shenker, and L. Zhang. *A Measurement-based Admission Control Algorithm for Integrated Services Packet Networks*. IEEE/ACM Transactions on Networking, 5(1):56–70, February 1997. (pp 57, 59, 60, 103, 176, 177)

[Jamin97b] S. Jamin and S. J. Shenker. *Measurement-based Admission Control Algorithms for Controlled-Load Service: A Structural Examination*. Technical Report CSE-TR-333-97, University of Michigan, April 1997. (pp 58, 59, 108, 109, 160, 167, 171, 176, 186, 206)

[Jamin97c] S. Jamin, S. J. Shenker, and P. B. Danzig. *Comparison of Measurement-based Admission Control Algorithms for Controlled-Load Service*. In Proceedings of IEEE INFOCOM'97, Kobe, Japan, April 1997. (pp 58, 59, 108, 140, 146, 148, 160, 167)

[Karlsson97] P. Karlsson and Å. Arvidsson. *On the Characteristics of WWW Traffic and the Relevance to ATM*. In Proceedings of the Fifth IFIP Workshop on Performance Modelling and Evaluation of ATM Networks, Ilkley, U.K., 1997. (p 165)

[Kelly96] Frank P. Kelly. *Notes on Effective Bandwidths*. In Frank P. Kelly, S. Zachary, and I. B. Ziedins, editors, *Stochastic Networks: Theory and Applications*, volume 4 of *Royal Statistical Society Lecture Note Series*, pages 141–168. Oxford University Press, 1996. (p 104)

[Keshav91] S. Keshav. *Congestion Control in Computer Networks*. PhD thesis, Department of EECS, University of California, Berkeley, CA, August 1991. (p 51)

[Key95] P. B. Key. *Connection Admission Control in ATM networks*. BT Technology Journal, 13(3), July 1995. (pp 58, 59, 68, 89, 145, 146, 157)

[Key99] P. Key, D. McAuley, P. Barham, and K. Laevens. *Congestion Pricing for Congestion Avoidance*. Technical Report MSR-TR-99-15, Microsoft Research, Microsoft Corporation, Cambridge, UK, December 1999. (p 49)

[Knightly96] E. W. Knightly. *Traffic Models and Admission Control for Integrated Services Networks*. PhD thesis, Department of EECS, University of California, Berkeley, CA, 1996. (pp 116, 151)

[Knightly98] E. W. Knightly and J. Qiu. *Measurement-Based Admission Control with Aggregate Traffic Envelopes*. In Proceedings of 10th Tyrrhenian International Workshop on Digital Communications, Ischia, Italy, September 1998. (pp 59, 68, 116, 117, 118, 157, 160, 176, 177, 206, 218)

[Knightly99] E. W. Knightly and N. B. Shroff. *Admission Control for Statistical QoS: Theory and Practice*. IEEE Network Magazine, 13(2):20–29, March 1999. (pp 56, 130)

[Kodialam00] M. Kodialam and T. V. Lakshman. *Minimum Interference Routing with Applications to MPLS Traffic Engineering*. In Proceedings of IEEE INFOCOM 2000, Tel Aviv, Israel, March 2000. (pp 243, 244)

[Kroner91] H. Kroner, G. Hebuterne, P. Boyer, and A. Gravey. *Priority Management in ATM Switching Nodes*. IEEE Journal on Selected Areas in Communications, 9(3):418–427, April 1991. (pp 53, 54, 217)

[Kumar98] V.P. Kumar, T.V. Lakshman, and D. Stiliadis. *Beyond best effort: Router architectures for the differentiated services of tomorrow's internet*. IEEE Communcations Magazine, 36(5):152–164, May 1998. (p 52)

[Lazar91] A. A. Lazar and G. Pacifici. *Control of Resources in Broadband networks with Quality of Service Guarantees*. "IEEE Communications Magazine, 29(10):66–73, September 1991. (p 61)

