A MEASUREMENT-BASED ADMISSION CONTROL ALGORITHM FOR INTEGRATED SERVICES PACKET NETWORKS

by

Sugih Jamin

A Dissertation Presented to the
FACULTY OF THE GRADUATE SCHOOL
UNIVERSITY OF SOUTHERN CALIFORNIA
In Partial Fulfillment of the
Requirements for the Degree
DOCTOR OF PHILOSOPHY
(Computer Science)

August 1996

To my Parents.
for their Love,
Support, and Trust.

Preface

Many designs for Integrated Services Packet Networks (ISPN) offer a bounded delay packet delivery service to support real-time applications. Networks achieve bounded delay by regulating their load and managing their resources. Admission control algorithm is the tool networks use to regulate their load. Previous work on admission control mainly focused on algorithms that compute the worst case theoretical queueing delay to guarantee either an absolute delay bound for all packets or a probabilistic bound on the statistical distribution tail of aggregate traffic. Since worst-case bounds are computed from parameterized source models, we call such algorithms parameter-based algorithms. Our own work proposes a *measurement-based* admission control algorithm for *predictive* service. Instead of guaranteeing an absolute or a numerically enforced probabilistic bound, predictive service promises a *reliable* bound. With the more relaxed bound, an admission control algorithm can operate without requiring a precise characterization of traffic; instead, it can use measured traffic characteristics. The reliance of our admission control algorithm on measurement dictates that it works well only when there is a high degree statistical multiplexing. Several researchers have discovered that network traffic is long-range dependent which rises and ebbs with possibly long ebb times. One dangerous implication of long-range dependent traffic on any measurement-based admission control algorithm is that the algorithm may allow too many new flows into the network during the ebb times, resulting in prolonged delay bound violations during the ensuing tides. In our simulations, besides traditional source models, we also use source models that exhibit long-range dependence, both in themselves and in their aggregation. As with most measurement-based control systems, there are several knobs that govern the degree of conservativeness of the measured values and resulting decisions. We will explore these and also look at some dynamic interactions between flows with different resource requirements. We will present results from a comparative study of several measurement-based admission control algorithms, and finally conclude this dissertation by pointing out several possible extensions to our work.

Appreciations

If one thought of the doctoral program as an apprenticeship in research, one could hardly ask for better teachers than those who taught and guided me. I thank my advisor, Peter Danzig, for the financial and moral support over the past six years, for patiently filling in the gaps in my knowledge of statistics and probability and queueing theories. And I thank him for the intellectual freedom that allowed me to explore on my own. Scott Shenker added rigor to my work. To him I owe my anchor, perspective, and approach to research. Lixia Zhang showed me how to scrutinize data. She taught me how to seek inward for answers and not look outward nor backward. I also thank her for arranging the Navy contract that partly supported my research. I thank Deborah Estrin for introducing me to the world of computer networking and the real world of research. John Silvester kindly served as the outside member of my committee and helped clarify my presentation. Sally Floyd and Walter Willinger patiently explained the intricacies of long-range dependent traffic to me. I would also like to thank Carolyn Hill for teaching me how to write coherently. Any mistakes, omissions, and blemishes of this work are, of course, attributable only to me and not to any of the illustrious masters above.

I thank Allyn Romanow and Allison Mankin for supporting my research. I have benefited from discussions, technical and otherwise, with Lee Breslau, Ron Frederick, Charley Liu, and Danny Mitzel. Finally, I thank my family and all my other friends for making life less dreary in these past seven years. I especially thank John Bell, who lured me into Computer Science and who has been my constant spiritual guide.

Kim Korner was a good teacher and a caring person whose untimely departure is sorely regretted.

Sponsor Acknowledgements

This research was supported in part by the Uniforum Research Award and by the Office of Naval Research Laboratory under contract number N00173-94-P-1205. This research also benefited from AFOSR award number F49620-93-1-0082, the NSF small-scale infrastructure grant, award number CDA-9216321, and equipment loan from Sun Microsystems, Inc. These funds and infrastructure awards gave me great independence to pursue my ideas, and I am thankful for the support.

Contents

Preface **iii**

1 Introduction **1**

2 Traditional Realtime Services **6**
2.1 Deterministic Bound 8
2.2 Probabilistic Bound: Equivalent Bandwidth 9

3 Relaxed Realtime Services **17**
3.1 Against Delay Bound 18
3.2 Unadvertised Bound 19
3.3 Predictive Service 20

4 Measurement-based Admission Control **23**
4.1 Framework 23
4.2 Worst-case Delay: Predictive Service 26
4.3 Worst-case Delay: Guaranteed Service 32
4.4 Equivalent Token Bucket Filter 33
4.5 The Admission Control Algorithm 34

5 A Simple Time-window Measurement Mechanism **36**
5.1 Measurement Process 36
5.2 Performance Tuning Knobs 39

6 Simulations **41**
6.1 Simulated Topologies 41
6.2 Source Models 42
6.3 Parameter Choices 48

7 On the Viability of the Algorithm **52**
7.1 Homogeneous Sources: The Single-hop Case 52
7.2 Homogeneous Sources: The Multi-hop Case 54
7.3 Heterogeneous Sources: The Single-hop Case 56
7.4 Heterogeneous Sources: The Multi-hop Case 57

8 Practical Deployment Issues **60**
8.1 Choosing a Window Size 60
8.2 Choosing a Utilization Target 64
8.3 Structural Limitations 64
8.4 If Peak Rate is Incoming Link Bandwidth 65

9 On Unequal Flow Rejection Rates **68**
9.1 Effect of Hop Count on Rejection Rates 68
9.2 Effect of Resource Requirements on Rejection Rates 70
9.3 A Quota Mechanism 71

10 Comparison of Admission Control Algorithms **75**
10.1 Five Admission Control Algorithms 76
10.2 Exponential-Weighted Moving Average 79
10.3 Simulation Results 80

11 Summary and Extensions **93**
11.1 A Better Estimator 94
11.2 Other Admission Criteria 94
11.3 Additional Issues 95

Bibliography **97**

List of Tables

6.1 Forty-five Host Pairs on TBone 44
6.2 Six Instantiations of the Three Source Models 49

7.1 Single-hop Homogeneous Sources Simulation Results 53
7.2 Multi-hop Homogeneous Sources Link Utilization 55
7.3 Single-hop, Single Source Model, Multiple Predictive Services Link Utilization 57
7.4 Single-hop, Multiple Source Models, Single Service Link Utilization . 57
7.5 Single-hop, Multiple Source Models, Multiple Predictive Services Link Utilization 58
7.6 Single- and Multi-hop, All Source Models, All Services Link Utilization 59

8.1 Effect of T and $\overline{L}$ 62

9.1 Efficacy of Quota Mechanisms 74
9.2 Benefits of Measurement-based Quota Mechanism 74

10.1 Six Instantiations of the Two Source Models 81
10.2 Single-hop Homogeneous Sources Simulation Results 82
10.3 Multiple-hop All Sources Simulation Results 90
10.4 Percentage Composition of Type of Admitted Flows 91

List of Figures

2.1 Example of Integrated Services Packet Network 7
2.2 Statistical multiplexing of three flows and their equivalent bandwidth. 10

4.1 Effect of new predictive flow on same priority traffic. 29
4.2 Effect of new predictive flow on lower priority traffic. 30
4.3 Effect of a guaranteed flow on predictive traffic. 32

5.1 Measuring delay. 37
5.2 Measuring rate. 38

6.1 The One-Link, Two-Link and Four-Link topologies 42
6.2 The TBone topology 43
6.3 Experienced delay of exp and poo sources. 45
6.4 On/off traffic model with token-bucket filter 48

8.1 Effect of T on Experienced Delay 63
8.2 Effect of T on Link Utilization 63

10.1 Distribution of experienced queueing delay of exp1 sources. 88
10.2 Distribution of experienced queueing delay of poo0 sources. 89
10.3 Distribution of experienced queueing delay of poo1 sources. 89
10.4 Distribution of experienced queueing delay of poo0 sources under the "Measured Sum" and bounded delay algorithms for different utilization target (ut) and delay bound (db). 90

Chapter 1

Introduction

The technical and regulatory developments of the past decade have created the possibility of merging digital telephony, multimedia transport, and data communication services into a single Integrated Services Packet Network (ISPN). From an economic perspective, an ISPN offering multiple service selections increases network's total utility by matching services closer to application needs [She95]. Chief among the services required by multimedia applications is bounded delay packet delivery. There have been many proposals for supporting bounded delay delivery in packet networks; see [OON88, FV90, GAN91] for a few representative examples. The ability of bounded delay services to achieve high utilization and also meet their service commitments depends crucially on their admission control algorithm. Conversely, the ability of an admission control algorithm to increase network utilization is ultimately constrained by the service commitments the network makes. A service model is a service commitments contract between the network and its users. Traditional real-time service provides a hard or absolute bound on the delay of every packet; in [FV90, CSZ92], this service model is called the *guaranteed*, or *deterministic guaranteed*, service. When a flow requests real-time service, it must characterize its traffic so that the network can make its admission control decision. Typically, sources are described by either peak and average rates [FV90] or a filter like a token bucket [OON88]; these descriptions provide upper bounds on the traffic that can be generated by the source. The admission control algorithm for guaranteed service uses *a priori* characterizations of sources to calculate the worst-case behavior of all the existing flows in addition to the incoming one. Calculating the worst-case delays may be very complex, but the underlying admission control principle is conceptually simple: does granting a new request for service cause the worst-case behavior of the network to violate any delay bound? (See [FV90] for an example of this approach.) Network utilization under this model is low when sources are bursty. However, network utilization can be increased if one can precisely characterize the offered traffic, such as when playing back recorded data [GKT95, WKZL96]; when flows carry live, bursty data, however, their traffic characterizations must necessarily be quite loose, in that the average behavior of the flows is significantly less than the upper bound of the traffic descriptions, and guaranteed service inevitably results in low utilization [ZF94].

A service model that promises a more relaxed delay bound than guaranteed service allows its admission control algorithm to admit more flows and attain a higher level of network utilization. There are many approaches to admission control that attempt to achieve higher utilization by weakening the degree of reliability of the delay bound. For instance, the probabilistic delay bound service described in [ZK94] does not provide for the worst-case scenario, instead it guarantees a bound on the probability of lost/late packets based on statistical characterization of traffic [VPV88]. In most cases the *a priori* characterization of flows is based on a statistical model [Hui88, SS91] or on a fluid flow approximation [GAN91, Kel91]). In this kind of approach, each flow is allotted an equivalent bandwidth that is larger than its average rate but less than its peak rate. If one can precisely characterize traffic *a priori*, this approach will increase network utilization. However, we think it will be quite difficult, if not impossible, to provide accurate and tight statistical models for each individual flow. For instance, the average bit rate produced by a given codec in a teleconference will depend on the participant's body movements. This can't possibly be predicted in advance with any degree of accuracy. Therefore the *a priori* traffic characterizations handed to admission control will inevitably be fairly loose upper bounds.

For this reason, we think that *measurement-based* admission control will play a key role in achieving high network utilization. The measurement-based admission control approach advocated in [CSZ92, JSZC92] uses the *a priori* source characterizations only for incoming flows (and those very recently admitted); it uses measurements to characterize those flows that have been in place for a reasonable duration. Therefore, network utilization does not suffer significantly if the traffic descriptions are not tight. For instance, if a source describes itself as conforming to a token bucket with a rate of 5 Mbps, but typically sends at an average rate of 1 Mbps, the measurement-based admission control approach does not indefinitely continue to set aside 5 Mbps for this flow, unlike the more traditional forms of admission control. Because it relies on measurements, and source behavior is not static in general, the measurement-based approach to admission control can never provide the completely reliable delay bounds needed for guaranteed, or even probabilistic, service. However, many real-time applications, such as *vat, ivs, nv*, and *vic*, have recently been developed for packet-switched networks. These applications can adapt to actual packet delays and are rather tolerant of delay bound violations; they do not need an absolutely reliable bound. For these *tolerant* applications, references [CSZ92, SCZ93] propose *predictive* service, which offers a fairly, but not absolutely, reliable bound on packet delivery times. The ability to occasionally incur delay violations gives admission control a great deal more flexibility, and is the chief advantage of predictive service. The measurement-based approaches to admission control can only be used in the context of predictive service and other more relaxed service commitments. Furthermore, when there are only a few flows present, the unpredictability of individual flow's behavior dictates that these measurement-based approaches must be very conservative—by using some worst-case calculation for example. Thus a measurement-based admission control algorithm can deliver significant gain in utilization only when there is a high degree of multiplexing.

The use of measurement in admission control algorithm has been mentioned in the literature prior and subsequent to this work. The authors of [HLP93, GKK95], for example, use measurements to determine admission, but the admission decisions are pre-computed based on the assumption that all sources are exactly described by one of a finite set of source models. This approach is clearly not applicable to a large, heterogeneous, and ever-changing application base, and is very different from our approach to admission control that is based on ongoing measurements. Using ongoing measurements of load in making admission decisions is suggested, but not fully developed nor explored, in [OON88]. Several recent papers, such as [SS91, AS94] use measurement to learn the parameters of certain assumed traffic distributions. The authors of [DJM96, Flo96a] use measurement of existing traffic in their calculation of equivalent bandwidth. In references [Hir91, CLG95], a neural network is used for dynamic bandwidth allocation. In [LCH95], the authors use pre-computed low frequency of flows to allocate bandwidth dynamically by renegotiation. Hardware implementation of measurement mechanisms are studied in [C^+91, WCKG94]. Incidentally, the work presented in this dissertation has been extended in [DKPS95] to support advance reservation. The authors of [DKPS95] have also replicated some of our results on their independently developed network simulator.

Several service models offering even more lax contractual agreement between the network and its users than predictive service have recently been proposed in the Internet Engineering Task Force (IETF). The *Controlled-load* service described in reference [Wro95] and *Committed-rate* service described in reference [BGK96] are examples of such service models. The service they provide do not in general involved an advertised quantitative service target such as loss rate or delay bound, rather they simply ensure that flows are alloted some reserved resources and experience low queueing delays. The minimal commitment made by these services makes them especially well suited to the decentralized and heterogeneous Internet. Our measurement-based admission control algorithm can thus also be used in conjunction with these more lax services.

In summary, when delay bound is strict, one can achieve high level of network utilization only when one has a very precise characterization of offered traffic. By relaxing the strictness of the delay bound, probabilistic service model can increase network utilization without requiring the tightest traffic characterization. Predictive service does not make any assumptions on source models and provides only reliable, not guaranteed, delay bound. Several new service models have recently been proposed that offer even more lax contractual agreement between the network and its users. We show in this dissertation that when measurement-based admission control algorithm is used in conjunction with services offering lax delay bound, and predictive service in particular, it can, at times, deliver order of magnitude higher level of network utilization than those achievable under parameter-based algorithm offering guaranteed service and still maintain reliable delay bound. Earlier versions of this work have been published as references [JSZC92, JDSZ95, JDSZ96].

Chapter 2

Traditional Realtime Services

Under the synchronous transfer mode (STM), sources send data at a constant bit rate (CBR). Sources with data rate higher than the constant bit rate must lower their quality either by dropping data or queueing it for later transmission. Sources with data rate less than the constant bit rate must pad their data. Constant bit rate leads to varying service quality or low network utilization. With the advent of asynchronous transfer mode (ATM), and on packet-switched networks such as the Internet, sources can transmit at variable bit rate (VBR), delivering constant service quality [VPV88]. Integrated services packet networks built on top of ATM or Internet technology allow packets from different types of VBR sources to be statistically multiplexed. Fig. 2.1 shows an example of an ISPN where packets from traditional data sources are multiplexed with packets from audio and video sources. The figure also shows a possible architecture of an ISPN switch, consisting of a realtime scheduler, such as the unified scheduler proposed in reference [CSZ92], an admission control algorithm, and a reservation protocol, such as the RSVP resource reservation protocol proposed in [Z$^+$93]. By allowing statistical multiplexing, an ISPN can increase network utilization; however, with statistical multiplexing, bursts of data could arrive simultaneously at a switch, leading to long queue and packet losses. Whereas the quality requirements of traditional data traffic are high throughput and short round-trip delay, the service requirements of realtime traffic are short queueing delay and

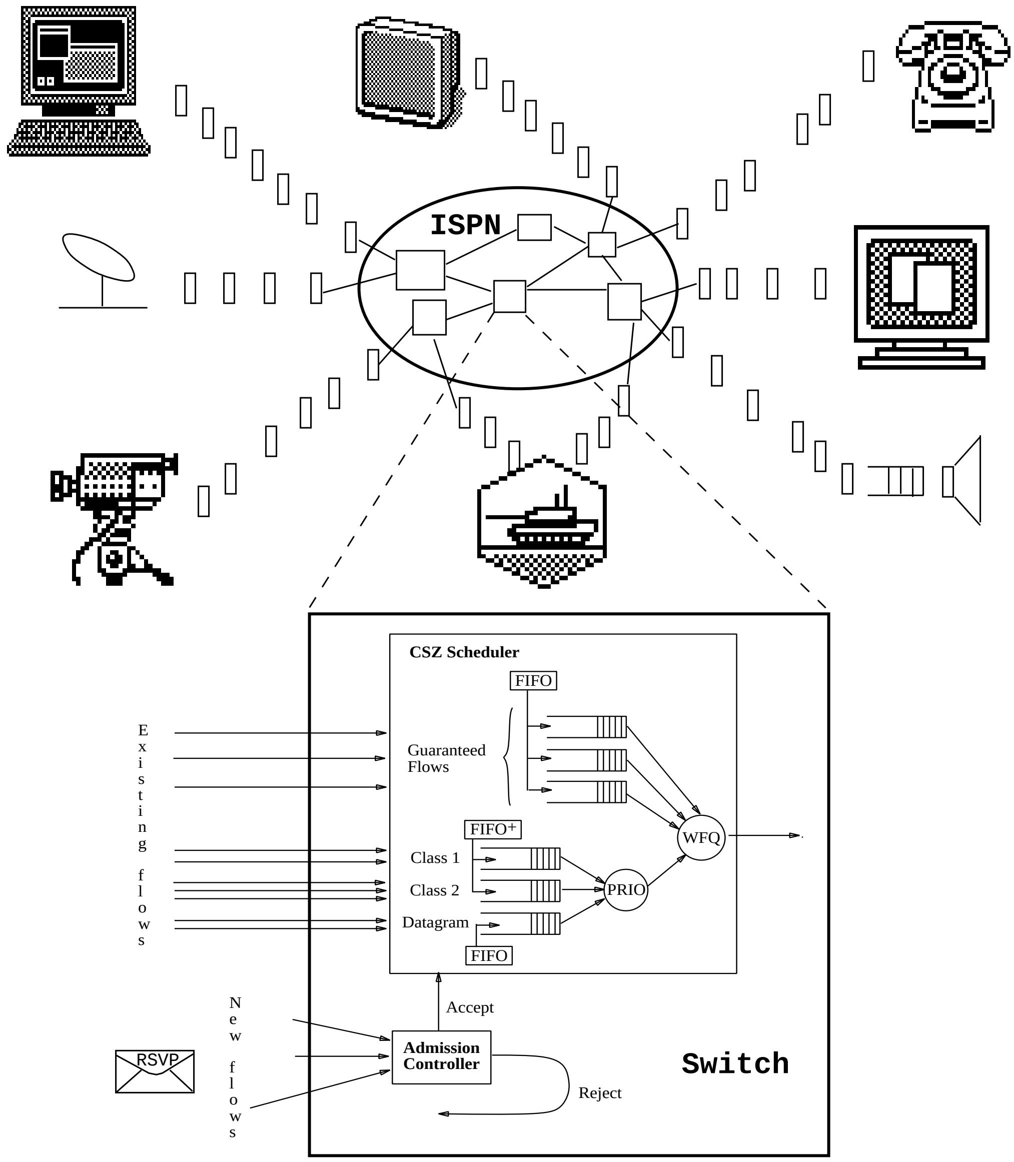


Figure 2.1: Example of Integrated Services Packet Network

small loss rate. The goal of all admission control algorithms is to meet users' quality of service requirements. In most cases, a secondary goal of an admission control algorithm is to meet users' requirements at as high a level of network utilization as feasible. The ability of an admission control algorithm to increase network utilization is ultimately constrained by the service commitments the network makes. We now look at the commitments different service models entail and the ensuing constraints put on the admission control algorithm.

2.1 Deterministic Bound

Traditional realtime service provides a hard or absolute bound on the delay of every packet; in the literature, this service model is called the *guaranteed* service. A *deterministic* guaranteed service provides for the worst-case requirements of flows. The worst-case requirements of flows are usually computed from parameterized models of traffic sources. The source models used for this computation may be very complex, but the underlying admission control principle is conceptually simple: does granting a new request for service cause the worst-case behavior of the network to violate any delay bound?

The admission control algorithms proposed in reference [KS85, KU93, OST88] require sources to provide peak rate characterization of their traffic. The algorithms then check that the sum of all peak rates is less than link capacity. If sources are willing to tolerate queueing delay, they can use a token bucket filter, instead of peak rate, to describe their traffic. The network ensures that the sum of all admitted flows' token rate is less than link bandwidth and the sum of all token bucket depths is less than available buffer space. This approach is proposed in [LV93]. In [ZF94], the authors presented an admission control algorithm for deterministic service based on calculation of maximum number of bits b^* that can arrive from a source during any interval τ[1]:

$$b^* = \min\left(\lceil(\tau \bmod \iota)p\rceil, \lceil\iota a\rceil\right) + \left\lceil\frac{\tau}{\iota}\right\rceil \lceil\iota a\rceil, \tag{2.1}$$

[1]We assume negligible packet size.

where ι is the averaging interval for a, the source's average rate, and p is the source's peak rate. Queueing delay per switch is then calculated as:

$$D^* = \frac{\max_{\forall \tau \geq 0} \left\{ \sum_{i=1}^{n} b_i^*(\tau) - \mu\tau \right\}}{\mu}, \tag{2.2}$$

μ being the link bandwidth. The admission control checks that D^* does not violate any delay bounds. This algorithm performs better than those requiring peak rate characterization and can achieve acceptable ($> 50\%$) link utilization when sources are not very bursty (peak-to-average ratio < 4) and the delay bound is not too tight (> 60 ms, per switch); when flows are bursty, however, deterministic service ultimately results in low utilization. In references [Gol91, RD91], the authors propose reshaping users' traffic according to network resources available at call setup time. While reshaping users' traffic according to available resources may increase network utilization, the reshaped traffic may not meet users' end-to-end quality requirements. Instead of imposing a traffic shaper at call setup time, authors of references [GKT95, WKZL96] propose characterizing different segments of a realtime stream and renegotiating the flow's resource reservation prior to the transmission of each segment. Renegotiation failure results in traffic from the next segment to be reshaped according to reservations already in place for the flow. This scheme may be applicable to video-on-demand application where the entire data stream is available for *a priori* characterization prior to transmission.

2.2 Probabilistic Bound: Equivalent Bandwidth

Statistical multiplexing is the interleaving of packets from different sources where the instantaneous degree of multiplexing is determined by the statistical characteristics of the sources. Contrast this to slotted Time Division Multiplexing (TDM), for example, where packets from a source is served for a certain duration at fixed intervals and the degree of multiplexing is fixed by the number of sources that can be fited into an interval. The probability density function (*pdf*) of statistically multiplexed independent sources is the convolution of the individual *pdf*'s; and the probability

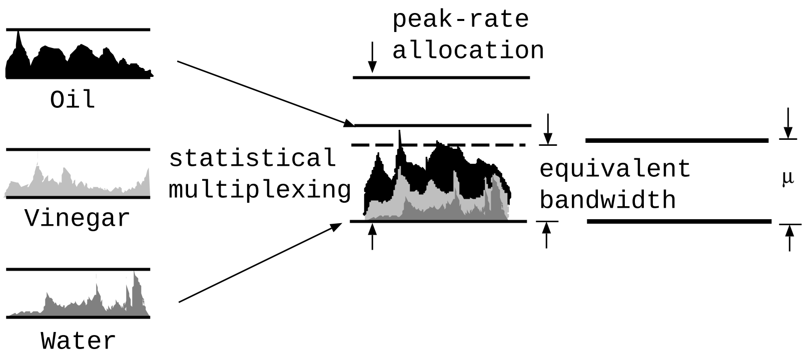


Figure 2.2: Statistical multiplexing of three flows and their equivalent bandwidth.

that the aggregate traffic will reach the sum of the peak rates is infinitesimally small (1e-48), much smaller than the loss characteristics of physical links. ATM network, for example, has a loss probability of 1e-8—in which case, guaranteeing a 1e-9 loss rate at the upper layer is sufficient [VPV88]. Hence networks that support statistical multiplexing can achieve higher level of utilization without sacrificing much on quality of service.

