Appears in *Performance Evaluation and Application of ATM Networks*, Kouvatsos, D. editor, Kluwer, 2000

Chapter 3

ADAPTIVE STATISTICAL MULTIPLEXING FOR BROADBAND COMMUNICATION

Timothy X Brown

Dept. of Electrical and Computer Engineering

University of Colorado, Boulder, CO 80309-0530

timxb@colorado.edu

Abstract Statistical multiplexing requires a decision function to classify which source combinations can be multiplexed through a given packet network node while meeting quality of service guarantees. This chapter shows there are no practical fixed statistical multiplexing decision functions that carry reasonable loads and rarely violate quality of service requirements under all distributions of source combinations. It reviews adaptive alternatives and presents statistical-classification-based decision functions that show promise across many distributions including difficult-to-analyze ethernet data, distributions with cross-source correlations, and traffic with mis-specified parameters.

Keywords: Asynchronous Transfer Mode, Quality of Service, Admission Control, Statistical Multiplexing, Adaptive Methods.

1. INTRODUCTION

Modern broadband services transport diverse sources—constant bit rate voice, variable-rate video, and bursty computer data—using packet-based protocols such as the asynchronous transfer mode (ATM). In Figure 3.1, packets arrive at a node from different sources and are multiplexed on an output link. Since the many traffic sources are uncoordinated and communication bandwidth is finite, links congest and the link introduces losses and delays. With enough congestion, delays grow, queues overflow, and service degrades. Unlike *best-effort* services such as the internet, we consider the case where traffic sources are given *quality of service* (QoS) guarantees. To be specific, this work focuses on packet-level QoS such as on the maximum delay, delay variation, or loss rate, rather than call-level QoS such as call blocking rates.

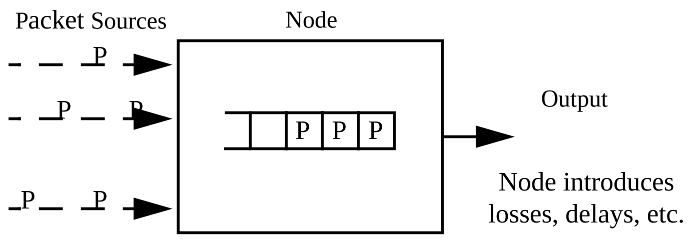


Figure 3.1 Network Node as a Black Box.

Providing packet-level QoS guarantees in broadband networks is a broad area of intense research (see [Gue99] and [Kni99] for an overview and extensive bibliography). While many aspects of QoS must be addressed, we would often like to answer a simple question: Under what conditions can a network meet a QoS guarantee. This chapter argues for new approaches to answering this question.

Standard multiplexing avoids congestion by rejecting a source combination if the total maximum source transmission rate would exceed the link bandwidth (so-called *peak-rate* multiplexing). This works well with constant bit rate sources. Variable rate and bursty sources generate packets at different rates over time. When many such sources are combined it is unlikely they all simultaneously communicate at their maximum rate. *Statistical multiplexing* exploits this fact to accept more sources and gain significantly higher utilizations. The key is an accurate *decision function* that classifies what combinations of sources can and can not be statistically multiplexed together on a given link while meeting QoS requirements. Successful statistical multiplexing is central to key tasks in broadband networks. For example, connection admission control avoids congestion by admitting new connections only when the new and existing connections will receive their requested QoS. A statistical multiplexing decision function could evaluate new connection requests for this purpose.

Statistical multiplexing can provide significant gains over peak rate allocation. If the utilization of a source type is low, for instance below 10%, then many such sources could be statistically multiplexed together providing up to 10 times greater network utilization. Deciding exactly how many such sources could be multiplexed together and still meet QoS requirements is part of the decision function design.

Two paths can be taken to developing the statistical multiplexing decision function as shown in Figure 3.2. The first path develops a model of the node function and traffic processes and then reduces this model to a decision function. This we denote the *fixed* method since the decision only applies to the modeled system. It is also fixed since the decision function is typically considered accurate for any combination of sources and therefore the same

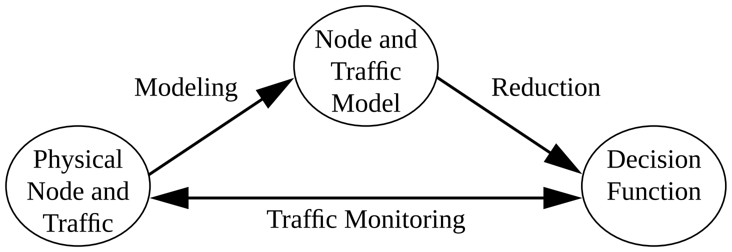


Figure 3.2 Two paths to developing statistical multiplexing decision functions.

without regard to the distribution of source combinations from the modeled traffic processes. The fixed method can fail if either the models are not accurate or if the model do not yield tractable decision functions and compromising simplifications are made.