[Lazar97] A. A. Lazar and M. Nandikesan. *A Real-Time Traffic Generation and QOS Monitoring System*. Technical Report 481-97-15, Center for Telecommunications Research, Columbia University, August 1997. (p 73)

[Le Gall91] D. Le Gall. *MPEG: a Video Compression Standard for Multimedia applications*. Communications of the ACM, 34(4):47–58, April 1991. (p 43)

[Leland94] W. E. Leland, M. S. Taqqu, W. Willinger, and D. V. Wilson. *On the Self-Similar Nature of Ethernet Traffic (extended version)*. IEEE/ACM Transactions on Networking, 2(1):1–15, February 1994. (pp 33, 35, 39)

[Leslie93] I. M. Leslie, D. R. McAuley, and D. L. Tennenhouse. *ATM Everywhere?* IEEE Network, 7(2):40–46, March 1993. (p 24)

[Leslie96] I. Leslie, D. McAuley, R. Black, T. Roscoe, P. Barham, D. Evers, R. Fairbairns, and E. Hyden. *The Design and Implementation of an Operating System to Support Distributed Multimedia Applications*. IEEE Journal on Selected Areas in Communications, 14(7):1280–1297, September 1996. (p 80)

[Lewis98] J. T. Lewis, R. Russell, F. Toomey, B. McGurk, S. Crosby, and I. Leslie. *Practical Connection Admission Control for ATM Networks Based on On-line Measurements*. Computer Communications, 21:1585–1596, March 1998. (pp 58, 59, 91, 125, 186)

[Li95] S. Q. Li, S. Chong, and C.-L. Hwang. *Link capacity allocation and network control by filtered input rate in high-speed networks*. IEEE/ACM Transactions on Networking, 3(1):10–25, February 1995. (p 60)

[Lin91] A. M. Lin and J. A. Silvester. *Priority Queueing Strategies and Buffer Allocation Protocols for Traffic Control at an ATM Integrated Broadband Switching System*. IEEE Journal on Selected Areas in Communications, 9(9):1524–1536, December 1991. (pp 53, 54)

[Lottor88] M. Lottor. *TCP port service Multiplexer (TCPMUX)*. RFC 1078, IETF, November 1988. (p 49)

[Mandelbrot74] B. B. Mandelbrot. *Intermittent turbulence in self-similar cascades: divergence of high moments and the dimension of the carrier*. Journal of Fluid Mechanics, 62:331–358, 1974. (p 35)

[Manthorpe96] S. Manthorpe. *STCP 3.2.6: TCP/ABR/ATM network simulator*. `http://icawww1.epfl.ch/~manthorp/stcp/`, September 1996. (p 72)

[Mathis96] M. Mathis, J. Mahdavi, S. Floyd, and A. Romanow. *TCP Selective Acknowledgement Options*. RFC 2018, IETF, October 1996. (p 49)

[Mathis97] M. Mathis, J. Semke, J. Mahdavi, and T. Ott. *Behavior of the TCP Congestion Avoidance Algorithm*. ACM Computer Communication Review, 27(3):67–82, July 1997. (pp 37, 38, 39)

[Matsufuru00] N. Matsufuru and R. Aibara. *Flexible QoS control using partial buffer sharing with UPC*. IEICE Transactions on Communications, E83-B(2):196–203, February 2000. (pp 54, 217)

[McAuley90] D. R. McAuley. *Protocol Design for High Speed Networks*. PhD thesis, University of Cambridge Computer Laboratory, January 1990. (p 24)

[McGregor00] A. J. McGregor, H-W Braun, and J. A. Brown. *The NLANR Network Analysis Infrastructure*. IEEE Communications Magazine, 38(5), May 2000. (p 61)

[Mena00] A. Mena and J. Heidemann. *An Empirical Study of Real Audio Traffic*. In Proceedings of IEEE INFOCOM 2000, Tel Aviv, Israel, March 2000. (p 38)

[Minoli79] D. Minoli. *Issues in packet voice communications*. Proceedings of the Institution of Electrical Engineers, 126(8):729–740, August 1979. (p 30)

[Molina27] E. C. Molina. *Application of the Theory of Probabilities to Telephone Trunking Problems*. The Bell System Technical Journal, 6(19):461–494, 1927. (pp 32, 166)

[Moore99] A. Moore and S. Crosby. *An experimental configuration for the evaluation of CAC algorithms*. Performance Evaluation Review, 27(3):43–54, December 1999.