Probabilistic guaranteed service exploits this statistical observation and does *not* provide for the worst-case, sum of peak rates, scenario. Instead, using the statistical characterization of the current and incoming traffic it guarantees a bound on the probability of lost packets:

$$\epsilon > \text{Prob(aggregate traffic} > \text{available bandwidth)}, \tag{2.3}$$

where ϵ is the desired loss rate. In environments where the available bandwidth is a portion of link capacity alloted to realtime traffic and realtime traffic is allowed to use the remaining bandwidth during overflow, ϵ simply bounds the overflow rate. We do not make the distinction between loss rate and overflow rate in the remainder of this dissertation. The aggregate traffic of the statistically multiplexed sources is called the *equivalent bandwidth* (or *effective bandwidth* or *equivalent capacity*) of the sources [WKFR90, Rob93]. In Fig. 2.2 we show the statistical multiplexing of the three flows, *water, vinegar*, and *oil*, and their bandwidth requirements according to peak rate allocation and equivalent bandwidth. Using the equivalent bandwidth method to compute the bandwidth requirement of the three flows, we decide that they can be served by a link with capacity μ. Whereas if we consider only peak rate allocation, we would not have admitted all three flows into the network. For switches with buffer, the probabilistic bound can be formulated as:

$$\epsilon > \text{Prob}((\text{aggregate traffic - available bandwidth})\ \tau > \text{buffer}), \tag{2.4}$$

where τ is a time interval.

We now look at the different approaches used to compute equivalent bandwidth. Let $X_{i,t}$ be the instantaneous arrival rate of flow i at time t. Assume that $X_{i,t}$'s are independent, identically distributed. Let $S_t = \sum_{i=1}^{n} X_{i,t}$ be the instantaneous arrival rate of n flows. We want S_t such that:

$$\epsilon > \text{Prob}(S_t > \mu) \tag{2.5}$$

where μ is the link bandwidth. S_t can be computed directly from aggregate traffic, or by summing up $X_{i,t}, i = 1 \ldots n$. If the switch has buffer, we can define $X_{i,\tau}$ to be the instantaneous arrival rate of flow i during time period τ. $X_{i,\tau}$'s are again assumed to be independent, identically distributed. Let $S_\tau = \sum_{i=1}^{n} X_{i,\tau}$ be the instantaneous arrival rate of n flows. And we want S_τ such that:

$$\epsilon > \text{Prob}((S_\tau - \mu)\tau > B), \tag{2.6}$$

where B is the buffer size.

Bernoulli Trials. In references [RS90, SS91], the authors model $X_{i,\tau}$ as Bernoulli random variables. The aggregate arrival rate is then the convolution of the Bernoulli variables. The bound on buffer overflow probability over τ can be calculated as:

$$\epsilon > \frac{\sum_{\kappa=0}^{\infty}(\kappa - \mu\tau)\varrho_1^* \star \cdots \star \varrho_n^*(\kappa)}{\sum_{\kappa=0}^{\infty} \kappa \varrho_1^* \star \cdots \star \varrho_n^*(\kappa)} \tag{2.7}$$

where κ is the number of bits, $\varrho_j^*(\kappa)$ is the probability that κ bits arrived from source j, $\star$ denotes convolution, and

$$\varrho_j^*(\kappa) = \begin{cases} \rho & \text{if} \quad \kappa = \mu\tau, \\ 1-\rho & \text{if} \quad \kappa = 0, \\ 0 & \text{otherwise.} \end{cases} \tag{2.8}$$

Binomial Distribution. The number of arrivals in a sequence of Bernoulli trials has a binomial distribution. Assuming sources are homogeneous two-state Markov processes, the convolution in Eqn. 2.7 reduces to a binomial distribution [KS89, RSKJ91, MSST91]. The bound on buffer overflow probability becomes:

$$\epsilon > \frac{\sum_{i=\mu/C+1}^{n} \varrho(i)(iC-\mu)}{nm} \tag{2.9}$$

where $\varrho(i)$ is the binomially distributed probability that i sources are active:

$$\varrho(i) = \binom{n}{i} \rho^i (1-\rho)^{n-i}, \tag{2.10}$$

and ρ is the probability of success in a Bernoulli trial. This computation results in overestimation of actual bandwidth for sources with short burst period because the buffer allows short bursts to be smoothed out and the approximation does not take this smoothing effect into account [GAN91]. In [ZK94], instead of a single binomial random variable, the authors used a *family* of time-interval-dependent binomial random variables, i.e. associated with each time interval is a binomial random variable that is stochastically larger than the actual bit rate generated. This method of modeling bit rate was first proposed by the author of reference [Kur92]. It allows a tighter bound on S_τ. The main drawback of modeling S_τ with binomial distribution is the cost of convoluting the arrival probabilities of heterogeneous sources. In [ZK94], for example, the authors suggest using the Fast Fourier Transform (FFT) to calculate the convolution. FFT has a complexity of $\Theta(n\widehat{B}\log\widehat{B})$, where n is the number of sources and $\widehat{B}$ the size of the largest burst from any source. Furthermore, when the number of sources multiplexed is small, this approximation of equivalent bandwidth underestimates the actual requirement [RSKJ91].

Fluid-flow Approximation. A fluid-flow model characterizes traffic as a Markov modulated continuous stream of bits with peak and mean rates. Let $\widehat{c}$ be the equivalent bandwidth of a source, as seen by a switch, computed using the fluid-flow approximation. In [GAN91], $\widehat{c}$ is computed using:

$$\widehat{c} \simeq \frac{\alpha n(1-\rho)p - B + \sqrt{[\alpha n(1-\rho)p - B]^2 + 4B\alpha n\rho(1-\rho)p}}{2\alpha n(1-\rho)}, \tag{2.11}$$

where ρ is the source's utilization (average/peak), p the source's peak rate, $\alpha = \ln(1/\epsilon)$, and B the switch's buffer size. This approximation assumes flows are not very bursty and have short average burst period.[2] When flows do not conform to this assumption the bandwidth requirement is overestimated [GAN91]. Equivalent bandwidth for more general source models have also be computed; see for example references [AMS82, Mit88, EM93, Kel91, KWC93, MP90]. Computing the equivalent bandwidth of a source using this method depends only on the flow's fluid-flow characteristics and not on the number nor characteristics of other existing flows. The computation of equivalent bandwidth for general sources, however, is computationally expensive ($O(n^3)$ is quoted in [Mit88], where n is the number of sources).

Gaussian Distribution. In references [VPV88, LPP90, GAN91, Sai92, AS94, SRLL94], S_t is approximated with a Gaussian distribution. Let $\widehat{C}_G$ be the equivalent capacity of the aggregate traffic. Given a desired loss rate, reference [GAN91] computes C_G using:

$$C_G = \nu + \alpha'\sigma \tag{2.12}$$

where, $\alpha' = \sqrt{-2\ln(\epsilon) - \ln(2\pi)}$, $\nu = \sum_{i=1}^{n} a_i$, and $\sigma^2 = \sum_{i=1}^{n} \sigma_i^2$, where ν and σ^2 are the average and variance of the aggregate traffic respectively and a_i and σ_i^2 are

[2] The length of a burst is measured relative to the buffer size of a switch.

those of each source. Hence the C_G bounds the right tail of the Gaussian distribution. This approximation tracks the actual bandwidth requirement well when there is a large number of sources (e.g. more than 10 homogeneous sources) with long burst period. When only a small number of sources are multiplexed, this approximation overestimates the required bandwidth. It also overestimates required bandwidth when sources have short bursts because short bursts are smoothed out by the switch buffer and the approximation does not take this into account. The authors of reference [GG93] use the minimum of the fluid-flow and Gaussian approximations, $\widehat{C} = \min\{\nu + \alpha'\sigma, \sum_{i=1}^{n} \widehat{c}_i\}$, in making admission control decisions.

Large Deviation Approximation. Originally proposed by the author of reference [Hui88], an approximation based on the theory of large deviation was later generalized in reference [Kel91] to handle resource with buffer. The theory of large deviation bounds the probability of rare events occuring. In this case, the rare event is $S_t > \mu$. The approximationn in references [Hui88, Kel91] are based on the Chernoff's bound, while the one in [Flo96a] is based on the Hoeffding's bound. The Hoeffding's bound does not require that $X_{i,t}$ be independent of $X_{i,t+\delta}$. Equivalent bandwidth computed using the Hoeffding's bound is given by:

$$C_H(\nu, \{p_i\}_{1 \leq i \leq n}, \epsilon) = \nu + \sqrt{\frac{ln(1/\epsilon)\sum_{i=1}^{n}(p_i)^2}{2}}, \tag{2.13}$$

where ν is the average arrival rate of the aggregate traffic and p_i source i's peak rate. Further approaches to admission control based on the theory of large deviation are presented in references [dVKW95, CT95, EMW95].

Poisson Distribution. The above approximations of equivalent bandwidth all assume high enough degree of statistical multiplexing. When the degree of statistical multiplexing is low, or when buffer space is small, approximations based on Gaussian distribution and theory of large deviation overestimate required bandwidth [GAN91, AS94, Flo96a], while approximations using both fluid-flow characterization and binomial distribution underestimate it [Fil89, NRSV91, RSKJ91]. In such cases, the authors of references [RSKJ91, Fil89] suggest calculating equivalent bandwidth by solving for an $M/D/1/B$ queue, assuming Poisson arrivals.

Measurement-based. Each approach to compute equivalent bandwidth above can be approximated by using measurement to determine the values of some of the parameters used. In reference [SS91], the authors proposes measuring the convolution of arrival probabilities used in Eqn. 2.7 instead of computing them:

$$\epsilon > \frac{\sum_{k=0}^{\infty}(k-\mu t)\widehat{\varrho}\star\varrho_{n+1}^{*}(k)}{\sum_{k=0}^{\infty}k\widehat{\varrho}\star\varrho_{n+1}^{*}(k)} \tag{2.14}$$

where $\widehat{\varrho}$ is the measured arrival probabilities of existing traffic and ϱ_{n+1}^{*} the arrival probability of the prospective source. The authors of reference [dVKW95] propose measuring the effective bandwidth of each source; while the authors of references [AS94, DJM96] propose measuring the mean and variance of traffic, assuming it has a Gaussian distribution. Given the unreliable nature of measurement, the authors of reference [DJM96] further provide an estimate of the measurement errors. In reference [Flo96a], the author proposes using measured arrival rates in the computation of equivalent bandwidth based on the Hoeffding's bound:

$$\widehat{C}_H(\widehat{\nu},\{p_i\}_{1\le i\le n},\epsilon) = \widehat{\nu} + \sqrt{\frac{ln(1/\epsilon)\sum_{i=1}^{n}(p_i)^2}{2}}. \tag{2.15}$$

In reference [LCH95], the authors propose measuring the spectral density of traffic. Bandwidth is provisioned according to the low frequency of traffic and buffer space according to the high frequency. The authors further suggest using a resource renegotiation method similar to the one mentioned in Section 2.1 to increase network utilization. This approach is appealing, however it is not clear what should the cut-off frequency be and how traffic spectral density can be computed on-line.

Two other methods to estimate equivalent bandwidth are specifically suited to measurement-based approach. The first is based on the Bayesian Estimation method. From a given initial load, and a set of recursive equations, one can estimate future load from successive measurements. This approach is presented in references

[WCKG94, GKK95]. The authors of reference [WCKG94] further describe a hardware implementation of the measurement mechanism. The second method is a table-driven method. An *admissible region* is a region of space within which service commitments are satisfied. The space is defined by the number of admitted flows from a finite set of flow types. The first approach to compute an admissible region uses simulation [DTVV90, HLP93, DLM93]. For a given number of flows from each flow type, simulate how many more flows of each type can be admitted without violating service commitments. Running such simulations repeatedly with a different set of initial flow mix, one eventually maps out the admissible region for the given flow types. The admissible region is encoded as a table and down-loaded to the switches. When a prospective flow makes a reservation, the admission control algorithm looks up the table to determine whether admittance of this flow will cause the network to operate outside the admissible region; if not, the flow is admitted. The major drawbacks of this method for doing admission control are: (1) it supports only a finite number of flow types, and (2) the simulation process can be computationally intensive. The authors of reference [GKK95] use a Bayesian method to pre-compute an admissible region for a set of flow types. The admissible threshold is chosen to maximize the reward of increased utilization against the penalty of lost packet. The computation assumes knowledge of link bandwidth, size of switch buffer space, flows' token bucket filter parameters, flows' burstiness, and the desired loss rate; it also assumes Poisson call arrival process and independent, exponentially distributed call holding times. However, the authors of [GKK95] claim that this algorithm is robust against fluctuations in the value of the assumed parameters. The measurement-based version of this algorithm ensures that the measured instantaneous load plus the peak rate of a new flow is below the admissible region. The authors of references [Hir91, CLG95] use a neural-network to learn the admissible region for a given set of flow types. In Chapter 10 we present a comparative study of a couple of the measurement-based admission control algorithms presented in this section next to our own.

Chapter 3

Relaxed Realtime Services

Traditional realtime service models have been designed on two assumptions: first, traffic sources can be well characterized by Markov chains, and second, receivers require rigid delay bound. Recent studies show that network traffic exhibits long-range dependence, a phenomenon not consistent with the first assumption. Long-range dependence has been observed in ISDN traffic [MH+91], telephone traffic [DMRW94], local-area ethernet traffic [LTWW94], wide-area Internet traffic [PF94, KM94], world-wide-web traffic [CB96], video traffic [BSTW95, GW94], and audio traffic [Flo96a]. Long-range dependent traffic has been shown to effect queue behavior and loss rate [LW91, ENW96, GB96] and its cause has been traced back to source processes with heavy-tailed ON and/or OFF times distributions [WTSW95, PKC94, Flo96a]. As to the second assumption, recent realtime applications specifically designed for the Internet, such as the *vat, ivs, nv*, and *vic* teleconferencing programs, can buffer received packets and adjust their playback point to adapt to experienced delay. Given such *adaptive playback* applications, the network is not required to provide absolute delay bound. This relaxation of delay bound enables the network to further increase utilization. Recognizing the above two trends and the heterogeneous and decentralized nature of the Internet, several "relaxed" realtime service models have been proposed in the literature. We review them in the remainder of this chapter.

3.1 Against Delay Bound

There is only one service model available on the current Internet: the best-effort service model. Under this model, neither packet delivery time nor loss rate is bounded. Some researchers believe that there will be such abundance of network bandwidth in the future that this service model will be sufficient to support realtime traffic. Without adding extra mechanism that will only slow down packet transmission, the network can attain high level of utilization by admitting all offered flows; users dissatisfied with network performance can leave the network, resulting in better performance for those who remain [WC93]. Where network bandwidth is not in abundance, applications should be written to adapt to available bandwidth. Applications that can gracefully adapt to heterogeneous environment are more robust and will survive those that cannot [Hui95]. Aside from the ability of video sources to adapt their compression ratio to available bandwidth [Cha86, GG91, YH91, GV93, WC93, KMR93, VC94], video sources can also be hierarchically encoded into separate levels. In references [MJV96, HS96], each level of the hierarchically encoded data is transmitted as a separate flow under the control of receivers. Depending on available bandwidth and receiver interest, more or fewer levels are actually transmitted. The ability of applications to adapt to available bandwidth thus further obviate the need for the network to guarantee a delay bound.

Next we question the meaning of a delay bound. Given the heterogeneous, decentralized, and non-deterministic nature of the Internet, is it even realistic to expect the network to guarantee a delay bound? If a flow is routed through a portion of the Internet that does not support delay bound, the guarantees provided by the rest of the path becomes contractually unenforceable. Such routing could happen either at flow setup time or during the flow's lifetime. Furthermore, how should the delay bound at each switch be chosen? Assuming "appropriate" per-hop bounds, one still has to determine the end-to-end delay bound; is this to be a simple sum of the per-hop bounds? Finally, if network traffic is indeed long-range dependent, traffic tides happen when sources burst simultaneously; at which time, the most effective control is to ensure sufficient bandwidth, lack of which will result in buffer overflow, for any reasonable buffer sizes. The purpose of buffer space in switches is merely to hold packets that arrive simultaneously because they were "jostled" a bit in upstream switches, not to reshape traffic tide [Flo96a, LCH95]. Except on networks where flows are isolated from each other, by means of a weighted fair queue for example, and buffer space is allocated for worst-case requirements, delay bound is inherently unenforceable.

Related to the question of how to provision for long-range dependent traffic, is how to choose the token bucket filter parameters to characterize a source. If long-range dependence in aggregate traffic is caused by ON/OFF sources with infinite variance ON times, for all bucket depths one chooses, there is a longer ON time than what the chosen bucket depth can accommodate. Furthermore, unless one is willing to tolerate long queues at the token bucket filters, flows must be assigned token rates very close to their peak rates. If flows are reserved bandwidth close to their peak rates, there will not be long queueing delay anyway.

3.2 Unadvertised Bound

While assuming infinite bandwith gives us a simple network architecture that is very appealing, it is still to be determined whether there really will be such abundance of bandwidth in the future. A more conservative version of the above model allow users to reserve a minimum bandwidth for each flow; traffic exceeding this minimum rate competes for the remaining capacity. The admission control algorithm checks that the sum of bandwidth requested does not exceed link capacity. Each flow may increase its transmission rate until it receives congestion feedback from the network. Upon congestion, the flow throttles back its offered load accordingly, down to the minimum bandwidth reserved [KMR93, WC93]. Alternatively, one could make reservations to ensure that the base level of an hierarchically encoded stream will be successfully transmitted. The service models discussed in this section recognizes the need of even adaptive applications to sometimes have some resources reserved. Conceding the difficulty of guaranteeing delay bound as raised in the previous section, however, they do not provide a contractually strict delay bound.

The *controlled-load* service model defined in reference [Wro95] "tightly approximates the behavior visible to applications receiving best-effort service *under unloaded conditions*" over the same path. Furthermore, applications requesting controlled-load service may assume that its packet loss rate is on the order of the transmission medium's error rate and that its typical experienced delay should be on the order of the path's transmission and propagation delays. More specifically, average packet queueing delay should be no greater than the flow's "burst time" and there should be minimal loss rate averaged over time-scales larger than "burst time"—where the "burst time" is defined as the time required to serve a flow's maximum burst at the flow's reserved rate. For a flow described by a token bucket filter, the "burst time" is b/r, where b is the token bucket depth and r its replenishment rate. Switches ensures adequate resources by doing admission control. While the specification of controlled-load service does not dictate specific quantitative values for service parameters such as delay bound or loss rate, operationally the admission control decisions must still be computed and evaluated based on meeting one or both of these constraints. In Chapter 10 we investigate five admission control algorithms that could support controlled-load service.

The *committed rate* service model described in reference [BGK96] provides resource reservation as in guaranteed service but without any delay or loss guarantee nor the ability to pre-compute end-to-end delay. Committed-rate differs from controlled-load service in that it allows for traffic policing and re-shaping, thereby more closely emulates a dedicated circuit.

3.3 Predictive Service

Adaptive playback applications do not require an absolute delay bound; however, they may still prefer an upper bound on the tail of their delay distributions. Even if the delay distribution has a very small median, it may be useful to some applications to have a bounded worst-case. A small dynamic range of a parameter should not be confused with the uselessness of that parameter. Furthermore, applications using the networks may have different levels of delay bound tolerance; by providing different levels of realtime service with order of magnitude difference in delay bound, a network can increase its portion of realtime traffic. While the heterogeneous and decentralized nature of the Internet does pose an implementation problem for services providing delay bound, predictive service can nevertheless be implemented on private internetworks or commercial portions of the Internet.

Unlike the controlled-load and committed-rate service models, predictive service offers a delay bound. Nevertheless, predictive service differs in two important ways from traditional guaranteed service: (1) the service commitment is somewhat less reliable, (2) while predictive service requires that sources be characterized by token bucket filters at admission time, the behavior of existing flows is determined by measurement rather than by *a priori* characterizations. It is important to keep these two differences distinct because while the first is commonplace, the second, i.e. the use of *measurement-based* admission control, is more novel. On the reliability of service commitment, we note that the definition of predictive service itself does not specify an acceptable level of delay violations. This is for two reasons. First, it is not particularly meaningful to specify a failure rate to a flow with a short duration [NK92]. Second, reliably ensuring that the failure rate never exceeds a particular level leads to the same worst-case calculations that predictive service was designed to avoid. Instead, the CSZ approach proposes that the level of reliability be a contractual matter between a network provider and its customers—not something specified on a per-flow basis. We presume that these contracts would only specify the level of violations over some large time scale (e.g. days or weeks) rather than over a few hundred packet times.[1] Hence the bound offered by predictive service is not a probabilistic bound. Probabilistic bounds, as discussed in Section 2.2, are based on the statistical characterizations of the traffic. Under probabilistic service, a flow can request any amount of bandwidth and thereby flexibly tune its resultant delay bound or statistical "loss" rate. In contrast, the delay bounds for predictive service are less flexible; each switch has a few predictive service classes which have pre-established target delay bounds.

[1] A network provider might promise to give its customers their money back if delay violations exceed some level over the duration of their flow, no matter how short the flow; however we contend that the provider cannot realistically assure that excessive violations will never occur.

These bounds will typically be chosen to be roughly an order of magnitude apart. Prospective flows can choose which class of predictive service they desire based on the delay bound they can tolerate. The validity of these bounds are assessed when making admission control decisions, based on actual measured characteristics of traffic, rather than the theoretical worst-case behavior. Since our measurement-based admission control algorithm does not rely on pre-existing measurements or computations, such as the ones in Section 2.2, predictive service is not limited to serve only a small and well-characterized set of traffic sources.

Chapter 4

Measurement-based Admission Control

Our admission control algorithm consists of two logically distinct aspects. The first aspect is the set of criteria controlling whether to admit a new flow; these are based on an approximate model of traffic flows and use measured quantities as inputs. The second aspect is the measurement process itself, which we will describe in Chapter 5. In this chapter we present the analytical underpinnings of our admission control criteria.

4.1 Framework

We have studied the behavior of our admission control algorithm mostly under the CSZ scheduling discipline [CSZ92], however we believe that the observations we made on our measurement-based admission control algorithm, and our methodology for studying such an algorithm, apply equally to other scheduling disciplines—for example, in Chapter 10 we apply our methodology and observations in studying several admission control algorithms for the controlled-load service model. While we believe that most future realtime applications written for asynchronous packet switched networks will be adaptive playback applications, we do not discount the need for guaranteed service. In the CSZ scheme, guaranteed service is provided by the weighted fair queueing (WFQ) algorithm described in [DKS89]—also known as the generalized processor sharing (GPS) algorithm in [PG93]. WFQ assigns a share of link capacity to each active flow; the admission control criterion is merely that the sum of the previously assigned bandwidths plus the bandwidth requested by the prospective flow does not exceed link capacity. The scheduling discipline for predictive service is a priority queue, as described in [CSZ92]; the scheduler attempts to minimize the maximal (minimax) delays actually experienced in each class, but does not guarantee an absolute maximum delay bound. Because of the minimax scheduler, we expect that for the same amount of bandwidth reserved, predictive service users will see lower delay than guaranteed service users. Under the CSZ model, a switch can support multiple levels of predictive service, each with its own delay bound. We envision that the delay bounds of different level of predictive service will be on the order of magnitude apart. In our scheme, the admission control algorithm at each switch enforces the queueing delay bound at that switch, assuming infinite buffer space. We leave the satisfaction of end-to-end delay requirements to the end systems. An end system could, for example, use adaptive source routing, such as the one proposed in reference [Bre95], to select a route that satisfies its end-to-end requirements. We also assume the existence of a reservation protocol, such as the one in [Z$^+$93], which the end systems could use to communicate their resource requirements to the network. We require that there be compelling incentives, such as quality of service based pricing (e.g. [CESZ93]), for users to always request the least costly quality of service satisfying their needs.

Sources requesting service must characterize the worst-case behavior of their flow. In [CSZ92] this characterization is done with a token bucket filter. A token bucket filter for a flow has two parameters: its token generation rate, r, and the depth of its bucket, b, i.e. no more than b tokens can be accumulated. Each token represents a single bit; sending a packet consumes as many tokens as there are bits in the packet. Without loss of generality, in this study we assume packets are of fixed size and that each token is worth a packet; sending a packet consumes one token. A flow is said to conform to its token bucket filter if no packet arrives when the token bucket is empty.

When the flow is idle or transmitting at a lower rate, tokens are accumulated up to b tokens. Thus flows that have been idle for a sufficiently long period of time can dump a whole bucket full of data back to back. For constant bit rate sources, one can set the token rate, r, to the peak traffic generation rate and let the bucket depth, b, be 1. In this case, the token-bucket filter precisely characterizes the traffic coming out of the sender. Many non-constant bit rate sources do not naturally conform to a token bucket filter with token rate less than their peak rates. The user, then, should pick a token bucket filter which looks like a reasonable upper bound on its behavior. It is conceivable that future real-time applications will have a module that can, over time, learn a suitable r and b to bound their traffic.