The second path assumes little about the traffic or node. Many protocols provide monitoring data or network simulations can generate data with samples of traffic source combinations and the observed QoS. Using methods described in this chapter, such samples can be combined directly into a decision function that classifies what combinations do and do not meet QoS requirements without developing any explicit analytical node and traffic models. This we denote the *adaptive* method since the decision can be modified by the actual behavior of the node and traffic. The adaptive decision function is accurate only after observing the network performance and as a result may have an initial period of low accuracy. But, with enough observation, the adaptive method has the potential to learn an optimal decision function.

A simple example will make the distinction clear. Given Poisson arrivals into a finite FIFO buffer with exponential service time (i.e. an $M/M/1$ queue) and a QoS requirement on the maximum blocking probability, the fixed approach would derive the relationship between total load and blocking. A decision threshold would be derived and only loads up to the threshold would be accepted. The adaptive method simply would use examples of the observed loss rate at different loads to set the threshold. The adaptive method applies equally if the arrival process, service time distribution or queueing discipline changed, whereas the fixed approach would do well only on certain models and then only if the model was known.

No particular source model is assumed in Figure 3.1. The sources could be homogeneous or heterogeneous, independent or correlated. No particular node model is assumed in Figure 3.1 either. The queues could be simple FIFO, or implement a more complex scheme such as multiple priority queues or weighted fair queueing. The service rate could be constant or vary over time. Feedback mechanisms may be in place such as for ABR traffic. This chapter treats statistical multiplexing decision functions that apply quite gen-

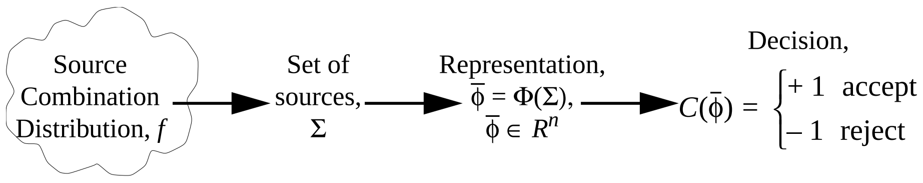

Source Combination Distribution, f
Set of sources, Σ
Representation, $\bar{\phi} = \Phi(\Sigma)$, $\bar{\phi} \in R^n$
Decision, $C(\bar{\phi}) = \begin{cases} +1 & \text{accept} \\ -1 & \text{reject} \end{cases}$

Figure 3.3 Relationship of formal elements to problem.

erally to a wide range of scenarios.

The body of this chapter is divided into four sections. Section 2 is a formal introduction to the statistical multiplexing decision function; the minimum necessary components; and metrics for evaluating the decision function effectiveness. Section 3 argues that any reasonable fixed controller either carries arbitrarily low loads relative to what is possible, is not robust to differing traffic structure, or does not treat artifacts of real networks such as inter-source correlations and misspecified parameters. Section 4 introduces adaptive statistical multiplexing and develops a theoretical foundation. Section 5 presents several experiments with adaptive multiplexing that show it has promise to be both robust and efficient across a variety of node types and traffic distributions including those with inter-source correlations and misspecified traffic parameters.

2. STATISTICAL MULTIPLEXING DECISION FUNCTIONS

The role of the decision function is to answer the question of whether the node can carry a set of sources and meet QoS guarantees. The set of sources need a description called a representation that can be used as inputs to the decision function. The performance of a decision function depends on the environment where it is applied. The environment is defined by the distribution of source combinations that will be seen by the controller. The elements of the statistical multiplexing decision function are shown in Figure 3.3. The rest of this section elaborates on these concepts.

2.1 DECISION MODEL

A source combination, Σ, consists of a number of sources. Such as three MPEG-2 video sources and five 10 BaseT ethernet links. The space for Σ depends on the application and will not be explicitly defined here. It is only necessary that a probability distribution, $f(\Sigma)$, can be defined from the space of possible distributions. Each source can generate packets according to its own traffic process. Since the number of sources is unbounded, the different traffic types vary greatly, and decisions must be made in a reasonable time; source combinations are described by an intermediate feature vector, $\bar{\phi} = \Phi(\Sigma) \in R^n$ for some fixed dimension n. The function, Φ, is the *representation*, with, for example, statistics of Σ such as the total load or the number of sources within different traffic classes.