[Morris00] Robert Morris. *Scalable TCP Congestion Control*. In Proceedings of IEEE INFOCOM 2000, Tel Aviv, Israel, March 2000. (p 234)

[Mortier00] R. Mortier, I. Pratt, C. Clark, and S. Crosby. *Implicit Admission Control*. IEEE Journal on Selected Areas in Communications, 18(12):2629–2639, December 2000. (p 84)

[Nagle85] J. Nagle. *On packet switches with infinite storage*. RFC 970, IETF, December 1985. (p 49)

[Ng99] T. S. E. Ng, D. C. Stephens, I. Stoica, and H. Zhang. *Supporting Best-Effort Traffic With Fair Service Curve*. In Proceedings of ACM SIGMETRICS'99, Atlanta, GA, June 1999. (p 50)

[Nichols98] K. Nichols, S. Blake, F. Baker, and D. Black. *Definition of the Differentiated Services Field (DS Field) in the IPv4 and IPv6 Headers*. RFC 2474, IETF, December 1998. (pp 102, 211, 221)

[Nichols99] K. Nichols, V. Jacobson, and L. Zhang. *A Two-bit Differentiated Services Architecture for the Internet*. RFC 2638, IETF, July 1999. (pp 23, 25, 102, 213)

[Niehaus97] D. Niehaus, A. Battou, A. McFarland, B. Decina, H. Dardy, V. Sirkay, and B. Edwards. *Performance Benchmarking of ATM Networks*. IEEE Communications, 35(8):134–143, August 1997. (pp 94, 95)

[NLANR95] NLANR (National Laboratory for Applied Network Research). *FIX-West Statistics*. Technical Report, June 1995. `http://www.nlanr.net/Flowsresearch/fixstats.21.6.html`. (p 37)

[NRL96] Nemesys Research Limited. *AVA Owner's Manual*. Technical Report, September 1996. (pp 44, 80)

[OPNET00] OPNET Optimum Network Performance. *OPNET Modeller*. Technical Report, December 2000. `http://www.opnet.com/products/home.html`. (p 73)

[Ortúzar94] J. Ortúzar and L. Willumsen. *Modelling Transport*. Wiley, 1994. ISBN 0-471-94193-X. (p 244)

[Padhye98] J. Padhye, V. Firoiu, D. Towsley, and J. Kurose. *Modeling TCP Throughput: A Simple Model and its Empirical Validation*. In Proceedings of ACM SIGCOMM'98, Vancouver, British Columbia, September 1998. (pp 38, 39, 235)

[Padhye99] J. Padhye, J. Kurose, D. Towsley, and R. Koodli. *A TCP-friendly rate adjustment protocol for continuous media flows over best-effort networks*. In Proceedings of ACM SIGMETRICS'99, Atlanta, GA, June 1999. (p 38)

[Pan93] D. Y. Pan. *Digital Audio Compression*. Digital Technical Journal, 5(2), Spring 1993. (p 38)

[Parekh93] A. K. Parekh and R. G. Gallager. *A Generalized Processor Sharing Approach to Flow Control in Integrated Services Networks: The Single-Node Case*. IEEE/ACM Transactions on Networking, 1(3):344–357, June 1993. (pp 50, 51)