When admitting a new flow, not only must the admission control algorithm decide whether the flow can get the service requested, but it must also decide if admitting the flow will prevent the network from keeping its prior commitments. Let us assume, for the moment, that admission control cannot allow *any* delay violations. Then, the admission control algorithm must analyze the worst-case impact of the newly arriving flow on existing flows' queueing delay. However, with bursty sources, where the token bucket parameters are very conservative estimates of the average traffic, delays rarely approach these worst-case bounds. To achieve a fairly reliable bound that is less conservative, we approximate the maximal delay of predictive flows by replacing the worst-case parameters in the analytical models with measured quantities. We call this approximation the *equivalent token bucket filter*. This approximation yields a series of expressions for the expected maximal delay that would result from the admission of a new flow. As mentioned above, in the CSZ architecture, switches serve guaranteed traffic with the weighted fair queueing (WFQ) scheduling discipline and serve predictive traffic with priority queueing. Hence, the computation of worst-case queueing delay is different for guaranteed and predictive services. In this chapter, we will first look at the worst-case delay computation of predictive service, then that of guaranteed service. Following the worst-case delay computations, we present the equivalent token bucket filter. We close this chapter by presenting the details of the admission control algorithm based on the equivalent token bucket filter approximations.

4.2 Worst-case Delay: Predictive Service

To compute the effect of a new flow on existing predictive traffic, we first need a model for the worst-case delay of priority queues. Cruz, in [Cru91], derived a tight bound for the worst-case delay, D_j^*, of priority queue level j. Our derivation follows Parekh's [Par92], which is a simpler, but looser, bound for D_j^* that assumes small packet sizes, i.e. the transmission time of each packet is sufficiently small (as compared to other delays) and hence can be ignored. This assumption of small packet sizes further allows us to ignore delays caused by the lack of preemption. Further, we assume that the aggregate rate, aggregated over all traffic classes, is within the link capacity ($\sum r_j \leq \mu$).

Theorem 1 *Parekh [Par92]: The worst-case class j delay, with* FIFO *discipline within the class and assuming infinite peak rates for the sources, is*

$$D_j^* = \frac{\sum_{i=1}^{j} b_i}{\mu - \sum_{i=1}^{j-1} r_i} \tag{4.1}$$

for each class j. Further, this delay is achieved for a strict priority service discipline under which class j has the least priority.

Proof: Let j be the session with the lowest priority at the first switch from the source of j. Session j dumps a bucket full of data at time t_0. We first prove that the delay seen by the last bit of session j's bucket is Eqn. 4.1. We then prove that this delay is the worst-case delay.[1]

Case 1: All higher priority sessions also dump their bucket full of data at time t_0. Session j's queue is empty. After dumping their bucket full of data, the higher priority sessions continue sending at their respective token rate, $r_i, 0 < i < j-1$.

[1]Interested readers may also refer to [Par92], Theorem 2.4 for an alternate proof.

The first bit of data from session j's bucket will be served only after all the packets of higher priority sessions have been served. The number of accumulated higher priority packets is $\sum_{i=1}^{j-1} b_i$; since all the higher priority sessions continue sending at their respective rate after dumping their bucket full of packets, the bandwidth left to serve $\sum_{i=1}^{j-1} b_i$ is $\mu - \sum_{i=1}^{j-1} r_i$. Thus the first bit of session j's bucket will be served at time:

$$t_s = t_0 + \frac{\sum_{i=1}^{j-1} b_i}{\mu - \sum_{i=1}^{j-1} r_i} \tag{4.2}$$

To determine the queueing delay experienced by the last bit of session j's packet, we first determine how long it takes to drain b_j at the source. In the worst case scenario, session j has infinite amount of data to transmit. The bucket drain rate is then C_j, the transmission rate of session j's source. Since the bucket is replenished at rate r_j, it takes τ_j time to drain the bucket. Note that we have accounted for the bucket replenishment rate here.

During session j's bucket draining time, it sends $C_j \tau_j$ amount of data onto the network. It takes

$$\frac{C_j \tau_j}{\mu - \sum_{i=1}^{j-1} r_i} \tag{4.3}$$

time to serve this data. But since the last bit of session j's packet does not arrived until time τ_j, and $\mu > r_j$, the delay seen by the last bit of session j's packet is only affected by the size of the session's bucket size. Hence Eqn. 4.1.

We now prove that Eqn. 4.1 is the maximal delay seen by all session j's packets.

Case 2: If session j's queue is *not* empty at time t_0, Some higher priority session must have dumped their data before t_0. Since $r_j < \mu - \sum_{i=1}^{j-1} r_i$, session j's queue could not have been longer than the amount of higher priority packets served before t_0. Thus the first bit of session j's bucket will see service earlier than t_s of Eqn. 4.2.

Case 3: Some higher priority sessions dump their data before t_0, and session j's queue is empty at time t_0. The first bit of session j's bucket will again see service earlier than t_s of Eqn. 4.2.

Case 4: Some upper priority sessions dump their data after t_0. This case reduces trivially to the case when all of them dump their data before or at t_0, i.e. case 1 or 2 above.

$\square$

The theorem says that the delay bound for class j is the one-time delay burst that accrues if the aggregate bucket of all classes 1 through j flows are simultaneously dumped into the switch and all classes 1 through $j-1$ sources continue to send at their reserved rates.

We now use Eqn. 4.1 as the base equation to model the effect of admitting a new flow α on existing predictive traffic. First we approximate the traffic from all flows belonging to a predictive class j as a single flow conforming to a $(\nu_j,\ b_j)$ token bucket filter. A conservative value for ν_j would be the aggregate reserved rate of all flows belonging to class j. Next, we recognize that there are three instances when the computed worst-case delay of a predictive class can change: (1) when a flow of the same class is admitted, (2) when a flow of a higher priority class is admitted, and (3) when a guaranteed flow is admitted. The switch priority scheduling isolates higher priority ($< k$) classes from a new flow of class k, so their worst-case delay need not be re-evaluated when admitting a flow of class k. In the remainder of this chapter, we compute each of the three effects on predictive traffic individually. At the end of these computations, we will observe that admitting a higher priority predictive flow "does more harm" to lower priority predictive traffic than admitting either a guaranteed flow or a predictive flow of the same priority.

In the equations below, we denote newly computed delay bound by $D^{*'}$. We denote the sum of guaranteed flows' reservation by ν_G. The link bandwidth available for serving predictive traffic is the nominal link bandwidth minus those reserved by guaranteed flows: $\mu - \nu_G$.

1. Effect of new predictive flow α on same priority traffic. We can model the effect of admitting a new flow α of predictive class k by changing the class's

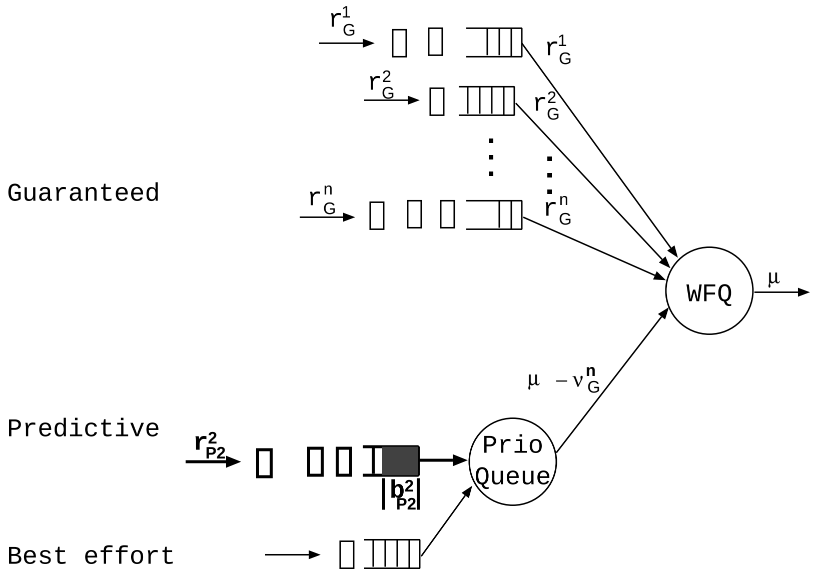


Figure 4.1: Effect of new predictive flow on same priority traffic.

token bucket parameters to $(\nu_k + r_k^\alpha,\ b_k + b_k^\alpha)$, where $(r_k^\alpha,\ b_k^\alpha)$ are the token bucket parameters of the new flow:

$$\begin{aligned} D_k^{*'} &= \frac{\sum_{i=1}^{k-1} b_i}{\mu - \nu_G - \sum_{i=1}^{k-1} \nu_i} + \frac{b_k + b_k^\alpha}{\mu - \nu_G - \sum_{i=1}^{k-1} \nu_i} \\ &= D_k^* + \frac{b_k^\alpha}{\mu - \nu_G - \sum_{i=1}^{k-1} \nu_i}. \end{aligned} \tag{4.4}$$

We see that the delay of class k grows by a term that is proportional to flow α's bucket size.

Fig. 4.1 depicts a CSZ scheduler with n number of guaranteed flows and one class 2 predictive flow. When the class 2 flow arrives, there is no other predictive flow in the system. Hence the worst-case delay seen by the new flow is the time it takes to drain the its bucket full of data (b_{P2}^2) at rate $\mu - \nu_G^n$, where ν_G^n is the sum of the n guaranteed flows' reserved rates. The new worst-case delay for class 2 is $D_{P2}^{*'} = b_{P2}^2/\mu - \nu_G^n$.

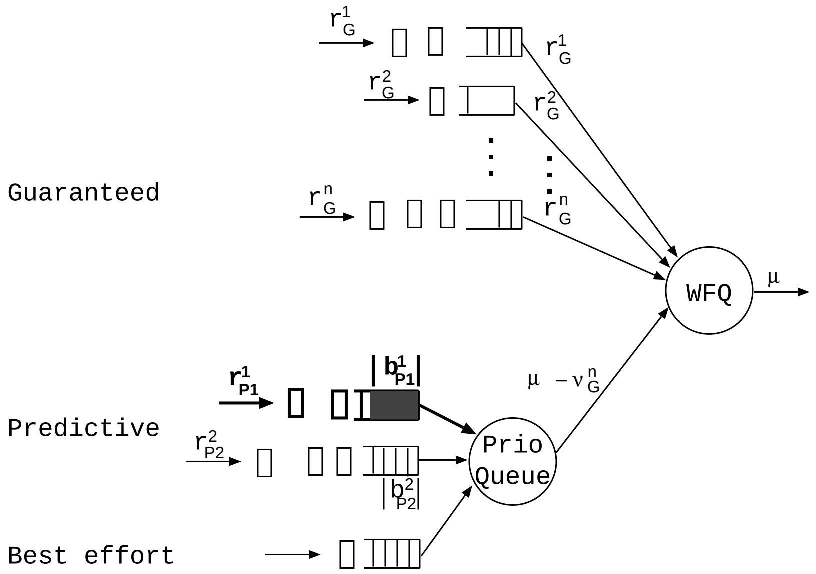


Figure 4.2: Effect of new predictive flow on lower priority traffic.

2. Effect of predictive flow α on lower priority traffic. We compute the new delay bound for class j, where j is greater than the requested class, k, directly from Eqn. 4.1, adding in the bucket depth b_k^α and reserved rate r_k^α of flow α.

$$\begin{aligned} D_j^{*'} &= \frac{\sum_{i=1}^{k-1} b_i + b_k + b_k^\alpha + \sum_{i=k+1}^{j} b_i}{\mu - \nu_G - \sum_{i=1}^{k-1} \nu_i - \nu_k - r_k^\alpha - \sum_{i=k+1}^{j-1} \nu_i} \\ &= D_j^* \frac{\mu - \nu_G - \sum_{i=1}^{j-1} \nu_i}{\mu - \nu_G - \sum_{i=1}^{j-1} \nu_i - r_k^\alpha} + \\ &\quad \frac{b_k^\alpha}{\mu - \nu_G - \sum_{i=1}^{j-1} \nu_i - r_k^\alpha}, \qquad k < j \leq K, \end{aligned} \tag{4.5}$$

where K is the number of predictive classes. The first term reflects a *squeezing* of the pipe, in that the additional bandwidth required by the new flow reduces the bandwidth available for lower priority flows. The second term is similar to the delay calculated above, and reflects the effect of the new flow's burstiness.

Fig. 4.2 adds a new class 1 flow to the system depicted in Fig. 4.1. The new flow is the highest priority predictive flow and there is no other flow of the same priority in service. The worst-case delay seen by this new flow is when it dumps its bucket full of data (b_{P1}^1). Maximal delay of class 1 is $D_{P1}^{*'} = b_{P1}^1/\mu - \nu_G^n$. Class 2 traffic must now wait for the class 1 queue to drain before it sees service. Hence the worst-case delay of class 2 traffic becomes:

$$D_{P2}^{*'} = D_{P2}^{*}\frac{\mu - \nu_G^n}{\mu - \nu_G^n - r_{P1}^1} + \frac{b_{P1}^1}{\mu - \nu_G^n - r_{P1}^1}. \tag{4.6}$$

3. Effect of a guaranteed flow α on predictive traffic. Again, we compute the new delay bound $D^{*'}$ for *all* predictive classes directly from Eqn. 4.1, adding in the reserved rate, r_G^α, of flow α.

$$\begin{aligned} D_j^{*'} &= \frac{\sum_{i=1}^{j} b_i}{\mu - \nu_G - \sum_{i=1}^{j-1}\nu_i - r_G^\alpha} \\ &= D_j^{*}\frac{\mu - \nu_G - \sum_{i=1}^{j-1}\nu_i}{\mu - \nu_G - \sum_{i=1}^{j-1}\nu_i - r_G^\alpha}, \qquad 1 \le j \le K. \end{aligned} \tag{4.7}$$

Notice how the new guaranteed flow simply squeezes the pipe, reducing the available bandwidth for predictive flows; new guaranteed flows do not contribute any delay due to their buckets because WFQ smooths out their bursts. Also observe that the first term of Eqn. 4.5 is equivalent to Eqn. 4.7: the impact of a new guaranteed flow is like adding a zero-size bucket, higher priority, predictive flow.

Fig. 4.3 shows a new guaranteed flow added to the system in Fig. 4.2. The new guaranteed flow does not effect existing guaranteed flows nor is it itself effected by any other flows. However, it does affect the bandwidth available to predictive traffic. The new maximal delay of class 1 and class 2 predictive services are respectively:

$$\begin{aligned} D_{P1}^{*'} &= D_{P1}^{*}\frac{\mu - \nu_G^n}{\mu - \nu_G^n - r_G^{n+1}}, \text{ and} \\ D_{P2}^{*'} &= D_{P2}^{*}\frac{\mu - \nu_G^n - r_{P1}^1}{\mu - \nu_G^n - r_G^{n+1} - r_{P1}^1}. \end{aligned} \tag{4.8}$$

Contrasting Eqns. 4.4, 4.5, and 4.7, we see that the experienced delay of lower priority predictive traffic increases more when a higher priority predictive flow is admitted than when a guaranteed flow or a same-priority predictive flow is admitted.

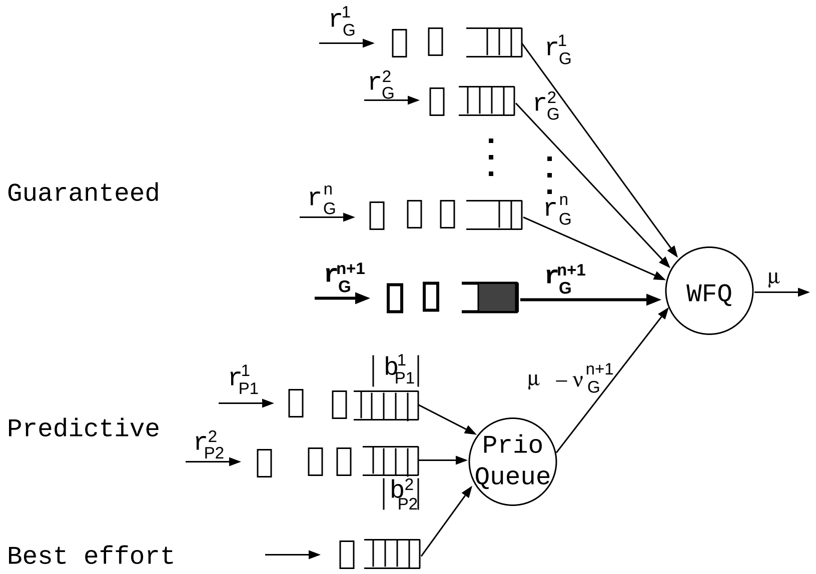


Figure 4.3: Effect of a guaranteed flow on predictive traffic.

The WFQ scheduler isolates predictive flows from attempts by guaranteed flows to dump their buckets into the network as bursts. In contrast, lower priority predictive traffic sees both the rates *and* the buckets of higher priority predictive flows. A higher priority predictive flow not only squeezes the pipe available to lower priority traffic, but also preempts it.

4.3 Worst-case Delay: Guaranteed Service

In reference [Par92], the author proves that in a network with arbitrary topology, the WFQ scheduling discipline provides guaranteed delay bounds that depend only on flows' reserved rates and bucket depths. Under WFQ, each guaranteed flow is isolated from the others. This isolation means that, as long as the total reserved rate of guaranteed flows is below the link bandwidth, new guaranteed flows cannot cause existing ones to miss their delay bounds. Hence, when accepting a new guaranteed flow, our admission control algorithm only needs to assure that (1) the new flow will not cause predictive flows to miss *their* delay bound (see Eqn. 4.7 above), and that (2) it will not over-subscribe the link: $\nu_G + r_G^\alpha \leq \upsilon\mu$, where μ is the link bandwidth and

υ is the utilization target (see Chapter 5.2 for a discussion on utilization target). In addition to protecting guaranteed flows from each other, WFQ also isolates (protects) guaranteed flows from all predictive traffic.

4.4 Equivalent Token Bucket Filter

The equations above describe the aggregate traffic of each predictive class with a single token bucket filter. How do we determine a class's token bucket parameters? A completely conservative approach would be to make them the sum of the parameters of all the constituent flows; when data sources are bursty and flows declare conservative parameters that cover their worst-case bursts, using the sum of declared parameters will result in low link utilization. Our algorithm is approximate and optimistic: we take advantage of statistical multiplexing by using measured values, instead of providing for the worst possible case, to gain higher utilization, risking that some packets may occasionally miss their delay bounds. In essence, we describe existing aggregate traffic of each predictive class with an *equivalent token bucket filter* with parameters determined from traffic measurement.

The equations above can be equally described in terms of current delays and usage rates as in bucket depths and usage rates. Since it is easier to measure delays than to measure bucket depths, we do the former. Thus, the measured values for a predictive class j are the aggregate bandwidth utilization of the class, $\widehat{\nu}_j$, and the experienced packet queueing delay for that class, $\widehat{D}_j$. For guaranteed service, we count the sum of all reserved rates, ν_G, and we measure the actual bandwidth utilization, $\widehat{\nu}_G$, of all guaranteed flows. Our approximation is based on substituting, in the above equations, the measured rates $\widehat{\nu}_j$ and $\widehat{\nu}_G$ for the reserved rates, and substituting the measured delays $\widehat{D}_j, j = 1 \ldots K$ for the maximal delays. We now use the previous computations and these measured values to formulate an admission control algorithm.

4.5 The Admission Control Algorithm

New Predictive Flow. If an incoming flow α requests service at predictive class k, the admission control algorithm:

1. Denies the request if the sum of the flow's requested rate, r_k^α, and current usage would exceed target link utilization:

$$\upsilon\mu > r_k^\alpha + \widehat{\nu}_G + \sum_{i=1}^{N} \widehat{\nu}_i, \tag{4.9}$$

2. Denies the request if admitting the new flow could violate the delay bound, D_k, of the same priority level:

$$D_k > \widehat{D}_k + \frac{b_k^\alpha}{\mu - \widehat{\nu}_G - \sum_{i=1}^{k-1} \widehat{\nu}_i}, \tag{4.10}$$

   or could cause violation of lower priority classes' delay bound, D_j:

$$\begin{aligned} D_j \quad > \quad & \widehat{D}_j \frac{\mu - \widehat{\nu}_G - \sum_{i=1}^{j-1} \widehat{\nu}_i}{\mu - \widehat{\nu}_G - \sum_{i=1}^{j-1} \widehat{\nu}_i - r_k^\alpha} + \\ & \frac{b_k^\alpha}{\mu - \widehat{\nu}_G - \sum_{i=1}^{j-1} \widehat{\nu}_i - r_k^\alpha}, \qquad k < j \leq K. \end{aligned} \tag{4.11}$$

New Guaranteed Flow. If an incoming flow α requests guaranteed service, the admission control algorithm:

1. Denies the request if either the bandwidth check in Eqn. 4.9 fails or if the reserved bandwidth of all guaranteed flows exceeds target link utilization:

$$\upsilon\mu > r_G^\alpha + \nu_G. \tag{4.12}$$

2. Denies the request if the delay bounds of predictive classes can be violated when the bandwidth available for predictive service is decreased by the new request:

$$D_j > \widehat{D}_j \frac{\mu - \widehat{\nu}_G - \sum_{i=1}^{j-1} \widehat{\nu}_i}{\mu - \widehat{\nu}_G - \sum_{i=1}^{j-1} \widehat{\nu}_i - r_G^{\alpha}}, \qquad 1 \leq j \leq K. \tag{4.13}$$

If the request satisfies all of these inequalities, the new flow is admitted.

Chapter 5

A Simple Time-window Measurement Mechanism

The formulae described in the previous chapter rely on the measured values $\widehat{D}_j$, $\hat{\nu}_G$, and $\hat{\nu}_j$ as inputs. We describe in this chapter the time-window measurement mechanism we use to measure these quantities. While we believe our admission control equations have some fundamental principles underlying them, we make no such claim for the measurement process. Our mechanism is extremely simple and could be replaced by a number of other approaches. We consider the simplicity of our approach an advantage in our study because it helps us isolate properties inherent to our admission control criteria from those induced by the measurement mechanism. Our measurement process uses the constants λ, S, and T; discussion of their roles as performance tuning knobs follows our description of the measurement process.

5.1 Measurement Process

We take two measurements: experienced delay and utilization. To estimate delay, we measure the queueing delay $\hat{d}$ of every packet. To estimate utilization, we sample the usage rate of guaranteed service, $\hat{\nu}_G^S$, and of each predictive class j, $\hat{\nu}_j^S$, over a sampling period of length S packet transmission units. Following we describe how

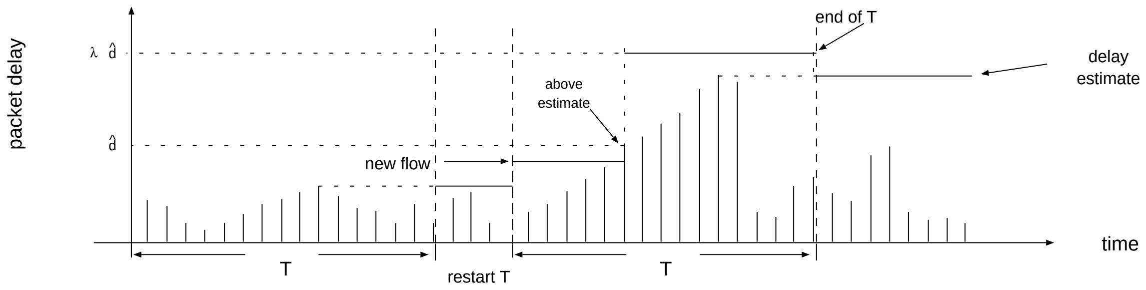


Figure 5.1: Measuring delay.

these measurements are used to compute the estimated maximal delay $\widehat{D_j}$ and the estimated utilization $\widehat{\nu}_G$ and $\widehat{\nu}_j$.