We define QoS at two levels. At the source level, $Q(\Sigma) = (Q_1(\Sigma), \ldots, Q_l(\Sigma))$ is a vector of l QoS metrics for Σ; for example, the cell loss rate and mean delay for this combination are two possible metrics. With non-homogeneous traffic classes this could be a vector of QoS values for each traffic class. The vector $Q(\Sigma)$ does not describe a particular instance of the node carrying Σ. It is the long term expected QoS metrics for the source combination Σ. In connection access control this is known as conservative control. An aggressive type controller might momentarily allow combinations that violate QoS if averaged over time the system meets QoS. This depends on the dynamics of the problem which we will not consider here, but is considered elsewhere [Mit98, Ton99].

For a given distribution of source combinations, $f(\Sigma)$, and representation, Φ, each feature vector, $\bar{\phi}$, has an associated QoS vector, $q(\bar{\phi}) = (q_1(\bar{\phi}), \ldots, q_l(\bar{\phi}))$ that is the average[1] over the source combinations having feature $\bar{\phi}$, e.g.:

$$q_i(\bar{\phi}) \;=\; \text{average of QoS metric } i \text{ at } \bar{\phi} \;=\; \frac{\int_{\{\Sigma \mid \Phi(\Sigma) = \bar{\phi}\}} Q_i(\Sigma) f(\Sigma) d\Sigma}{\int_{\{\Sigma \mid \Phi(\Sigma) = \bar{\phi}\}} f(\Sigma) d\Sigma}. \quad (3.1)$$

We formulate the QoS requirements in terms of the QoS metric vector and a threshold vector $\tau = (\tau_1, \ldots, \tau_l)$:

$$q_i(\bar{\phi}) < \tau_i \text{ for all } i. \quad (3.2)$$

These notions of QoS are quite general. Most QoS requirements can be put in this form (e.g. if $x(\bar{\phi})$ is the expected delay, a requirement on delay between 1ms and 2ms can be represented by $q_i(\bar{\phi}) = |x(\bar{\phi}) - 1.5|$ and $\tau_i = 0.5$ or by $q_i(\bar{\phi}) = x(\bar{\phi})$, $q_{i+1}(\bar{\phi}) = -x(\bar{\phi})$, $\tau_i = 2$, and $\tau_{i+1} = -1$).

Having defined the representation and QoS, we turn to the decision function. The decision function can be treated as a classifier, $C(\bar{\phi}) \in \{+1, -1\}$, that classifies which $\bar{\phi}$ meet and don't meet QoS requirements. If $C(\bar{\phi}) = -1$ we say the classifier *rejects* the source combination, otherwise it *accepts* it. The optimal classifier accepts a source combination if and only if $q(\bar{\phi})$ meets the QoS requirements (3.2). Noting $q(\bar{\phi})$ is implicitly a function of the source distribution, $f(\Sigma)$, the optimal decision function is defined as:

1. Other criteria could be defined such as the infimum (i.e. worst case) QoS of all Σ that have $\bar{\phi}$ as a representation.

$$C_f(\bar{\phi}) = \begin{cases} +1 & \text{if } q(\bar{\phi}) \text{ meets all QoS requirements} \\ -1 & \text{o.w.} \end{cases}. \qquad (3.3)$$

To be clear, the optimal classifier depends on the space of sources, their distribution, the representation, the QoS metrics, and the QoS requirements. It also depends on the source traffic processes, the interaction of the traffic processes, and the functionality of the network node. The traffic processes and node functionality are fixed but not necessarily known. Their effect is captured by the QoS, $q(\bar{\phi})$.

2.2 PERFORMANCE METRICS

Ideally the decision function is defined by (3.3). How does a given classifier, C, compare with C_f? A classifier can misclassify by rejecting combinations that would have been accepted by the optimal classifier, or by accepting combinations that do not meet QoS. The first reduces the utilization of the network. The second increases the fraction of the source combinations accepted that violate QoS guarantees. We define two performance measures of a given classifier C to capture these notions. Each of these measures is defined in terms of a specific source distribution, or independent of the source distribution.