[Parekh94] A. K. Parekh and R. G. Gallager. *A Generalized Processor Sharing Approach to Flow Control in Integrated Services Networks: The Multiple-Node Case*. IEEE/ACM Transactions on Networking, 2(2):137–150, April 1994. (pp 50, 51)

[Pawlikowski90] K. Pawlikowski. *Steady-State Simulation of Queueing Processes: A Survey of Problems and Solutions*. ACM Computer Surveys, 22(2):123–170, June 1990. (pp 90, 187)

[Paxson95] V. Paxson and S. Floyd. *Wide-Area Traffic: The Failure of Poisson Modeling*. IEEE/ACM Transactions on Networking, 3(3):226–244, June 1995. (pp 21, 33, 35, 40, 165, 244)

[Perros96] H. G. Perros and K. M. F. Elsayed. *Call admission control schemes: A review*. IEEE Magazine on Communications, 34(11):82–91, November 1996. (p 56)

[Postel80] J. Postel. *User Datagram Protocol*. RFC 768, IETF, August 1980. (p 38)

[Postel81a] J. Postel. *Internet Protocol*. RFC 791, IETF, September 1981. (p 24)

[Postel81b] J. Postel. *Transmission Control Protocol*. RFC 793, IETF, September 1981. (pp 24, 49)

[Qiu98a] J. Qiu. *Measurement-Based Admission Control in Integrated-Services Networks*. Master's thesis, Electrical and Computer Engineering, Rice University, Houston, Texas, April 1998. (pp 117, 153)

[Qiu98b] J. Qiu and E. W. Knightly. *QoS Control via Robust Envelope-Based MBAC*. In Proceedings of 7th IEEE/IFIP Workshop on Quality of Service IWQoS, Napa, CA, May 1998. (pp 68, 116, 160, 176, 218)

[Ramakrishnan90] K. K. Ramakrishnan and R. Jain. *A Binary Feedback Scheme for Congestion Avoidance in Computer Networks*. ACM Transactions on Computer Systems, 8(2):158–181, May 1990. (p 49)

[Ramakrishnan99] K. Ramakrishnan and S. Floyd. *A Proposal to add Explicit Congestion Notification (ECN) to IP*. RFC 2481, IETF, January 1999. (p 49)

[Rexford96] J. L. Rexford, A. G. Greenberg, and F. G. Bonomi. *Hardware-efficient fair queueing architectures for high-speed networks*. In Proceedings of IEEE INFOCOM'96, San Francisco, CA, March 1996. (p 51)

[Riedi97] R. Riedi and J. Lévy Véhel. *TCP traffic is multifractal: a numerical study*. Technical Report Inria research report, no. 3129, Project Fractales, INRIA Rocquencourt, March 1997. (p 35)

[Riedi98] R. Riedi, M. Crouse, V. Ribeiro, and R. Baraniuk. *A multifractal wavelet model with application to TCP network traffic*. IEEE Transactions on Information Theory, 45(4):992–1018, 1998. (p 35)

[Risso99] F. Risso and P. Gevros. *Operational Performance Issues of a CBQ router*. ACM Computer Communication Review, 29(5):47–58, October 1999. (p 73)

[Roberts92] J. W. Roberts, editor. *Performance evaluation and design of multiservice networks — COST 224 Final Report*. Commission of the European Communities, Directorate-General, Telecommunications, Information Industries and Innovation, L-2920 Luxembourg, 1992. (p 53)

[Roberts94] J. W. Roberts. *Virtual Spacing for Flexible Traffic Control*. International Journal of Communication Systems, 7:307–318, 1994. (p 51)

[Rooney98] S. Rooney, J. E. van der Merwe, S. Crosby, and I. Leslie. *The Tempest, a Framework for Safe, Resource Assured, Programmable Networks*. IEEE Communications Magazine, 36(10):42–53, October 1998. (p 212)

[Rosen99] E. Rosen and Y. Rekhter. *BGP/MPLS VPNs*. RFC 2547, IETF, March 1999. (p 212)