Measuring delay. The measurement variable $\widehat{D_j}$ tracks the estimated maximum queueing delay for class j. We use a measurement window of T packet transmission units as our basic measurement block. As shown in Fig. 5.1, the value of $\widehat{D_j}$ is updated on three occasions. At the end of the measurement block, we update $\widehat{D_j}$ to reflect the maximal packet delay seen in the previous block. Whenever an individual delay measurement exceeds this estimated maximum queueing delay, we know our estimate is wrong and immediately update $\widehat{D_j}$ to be λ times this sampled delay. The parameter λ allows us to be more conservative by increasing $\widehat{D_j}$ to a value higher than the actual sampled delay. Finally, we update $\widehat{D_j}$ whenever a new flow is admitted, to the value of projected delay from our admission control equations. Algebraically, the updating of $\widehat{D_j}$ is as follows:

$$\widehat{D}'_j = \begin{cases} MAX(\widehat{d}), & \text{of past T measurement window,} \\ \lambda\widehat{d}, & \text{if } \widehat{d} > \widehat{D}_j, \\ \text{Right side of} & \text{when adding a new flow, depending} \\ \text{Eqn. 4.10, 4.11,} & \text{on the service and class requested by} \\ \text{or 4.13,} & \text{the flow.} \end{cases} \tag{5.1}$$

Measuring rate. The measurement variables $\widehat{\nu}_G$ and $\widehat{\nu}_j$ track the highest sampled aggregate rate of guaranteed flows and each predictive class respectively (heretofore,

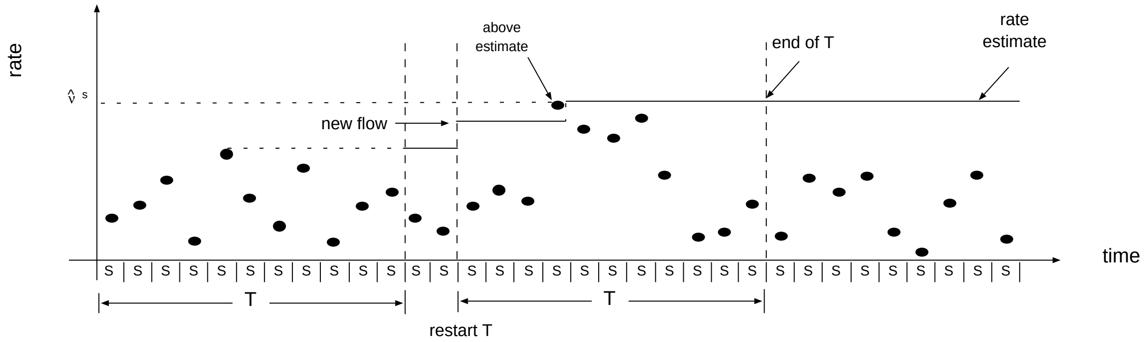


Figure 5.2: Measuring rate.

we will use "$\widehat{\nu}$" as a shorthand for "$\widehat{\nu}_G$ and/or $\widehat{\nu}_j$," and "$\widehat{\nu}^S$" for "$\widehat{\nu}_G^S$ and/or $\widehat{\nu}_j^S$.") As shown in Fig. 5.2, the value of $\widehat{\nu}$ is updated on three occasions. At the end of the measurement block, we update $\widehat{\nu}$ to reflect the maximal sampled utilization seen in the previous block. Whenever an individual utilization measurement exceeds $\widehat{\nu}$, we immediately update $\widehat{\nu}$ with the new sampled value. Finally, we update $\widehat{\nu}$ upon admission of new flows. Algebraically, the updating of $\widehat{\nu}$ is as follows:

$$\widehat{\nu}' = \begin{cases} MAX(\widehat{\nu}^S), & \text{of past } T \text{ measurement window,} \\ \widehat{\nu}^S, & \text{if } \widehat{\nu}^S > \widehat{\nu}\text{, where } \widehat{\nu}^S \text{ is the average rate} \\ & \text{over } S \text{ averaging period,} \\ \widehat{\nu} + r^\alpha, & \text{when adding a new flow } \alpha. \end{cases} \tag{5.2}$$

The measured rate of guaranteed traffic is capped at the sum of guaranteed reserved rate ($\widehat{\nu}'_G = MIN(\widehat{\nu}_G, \nu_G)$).

When a flow leaves the network, we do not explicitly adjust the measured values; instead we allow the measurement mechanism to adapt to the observed traffic automatically. We do, however, subtract the reserved rate of a departing guaranteed flow from the sum of all guaranteed reserved rate, ν_G.

5.2 Performance Tuning Knobs

We now look at the constants used in the algorithm.

υ: In a simple $M/M/1$ queue, the variance in delay diverges as the system approaches full utilization. A measurement-based approach is doomed to fail when delay variations are exceedingly large, which will occur at very high utilization. It is thus necessary to identify a *utilization target* and require that the admission control algorithm strive to keep link utilization below this level.

The appropriate utilization target of any given link depends on the characteristics of the traffic flowing through it. If each source's rate is small compared to link capacity (small grain size) and bursts are short, the link's utilization target can be set higher. Bursty sources with big, long bursts or long idle periods will require a lower link utilization target. In this study, we set utilization target at 90% capacity.

λ: In our simulations, a single instance of packet delay above the current estimate typically indicate that subsequent delays are likely to be even larger; so when a packet's queueing delay, $\widehat{d}$, is higher than its class's estimated maximal delay $\widehat{D_j}$, we back off our delay estimate to a much larger value, $\lambda\widehat{d}$. In this study, we use $\lambda = 2$.

S: The averaging period S in Eqn. 5.2 controls the sensitivity of our rate measurement. The smaller the averaging period, the more sensitive we are to bursts; the larger the averaging period, the smoother traffic appears. An S that captures individual bursts may make the admission control more conservative than desired. In this study we use S of at least 100 packet transmission times.

T: Once $\widehat{D}$ or $\widehat{\nu}$ is increased, their values stay high until the end of their respective measurement window T. The size of T controls the adaptability of our measurement mechanism to drops in traffic load. Smaller T means more adaptability, but larger T results in greater stability. The window size for load measurement should allow for enough utilization samples, i.e. T should be several times S. The measurement windows of the load and the delay can be maintained independently. When we admit a new flow and add its worst case effect to the measured values, we also restart the measurement windows to give the measurement mechanism one whole window to gather information on the new flow.

Chapter 6

Simulations

Admission control algorithms for guaranteed service can be verified by formal proof. Measurement based admission control algorithms can only be verified through experiments on either real networks or a simulator. We have tested our algorithm through simulations on a wide variety of network topologies and driven with various source models; we describe a few of these simulations in the following chapters. In each case, we were able to achieve a reasonable degree of utilization (when compared to guaranteed service) and a low delay bound violation rate (we try to be very conservative here and always aim for *no* delay bound violation over the course of all our simulations). Before we present the results from our simulations, we first present the topologies and source models used in these simulations.

6.1 Simulated Topologies

We run our simulations on four topologies: the One-Link, Two-Link, Four-Link, and TBone topologies depicted in Figs. 6.1(a), (b), (c), and 6.2 respectively. In the first three topologies, each host is connected to a switch by an infinite bandwidth link. The connection between switches in these three topologies are all 10 Mbps links, with infinite buffers. In the One-Link topology, traffic flows from HostA to HostB. In the Two-Link case, traffic flows between three host pairs (in source–destination order): HostA–HostB, HostB–HostC, HostA–HostC. Flows are assigned to one of

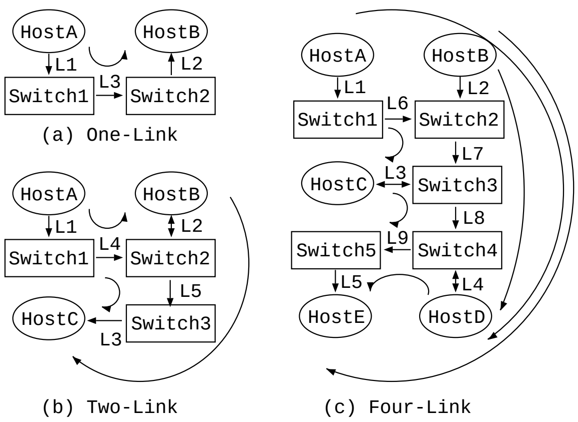


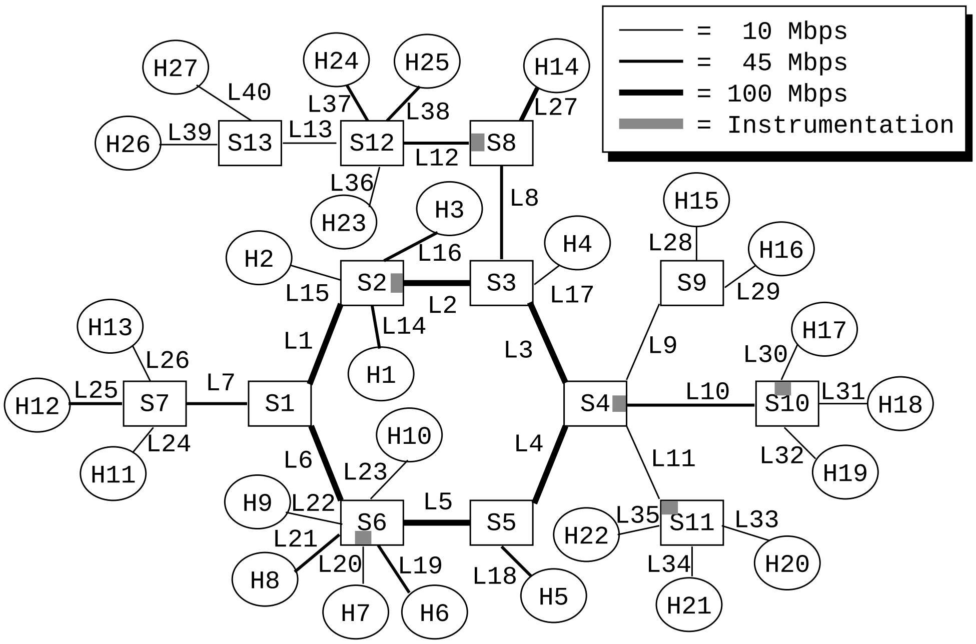
Figure 6.1: The ONE-LINK, TWO-LINK and FOUR-LINK topologies

these three host pairs with uniform probability. In the FOUR-LINK topologies, traffic flows between six host pairs: HostA–HostC, HostB–HostD, HostC–HostE, HostA–HostD, HostB–HostE, HostD–HostE; again, flows are distributed among the six host pairs with uniform probability. In Fig. 6.1, these host pairs and the paths their packets traverse are indicated by the directed curve lines.

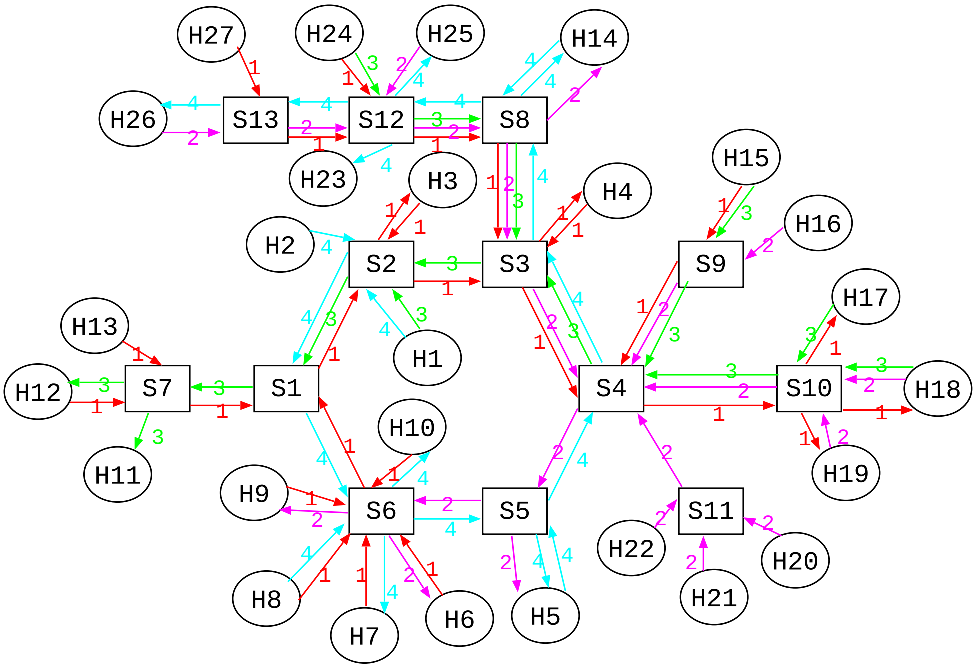
The TBONE topology consists of 10, 45, and 100 Mbps links as depicted in Fig. 6.2(a). Traffic flows between 45 host-pairs following four major "currents" as shown in Fig. 6.2(b): the numbers 1, 2, 3, 4 next to each directed edge in the figure denote the "current" present on that edge. The 45 host-pairs are listed in Table 6.1. Flows between these host-pairs ride on only one current, for example flows from host H1 to H26 ride on current 4. In Fig. 6.2(a), a checkered box on a switch indicates that we have instrumented the switch to study traffic flowing out of that switch onto the link adjacent to the checkered box.

6.2 Source Models

We currently use three kinds of source model in our simulations. All of them are ON/OFF processes. They differ in the distribution of their ON and OFF times and call holding time (CHT, which we will also call "flow duration" or "flow lifetime"). One of these is the two-state Markov process used widely in the literature. Recent



(a) TBone topology



(b) Four traffic "currents" on TBone

Figure 6.2: The TBone topology

Table 6.1: Forty-five Host Pairs on TBone

Source	Destination(s)	Source	Destination(s)
H1	H5, H7, H11, H12, H14, and H26	H14	H23 and H25
		H15	H11 and H17
H2	H10 and H25	H16	H5 and H9
H3	H4 and H19	H17	H12
H4	H18	H18	H5, H6, and H11
H5	H14 and H25	H19	H5
H6	H18	H20	H5
H7	H17	H21	H9
H8	H4, H5, H26	H22	H6
H9	H3 and H19	H24	H12 and H17
H10	H3 and H18	H25	H6 and H14
H12	H4	H26	H9 and H14
H13	H17	H27	H4

studies ([LTWW94, DMRW94, PF94, KM94, GW94, BSTW95]) have shown that network traffic often exhibits long-range dependence (LRD), with the implications that congested periods can be quite long and a slight increase in the number of active connections can result in large increase in packet loss rate [PF94]. The authors of reference [PF94, Flo96a] further call attention to the possibly damaging effect long-range dependent traffic might have on measurement-based admission control algorithms. To investigate this and other LRD related questions, we augment our simulator with two LRD source models.

EXP Model. Our first model is an ON/OFF model with exponentially distributed ON and OFF times. During each ON period, an exponentially distributed random number of packets, with average N, are generated at fixed rate p packet/sec. Let I milliseconds be the average of the exponentially distributed OFF times, then the average packet generation rate a is given by $1/a = I/N + 1/p$. The EXP1 model described in the next section is a model for packetized voice encoded using ADPCM at 32 Kbps.

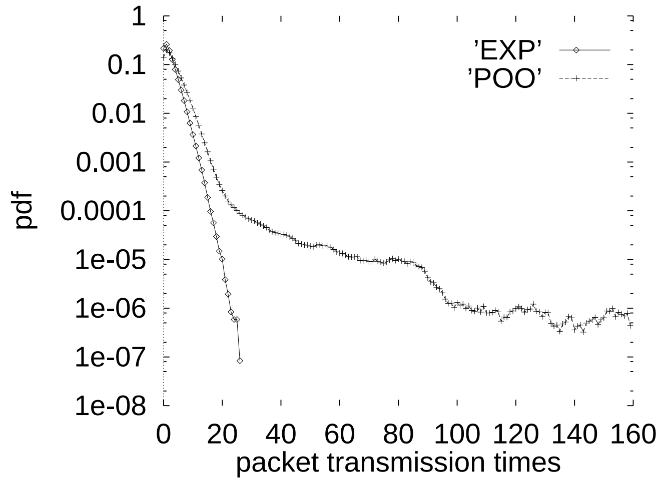


Figure 6.3: Experienced delay of EXP and POO sources.

LRD: Pareto-ON/OFF. Our next model is an ON/OFF process with Pareto distributed ON and OFF times (for ease of reference, we call this the *Pareto*-ON/OFF model). During each ON period, a Pareto distributed number of packets, with mean N and Pareto shape parameter β, are generated at peak rate p packet/sec. The OFF times are also Pareto distributed with mean I milliseconds and shape parameter γ. Pareto shape parameter less than 1 gives data with infinite mean; shape parameter less than 2 results in data with infinite variance. The Pareto location parameter is $mean * (shape - 1)/shape$. Each *Pareto*-ON/OFF source by itself does not generate LRD series. However, the aggregation of them does [WTSW95]. The Hurst parameter that reflects the degree of long-range dependency of a time-series is determined by the heavier tailed of the ON or OFF time distribution. If β is the shape parameter of the heavier tailed Pareto distribution, the Hurst parameter of the aggregate traffic is $H = (3 - \beta)/2$ [CB96].

Fig. 6.3 shows the experienced delay of 250 EXP sources and 330 POO sources at the bottleneck link of the ONE-LINK topology. Both source models have a peak rate of 64 Kbps, average idle time of 325 msec., and average burst length of 20 packets. We can see that even though both source models have peak to average rate of 2, the delay distribution of POO sources has a much heavier tail.

LRD: Fractional ARIMA. We use each number generated by the *fractional autoregressive integrated moving average* (fARIMA) process ([HR89]) as the number of fixed-size packets to be sent back to back in each ON period. Interarrivals of ON periods are of fixed length. For practical programming reasons, we generate a series of 15,000 fARIMA data points at the beginning of each simulation. Each fARIMA source then picks a uniformly distributed number between 1 and 15,000 to be used as its index into that series. On reaching the end of the series, the source wraps around to the beginning. This method is similar to the one used by the authors of [GW94] to simulate data from several sources using one variable bit rate (VBR) video trace.

Let $\{X_t\}$ denote data points from a time-series. An ARMA(p, q) process has the form:

$$X_t = \phi_1 X_{t-1} + \phi_2 X_{t_2} + \ldots + \phi_p X_{t-p} + \alpha_t - \theta_1 \alpha_{t-1} - \theta_2 \alpha_{t-2} - \ldots - \theta_q \alpha_{t-q}, \quad (6.1)$$

where the α_t are uncorrelated Gaussian noise, the $\phi_j, j = 1 \ldots p$, are the autoregressive weights and the $\theta_j, j = 1 \ldots q$, are the moving average weights. The ARMA process is stationary if $-1/2 < \phi_1 < 1/2$ ([BJ76], p. 76). Next define a lag operator B as $X_{t-1} = BX_t$, and the difference operator ∇ as $(X_t - X_{t-1}) = \nabla X_t$; hence $\nabla X_t = (1 - B)X_t$. Let $\Phi(B) = (1 - \phi_1 B - \ldots - \phi_p B^p)$ and $\Theta(B) = (1 - \theta_1 B - \ldots - \theta_q B^q)$. Then an ARIMA$(p, d, q)$ process is defined as:

$$\Phi(B)\nabla^d X_t = \Theta(B)\alpha_t. \quad (6.2)$$

A fractional ARIMA process has a d of fractional value. A fARIMA(p, d, q) process with $0 < d < 1/2$ generates long-range dependent series with Hurst parameter $H = d + 1/2$ [Hos81, LTWW94]. Hence the fARIMA model takes three parameters: the autoregressive process order with the corresponding set of weights, the degree of integration, and the moving average process order with the corresponding set of weights, it also requires an innovation with a Gaussian marginal distribution. The marginal distribution of a fARIMA generated series is also Gaussian; whereas VBR video traces exhibit low average with high peaks. Thus we can *not* use the fARIMA output to model traffic from a single VBR video source. Nevertheless, simulation results in [GW94] indicate that aggregation of fARIMA generated series may well model aggregate VBR video traffic—such as that coming from a subnetwork. In our simulations, we first generate a normally distributed innovation with mean N and standard deviation s packets. If the minimum of the fARIMA output is less than zero, we shift the whole series by adding the absolute value of its minimum to every number in the series. This way of obtaining non-negative series is also used in [AM95]. Note that this shifting process constrains the maximum value of the generated series to be always twice its average. The Whittle maximum likelihood estimator [Ber94] confirms that our shifting, cropping, and overlaying of the fARIMA generated series do not destroy its long-range dependent characteristic.

To ease discussion on the effect of different source models on traffic characteristics, it is useful to define the following additional notations and concepts: let ρ be a flow's *density* (the ratio of its average to peak rates, a/p), R its *grain size* (the ratio of its peak rate to link bandwidth, p/μ), and $1/\rho$ its *burstiness*. Aggregation of flows with $\rho \approx 1$ results in smooth traffic and reliable measurement. Bursty flows with short bursts ($\rho \ll 1, N/\mu \ll D_j$) will have their bursts smoothed out by the switch's buffer, resulting, again, in reliable measurement values. Bursty flows with large bursts, e.g. Pareto distributed ON time, but small grain size ($\rho \ll 1, N/\mu \gg D_j, \text{and } R \leq 0.1\mu$) still allow for large degree of statistical multiplexing. However bursty flows with long burst *and* large grain size ($\rho \ll 1, N/\mu \gg D_j, \text{and } R > 0.1\mu$) might best be alloted their own guaranteed bandwidth.

In addition to each source's characteristics of density and grain size, network traffic dynamics is also shaped by the arrival pattern and duration of flows. Our simulator allows us to drive each simulation with a number of flow generators; for each generator, we can specify its start and stop times, the average flow interarrival time, the maximum number of concurrently active flows, and the mix of transport protocol, source model, token bucket filter, and service request ascribed to each flow. We give exponentially distributed lifetimes to the EXP model, following [Mol27]. The duration of for LRD sources, however, are taken from a lognormal distribution,

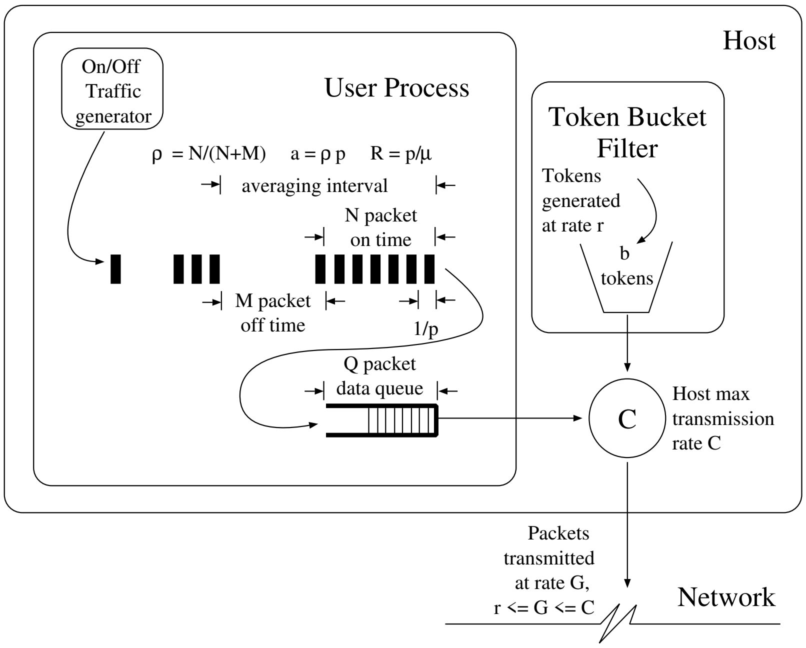


Figure 6.4: ON/OFF traffic model with token-bucket filter

following [Bol94, DMRW94]. The interarrival times of all flows are exponentially distributed [PF94].

6.3 Parameter Choices

Fig. 6.4 shows a packet-arrival depiction of an ON/OFF source in the context of a host with token-bucket filter. To make a given traffic generation source conform to a particular token bucket filter, a host can queue packets arriving at an empty bucket until more tokens are available. If the data queue length (Q) is 0, packets that arrive at an empty token bucket are immediately dropped. We chose six instantiations of the above three source models, as summarized in Table 6.2. In the table, $p = \infty$ means that after each OFF time, packets for the next ON period are transmitted back to back. (On real networks, packets are sent back to back when the applications generate traffic faster than the network can transmit it.) In the same table, we also list the settings of the token bucket parameters assigned to each source. Column 8 of the table, labeled *cut rate*, indicates the average number of packets that would have

Table 6.2: Six Instantiations of the Three Source Models

Model Name	Model's Parameters				Token Bucket Parameters				Bound (ms)	
	p pkt/ sec	I msec	N pkts	p/a	r tkn/ sec	b tkns	cut rate	max qlen	D^*	D_j
EXP1	64	325	20	2	64	1	0	0	16	16
EXP2	1024	90	10	10	320	50	2.1e-3	17	160	160
EXP3	∞	684	9	∞	512	80	9.4e-5	1	160	160
				β						
POO1	64	2925	20	1.2	64	1	0	0	16	16
POO2	256	360	10	1.9	240	60	4.5e-5	220	256	160
				s						
fARIMA ({0.75}, 0.15, -)	∞	125	8	13	1024	100	1.1e-2	34	100	160

been dropped by each flow's token bucket filter *over* the total number of packets sent by the flow, had the data queue length been 0 (i.e. packets are immediately dropped upon arriving at an empty token bucket). Column 9, labeled *max qlen*, shows the maximum data queue length a flow can expect to see if the data queue has infinite length. We assign each flow a data queue with infinite length in all our simulations (i.e. packets that arrive at an empty token bucket are always queued, and the queue never overflows). Recall that in this study we use fixed packet size and each of our token is worth 1 Kbits of data, which is also our packet size.

The shape parameter of the Pareto distributed ON time (β) of the Pareto-ON/OFF sources are selected following the observations in [WTSW95]. According to the same reference, the shape parameter of the Pareto distributed OFF time (γ) stays mostly below 1.5; in this study we use $\gamma = 1.1$ for all POO sources. For the POO1 model, we use a token bucket rate equals to the source's peak rate such that the token bucket filter does not reshape the traffic. For the POO2 model, some of the generated packets were queued; this means during some of the source's alleged "OFF" times, it may actually still be draining its data queue onto the network. Thus for the POO2 model, the traffic seen on the wire may not be Pareto-ON/OFF.