[Ross95] K. W. Ross. *Multiservice Loss Models for Broadband Telecommunications Networks*. Telecommunication Networks & Computer Systems Series. Springer–Verlag, London, UK, 1995. (p 39)

[Ryu96] B. K. Ryu and A. Elwalid. *The Importance of Long-range Dependence of VBR Video Traffic in ATM Traffic Engineering: Myths and Realities*. In Proceedings of ACM SIGCOMM'96, Stanford, CA, August 1996. (pp 36, 50)

[Sahu99] S. Sahu, P. Nain, D. Towsley, C. Diot, and V. Firou. *On Achievable Service Differentiation with Token Bucket Marking for TCP*. Technical Report UMASS CMPSCI TECH REPORT 99-72, University of Massachusetts, 1999. (pp 54, 235)

[Saito91] H. Saito and K. Shiomoto. *Dynamic call admission control in ATM networks*. IEEE Journal on Selected Areas in Communications, 9(7):982–989, September 1991. (pp 60, 121, 122)

[Saito92] H. Saito. *CAC in an ATM network using upper bound cell loss probability*. IEEE Transactions on Communications, 40:1512–1521, September 1992. (p 121)

[Sariowan95] H. Sariowan, R. L. Cruz, and G. C. Polyzos. *Scheduling for quality of service guarantees via service curves*. In Proceedings of the International Conference on Computer Communications and Networks (ICCCN), Las Vegas, NV, September 1995. (pp 52, 237)

[Schlembach00] J. Schlembach, A. Skoe, P. Yuan, and E. W. Knightly. *Design and Implementation of a Scalable Admission Control*. IEEE/ACM Transactions on Networking, October 2000. Submitted for publication. (p 102)

[Shenker94] S. Shenker. *Making greed work in networks: A game theoretical analysis of switch service disciplines*. In Proceedings of ACM SIGCOMM'94, London, UK, August 1994. (p 50)

[Shiomoto95] K. Shiomoto and S. Chaki. *Adaptive connection admission control using real-time traffic measurements in ATM networks*. IEICE Transactions on Communications, E78-B(4):458–464, April 1995. (p 60)

[Shiomoto99] K. Shiomoto, N. Yamanaka, and T. Takahashi. *Overview of Measurement-Based Connection Admission Control Methods in ATM Networks*. IEEE Communication Surveys, pages 2–13, First Quarter 1999. (pp 60, 103, 156)

[Shreedhar95] M. Shreedhar and G. Vargese. *Efficient fair queueing using deficit round robin*. In Proceedings of ACM SIGCOMM'95, Cambridge, MA, August 1995. (p 51)

[Stephens99] D. C. Stephens, J. C. R. Bennett, and H. Zhang. *Implementing Scheduling Algorithms in High-Speed Networks*. IEEE Journal on Selected Areas in Communications, 17(6):1145–1158, June 1999. (pp 51, 52, 215, 218)

[Stevens97] W. Stevens. *TCP Slow Start, Congestion Avoidance, Fast Retransmit, and Fast Recovery Algorithms*. RFC 2001, IETF, January 1997. (p 49)

[Stiliadis96] D. Stiliadis and A. Verma. *Design and analysis of frame based fair queueing: A new traffic scheduling algorithm for packet-switched networks*. In Proceedings of ACM SIGMETRICS'96, Philadelphia, PA, 1996. (p 51)

[Stine90] R.H. Stine. *FYI on a network management tool catalog: Tools for monitoring and debugging TCP/IP internets and interconnected devices*. RFC 1147, IETF, April 1990. (pp 73, 82)

[Stoica97] I. Stoica, H. Zhang, and T. Ng. *A hierarchical fair service curve algorithm for link-sharing*. In Proceedings of ACM SIGCOMM'97, Cannes, France, September 1997. (pp 50, 52, 209, 237)