When a flow with token bucket parameters (r, b) requests guaranteed service, the maximal queueing delay (ignoring terms proportional to a single packet time) is given by b/r [Par92]. Column 10 of the table, labeled D^*, lists the guaranteed delay bound for each source given its assigned token bucket filter. Column 11, labeled D_j, lists the predictive delay bound assigned to each source. We simulate only two classes of predictive service. A predictive bound of 16 msecs. means first class predictive service, 160 msecs. second class. We have chosen the token bucket parameters so that, in most cases, the delay bounds given to a flow by predictive and guaranteed services are the same. This facilitates comparison between the utilization levels achieved with predictive and guaranteed services. In the few cases where the delays are not the same, such as in the POO2 and fARIMA cases, the utilization comparison is less meaningful. In the POO2 case, for example, the predictive delay bound is smaller than the guaranteed bound, so the utilization gain we find here understates the true gain.

For the fARIMA source, we use an autoregressive process of order 1 (with weight 0.75) and degree of integration 0.15 (resulting in a generated series with Hurst parameter 0.65). The first order autoregressive process with weight 0.75 means our fARIMA traffic also has strong short-range dependence. The interarrival time between ON periods is 1/8th of a second. The Gaussian innovation fed to the fARIMA process has a mean of 8 packets with standard deviation 13.

Except for simulations on the TBone topology, flow interarrival times are exponentially distributed with an average of 400 milliseconds. Because of system memory limitation, we set the average flow interarrivals of simulations on the TBone topology to 5 seconds. The average holding time of all EXP sources is 300 seconds. The POO and fARIMA sources have lognormal distributed holding times with median 300 seconds and shape parameter 2.5. We run simulations with Markov-ON/OFF (EXP) sources for 3000 seconds simulated time. The data presented are obtained from the later half of each simulation. By visual inspection, we determined that 1500 simulated seconds is sufficient time for the simulations to warm up. Simulations involving Pareto-ON/OFF sources require a longer warmup period and a longer simulation time for the LRD effect to be seen, thus we run them for 5.5 hours simulation time, with reported data taken from the later 10000 seconds.

Chapter 7

On the Viability of the Algorithm

In this chapter, we show that measurement-based admission control algorithm, when used with predictive service, indeed yields higher level of link utilization than that achieveable under parameter-based algorithms with guaranteed service. We provide supporting evidence from results of simulations with both homogeneous and heterogeneous traffic sources, on both single-hop and multi-hop networks. Depending on traffic burstiness, the utilization gain ranges from twice to order of magnitude.

7.1 Homogeneous Sources: The Single-hop Case

By homogeneous sources we mean sources that not only employ just one kind of traffic model, but also ask for only one kind of service. For this and all subsequent single-hop simulations, we use the topology depicted in Fig. 6.1(a). For each source, we run two kinds of simulation. The first has all sources requesting guaranteed service. The second has all sources requesting predictive service. The results of the simulations are shown in Table 7.1. The column labeled "%Util" contains the link utilization of the bottleneck link, L3. The "#Actv" column contains a snapshot of the average number of active flows concurrently running on that bottleneck link. The "$\lceil d_j \rceil$" column contains the maximum experienced delay of predictive class j packets. The "$\overline{L}/T$" column lists the ratio of average flow duration to measurement window used with each source model. We repeat the predictive service simulations

Table 7.1: Single-hop Homogeneous Sources Simulation Results

Model Name	Guaranteed %Util	Guaranteed #Actv	Predictive %Util	Predictive #Actv	$\lceil d_j \rceil$	$\overline{L}/T$
EXP1	46	144	79	250	3	60
EXP2	28	28	75	75	42	300
EXP3	2	18	54	406	33	600
POO1	7	144	78	1539	8	60
POO2	3	38	72	965	8	60
fARIMA	55	9	81	13	72	60

nine times, each time with a different random seed, to obtain confidence intervals. We find the confidence interval for the all the numbers to be very tight, less than one least significant digit in most cases.

As mentioned in Chapter 6.2, we consider the performance of our admission control algorithm "good" if there is *no* delay bound violation during a simulation run. Even with this very restrictive requirement, one can see from Table 7.1 that predictive service consistently allows the network to achieve higher level of utilization than guaranteed service does. The utilization gain is not large when sources are smooth. For instance, the source model EXP1 has a peak rate that is only twice its average rate. Consequently, the data only shows an increase in utilization from 46% to 80%. (One can argue that the theoretical upper bound in the utilization increase is the peak to average ratio.) In contrast, bursty sources allow predictive service to achieve several orders of magnitude higher utilization compared to that achievable under guaranteed service. Source model EXP3, for example, is a very bursty source; it has an infinite peak rate (i.e. sends out packets back to back) and has a token bucket of size 80. The EXP3 flows request reservations of 512 Kbps, corresponding to the token bucket rate at the sources. Under guaranteed service, only 18 flows can be admitted to the 10 Mbps bottleneck link (with 90% utilization target). The actual link utilization is only 2%.[1] Under predictive service, 406 flows are served on the average, resulting in actual link utilization of 54%.

In this homogeneous scenario with only one class of predictive service and constantly oversubscribed link, our measurement-based admission control algorithm easily adapts to LRD traffic between the coming and going of flows. The utilization increased from 7% to 78% and from 3% to 72% for the POO1 and POO2 sources respectively. The utilization gain for the fARIMA sources was more modest, from 55% to 81%. This is most probably because the source's maximum ON time is at most twice its average (an artifact of the shifting we do, as discussed in Chapter 6.2, to obtain non-negative values from the fARIMA generated series). In all cases, we are able to achieve high levels of utilization without incurring delay violations. To further test the effect of long OFF times on our measurement-based algorithm, we simulated POO1 sources with infinite duration. With utilization target of 90% link capacity, we do see a rather high percentage of packets missing their delay bound. Lowering the utilization target to 70%, however, provide us enough room to accommodate traffic bursts. Thus for these scenarios, we see no reason to conclude that LRD traffic poses special challenges to our measurement-based approach.

7.2 Homogeneous Sources: The Multi-hop Case

Next we run simulations on multi-hop topologies depicted in Figs. 6.1(b) and (c). The top half of Table 7.2 shows results from simulations on the TWO-LINK topology. The utilization numbers are those of the two links connecting the switches in the topology. The source models employed here are the EXP1, EXP3, and POO2 models, one per simulation. The bottom half of Table 7.2 shows the results from simulating source models EXP2, POO1, and fARIMA on the FOUR-LINK topology. For each source model, we again run one simulation where all sources request guaranteed service, and another one where all sources request *one* class of predictive service.

[1] Parameter-based admission control algorithms may not need to set a utilization target and thus can achieve a somewhat higher utilization; for the scenario simulated here, two more guaranteed flows could have been admitted.

Table 7.2: Multi-hop Homogeneous Sources Link Utilization

Topology	Link Name	Model Name	Guaranteed %Util	Predictive %Util	Predictive $\lceil d_j \rceil$
Two-Link	L4	exp1	45	67	2
		exp3	2	44	20
		poo2	3	59	7
	L5	exp1	46	78	3
		exp3	2	58	30
		poo2	3	70	17
Four-Link	L6	exp2	17	42	6
		poo1	4	31	1
		fARIMA	38	54	36
	L7	exp2	28	71	31
		poo1	7	66	2
		fARIMA	55	77	40
	L8	exp2	28	72	24
		poo1	8	75	7
		fARIMA	53	74	29
	L9	exp2	28	71	31
		poo1	8	59	2
		fARIMA	53	80	44

The most important result to note is that, once again, predictive service yields reasonable levels of utilization without incurring any delay violations. The utilization levels, and the utilization gains compared to guaranteed service, are roughly comparable to those achieved in the single hop case.

7.3 Heterogeneous Sources: The Single-hop Case

We now look at simulations with heterogeneous sources. For each of the simulation, we use two of our six source model instantiations. Each source is given the same token bucket as listed in Table 6.2 and, when requesting predictive service, requests the same delay bound as listed in the said table. We run three kinds of simulation with heterogeneous sources: (1) single source model requesting multiple levels of predictive service, (2) multiple source models requesting a single class of predictive service, and (3) multiple source models requesting multiple levels of predictive service. In all cases, we compare the achieved utilization with those achieved under guaranteed service. For the first and third cases, we also experiment with sources that request both guaranteed and predictive services. When multiple source and/or service models are involved, each model is given an equal probability of being assigned to the next new flow. In all these simulations, the experience delays are all within their respective bounds.

Table 7.3 shows the utilization achieved when flows with the same source model request: two classes of predictive service (PP), guaranteed and one predictive class (GP), and guaranteed and two predictive classes (GPP). In the GP case, flows request the predictive class "assigned" to the source model under study (see Table 6.2). In the other cases, both predictive classes are requested. Compare the numbers in each column of Table 7.3 with those in the "%Util" column of Table 7.1 under guaranteed service. The presence of predictive traffic invariably increases network utilization.

Next we look at the simulation results of multiple source models requesting a single service model. Table 7.4 shows the utilization achieved for selected pairings of the models. The column headings name the source model pairs. The first row shows

Table 7.3: Single-hop, Single Source Model, Multiple Predictive Services Link Utilization

Model	PP	GP	GPP
EXP1	77	77	–
EXP2	71	70	–
EXP3	31	31	–
POO1	70	69	69
POO2	60	57	–
fARIMA	79	79	78

Table 7.4: Single-hop, Multiple Source Models, Single Service Link Utilization

Service	EXP1–POO1	EXP2–EXP3	EXP2–POO2	EXP2–fARIMA	EXP3–fARIMA	POO2–fARIMA
Guaranteed	15	21	5	38	18	32
Predictive	75	70	63	79	81	69

the utilization achieved with guaranteed service, the second predictive service. We let the numbers speak for themselves.

Finally in Table 7.5 we show utilization numbers for flows with multiple source models requesting multiple service models. The first row shows the utilization achieved when all flows asked only for guaranteed service. The second row shows the utilization when half of the flows requests guaranteed service and the other half requests the predictive service suitable for its characteristics (see Table 6.2). And the last row shows the utilization achieved when each source requests a predictive service suitable to its characteristics.

7.4 Heterogeneous Sources: The Multi-hop Case

We next run simulations with all six source models on all our topologies. In Table 7.6 we show the utilization level of the bottleneck links of the different topologies. Again, contrast the utilization achieved under guaranteed service alone with those under both

Table 7.5: Single-hop, Multiple Source Models, Multiple Predictive Services Link Utilization

Service	EXP1–EXP2	EXP1–fARIMA	EXP1–POO2	EXP2–POO1	EXP3–POO1	POO1–fARIMA
Guaranteed	43	50	29	10	7	23
Guar./Pred.	73	74	65	61	51	65
Predictive	75	78	65	62	60	65

guaranteed and predictive services. The observed low predictive service utilization on link L6 is not due to any constraint enforced by its *own* admission decisions, but rather is due to lack of traffic flows caused by rejection of multi-hop flows by later hops, as we will explain in Chapter 9. Utilization gains on the TBone topology are not so pronounced as on the other topologies. This is partly because we are limited by our simulation resources and cannot drive the simulations with higher offered load. Recall that flow interarrivals on simulations using the TBone topology have an average of 5 seconds, which is order of magnitude larger than the 400 milliseconds used on the other topologies.

Our results so far indicate that a measurement-based admission control algorithm can provide reasonably reliable delay bounds at significant utilization gains. These conclusions appear to hold not just for single hop topologies and smooth traffic sources, but also for multi-hop configurations and long-range dependent traffic. We cannot, within reasonable time, verify our approach in an exhaustive and comprehensive way, but our simulation results are encouraging.

Table 7.6: Single- and Multi-hop, All Source Models, All Services Link Utilization

Topology Name	Link Name	Guaranteed %Util	Guaranteed and Predictive %Util	$\lceil d_1 \rceil$	$\lceil d_2 \rceil$
ONE-LINK	L3	24	66	3.	45.
TWO-LINK	L4	15	72	2.	54.
	L5	21	72	2.	41.
FOUR-LINK	L6	19	47	1.	36.
	L7	24	70	2.	46.
	L8	20	72	2.	49.
	L9	18	75	1.	53.
TBONE	L2	9	14	0.02	0.15
	L10	17	31	0.15	5.35
	L11	27	32	0.37	21.9
	L12	22	23	0.1	5.84
	L20	8	21	0.22	16.6
	L30	32	52	0.49	34.7

Chapter 8

Practical Deployment Issues

In this chapter we consider several practical issues related to deployment of our algorithm. In particular, we look at the effect of different measurement window (T) settings on the behavior of the admission control algorithm. We show that a smaller T, relative to flow lifetime (L), yields higher utilization but less reliable delay bound, while a larger one provides more stable delay estimate at lower utilization. We also present a few sample path snapshots illustrating the effect of T.

8.1 Choosing a Window Size

Varying the measurement window size, T, has two related effects on the admission control algorithm. First, since T is the length of the measurement block used to determine how long we keep the previous maximal packet delay and sampled utilization, increasing T makes these estimates more conservative, which in turn makes the admission control algorithm itself more conservative. Thus, larger T means fewer delay violations and lower link utilization. Second, T also controls how long we continue to use our calculated estimate of the delay and utilization induced by a newly admitted flow. Recall that whenever a new flow is admitted, we artificially increase the measured values to reflect the worst-case expectations, and then restart the measurement window. Thus, we are using the calculated effects of new flows rather than the measured effects until we survive an entire T period without any new flow arrival.

Let $\overline{r}$ be the average flow reservation rate, and μ the link bandwidth (for convenience, assume we only perform bandwidth check (Eqn. 4.9) and $\upsilon = 1$), we will admit at most $A = \mu/\overline{r}$ number of flows for every T. Thus at the end of its average lifetime, $\overline{L}$, an average flow would have seen approximately

$$F = A * \overline{L}/T \tag{8.1}$$

number of flows. If the average rate of an average flow is $\widehat{\overline{r}}$, ideally we want $F * \widehat{\overline{r}}$, a link's stable utilization level, to be near μ. However, flows also depart from the network. The expected number of flow departures during the period T depends on the number of admitted flows and their duration. If this number of departures is significant, a flow will see a much smaller number of flows during its lifetime, i.e. the stable $F * \widehat{\overline{r}}$ becomes *much* smaller than μ. For the same average reservation rate, $\overline{r}$, and a given T, the size of the stable F is determined by the average flow duration, $\overline{L}$. A shorter average flow duration means more departure per T. In the long run, we aim for $F * \widehat{\overline{r}} \approx \mu$, or equivalently, $\overline{L}/T \approx \overline{r}/\widehat{\overline{r}}$. If all flows use exactly what they reserved, we have $\overline{L}/T = 1$, meaning that we should not try to give away the flows' reservations.

In other words, T has two related effects on the admission control algorithm: (1) too small a T results in more delay violations and lower link utilization, (2) too long a T depresses utilization by keeping the artificially heighten measured values for longer than necessary. While the first effect is linked to flow duration only if the flow exhibits long-range dependence, the second effect is closely linked to the average flow duration in general. Note that when T is infinite, we only use our computed values, which are conservative bounds, and ignore the measurements entirely. That is, we will never suffer any delay violations at a given hop if we use an infinite value for T. Thus, the parameter T always provides us with a region of reliability. We now present some illustrative simulation results on the importance of the $\overline{L}/T$ ratio. These results are meant to be canonical illustrations, thus we do not provide the full details of the simulations from which they are obtained.

Table 8.1: Effect of T and $\overline{L}$

(a) Varying T, $\overline{L} = 300$ secs.

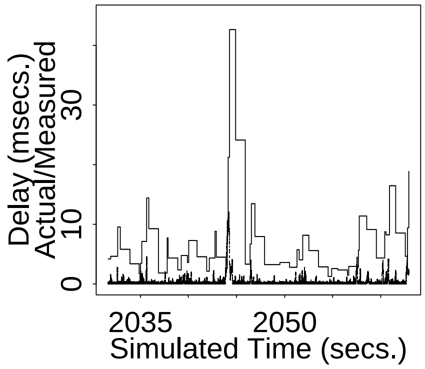
T	%Util	$\lceil d_j \rceil$
1e4	82	25
5e4	81	22
1e5	77	15
2e5	75	13
5e5	68	5

(b) Varying $\overline{L}$ (secs)

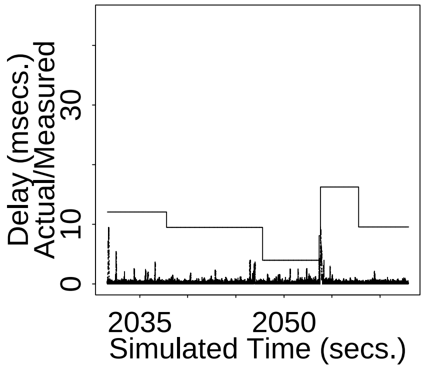
$\overline{L}$	T = 1e4		T = 1e5	
	%Util	$\lceil d_j \rceil$	%Util	$\lceil d_j \rceil$
3000	86	48	82	24
900	84	32	80	16
300	82	25	77	15
100	81	21	76	11
30	78	15	69	7

In Table 8.1(a) we show the average link utilization and maximum experienced delay from simulations of flows with average duration of 300 seconds. We varied the measurement window, T, from $1e4$ packet times to $5e5$ packet times. Notice how smaller T yields higher utilization at higher experienced delay and larger T keeps more reliable delay bounds at the expense of utilization level. Next we fixed T and varied the average flow duration. Table 8.1(b) shows the average link utilization and maximum experienced delay for different values of average flow duration with T fixed at $1e4$ and $1e5$. We varied the average flow duration from 3000 seconds (practically infinite, given our simulation duration of the same length) to 30 seconds. Notice how longer lasting flows allow higher achieved link utilization while larger measurement periods yield lower link utilization. Link utilization is at its highest when the $\overline{L}/T$ ratio is the largest and at its lowest when this ratio is the smallest. On the other hand, the smaller $\overline{L}/T$ ratio means lower experienced delay and larger $\overline{L}/T$ means the opposite—thus lowering the $\overline{L}/T$ ratio is one way to decrease delay violation rate.

In Figs. 8.1 and 8.2 we provide sample path snapshots showing the effect of T on delay and link utilization. We note however, a T that yields artificially low utilization when used in conjunction with one source model may yield appropriate utilization when used with burstier sources or sources with longer burst time.

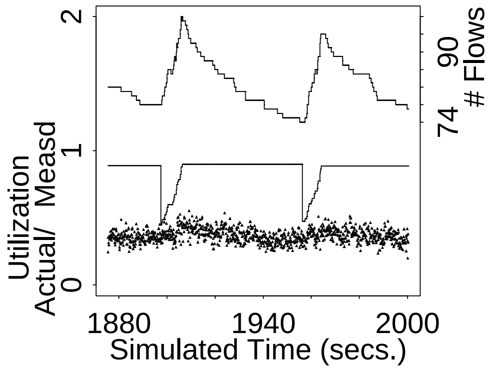


(a) Smaller T

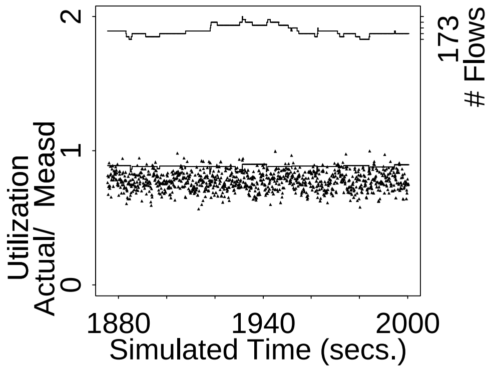


(b) Larger T

Figure 8.1: Effect of T on Experienced Delay



(a) Larger T



(b) Smaller T

Figure 8.2: Effect of T on Link Utilization

8.2 Choosing a Utilization Target

Imagine now that flows have infinite duration; by Eqn. 8.1, the number of admissible flows would also be infinite. In practice, this means flows will be admitted until the link reaches 100% utilization. As we noted in Chapter 5, variance in delay diverges in a simple $M/M/1$ queue as the system approaches full utilization. Obviously, we need to prevent the network from reaching such high load by instituting a maximum utilization target. Sources with small grain size and short bursts will allow an higher utilization target. High density sources with long bursts will require a lower utilization target. In Chapter 10 we will study several attempts to set the utilization target, both formal and *ad-hoc*.

8.3 Structural Limitations

As we mentioned in Chapter 1, when there are only a few flows present, or when a few large-grain flows dominate the link bandwidth, the unpredictability of individual flow's behavior dictates that a measurement-based admission control algorithm must be very conservative. One may need to rely less on measurements and more on the worst-case parameters furnished by the source, and perform the following bandwidth check instead of Eqn. 4.9:

$$\upsilon\mu > \widetilde{\nu}_G + \sum_{i=1}^{K} \widetilde{\nu}_i, \tag{8.2}$$

where,

$$\begin{aligned} \widetilde{\nu}_G &= \widehat{\nu}_G + \kappa(\mathrm{MAX}(0, \nu_G - \widehat{\nu}_G)), \\ \widetilde{\nu}_j &= \widehat{\nu}_j + \kappa(\mathrm{MAX}(0, \nu_j - \widehat{\nu}_j)),\ j = 1 \ldots K, \end{aligned}$$

ν_G is the sum of all reserved guaranteed rates, ν_j is the sum of all reserved rates in class j, K is number of predictive classes, and κ is a fraction between 0 and 1.

For $\kappa = 1$, we have the completely conservative case. Similarly, one could do the following delay check:

$$D_j = \frac{\sum_{i=1}^{j} \kappa \sum_{\alpha \in \{i\}} b_i^{\alpha}}{\mu - \kappa \nu_G - \kappa \sum_{i=1}^{j-1} \nu_i}. \tag{8.3}$$

for every predictive class j for which one needs to do a delay check as determined in Chapter 4.5.

Even with a high enough degree of statistical multiplexing, a flow might become idle for prolonged periods of time such that the measurement mechanism becomes oblivious to it. When the idle flow resumes transmission, delay bound violations could ensue. We recognize two kinds of idle of times: (1) those of time-scale larger than the average flow duration, (2) those that are some small multiples of T. Examples of the first are flows with advance or dynamic reservations. This would require non-measurement based mechanism to accommodate them and is not part of our current research. The second kind may be common in two-way conversations or database lookup applications. One could either make a separate reservation for each burst of activity, risking admission control failure, or make some portion of each flow's reservation not subject to the measurement process. The latter approach is adopted in reference [C+91].

We should also note that our measurement-based approach is vulnerable to spontaneous correlation of sources, such as when all the TV channels air coverage of a major event. Each source model used in this study has uncorrelated ON and OFF times. The ON and OFF times between sources are also not correlated. If all flows suddenly burst at the same time, delay violations will result. We are not aware of any way to prevent this kind of delay violation, since the network cannot predict such correlations beforehand. Instead, we rely on the uncorrelated nature of statistically multiplexed flows to render this possibility a very unlikely event.

8.4 If Peak Rate is Incoming Link Bandwidth

Eqn. 4.1 is an upper bound on the worst-case delay of a class, assuming infinite source rate. In reality, the peak rate of traffic arriving at a switch is bounded by the bandwidth of the incoming link. In this section we consider the effect of incoming link bandwidth on our algorithm. By expanding the last term of Eqn. 4.1 and applying the distributive law we get:

$$\begin{aligned} D_j^* &= \frac{\sum_{i=1}^{j-1} b_i}{\mu - \sum_{i=1}^{j-1} r_i} + \frac{C_j \frac{b_j}{C_j - r_j}}{\mu - \sum_{i=1}^{j-1} r_i} - \frac{(\mu - \sum_{i=1}^{j-1} r_i)\frac{b_j}{C_j - r_j}}{\mu - \sum_{i=1}^{j-1} r_i} \\ &= \frac{\sum_{i=1}^{j-1} b_i}{\mu - \sum_{i=1}^{j-1} r_i} + \frac{\left[C_j - (\mu - \sum_{i=1}^{j-1} r_i)\right]\frac{b_j}{C_j - r_j}}{\mu - \sum_{i=1}^{j-1} r_i}. \end{aligned} \tag{8.4}$$

Substituting μ_{in}, the incoming link bandwidth, for C_j, the source's peak rate, and combining the two terms, the equation becomes:

$$D_j^* = \frac{\sum_{i=1}^{j-1} b_i + \frac{\mu_{in} - (\mu - \sum_{i=1}^{j-1} r_i)}{\mu_{in} - r_j} b_j}{\mu - \sum_{i=1}^{j-1} r_i}. \tag{8.5}$$

As mentioned in the proof of Theorem 1, we require $\mu > \sum_{i=1}^{j} r_i$ at all switches, hence $r_j < \mu_{in}$. If $\mu - \sum_{i=1}^{j-1} r_i \geq \mu_{in} > r_j$, b_j will not be queued. Hence the worst-case delay for $\mu_{in} < \mu$ is:

$$D_j^* = \frac{\sum_{i=1}^{j-1} b_i + \frac{\left[\mu_{in} - (\mu - \sum_{i=1}^{j-1} r_i)\right]^+}{\mu_{in} - r_j} b_j}{\mu - \sum_{i=1}^{j-1} r_i}, \tag{8.6}$$

where:

$$[x]^+ = \begin{cases} x, & x > 0, \\ 0, & x \leq 0. \end{cases}$$

Applying Eqn. 8.6 to our admission control algorithm, a prospective predictive flow α of class k is denied admittance if the delay bound of the same priority traffic, D_k, is violated:

$$D_k > \widehat{D}_k + \frac{\frac{\left[\mu_{in} - (\mu - \widehat{\nu}_G - \sum_{i=1}^{k-1} \widehat{\nu}_i)\right]^+}{\mu_{in} - r_j} b_k^\alpha}{\mu - \widehat{\nu}_G - \sum_{i=1}^{k-1} \widehat{\nu}_i}, \tag{8.7}$$

or if lower priority classes' delay bounds, D_j's, are violated:

$$D_j > \widehat{D}_j \frac{\mu - \widehat{\nu}_G - \sum_{i=1}^{j-1} \widehat{\nu}_i}{\mu - \widehat{\nu}_G - \sum_{i=1}^{j-1} \widehat{\nu}_i - r_k^\alpha} + \frac{\frac{\left[\mu_{in} - (\mu - \widehat{\nu}_G - \sum_{i=1}^{k-1} \widehat{\nu}_i)\right]^+}{\mu_{in} - r_j} b_k^\alpha}{\mu - \widehat{\nu}_G - \sum_{i=1}^{j-1} \widehat{\nu}_i - r_k^\alpha}, \qquad k < j \leq K. \tag{8.8}$$

However, if $\mu_{in} > \mu$, Eqn. 4.1 applies. Hence we do not use Eqns. 8.7 and 8.8 in our admission control algorithm.