[Stoica98] I. Stoica, S. Shenker, and H. Zhang. *Core-Stateless Fair Queueing: A Scalable Architecture to Approximate Fair Bandwidth Allocations in High Speed Networks*. In Proceedings of ACM SIGCOMM'98, Vancouver, British Columbia, September 1998. (p 50)

[Stoica99] I. Stoica and H. Zhang. *Providing Guaranteed Services Without Per Flow Management*. In Proceedings of ACM SIGCOMM'99, Cambridge, MA, August 1999. (p 102)

[Suter99] B. Suter, T. V. Lakshman, D. Stiliadis, and A. K. Choudhury. *Buffer Management Schemes for Supporting TCP in Gigabit Routers with Per-Flow Queueing*. IEEE Journal on Selected Areas in Communications, 16(6):1159–1169, September 1999. (p 54)

[Tedijanto93] T. E. Tedijanto and L. Gün. *Effectiveness of dynamic bandwidth management mechanisms in ATM networks*. In Proceedings of IEEE INFOCOM'93, San Francisco, CA, 1993. (pp 60, 122)

[Tse99] D. Tse and M. Grossglauser. *A Time-Scale Decomposition Approach to Measurement-Based Admission Control*. In Proceedings of IEEE INFO-

COM'99, New York, NY, March 1999. (pp 57, 58, 63, 68, 103, 117, 118, 119, 157, 170)

[Tutschku98] K. Tutschku and P. Tran-Gia. *Spatial traffic estimation and characterization for mobile communication network design*. IEEE Journal on Selected Areas in Communications, 16(5):804–811, June 1998. (p 245)

[van den Berg00] H. van den Berg and M. Mandjes. *Admission control in integrated networks: overview, evaluation and research perspectives*. Telecommunication Systems, 2000. To be published. (p 60)

[Veres00] A. Veres and M. Boda. *The Chaotic Nature of TCP Congestion Control*. In Proceedings of IEEE INFOCOM 2000, Tel Aviv, Israel, March 2000. (p 37)

[Vicari00] N. Vicari and S. Köhler. *Measuring Internet User Traffic Behavior Dependent on Access Speed*. In Proceedings of the ITC Specialist Seminar on IP Traffic Modeling, Measurement and Management, Monterey, CA, September 2000. (pp 33, 40)

[Vicari97] N. Vicari. *Measurement and Modeling of WWW-Sessions*. Technical Report 184, Institute of Computer Science, University of Würzburg, September 1997. (p 46)

[Warfield94] R. Warfield, S. Chan, A. Konheim, and A. Guillaume. *Real-Time Traffic Estimation in ATM Networks*. In Proceedings of 14th International Teletraffic Congress (ITC14), Antibes, France, June 1994. (p 68)

[Willinger95] W. Willinger, M. S. Taqqu, R. Sherman, and D. V. Wilson. *Self-Similarity Through High-Variability: Statistical Analysis of Ethernet LAN Traffic at the Source Level*. In Proceedings of ACM SIGCOMM'95, Cambridge, MA, August 1995. (pp 42, 46)

[Wroclawski97] J. Wroclawski. *Specification of the Controlled-Load Network Element Service*. RFC 2211, IETF, September 1997. (pp 50, 102, 144)

[Wu98] Y. Wu. *Fair-Sharing of Link and Buffer*. IEICE Transactions on Communications, E81-B(5):1025–1028, May 1998. (p 50)

[Zhang91] L. Zhang. *Virtual Clock: A new traffic control algorithm for packet switching*. ACM Transactions on Computer Systems, 9(2):101–124, May 1991. (p 51)

[Zhang95] H. Zhang and E.W. Knightly. *A New Approach to Support Delay-Sensitive VBR Video in Packet-Switched Networks*. In 5th International Workshop on Network and Operating Systems Support for Digital Audio and Video (NOSSDAV'95), Durham, NH, 1995. (p 214)