Chapter 9

On Unequal Flow Rejection Rates

Most of the admission control algorithms in the literature are based on the *violation prevention* paradigm: each switch decides to admit a flow if and only if the switch can still meet all of its service commitments. In other words, the *only* criteria considered by admission control algorithms based on the violation prevention paradigm is whether any service commitments will be violated as a result of a new admission. In this section we discuss some policy or allocation issues that arise when not all flows are completely equivalent. When flows with different characteristics—either different service requests, different holding times, or different path lengths—compete for admission, admission control algorithms based purely on violation prevention can sometimes produce equilibria with some categories of flows experiencing higher rejection rate than other categories do. In particular, we identify two causes of unequal rejection rate: (1) flows traversing a larger number of hops have a higher chance of being rejected by the network, and (2) flows requesting more resources are more likely to be rejected by the network.

9.1 Effect of Hop Count on Rejection Rates

As expected, when the network is as loaded as in our simulations, multi-hop flows face an increased chance of being denied service by the network. For example, in our simulation with homogeneous sources on the TWO-LINK network, as reported in

Table 7.2, more than 75% of the 700 new EXP1 sources admitted under guaranteed service are single-hop flows. This is true for both of the bottleneck links. A somewhat smaller percentage of the more than 1000 flows admitted under predictive service are single-hop flows. This effect is even more pronounced for sources that request larger amount of resources, e.g. the POO2 or the fARIMA sources. And it is exacerbated by sources with longer lifetimes: with fewer departures from the network, new flows see an even higher rejection rate.

Aside from disparity in the kinds of flow present on the link, this phenomenon also affects link utilization; upstream switches (switches closer to source hosts) could yield lower utilization than downstream switches. We observe two causes to this: (1) switches that carry only multi-hop flows could be starved by admission rejections at downstream switches. The utilization numbers of link L6 in both Tables 7.2 and 7.6 are consistently lower than the utilization of the other links in the FOUR-LINK topology. Notice that we set these simulations up with no single hop flow on link L6. The low utilization is thus not due to the constraint put on by link L6's *own* admission decisions, but rather is due to multi-hop flows being rejected by downstream switches. (2) Non-consummated reservations depress utilization at upstream switches; to illustrate: a flow admitted by an upstream switch is later rejected by a downstream switch; meanwhile, the upstream switch has increased its measurement estimates in anticipation of the new flow's traffic, traffic that never come. It takes time (to the expiration of the current measurement window) for the increased values to come back down. During this time, the switch cannot give the reserved resources away to other flows. We can see this effect by comparing the utilization at the two bottleneck links of the TWO-LINK topology as reported in Table 7.2. Note, however, even with the presence of this phenomenon, the utilization achieved under predictive service with our measurement-based admission control algorithm still outperforms those achieved under guaranteed service.

9.2 Effect of Resource Requirements on Rejection Rates

Sources that request smaller amount of resources can prevent those requesting larger amount of resources from entering the network. For example, in the simulation using the EXP2–EXP3 source pair reported in Table 7.4, 80% of the 577 new guaranteed flows admitted after the simulation warmup period were EXP2 flows, which are less resource demanding. In contrast, 40% of flows admitted under predictive service with our measurement-based admission control algorithm were the more resource demanding EXP3 flows. Another manifestation of this case is when there are sources with large bucket sizes trying to get into a high priority class. Because the delay of a lower priority class is affected by *both* the rate and bucket size of the higher priority flow (as explained in Chapter 4.2), the admission control algorithm is more likely to reject flows with a large bucket size and high priority than those with a smaller bucket size or low priority. We see this phenomenon in the simulation of source model EXP3 reported in Table 7.3. When all sources request either of the two classes of predictive service with equal probability, of the 1162 flows admitted after the simulation warmup period, 83% were of class 2. When sources request guaranteed or second class predictive service, only 8% of the 1137 new flows ends up being guaranteed flows. In both of these scenarios, the link utilization achieved is 31%, which is lower than the 62% achieved when all flows request only class 2 predictive service (see Table 7.1), but still order of magnitude higher than the 2% achieved when all flows request only guaranteed service (again, see Table 7.1).

We consider the unequal rejection rate phenomenon a policy issue (or rather, several policy issues) because there is no delay violations and the network is still meeting all its service commitments (which is the original purpose of admission control); the resulting allocation of bandwidth is, however, very uneven and might not meet some policy requirements of the network. We want to stress that this unequal rejection rate phenomenon arises in *all* admission control algorithms based on the *violation prevention* paradigm. In fact, our data shows that these uneven allocations occur in sharper contrast when all flows request guaranteed service, when admission control is a simple bandwidth check. In Chapter 10, we present further evidence that this phenomenon occurs under other admission control algorithms. Clearly, when possible service commitment violations is the only admission control criteria, one cannot ensure that policy goals will be met.

9.3 A Quota Mechanism

One possible approach to control the allocation of resources to flows of differing requirements is by instituting a quota policy. In this section we provide a simple mechanism by which quota policies may be implemented. We hasten to note, however, that we do not intend to study the fair allocation of resources by various quota policies. We refer the interested readers to references [KS85, KU93], in which the authors study allocation strategies for two types of flows that reduce blocking probability, or to reference [DM96], in which the authors propose a game-theoretic approach to ensure fairness to various supported flow types.

Flow opportunity cost metric. To design a quota mechanism, we must first define a *flow opportunity cost metric* that would allow us to compare the resource requirements of one flow to that of anoother. We require the following characteristics of the metric: (1) it must be a function of the flow's token bucket parameters, (r, b), and the requested delay bound, D, (2) the metric of a flow must be independent of existing traffic, and (3) the metric must support arbitrarily complex quota policies. On a switch with FIFO scheduling discipline, we know from Eqn. 4.1 that the largest demand a flow places on the switch is to serve its bucket full of data. To meet the requested delay bound, the switch must serve the flow at rate $\geq b/D$. Thus the worst-case rate required by a flow is: $\mathrm{MAX}(b/D, r)$. A simple opportunity cost metric is the ratio between this rate and link bandwidth:

$$\eta = \frac{\mathrm{MAX}(b/D, r)}{\mu}. \tag{9.1}$$

For example the η of EXP1 source model defined in Section 6.3 on a 10 Mbps link is 6.25e-3, and the η of POO2 source model is 3.75e-2.

Expressing quota policy. Once we have a metric to compare the resource requirement of various flows, we can use them in an expression of quota policy. Instead of allocating bandwidth to specific η values, we specify quota policy for different "η-classes." An η-class is a range of η values. η-classes should be set at least an order of magnitude apart. Thus a simple sample quota policy could be:

$$x\%(\eta < 0.01) + y\%(0.01 < \eta < 0.1) + z\%(\eta > 0.1). \tag{9.2}$$

The above policy allots $x\%$ of capacity to flows with η values less than 0.01, $y\%$ capacity to η values between 0.01 and 0.1, and $z\%$ to η values larger than 0.1. On a system with EXP1 and POO2 sources, we could allocate 30% of link bandwidth to EXP1 sources and 50% to POO2 sources by the following quota policy:

$$30\%(\eta < 0.01) + 50\%(0.01 < \eta < 0.1). \tag{9.3}$$

The remaining 20% of link bandwidth will be allocated to various flows at the discretion of the admission control algorithm.

A worst-case quota mechanism. A straight forward implementation of a quota policy would be to partition link capacity according to the percentages expressed in the quota policy and to assign each portion to the corresponding η-class. Available bandwidth of each η-class is adjusted upon the arrival and departure of flows of that class. When an η-class exhausts its available bandwidth, no more flow of that class will be admitted until more bandwidth becomes available. The accounting of available bandwidth may be done fractionally according to the declared worst-case requirements of each flow.

A measurement-based quota mechanism. A quota mechanism that enforces quota conformance by the declared worst-case requirements of flows could result in low utilization. A *measurement-based* extension of the above algorithm allows an η-class that has exhausted its worst-case quota allotment to borrow from the pool of *measured* available bandwidth. Borrowing is permitted as long as there is enough left-over bandwidth to support both the borrowed amount and the non-consummated quotas of the other η-classes. The accounting of an η-class that borrows bandwidth shows a deficit. When a flow of an η-class with deficit leaves the network, the quota count of that class is incremented as usual. As long as an η-class is in deficit, flow admittance to that class must ensure enough provisioning for the non-consummated quotas of the other classes.

Simulation results. We run some simulations to evaluate the efficacy of the above two quota mechanisms. In our evaluation we test our ability to control the resulting mixture of a link's traffic such that flows requesting large amount of resources are not unintentionally discriminated. We further require that our quota mechanism does not unduly lower link utilization. All the results reported here are from simulations on the One-Link topology of Fig. 6.1(a). In the tables below, the NQ rows contain results from simulations with no quota mechanism, the WQ rows contain results from the worst-case quota mechanism, and the MQ rows contain results from the measurement-based quota mechanism. In all scenarios that implement a quota mechanism, there are two η-classes. The tuple following WQ or MQ contains the percentages of bandwidth that constitute the quota policy for the two classes. For each simulation, we report the number of concurrently active flows in each η-class after the warmup period. The second column of all the tables show the number of concurrently running flows in the first η-class, the third column the second. On all tables, the "%Util" column shows the average link utilization after the warmup period.

For the first set of simulations, we use source models EXP1 and fARIMA. Table 9.1 shows the simulation results. On the 10 Mbps bottleneck link of the One-link topology, the η of fARIMA sources is 0.1. This η-class is alloted 80% of the utilization target (or 90% of link bandwidth), which allows accommodation of 7 fARIMA flows. The table shows that this quota is honored in both WQ and MQ cases, and that there are more fARIMA flows when quota is instituted. The WQ case shows low utilization, as predicted, and the measurement extension does increase utilization back to the level achieved without quota. For this particular source models, the utilization gain of the measurement-based quota mechanism benefits only the EXP1 sources; nevertheless,

Table 9.1: Efficacy of Quota Mechanisms

Scheme	EXP1	fARIMA	%Util
NQ	184	3	78.23
WQ(20, 80)	28	7	51.58
MQ(20, 80)	121	7	79.26

Table 9.2: Benefits of Measurement-based Quota Mechanism

Scheme	EXP1	EXP3	%Util
NQ	190	43	67.86
WQ(50, 50)	71	8	23.57
WQ(10, 90)	58	10	19.37
MQ(50, 50)	204	44	68.63
MQ(20, 80)	198	49	68.
MQ(10, 90)	189	60	68.35

the quota policy is met. In simulations with the EXP1 and EXP3 source models, both kinds of sources benefit from the measurement-based quota mechanism, as shown in Table 9.2.

From the simulation results, we conclude that given a metric with which to compare the resource requirement of various flows, we can implement a quota mechanism to regulate the traffic mix of a link. A quota policy can then be instituted on top of this mechanism to prevent the exclusion of resource demanding flows. The measurement-based quota mechanism restore the achievable link utilization lowered by the worst-case quota mechanism.

Chapter 10

Comparison of Admission Control Algorithms

In this chapter, we report on a comparative study of five admission control algorithms to support the controlled-load service model described in Chapter 3. Since the role of admission control is to ensure that service commitments are not violated, the main criterion used in evaluating any admission control algorithm must be how well it fulfills this role. The simplest way to ensure complete conformance to commitments made is to give each flow enough resources to meet its worst-case requirements. For bursty sources, however, this scheme ultimately results in low network utilization. Hence, the second evaluation criterion is how high a level of network utilization an admission control algorithm can achieve while still meeting its service commitments. The third evaluation criterion is the implementation and operational costs of an algorithm. An algorithm that can achieve high level of utilization without violating any service commitments would not be useful if it cannot be implemented in a cost effective manner, or if it cannot drive a fast link. We only consider the first two criteria in this study. Since admission control is a session-level, not packet-level, control mechanism, we do not expect its implementation or operational cost to be a prohibitive factor. On the other hand, doing measurement could be operationally expensive.

Our evaluation of the algorithms involve simulating them under various scenarios. We have tried to make the simulation environments under which we investigate the behavior of the various algorithms as comparable as possible, but this does not mean the operating conditions would not be unfairly disadvantageous to any particular algorithm. In our evaluation of these algorithms we try to answer the question: which algorithm provides the highest level of network utilization at the lowest packet loss rate or experienced delay? The main conclusion of our study is: To satisfy controlled-load service commitments, the admission control algorithm must identify a link utilization target conditioned upon the characteristics of observed traffic. Formal attempts to compute this utilization target, such as the ones found in references [GKK95, Flo96a], that rely solely on peak rate and token bucket filter characterization of sources and do not take into account sources' burst length and idle times distributions can be either too optimistic or too conservative. In an environment where best-effort traffic continues to constitute a large fraction of bandwidth, *ad-hoc* methods of engineering the utilization target shall perform very well.

10.1 Five Admission Control Algorithms

Simple Sum. The first admission control algorithm simply ensures that the sum of requested resources does not exceed link capacity. Let ν be the sum of reserved rates, μ the link bandwidth, α the name of a flow requesting admission, and r^α the rate requested by flow α. This algorithm accepts the new flow if the following check succeeds:

$$\nu + r^\alpha < \mu. \tag{10.1}$$

We call this the "Simple Sum" algorithm. This is the simplest admission control algorithm and hence is being most widely implemented by switch and router vendors. Often, to ensure low queueing delay called for by controlled-load service, an approximation of the weighted fair queueing (WFQ) scheduling discipline is implemented with this admission control algorithm. WFQ assigns each flow its own queue served at its own reserved rate, thereby isolating flows from each other's bursts. We use WFQ with the "Simple Sum" admission control algorithm in this study—incidentally, this setup also satisfies the *committed rate* service model described in [BGK96]. For the other, measurement-based algorithms, we use first-in-first-out (FIFO) scheduling discipline.

Measured Sum. Whereas the "Simple Sum" algorithm ensures that the sum of existing reservations plus a newly incoming reservation does not exceed capacity, the "Measured Sum" algorithm uses measurement to estimate the load of existing traffic. This algorithm admits the new flow if the following test succeeds:

$$\widehat{\nu} + r^{\alpha} < \upsilon\mu, \tag{10.2}$$

where υ is a user-defined utilization target as explained in Chapter 5, and $\widehat{\nu}$ the measured load of existing traffic. We let $\upsilon = 0.9$ except otherwise noted. The measurement mechanism is the time-window measurement mechanism described in Chapter 5. Upon admission of a new flow, the load estimate is increased with:

$$\widehat{\nu}' = \widehat{\nu} + r^{\alpha}. \tag{10.3}$$

Admissible Region. The second measurement-based algorithm as proposed by the authors of reference [GKK95] computes an admissible region that maximizes the reward of utilization against the penalty of packet loss. Given link bandwidth, switch buffer space, a flow's token bucket filter parameters, the flow's burstiness, and desired probability of actual load exceeding bound, one can compute an admissible region for a specific set of flow types, beyond which no more flow of those particular types should be accepted. The computation of the admissible region assumes Poisson call arrival process and independent, exponentially distributed call holding times. However, the authors of [GKK95] claim that this algorithm is robust against fluctuations in the value of the assumed parameters. We refer the interested readers to [GKK95] for the computation of the admissible region. The measurement-based version of this algorithm ensures that the measured instantaneous load plus the peak rate of a new flow is below the admissible region. Even though reference [GKK95] does not specify adjusting measured load upon admittance of a new flow, we adjust the measured load according to the admission check by adding the new flow's peak rate (p^α) to it upon admitting a new flow α:

$$\widehat{\nu}' = \widehat{\nu} + p^\alpha. \tag{10.4}$$

For flows described by a token bucket filter (r, b) but not peak rate, we derive their peak rates ($\widehat{p}$) from the token bucket parameters using the equation:

$$\widehat{p} = r + b/U, \tag{10.5}$$

where U is a user-defined averaging period [Flo96a]. If a flow is rejected, the admission control algorithm does not admit another flow until an existing one leaves the network. In the remainder of this chapter, we use the terms "utilization target" and "utilization threshold" interchangeably with "admissible region."

Equivalent Bandwidth. The third measurement-based algorithm computes the equivalent bandwidth for a set of flows using the Hoeffding bounds, as explained in Section 2.2. To recapitulate, the equivalent bandwidth of a set of flows is the bandwidth $C(\epsilon)$ such that the stationary bandwidth requirement of the set of flows exceeds this value with probability at most ϵ. We call ϵ the "loss rate" in the remainder of the chapter; however, as pointed out in Section 2.2, in an environment where large portion of traffic is best-effort traffic, realtime traffic rate exceeding its equivalent bandwidth is not lost but simply encroaches upon best-effort traffic. In such an environment, ϵ is more appropriately called the "overflow rate." We make no such distinction in the remainder of this chapter. Following [GKK95], we use ϵ = 1e-12, except otherwise noted. The admission control check when a new flow α requests admission is:

$$\widehat{C}_H + p^\alpha \leq \upsilon\mu, \tag{10.6}$$

where $\widehat{C}_H$ is defined in Eqn. 2.15. In reference [Flo96a], the author mentions that instead of measuring average arrival rate, measuring average bandwidth actually used would be sufficient. We use measured arrival rate in our study of this algorithm and measured actual bandwidth usage for the other algorithms. Upon admission of a new flow, the load estimate is increased using Eqn. 10.4. Again, if a flow's peak rate is unknown, it is derived from its token bucket filter parameters (r, b) using Eqn. 10.5. Similar to the algorithm in [GKK95], if a flow is denied admission, no other flow of similar type will be admitted until an existing one departs.

Bounded Delay. The last measurement-based algorithm we consider is our own algorithm. Whereas the previous four algorithms bound bandwidth usage in their admission decisions, our algorithm bounds both bandwidth usage and experienced delay. Since controlled-load service consists only of one service level, we use only a subset of our algorithm: when a new flow α requests admission to the network, we use the "Measured Sum" algorithm above to check that the bandwidth requirements of admitted flows will be met; then we check that the delay bound (D) of existing traffic will not be violated by the admittance of the new flow. Presumably the delay bound of a flow is defined as $D = b/r$, where r and b are its token bucket parameters. The flow α is denied admission if it fails the following check:

$$\widehat{D} + \frac{b^\alpha}{\mu} < D, \tag{10.7}$$

where $\widehat{D}$ is the measured delay. Upon admittance of a new flow, we adjust both the load measure (using Eqn. 10.3) and the delay measure, by adding b^α/μ to the delay estimate.

We would like to remind the readers that while the admission control algorithms described here are based on meeting quality of service constraints of either loss rate or delay bound, the specific values used by the admission control algorithms are not advertised to the users of controlled-load service.

10.2 Exponential-Weighted Moving Average

We use the same time-window measurement mechanism described in Chapter 5 to measure network load with all but the equivalent bandwidth based admission control algorithms. With the equivalent bandwidth based admission control algorithm, we use an exponential-weighted moving average method to estimate the average arrival rate as suggested in reference [Flo96a]. The average arrival rate ($\widehat{\nu}^S$) is measured once every S sampling period. The average arrival rate is then computed using an infinite impulse response function with weight w, which we set to 2e-3 in this study:

$$\widehat{\nu}' = (1 - w) * \widehat{\nu} + w * \widehat{\nu}^S. \tag{10.8}$$

If the traffic arrival rate changes abruptly from 0 to 1 and then remains at 1, a w of 2e-3 allows the estimate to reach 75% of the new rate after 10 sampling periods. A larger w makes the averaging process more adaptive to load changes; a smaller w gives a smoother average by keeping a longer history. Recall that the equivalent bandwidth based admission control algorithm requires peak rate policing and derives a flow's peak rate from its token bucket parameters using Eqn. 10.5 when the peak rate is not explicitly specified. The author of [Flo96a] suggests that U should be set smaller than S, the sampling period of the measurement mechanism [Flo96b]. In this study, we let $U = S$ to reflect the peak rate seen by the measurement mechanism. A smaller S not only makes the measurement mechanism more sensitive to bursts, it also makes the peak rate derivation more conservative. A larger S may result in lower averages, however it also means that the measurement mechanism keeps a longer history because the averaging process (Eqn. 10.8) is invoked less often.

10.3 Simulation Results

We run our simulations on the One-Link and Four-Link topologies, described in Chapter 6, with the Exponential-ON/OFF (EXP) and Pareto-ON/OFF (POO) source models. Table 10.1 summarizes the six instantiation of the two models. Sources EXP1, EXP2, EXP3, POO1, and POO2 are the same ones we have been using throughout the dissertation. Columns in Table 10.1 that have the same names as the ones in Table 6.2 have the same meaning. We refer the readers to Section 6.3 for their descriptions.

Table 10.1: Six Instantiations of the Two Source Models

Model Name	Model Parameters				TB Filter			Switch Parameters				
	p pkt/ sec	I msec	N pkts	p/a	r tkn/ sec	b tkns	max qlen	D^* msec	D msec	Υ (%)	S ptt	$\widehat{p}$ pkt/ sec
EXP1	64	325	20	2	64	1	0	16	16	97	5e3	–
EXP2	1024	90	10	10	320	50	17	160	160	41	1e3	832
EXP3	∞	684	9	∞	512	80	1	160	160	–	5e2	2150
				β								
POO0	64	325	20	1.2	64	1	0	16	16	97	5e3	–
POO1	64	2925	20	1.2	64	1	0	16	16	97	5e3	–
POO2	256	360	10	1.9	240	60	220	256	160	41	5e3	363

The maximal delay for each source, listed in column 8, is also the "burst time" queueing delay acceptable under the definition of controlled-load service, given its assigned token bucket filter. Again, we have chosen the token bucket parameters such that, in most cases, the delay bounds given to a flow will be the same as its "burst time" queueing delay. This facilitates analyzing the performance of the algorithms under controlled-load service. For each simulation with measurement-based admission control algorithm, we size the buffer at the switches with enough space to accommodate the delay bound (D). For example, simulations with EXP1 source, given a link speed of 10 Mbps, use a buffer size of 160 packets. In simulations with multiple source models having different delay bound requirements, we use the maximum of the required buffer sizes; for example, in a simulation with both EXP1 and EXP2 models, we use a buffer size of 1600 packets. Simulations with the parameter-based admission control algorithm assume infinite buffer size. Column 10, labeled Υ, contains the utilization threshold used when simulating each of the source model with the admissible region based admission control algorithm. This should not be confused with the utilization target used with the other measurement-based admission control schemes, where the value is set to 90% link bandwidth. When we simulate more than

Table 10.2: Single-hop Homogeneous Sources Simulation Results

Model	Simple Sum		Measured Sum		Adm. Rgn.		Eqv. Bw.	
Name	%Util	#Actv	%Util	#Actv	%Util	#Actv	%Util	#Actv
EXP1	46	144	79	250	85	266	57	178
EXP2	28	28	75	74	19	18	8	8
EXP3	2	18	54	406	–	–	0.1	1
POO0	39	144	86	330	92	355	56	213
POO1	7	144	78	1539	83	1629	31	616
POO2	3	38	72	965	26	347	1	13

one source models with the admissible region based admission control algorithm, we use the most conservative of the utilization threshold. For example, in a simulation with both EXP1 and EXP2 sources, we use a utilization threshold of 41%. The next column, labeled S, gives the sampling period used with the measurement mechanisms in packet transmission time (*ptt*). For the time-window mechanism, the window size is $10 * S$. Both the equivalent bandwidth and admissible region based algorithms take ϵ as a parameter in their admission computation. Following [GKK95], we use $\epsilon =$ 1e-12, except where otherwise noted. Both the equivalent bandwidth and admissible region based algorithms also need to derive a flow's peak rate using Eqn. 10.5 when the flow's token bucket depth is greater than 1. The last column, labeled $\widehat{p}$, contains the derived peak rates. Note that for source POO2, the derived peak rate is larger than the actual peak rate. We also look at using the token rate (r) as the peak rate in our simulations of the equivalent bandwidth and admissible region based algorithms below. Flow interarrival times, durations, and warmup periods are as explained in Section 6.3.

The single-hop, homogeneous sources case. We now present simulation results from simulations on the ONE-LINK topology. A summary of the results is presented in Table 10.2. Each row of the table contains results from up to six simulations using the source model named at the leftmost column and the admission control algorithm indicated at the head of the columns. The "%Util" columns list the average utilization achieved at the bottleneck link of the ONE-LINK topology. The "#Active" columns list the average number of concurrently running flows in steady state.

The first two columns of Table 10.2 show results from simulation using the "Simple Sum" parameter-based admission control algorithm. There are no lost packets. The second set of two columns show results from the "Measured Sum" algorithm. For the scenarios simulated here where there is only a single-level of service and the delay bounds are very loose, we do not see any discernible difference between results from "Measured Sum" and those from bounded delay based algorithms, hence we do not show results from the bounded delay algorithm. Except for the POO0 cases, where both the "Measured Sum" and bounded delay algorithms give a loss rate on the order of 1e-7, simulations with other source models using these two algorithms do not result in any loss. For both algorithms, we can achieve no loss with POO0 sources if we reduce the utilization target to 80% of link bandwidth; in which case, the average link utilization achieved is 77% and the average number of concurrently served flows is 297. Alternatively, we could also achieve no loss with POO0 sources under the bounded delay algorithm by maintaining utilization target at 90% link bandwidth, but reducing delay bound to 8 ms, keeping buffer space at 160 packets; in this case, the achievable average link utilization is 80%, the average number of concurrently active flows is 251.

The next two columns, under the heading "Adm. Rgn." give the results of simulations with the admissible region algorithm; here the peak rate of sources with token bucket greater than 1 is derived from their token bucket parameters using Eqn. 10.5. We do not study the performance of this algorithm for EXP3 source because the utilization threshold for this model comes out to be 0 using the computation provided in [GKK95]. The loss rate for sources POO0 and POO1 are 1e-4 and 1e-6 respectively, much higher than ϵ of 1e-12 used to compute the utilization targets. While the performance of this algorithm is impressive for the EXP1 source, it results in too many losses for the POO0 and POO1 sources. Since the EXP1 and POO0 sources have mostly the same characteristics except for the distribution of their ON and OFF times, we conclude that by not taking these into account, the admissible region algorithm becomes overly optimistic when given sources with heavy-tailed ON and OFF times distributions. As the grain size of flows, i.e. the ratio p/μ, becomes larger, this algorithm becomes more conservative. For the EXP2 and POO2 sources, the achievable utilization is only 25% to 36% of the "Measured Sum" utilization; the same measurement mechanism is used with both algorithms. We next consider the effect of peak rate derivation on the performance of the algorithm. We run some simulations using the admissible region algorithm where the peak rate is assumed to be the token bucket rate, ignoring the bucket depth. The utilization thresholds used with the EXP2 and POO2 sources in these simulations are maintained at 41% of link bandwidth, as in the previous case. The performance of the algorithm using this more lax "peak rate" does not improve much: for EXP2 model, the average number of concurrently served flows becomes 25 at average link utilization of 25%, for POO2 model, the numbers are 395 and 29% respectively. The limitation to the average number of admissible flows is inherent in the computation of the utilization threshold.

Analysis of the equivalent bandwidth algorithm. The two columns of Table 10.2 under the heading "Eqv. Bw." show results from simulations using the equivalent bandwidth based admission control algorithm; here the peak rate of sources with token bucket depth greater than 1 is similarly derived using Eqn. 10.5. Comparing the "Eqv. Bw." columns against results from the other measurement-based algorithms, one sees that even though it is measurement-based, the performance of this algorithm is not much better than the parameter-based "Simple Sum" one. To better understand the equivalent bandwidth algorithm, we take a closer look at Eqn. 2.15:

$$\widehat{C}_H(\widehat{\nu}, \{p_i\}_{1\leq i\leq n}, \epsilon) = \widehat{\nu} + \sqrt{\frac{ln(1/\epsilon)\sum_{i=1}^{n}(p_i)^2}{2}}. \tag{10.9}$$

One realizes that a smaller $\widehat{\nu}$, i.e. an estimator that more closely tracks actual utilization, will give a smaller estimated equivalent bandwidth, resulting in higher flow admittance rate. Increasing the sampling frequency of the exponential averaging process, i.e. using smaller S, makes the estimator more adaptive and gives us an estimate that is closer to actual utilization. Indeed in a simulation of EXP1 sources, using S of 1e2, we see 182 concurrently active flows, achieving link utilization of 58%. Can we do better with an even more accurate measurement mechanism? Notice that in scenarios with homogeneous sources like we have here, knowing the peak (p) and average (a) rates of the sources, we can deterministically compute the average number of flows admissible under the equivalent bandwidth method by solving for n in the quadratic equation:

$$C_H = na + \sqrt{\frac{ln(1/\epsilon)np^2}{2}}. \tag{10.10}$$

The number of admissible flow is the n that also satisfies $C_H - na > 0$. Achievable utilization is then na/μ. For the EXP1 source and $C_H = 0.9 \cdot 10$Mbps, the admissible number of flow is $n = 186$, with achievable utilization $na/\mu = .58$. This means that independent of the accuracy of the measurement mechanism, the equivalent bandwidth method cannot admit more than 186 flows. Even an off-line, *post facto* re-run of the simulations using actual utilization as the "measured" average arrival rate will not result in higher number of admitted flows.

One could admit more flows into the network if actual aggregate utilization of n flows is lower than na, where a is the sources' declared average rate. Recall that the POO0 model has the same peak and average rates as EXP1 model but that it has heavy tailed ON and OFF times distributions, which leads to burstier aggregate traffic. Table 10.2 shows that the burstier aggregate traffic results in more POO0 flows being admitted under all measurement-based algorithms when compared to the number of admitted EXP1 flows. But notice also that for the equivalent bandwidth case, even with the higher number of admitted flows, achievable utilization remains below 58%. This can be explained by analyzing Eqn. 10.10. Let Ω be the second term of Eqn. 10.10, i.e. $\Omega = \sqrt{\frac{ln(1/\epsilon)\sum_{i=1}^{n}(p_i)^2}{2}}$. Since the admission algorithm requires that $\widehat{C}_H < \upsilon\mu$, achievable utilization $\widehat{\nu}$ is constrained by

$$\widehat{\nu} \leq \upsilon\mu - \Omega. \tag{10.11}$$

Hence admitting 213 POO0 sources constrains link utilization to 56%, which is indeed the utilization achieved for the simulation. Similarly for POO1 sources, admitting 616 flows constrains achievable utilization to below 32%.

Aside from lowering $\widehat{\nu}$—either by using a better estimator or having sources send aggregate traffic below the computed average, we could also increase admittance rate by lowering Ω. The two variables in Ω are ϵ and the sources' peak rate (p). By lowering ϵ from 1e-12 to 1e-9, we can admit 192 EXP1 flows, achieving 61% utilization, up from 182 and 58% respectively. These increases are in close agreement with computation using Eqn. 10.10. For $\epsilon =$ 1e-1, Eqn. 10.10 gives 254 EXP1 flows at 79% link utilization, which also applies to POO0 sources. For POO1 sources, similar ϵ, Eqn. 10.10 gives 68% link utilization for 1086 flows. A closer investigation of Eqns. 10.10 and 10.11 reveals that achievable link utilization is bounded by $\widehat{\nu} = na$ when n is small and $\widehat{\nu} = \upsilon\mu - \Omega$ when n is large. For the EXP1 source, $\epsilon =$ 1e-12, the intersection of the two lines $\widehat{\nu} = na$ and $\widehat{\nu} = \upsilon\mu - \Omega$ is at $n = 187$ and $\widehat{\nu} = 59.8\%$, meaning that we can never hope to admit more than 187 EXP1 flows or achieve utilization higher than 59.8% link bandwidth, if we insist on $\epsilon =$ 1e-12. In effect, Ω acts as a safety zone to accommodate traffic that bursts beyond the measured average.

In the case of simulations with EXP2, EXP3 and POO2 sources, the flows' peak rates are derived from their token bucket parameters using the Eqn. 10.5. We showed the derived peak rates for the three sources in Table 6.2 and pointed out that in the POO2 case the derived peak rate is actually higher than the actual peak rate. In reference [Flo96a], the author suggests that the token bucket parameters be set with a small bucket depth and peak rate as token rate. The token bucket is thus only intended to "accommodate small variations in packet delay that accumulate in the network." To see how a less conservative peak rate effects the performance of the algorithm on the EXP2, EXP3, and POO2 sources, we simulate them with the token bucket rate of each as its peak rate, ignoring the token bucket depths. With token rate as peak rate, simulation with EXP2 sources achieve average link utilization of 57%, serving 56 concurrent flows; for EXP3 sources, the achieved average link utilization is 3%, with 21 concurrent flows; and for POO3 sources, the numbers are 8% and 102 flows respectively. While the performance of the algorithm improves by order of magnitude compared to the original case, they are scantly better compared to results from the other measurement-based algorithms. Next we experiment with ϵ = 1e-1, using the token rate as peak rate, for sources EXP2 and POO2. For EXP2, Eqn. 10.10 gives 63 flows at 63% utilization; POO2 results in 213 flows at 53% utilization. Note that these numbers are still lower than those achieved with the "Measured Sum" algorithm and we cannot relax any parameters further to increase them. We conclude that the equivalent bandwidth based method is inherently conservative. Incidentally, this exercise also points out the difficulty of deriving peak rate from the token bucket parameters. To be safe, the averaging period U in Eqn. 10.5 should be smaller than or equal to S, the measurement sampling period; on the other hand, too small a U could results in practically infinite peak rate when the bucket depth is large. Due to its conservativeness, we never experience packet loss with any of the simulations involving the equivalent bandwidth algorithm.

Our next attempt to improve the performance of the equivalent bandwidth algorithm introduces a gambling factor to Eqn. 10.11:

$$\widehat{\nu} \leq \upsilon\mu - (1 - \kappa)\Omega. \tag{10.12}$$

For $\kappa = 0$, we keep the original Ω safety zone. If, after an observation at time-scales of days or weeks, we decide that our traffic is not that bursty and we can safely increase link utilization, we can increase κ. For example, in a simulation with EXP1 sources, ϵ = 1e-12, setting $\kappa = 0.8$, we are able to admit 254 flows, achieving 81% link utilization with no lost packets. We hasten to add, however, introducing κ to the equivalent bandwidth equation destroys its rigorousness and makes it as *ad-hoc* as setting the utilization target of the "Measured Sum" algorithm.

Analyzing the experienced queueing delay. Aside from achievable utilization and loss rate, one might also be interested in packets' experienced delay under the

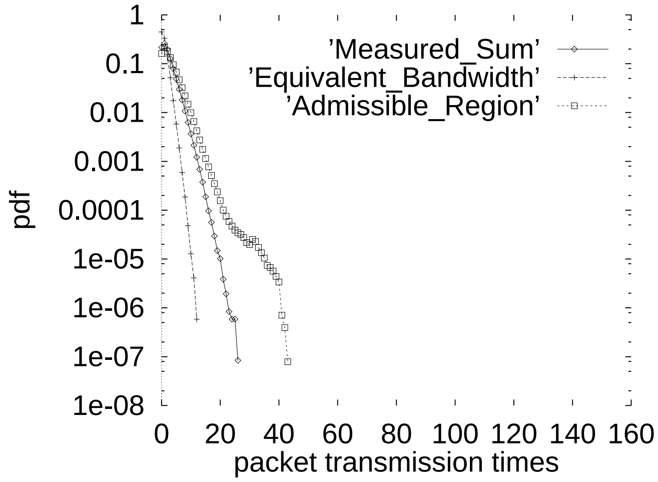


Figure 10.1: Distribution of experienced queueing delay of EXP1 sources.

various algorithms. In this section, we look at the distribution of experienced queueing delay under the "Measured Sum," admissible region, and equivalent bandwidth algorithms. Figs. 10.1, 10.2, and 10.3 show these distributions at the switch connected to the bottleneck link in topology ONE-LINK, for sources EXP1, POO0, and POO1 respectively. Recall that under the definition of controlled-load service model, the acceptable "average burst length" queueing delay for all three source models is 16 msecs. Comparing the experienced delay of EXP1 and POO0 under the admissible region admission control algorithm, one can see from Figs. 10.1 and 10.2 that for the same peak rate and degree of burstiness, POO0 sources must certainly be allowed a smaller utilization target than that used with EXP1 sources. However, as we pointed out earlier, the admissible region computation in [GKK95] does not take into account the distributions of ON and OFF times, resulting in massive losses for POO0 and POO1 sources under that algorithm. The burstier POO1 source gives us a shorter delay tail; note, however, that the distribution of POO1 experienced delays still exhibits a longer tail than that of EXP1 sources, attesting to the LRD effect on experienced queueing delay. Given the acceptable queueing delay of 16 ms, and the capability of the "Measured Sum" algorithm to exploit this bound to achieve a high level of link utilization without experiencing any loss, we think that the equivalent bandwidth based algorithm is too conservative. We mentioned earlier that even though we experience loss

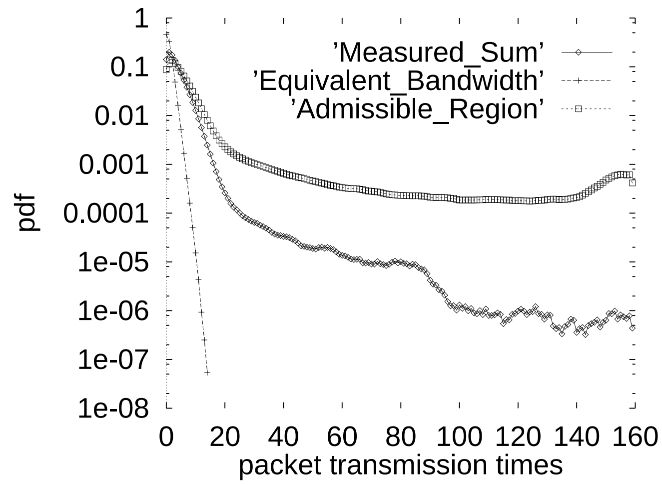


Figure 10.2: Distribution of experienced queueing delay of POO0 sources.

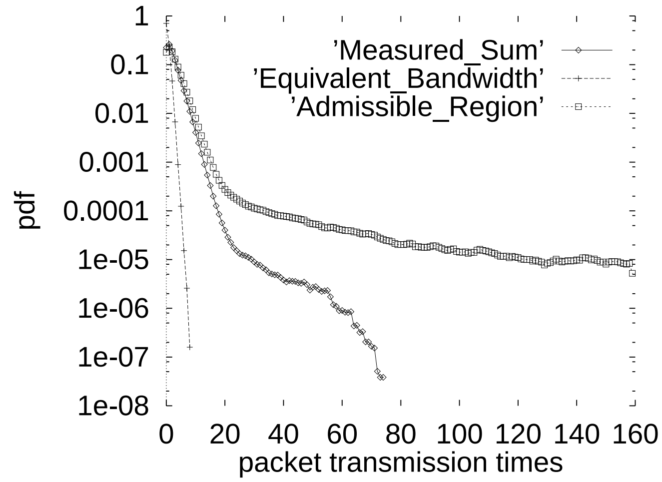


Figure 10.3: Distribution of experienced queueing delay of POO1 sources.

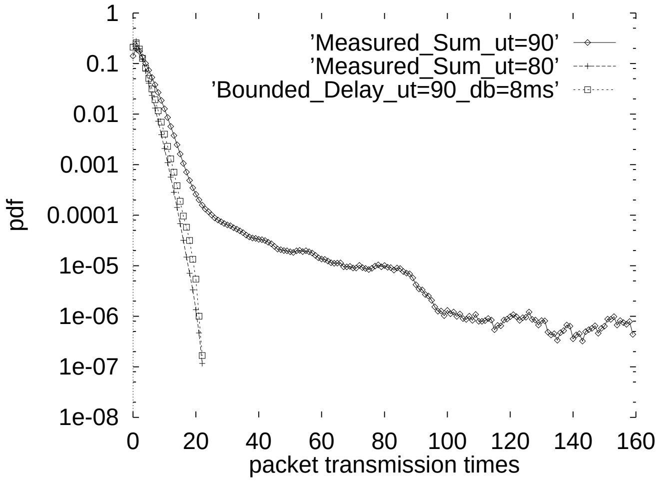


Figure 10.4: Distribution of experienced queueing delay of POO0 sources under the "Measured Sum" and bounded delay algorithms for different utilization target (ut) and delay bound (db).

Table 10.3: Multiple-hop All Sources Simulation Results

Link	Measured Sum		Adm. Rgn. 1		Adm. Rgn. 2		Equivalent Bw.	
	%Util	#Actv	%Util	#Actv	%Util	#Actv	%Util	#Actv
L6	47	282	19	117	48	285	33	237
L7	79	485	30	198	81	470	57	392
L8	77	469	29	189	79	454	60	404
L9	77	469	31	203	80	467	60	407

rate of 1e-7 for POO0 sources with the "Measured Sum" algorithm when the utilization target is set at 90% link bandwidth, we suffer no loss both when the utilization target is set to 80% and when we use the bounded delay algorithm with delay bound set to 8 ms. Fig. 10.4 shows the experienced delay of the three cases.

The multiple-hop, heterogeneous sources case. Table 10.3 contains the average link utilization and average number of connections of the four links in the FOUR-LINK topology from simulations where we run all six sources, with the choice of sources uniformly distributed. All the simulations use a sampling period of 1e3

Table 10.4: Percentage Composition of Type of Admitted Flows

Algorithm	EXP1	EXP2	EXP3	POO0	POO1	POO2
Measured Sum	21%	9	12	21	22	15
Admissible Rgn. 1	20	10	14	19	21	16
Admissible Rgn. 2	19	12	15	19	19	16
Equivalent Bw.	25	1	6	25	26	18

packet transmission times and buffer space for 1600 packets. For sources with token bucket depth greater than 1, we use the token bucket rate as the peak rate, ignoring the bucket depth. The table shows that the equivalent bandwidth based algorithm is, again, rather conservative in this scenario. The "Adm. Rgn. 1" scenario uses a utilization target of 41%, whereas the "Adm. Rgn. 2" scenario uses 97%. None of the simulation suffers any packet loss. Link L6 consistently achieves lower utilization than the other links. We called this the *under-representation* phenomenon in Chapter 9, and attributed its cause to un-consummated reservations when multi-hop flows admitted by the switch attached to L6 are rejected by one of the downstream switches. To better understand why, when compared to the "Adm. Rgn. 2" numbers, the larger number of flow counts under the "Measured Sum" algorithm results in lower utilization, we investigate the mix of admitted flows. Again, we do not include results from bounded delay algorithm because they are practically identical to the "Measured Sum" results. Table 10.4 shows the composition of the type of admitted flows, in percentages. We can immediately see that under "Admissible Rgn. 2," more flows with deeper bucket depth are admitted, resulting in higher utilization, even at a lower flow count, compared to the numbers of "Measured Sum." Table 10.4 also confirms our earlier observation that more resource demanding flows can suffer from another form of *under-representation*, where they are discriminated against by the network. This problem is even more pronounced in the "Equivalent Bw." case where the peak rates of all currently admitted flows are used in every admission decision. While the performance of the admissible region algorithm is excellent when the utilization threshold is 97%, the choice of this utilization threshold is not from computation in [GKK95], rather it is a "best case," though *ad-hoc*, choice for this scenario—hence does not allow the load estimation error to be quantified and assessed any more more rigorously than under the "Measured Sum" method.

The "Measured Sum" method seems to work as well as the bounded delay algorithm under the scenarios simulated here. The admissible region based algorithm suggested by the authors of reference [GKK95] is either too conservative when flows' grain size is large, or too optimistic when the flows' have heavy-tailed ON and OFF times distributions. The equivalent bandwidth based algorithm found in [Flo96a] is inherently conservative. In general, while it is clear that admission control algorithm for controlled-load service should have a utilization target, it is still not clear how to compute this bound from observed traffic characteristics. Computing equivalent bandwidth or admissible region, taking into account only the sources' peak rate and token bucket filter parameters, does not seem sufficient. One must also take into account the sources' burst lengths and idle times distributions. The "Simple Sum" method used in conjunction with WFQ scheduling discipline favored by router vendors for its implementation simplicity gives the worst performance in terms of link utilization. However, we have not studied the implementation and operational costs of the various admission control algorithms; when these are taken into account, one might not be able to implement anything more complicated than the "Simple Sum" algorithm, given current hardware technology.

Chapter 11

Summary and Extensions

In this dissertation we presented a measurement-based admission control algorithm that consists of two logically distinct pieces, the *criteria* and the *estimator*. The admission control criteria are based on an equivalent token bucket filter model, where each predictive class aggregate traffic is modeled as conforming to a single token bucket filter. This enables us to calculate worst case delays in a straightforward manner. The estimator produces measured values we use in the equations representing our admission control criteria. We have shown that even with the simplest measurement estimator, it is possible to provide a reliable delay bound for predictive service using our measurement-based admission control algorithm. We have also shown that our measurement-based admission control can be used with controlled-load service to provide the illusion of lightly loaded network. Thus we conclude that for those applications willing to tolerate delay violations, services with more relaxed commitments than those provided by guaranteed service are viable alternatives. For bursty sources, in particular, measurement-based admission control algorithm with the more relaxed services can achieve a level of network utilization significantly higher than those achievable under guaranteed service. Finally, we now identify three broad categories of possible extensions to our work:

11.1 A Better Estimator

It is essential for an admission control algorithm to set aside some slack bandwidth to accommodate sudden increases in traffic flow. The amount of bandwidth set aside for such purpose could be decided based on historical data such as we have done with our utilization target. A less *ad-hoc* method would be to compute the equivalent bandwidth of the aggregate traffic as proposed by the authors cited in Section 2.2. Unfortunately, because of the assumptions made by the different approaches to compute equivalent bandwidth, the resulting numbers are either too conservative for certain types of flow or too optimistic for other types of flow. Or the approach would be too restrictive and support only specific types of flow. Recent works on spectral analysis of network traffic, such as [LCH95], have identified the low frequency of traffic as a good indicator of bandwidth requirement, and the high frequency as indicator of buffer space requirement. However, other researchers have also called into question the reliability of traffic autocorrelation in predicting queueing behavior [HH96]. It is interesting to pursue how and when one might be able to peruse the spectral density of traffic in estimating adequate resource provisioning.

Another approach to a better estimator is to bound the error rates of the estimates. Reference [DJM96] contains such an approach. Given the reliance of estimating error rates on traffic characteristics, we doubt that this would be a promising approach. Of more interest to us is a fast implementation of the estimator, either in hardware or in software. References [C+91, WCKG94] contain possible hardware implementations of the estimator. Finally, we would like to implement an higher order mechanism to automatically tune the parameters of our algorithm over large time-scales.

11.2 Other Admission Criteria

In Chapter 9, we identify two kinds of flow under-representation problem: (1) flows with large resource requirements are discriminated at admission time, and (2) flows with multiple hops run a larger chance of being rejected by the network. The first of these discriminations is local to the decision of a switch, the second requires cooperation between switches. We showed in Chapter 9 that these problems are always present when service violation prevention is the only criterion used in making admission decisions, both parameter-based and measurement-based. To address the first kind of discrimination, we adopted in Chapter 9 another criterion wherein different kinds of flow are alloted their own quota. We also introduced a measurement-based quota mechanism that allows network administrators to control the traffic mix on their links. This mechanism relies on a flow opportunity cost metric to compare the resource requirement of one flow against that of others. Both a more accurate estimate of flows' actual opportunity cost and a more sophisticated quota mechanism would be interesting extensions to our work. We have not begun to address the second discrimination problem. To explore it is of of immediate interest to us.

11.3 Additional Issues

In this section we present three additional issues that could effect a measurement-based admission control algorithm.

Stability of adaptive playback point. Guaranteed service provides an absolute delay bound from which one can compute the bound on a packet's end-to-end delay. Predictive and controlled-load services do not provide such bound. While adaptive applications can adjust their playback point to accommodate variations in packets' end-to-end delay, one would still prefer a stable playback point. An interesting research project would be to study the stability of playback points for applications receiving predictive and controlled-load services. Would one be able to compute a stable end-to-end delay distribution given delay distributions at the switches along a flow's path?

Link sharing. In references [FJ95, SCZ93], the authors identify the need to partition link bandwidth into portions that are then sold to separate entities. This service is called *link-sharing* in the literature. To provide this service, the admission control algorithm must ensure that realtime traffic from each entity does not overflow its alloted portion. As we mentioned in this dissertation, a measurement-based admission control algorithm can deliver high degree of utilization gain only when there is an high degree of statistical multiplexing. The reliability of traffic estimates also depend on high degree of statistical multiplexing. Link-sharing lowers the degree of statistical multiplexing by segregating traffic into different partitions. One possible approach to regain an higher degree of statistical multiplexing is for the scheduler and traffic estimators to ignore link partitioning once a realtime flow has been determined by the admission control algorithm to conform to its entity's share. We plan to experiment with this architecture.

Preemptible Service. In Chapter 3, we mentioned sources that can transmit at variable bandwidth either by changing their compression ratio or by transmitting fewer levels of their hierarchically encoded data. In reference [HS96], each level of hierarchically encoded data is sent as a separate flow with resource reservation. To support variable bandwidth sources requires reconsideration of our measurement-based admission control algorithm. In the case where a source can adjust its transmission rate based on congestion feedback, our traffic estimates must ignore the extra traffic generated by the source when network is not congested. In the case where sources reserve bandwidth for each of their hierarchically encoded data, and adjust the number of levels they transmit based on congestion feedback, we must prioritize our dropping policy. One possible solution is to give lower scheduling priorities to higher levels traffic. However, this could cause massive packet reordering if the different levels of priority are scheduled by strict priority. We think the right approach is to transmit traffic from all hierarchies in the same level of scheduling priority and rely on packet dropping policy to drop the highest layer traffic first.

To facilitate prioritized dropping policy, we intend to introduce a meta service model: the preemptible service model. It is a meta service model in that it must be used in conjunction with one of the other service models mentioned previously (which we will call the base service). Packets of a preemptible flow will be dropped first before packets from non-preemptible flows of the same scheduling priority. A preemptible flow may also be completely dropped from service. We can support multiple levels of preemptible service with decreasing dropping priorities. An extra benefit of preemptible service is that one can admit preemptible flows that would otherwise be rejected because of quota or link share violation. Preemptible flows could also be dropped upon sudden surges of traffic or arrivals of an advance reservation start time. Note that when network is not congested, preemptible flows receive the same service as non-preemptible flows requesting the same base service. Hence preemptible service is not best-effort service. It is an interesting problem to study how a measurement mechanism must be designed to support preemptible flows. Do we include packets from preemptible flows in our measurement? If not, how shall we subtract them out, especially when measuring delay? If so, how do we recognize whether we can admit more non-preemptible flows given current load of preemptible traffic?

Bibliography

[AM95] A. Adas and A. Mukherjee. "On Resource Management and QoS Guarantees for Long Range Dependent Traffic. *Proc. of IEEE INFOCOM 1995*, Apr. 1995.

[AMS82] D. Anick, D. Mitra, and M.M. Sondhi. "Stochastic Theory of a Data-Handling System with Multiple Sources". *The Bell System Technical Journal*, 61(8):1871–1895, Oct. 1982.

[AS94] S. Abe and T. Soumiya. "A Traffic Control Method for Service Quality Assurance in an ATM Network". *IEEE Journal of Selected Areas in Communication*, 12(2):322–331, Feb. 1994.

[Ber94] J. Beran. *Statistics for Long-Memory Processes.* New York: Chapman & Hall, 1994.

[BGK96] F. Baker, R. Guérin, and D. Kandlur. *Specification of the Committed Rate Quality of Service.* Internet-Draft, Jun. 1996.

[BJ76] G.E.P. Box and G.M. Jenkins. *Time Series Analysis: Forecasting and Control.* New Jersey: Prentice Hall, 1976.

[Bol94] V.A. Bolotin. "Modeling Call Holding Time Distributions for CCS Network Design and Performance Analysis". *IEEE Journal of Selected Areas in Communication*, 12(3):433–438, Apr. 1994.

[Bre95] L. Breslau. *Adaptive Source Routing of Real-Time Traffic in Integrated Services Networks.* PhD thesis, USC, 1995.

[BSTW95] J. Beran, R. Sherman, M.S. Taqqu, and W. Willinger. "Long-range Dependence in Variable-Bit-Rate Video Traffic". *IEEE Transactions on Communications*, 43:1566–1579, 1995.

[C+91] M. Conti et al. “Interconnection of Dual Bus MANs: Architecture and Algorithms for Bandwidth Allocation”. *Journal of Internetworking: Research and Experience*, 2(1):1–22, March 1991.

[CB96] M.E. Crovella and A. Bestavros. “Self-Similarity in World Wide Web Traffic: Evidence and Possible Causes”. *Proc. of ACM SIGMETRICS'96*, May 1996.

[CESZ93] R. Cocchi, D. Estrin, S.J. Shenker, and L Zhang. “Pricing in Computer Networks: Motivation, Formulation, and Example”. *ACM/IEEE Transactions on Networking*, 1(6):614–627, Dec. 1993.

[Cha86] P-C. Chang. *Predictive, Hierarchical and Transform Vector Quantization for Speech Coding*. PhD thesis, Stanford University, 1986.

[CLG95] S. Chong, S-Q. Li, and J. Ghosh. “Predictive Dynamic Bandwidth Allocation for Efficient Transport of Real-Time VBR Video over ATM”. *IEEE Journal of Selected Areas in Communication*, 13(1):12–23, Jan. 1995.

[Cru91] R.L. Cruz. “A Calculus for Network Delay, Part I: Network Elements in Isolation”. *IEEE Transactions on Information Theory*, 37(1):114–131, Jan. 1991.

[CSZ92] D.D. Clark, S.J. Shenker, and L. Zhang. “Supporting Real-Time Applications in an Integrated Services Packet Network: Architecture and Mechanism”. *Proc. of ACM SIGCOMM '92*, pages 14–26, Aug. 1992. **URL** ftp://parcftp.parc.xerox.com/pub/net-research/csz_sigcomm92.ps.

[CT95] C.-S. Chang and J.A. Thomas. “Effective Bandwidth in High-Speed Digital Networks”. *IEEE Journal of Selected Areas in Communication*, 13(6):1091–1100, Aug. 1995.

[DJM96] Z. Dziong, M. Juda, and L.G. Mason. “A Framework for Bandwidth Management in ATM Networks — Aggregate Equivalent Bandwidth Estimation Approach”. *Submitted for publication*, 1996.

[DKPS95] M. Degermark, T. Köhler, S. Pink, and O. Schelén. “Advance Reservations for Predicted Service”. *Proc. 5th Int'l Network and Operating Systems Support for Digial Audio and Video Workshop*, pages 3–14, Apr. 1995.

[DKS89] A. Demers, S. Keshav, and S.J. Shenker. "Analysis and Simulation of a Fair Queueing Algorithm". *Proc. of ACM SIGCOMM '89*, pages 1–12, Sept. 1989.

[DLM93] Z. Dziong, K-Q. Liao, and L. Mason. "Effective Bandwidth Allocation and Buffer Dimensioning in ATM Based Networks with Priorities". *Computer Networks and ISDN Systems*, 25:1065–1078, 1993.

[DM96] Z. Dziong and L.G. Mason. "Fair-Efficient Call Admission Control Policies for Broadband Networks—A Game Theoretic Framework". *ACM/IEEE Transactions on Networking*, 4(1):123–136, Feb. 1996.

[DMRW94] D.E. Duffy, A.A. McIntosh, M. Rosenstein, and W. Willinger. "Statistical Analysis of CCSN/SS7 Traffic Data from Working CCS Subnetworks". *IEEE Journal of Selected Areas in Communication*, 12(3):544–551, Apr. 1994.

[DTVV90] M. Decina, T. Toniatti, P. Vaccari, and L. Verri. "Bandwidth Assignment and Virtual Call Blocking in ATM Networks". *Proc. of IEEE INFOCOM '90*, pages 881–888, 1990.

[dVKW95] G. de Veciana, G. Kesidis, and J. Walrand. "Resource Management in Wide-Area ATM Networks Using Effective Bandwidths". *IEEE Journal of Selected Areas in Communication*, 13(6):1081–1090, Aug. 1995.

[EM93] A.I. Elwalid and D. Mitra. "Effective Bandwidth of General Markovian Traffic Sources and Admission Control of High Speed Networks". *ACM/IEEE Transactions on Networking*, 1(3):329–343, Jun. 1993.

[EMW95] 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 of Selected Areas in Communication*, 13(6):1115–1127, Aug. 1995.

[ENW96] A. Erramilli, O. Narayan, and W. Willinger. "Experimental Queueing Analysis with Long-Range Dependent Packet Traffic". *ACM/IEEE Transactions on Networking*, 4(2):209–223, Apr. 96.

[Fil89] J. Filipiak. "Structured Systems Analysis Methodology for Design of an ATM Network Architecture". *IEEE Journal of Selected Areas in Communication*, 7(8):1263–1273, Oct. 1989.

[FJ95] S. Floyd and V. Jacobson. "Link-Sharing and Resource Management Models for Packet Networks". *ACM/IEEE Transactions on Networking*, 3(4):365–386, Aug. 1995.

[Flo96a] S. Floyd. "Comments on Measurement-based Admissions Control for Controlled-Load Service". Submitted to *Computer Communication Review*, 1996. **URL** ftp://ftp.ee.lbl.gov/papers/admit.ps.Z.

[Flo96b] S. Floyd. Personal communication. E-mail, Jun. 29, 1996.

[FV90] D. Ferrari and D.C. Verma. "A Scheme for Real-Time Channel Establishment in Wide-Area Networks". *IEEE Journal of Selected Areas in Communication*, 8(3):368–379, 1990.

[GAN91] R. Guérin, H. Ahmadi, and M. Naghshineh. "Equivalent Capacity and Its Application to Bandwidth Allocation in High-Speed Networks". *IEEE Journal of Selected Areas in Communication*, 9(7):968–981, Sept. 1991.

[GB96] M. Grossglauser and J-C. Bolot. "On the Relevance of Long-Range Dependence in Network Traffic". *Proc. of ACM SIGCOMM '96*, 1996. **URL** http://www.inria.fr/rodeo/personnel/mgross/WWW/Papers/sigcomm96.ps.gz.

[GG91] M. Gilge and R. Gusella. "Motion Video Coding for Packet-Switching Networks — An Integrated Approach". *Proceedings of the SPIE Conference on Visual Communications and Image Processing*, Nov. 1991.

[GG93] L. Gün and R. Guérin. "Bandwidth Management and Congestion Control Framework of the Broadband Network Architecture". *Computer Networks and ISDN Systems*, 26:61–78, 1993.

[GKK95] R.J. Gibbens, F.P. Kelly, and P.B. Key. "A Decision-Theoretic Approach to Call Admission Control in ATM Networks.". *IEEE Journal of Selected Areas in Communication*, 13(6):1101–1114, Aug. 1995.

[GKT95] M. Grossglauser, S. Keshav, and D. Tse. "RCBR: A Simple and Efficient Service for Multiple Time-Scale Traffic". *Proc. of ACM SIGCOMM '95*, pages 219–230, 1995.

[Gol91] S.J. Golestani. "A Framing Strategy for Congestion Management". *IEEE Journal of Selected Areas in Communication*, 9(7):1064–1077, Sept. 1991.

[GV93] M. Garrett and M. Vetterli. "Joint Source/Channel Coding of Statistically Multiplexed Real-Time Services on Packet Networks". *ACM/IEEE Transactions on Networking*, 1(1):71–80, Feb. 1993.

[GW94] M. Garrett and W. Willinger. "Analysis, Modeling and Generation of Self-Similar VBR Video Traffic". *Proc. of ACM SIGCOMM '94*, pages 269–279, Sept. 1994.

[HH96] B. Hajek and L. He. "On Variations of Queue Response for Inputs with Identical Mean and Autocorrelation Functions". *Proc. Conference on Information Sciences and Systems*, 1996. **URL** http://tesla.csl.uiuc.edu:80/ hajek/Papers/Qvar.ps.

[Hir91] A. Hiramatsu. "Integration of ATM Call Admission Control and Link Capacity Control by Distributed Neural Network". *IEEE Journal of Selected Areas in Communication*, 9(7):1131–1138, Sept. 1991.

[HLP93] J.M. Hyman, A.A. Lazar, and G. Pacifici. "A Separation Principle Between Scheduling and Admission Control for Broadband Switching". *IEEE Journal of Selected Areas in Communication*, 11(4):605–616, May 1993.

[Hos81] J.R.M. Hosking. "Fractional Differencing". *Biometrika*, 68(1):165–176, 1981.

[HR89] J. Haslett and A.E. Raftery. "Space-time Modelling with Long-memory Dependence: Assessing Ireland's Wind Power Resource". *Applied Statistics*, 38(1):1–50, 1989.

[HS96] D. Hoffman and M. Speer. "Hierarchical Video Distribution over Internet-Style Networks". *ICIP*, 1996.

[Hui88] J.Y. Hui. "Resource Allocation for Broadband Networks". *IEEE Journal of Selected Areas in Communication*, 6(9):1598–1608, Dec. 1988.

[Hui95] C. Huitema. To share rather than to pay. Panel Discussion on *Reservation or No Reservation*, Infocom-95, 1995.

[JDSZ95] S. Jamin, P. B. Danzig, S. J. Shenker, and L. Zhang. "A Measurement-based Admission Control Algorithm for Integrated Services Packet Networks". *Proc. of ACM SIGCOMM '95*, pages 2–13, 1995.
URL http://netweb.usc.edu/jamin/admctl/sigcomm95.ps.Z.

[JDSZ96] S. Jamin, P. B. Danzig, S. J. Shenker, and L. Zhang. "A Measurement-based Admission Control Algorithm for Integrated Services Packet Networks (Extended Version)". *ACM/IEEE Transactions on Networking*, 1996. **URL** http://netweb.usc.edu/jamin/
admctl/ton96.ps.Z.

[JSZC92] S. Jamin, S.J. Shenker, L. Zhang, and D.D. Clark. "An Admission Control Algorithm for Predictive Real-Time Service (Extended Abstract)". *Proc. 3rd Int'l Network and Operating Systems Support for Digial Audio and Video Workshop*, Nov. 1992.
URL http://netweb.usc.edu/jamin/admctl/nossdav92.ps.Z.

[Kel91] F.P. Kelly. "Effective Bandwidths at Multi-Class Queues". *Queueing Systems*, 9:5–16, 1991.

[KM94] S.M. Klivansky and A. Mukherjee. On long-range dependence in nsfnet traffic. Technical Report GIT-CC-94-61, Gerogia Insititue of Technology, December 1994.

[KMR93] H. Kanakia, P.P. Mishra, and A. Reibman. "An Adaptive Congestion Control Scheme for Real-Time Packet Video Transport". *Proc. of ACM SIGCOMM '93*, pages 20–31, Sept. 1993.

[KS85] B. Kraimeche and M. Schwartz. "Analysis of Traffic Access Control Strategies in Integrated Service Networks". *IEEE Transactions on Communications*, COM-33(10):1085–1093, Oct. 1985.

[KS89] T. Kamitake and T. Suda. "Evaluation of an Admission Control Scheme for an ATM Network Considering Fluctuations in Cell Loss Rate". *Proc. of IEEE GLOBECOMM '89*, pages 1774–1780, 1989.

[KU93] Y-H. Kim and C-K. Un. "Analysis of Bandwidth Allocation Strategies with Access Control Restrictions in Broadband ISDN". *IEEE Transactions on Communications*, 41(5):771–781, May 1993.

[Kur92] J. Kurose. "On Computing Per-session Performance Bounds in High-Speed Multi-hop Computer Networks". *Proc. of ACM SIGMETRICS'92*, pages 128–139, Jun. 1992.

[KWC93] G. Kesidis, J. Walrand, and C-S. Chang. "Effective Bandwidths for Multiclass Markov Fluids and Other ATM Sources". *ACM/IEEE Transactions on Networking*, 1(4):424–428, Aug. 1993.

[LCH95] S-Q. Li, S. Chong, and C-L. Hwang. "Link Capacity Allocation and Network Control by Filtered Input Rate in High-Speed Networks". *ACM/IEEE Transactions on Networking*, 3(1):10–25, Feb. 1995. **URL** http://mocha.ece.utexas.edu/ sanqi/papers/link-cap.ps.

[LPP90] A. Lombardo, S. Palazzo, and D. Panno. "A Framework for Sharing Bandwidth Resources Among Connectionless and Connection-Oriented Services in B-ISDNS". *7th International Teletraffic Congress*, Sept. 1990.

[LTWW94] W.E. Leland, M.S. Taqqu, W. Willinger, and D.V. Wilson. "On the Self-Similar Nature of Ethernet Traffic (Extended Version)". *ACM/IEEE Transactions on Networking*, 2(1):1–15, Feb. 1994.

[LV93] S.H. Low and P.P. Varaiya. "A New Approach to Service Provisioning in ATM Networks". *ACM/IEEE Transactions on Networking*, 1(5):547–553, Oct. 1993.

[LW91] W.E. Leland and D.V. Wilson. "High Time-Resolution Measurement and Analysis of LAN Traffic: Implications for LAN Interconnection". *Proc. of IEEE INFOCOM '91*, 1991.

[MH+91] K. Meier-Hellstren et al. "Traffic Models for ISDN Data Users: Office Automation Applications". *International Teletraffic Congress13th*, pages 167–172, Jun. 1991.

[Mit88] D. Mitra. "Stochastic Theory of a Fluid Model of Producers and Consumers Coupled by a Buffer". *Advance Applied Probability*, 20:646–676, 1988.

[MJV96] S. McCanne, V. Jacobson, and M. Vetterli. "Receiver-driven Layered Multicast". *Proc. of ACM SIGCOMM '96*, Sep. 1996. **URL** ftp://ftp.ee.lbl.gov/papers/mccanne-sigcomm96.ps.gz.

[Mol27] E.C Molina. "Application of the Theory of Probability to Telephone Trunking Problems". *The Bell System Technical Journal*, 6:461–494, 1927.

[MP90] N.M. Mitrou and D.E. Pendarakis. "Cell-Level Statistical Multiplexing in ATM Networks: Analysis, Dimensioning and Call-Acceptance Control w.r.t. QOS Criteria". *7th International Teletraffic Congress*, Sept. 1990.

[MSST91] T. Murase, H. Suzuki, S. Sato, and T. Takeuchi. "A Call Admission Control Scheme for ATM Networks Using a Simple Quality Estimate".

IEEE Journal of Selected Areas in Communication, 9(9):1461–1470, Dec. 1991.

[NK92] R. Nagarajan and J. Kurose. "On Defining, Computing, and Guaranteeing Quality-of-Service in High-Speed Networks". *Proc. of IEEE INFOCOM '92*, 1992.

[NRSV91] I. Norros, J.W. Roberts, A. Simonian, and J.T. Virtamo. "The Superposition of Variable Bit Rate Sources in an ATM Multiplexer". *IEEE Journal of Selected Areas in Communication*, 9(3):378–387, Apr. 1991.

[OON88] H. Ohnishi, T. Okada, and K. Noguchi. "Flow Control Schemes and Delay/Loss Tradeoff in ATM Networks". *IEEE Journal of Selected Areas in Communication*, 6(9):1609–1616, Dec. 1988.

[OST88] S. Ohta, K-I. Sato, and I. Tokizawa. "A Dynamically Controllable ATM Transport Network Based on the Virtual Path Concept". *Proc. of IEEE GLOBECOMM '88*, pages 1272–1276, 1988.

[Par92] A.K. Parekh. *A Generalized Processor Sharing Approach to Flow Control in Integrated Services Networks*. PhD thesis, MIT, Lab. for Information and Decision Systems, Tech. Report LIDS-TR-2089 1992. Parts of this thesis were also published with R.G. Gallager in the *ACM/IEEE Transactions on Networking*, 1(3):344-357 and 2(2):137-150.

[PF94] V. Paxson and S. Floyd. "Wide-Area Traffic: The Failure of Poisson Modeling". *Proc. of ACM SIGCOMM '94*, pages 257–268, Aug, 1994. An extended version of this paper is available as **URL** ftp://ftp.ee.lbl.gov/papers/poisson.ps.Z.

[PG93] A.K. Parekh and R.G. Gallager. "A Generalized Processor Sharing Approach to Flow Control in Integrated Services Networks: The Single-Node Case". *ACM/IEEE Transactions on Networking*, 1(3):344–357, 1993.

[PKC94] K. Park, G. Kim, and M. Crovella. On the relationship between file sizes, transport protocols, and self-similar network traffic. Technical report, Boston University, Mar. 1994. **URL** http://cs-www.bu.edu/faculty/crovella/paper-archive/files-protocols.ps.

[RD91] G. Ramamurthy and R.S. Dighe. "Distributed Source Control: A Network Access Control for Integrated Broadband Packet Networks". *IEEE Journal of Selected Areas in Communication*, 9(7):990–1002, Sep. 1991.

[Rob93] J.W. Roberts. "Traffic Control in B-ISDN". *Computer Networks and ISDN Systems*, 25:1055–1064, 1993.

[RS90] C. Rasmussen and J. Sorensen. "A Simple Call Acceptance Procedure in an ATM Network". *Computer Networks and ISDN Systems*, 20:197–202, 1990.

[RSKJ91] C. Rasmussen, J. Sorensen, K.S. Kvols, and S.B. Jacobsen. "Source Independent Call Acceptance Procedures in ATM Networks". *IEEE Journal of Selected Areas in Communication*, 9(3):351–358, Apr. 1991.

[Sai92] H. Saito. "Call Admission Control in an ATM Network Using Upper Bound of Cell Loss Probability". *IEEE Transactions on Communications*, 40(9):1512–1521, Sept. 1992.

[SCZ93] S.J. Shenker, D.D. Clark, and L. Zhang. *A Scheduling Service Model and a Scheduling Architecture for an Integrated Services Packet Network.* **URL** ftp://parcftp.parc.xerox.com/pub/net-research/archfin.ps, 1993.

[She95] S. Shenker. "Fundamental Design Issues for the Future Internet". *IEEE Journal of Selected Areas in Communication*, 13(7):1176–1188, Sep. 1995.

[SRLL94] C. Shim, I. Ryoo, J. Lee, and S-B. Lee. "Modeling and Call Admission Control Algorithm of Variable Bit Rate Video in ATM Networks". *IEEE Journal of Selected Areas in Communication*, 12(2):332–344, Feb. 1994.

[SS91] H. Saito and K. Shiomoto. "Dynamic Call Admission Control in ATM Networks". *IEEE Journal of Selected Areas in Communication*, 9(7):982–989, Sept. 1991.

[VC94] M. Viswanath and P. Chou. An efficient algorithm for hierarchical compression of video. preprint, 1994.

[VPV88] W. Verbiest, L. Pinnoo, and B. Voeten. "The Impact of the ATM Concept on Video Coding". *IEEE Journal of Selected Areas in Communication*, 6(9):1623–1632, Dec. 1988.

[WC93] I. Wakeman and J. Crowcroft. "A Combined Admission and Congestion Control Scheme for Variable Bit Rate Video". *The Journal of Distributed Systems Engineering*, 1993.

[WCKG94] R. Warfield, S. Chan, A. Konheim, and A. Guillaume. "Real-Time Traffic Esitmation in ATM Networks". *International Teletraffic Congress*, June 1994.

[WKFR90] G. Woodruff, R. Kositpaiboon, G. Fitzpatrick, and P. Richards. "Control of ATM Statistical Multiplexing Performance". *Computer Networks and ISDN Systems*, 20:351–360, 1990.

[WKZL96] D.E. Wrege, E.W. Knightly, H. Zhang, and J. Liebeherr. "Deterministic Delay Bounds for VBR Video in Packet-Switching Networks: Fundamental Limits and Trade-Offs". *ACM/IEEE Transactions on Networking*, 4(3):352–363, Jun. 1996.

[Wro95] J. Wroclawski. *Specification of the Controlled-Load Network Element Service*. Internet-Draft, Nov. 1995.

[WTSW95] 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". *Proc. of ACM SIGCOMM '95*, pages 100–113, Aug. 1995.

[YH91] N. Yin and M.G. Hluchyj. "A Dynamic Rate Control Mechanism for Source Coded Traffic in a Fast Packet Network". *IEEE Journal of Selected Areas in Communication*, 9(7):1003–1012, Sept. 1991.

[Z^+93] L. Zhang et al. *Resource ReSerVation Protocol (RSVP)* Internet-Draft. **URL** ftp://ds.internic.net/internet-drafts/, Oct. 1993.

[ZF94] H. Zhang and D. Ferrari. "Improving Utilization for Deterministic Service in Multimedia Communication". *IEEE International Conference on Multimedia Computing and Systems*, 1994.

[ZK94] H. Zhang and E.W. Knightly. "Providing End-to-End Statistical Performance Guarantee with Bounding Interval Dependent Stochastic Models". *Proc. of ACM SIGMETRICS'94*, pages 211–220, May. 1994.