Let $U(\Sigma)$ be the utilization of the node output link with source combination Σ. The utilization can be defined quite generally as carried load, generated revenue, etc. The distribution dependent efficiency is

$$E_f(C) = \frac{\text{avg. utilization with } C}{\text{avg. utilization with } C_f} = \frac{\int_{\{\Sigma | C(\Phi(\Sigma)) = +1\}} U(\Sigma) f(\Sigma) d\Sigma}{\int_{\{\Sigma | C_f(\Phi(\Sigma)) = +1\}} U(\Sigma) f(\Sigma) d\Sigma}. \qquad (3.4)$$

$E_f(C) > 1$ is possible if the classifier accepts source combinations that do not meet QoS requirements. If numerator and denominator are both zero then by definition $E_f(C) = 1$. For a given distribution, a classifier can have a high efficiency if it rarely rejects source combinations that meet QoS or the utilization of rejected source combinations is low. The distribution free efficiency,

$$E(C) = \inf_f \{E_f(C)\}, \qquad (3.5)$$

is the worst case efficiency over all source distributions.

The fraction of correct accepts for a given distribution is

$$R_f(C) = \frac{\text{pr. } C \text{ correctly accepts}}{\text{probability } C \text{ accepts}} = \frac{\int_{\{\Sigma | C(\Phi(\Sigma)) = C_f(\Phi(\Sigma)) = +1\}} f(\Sigma) d\Sigma}{\int_{\{\Sigma | C(\Phi(\Sigma)) = +1\}} f(\Sigma) d\Sigma} \qquad (3.6)$$

One minus $R_f(C)$ is how often the classifier falsely accepts a source combination and violates QoS requirements. If numerator and denominator are both zero, then $R_f(C) = 1$. The robustness,

$$R(C) = \inf_f \{R_f(C)\}, \tag{3.7}$$

is the fraction of correct accept decisions in the worst case.

This section emphasizes statistical multiplexing decision functions classify a feature space which is defined via a representation function on the space of source combinations. The performance of this classification is defined by the types of errors relative to the source combination distribution. Since this distribution may not be known a priori, we have also considered the worst case performance over all distributions.

3. FIXED DECISION FUNCTIONS

There exist many proposed QoS decision functions either explicitly or implicitly in terms of admission control or equivalent bandwidth strategies. Typically these are based on assumed models of the traffic and node function. These we denote as fixed decision functions because they are designed to apply to any distribution of traffic from a given class of traffic models. Formally, we define a fixed decision function as a classification function that depends on the space of possible source combinations, the representation function, the source traffic processes, and node function; but is independent of the distribution of source combinations. This section focuses on the distribution of sources and source types and the representation function. For any decision function there exists source distributions for which the method is optimal in the sense of having maximal utilization relative to what is possible (efficiency) and correctly classifying the source combinations realized from this distribution (robustness). This section demonstrates that for any reasonable fixed decision function there exist source distributions for which the function either has zero efficiency or zero robustness. This argument is theoretical. We also show that in practice, fixed decision functions are fragile to realistic variations from typical assumed models.

3.1 THE REPRESENTATION

This section discusses the representation's role in the fixed classifier's efficiency and robustness. A representation, $\Phi(\Sigma)$, is *separable* if for all Σ_1 and Σ_2 where $\Phi(\Sigma_1) = \Phi(\Sigma_2)$: either both $Q(\Sigma_1)$ and $Q(\Sigma_2)$ meet or both do not meet QoS requirements. Appendix A shows if a representation is not separable, then there is always some distribution of source combinations that has either zero efficiency, or only accepts source combinations violating QoS requirements (zero robustness). Therefore, a good classifier over many source distributions requires a good representation.

One might ask if separable representations exist. At one extreme, if $\Phi(\Sigma_1) = \Phi(\Sigma_2)$ only if $\Sigma_1 = \Sigma_2$ then by definition Φ is separable. As an example that this is always possible, define every source by listing every packet's arrival time in order starting with the first packet. By mathematical artifices such as a diagonal counting of these arrival times, and interleaving of the decimal digits of these arrival times a single (albeit infinite precision) real number can uniquely represent any source combination.

At the other extreme $\Phi(\Sigma) = 1$ if Σ meets all its QoS requirements $\Phi(\Sigma) = 0$ otherwise is also separable: that is, the representation function is the decision function. In any real system the representation lies between these extremes and furthermore is fixed by existing protocol or hardware limitations. The next section will look at typical representations and show they are not separable. Further, several examples will make clear the resulting low efficiency or robustness is likely to be observed in practice.

3.2 CONVENTIONAL REPRESENTATIONS

The most robust decision function is based on simply the peak rates of the sources. Others such as the stationary (zero-buffer) approach in [Gue91] use both the peak and average rate. Although for CBR traffic they are optimal, the efficiency is zero or low with bursty video and data sources [DeP92]. For example, with the Ethernet data of Figure 3.7 the utilizations are much less than 1% for a 10Mbps link bandwidth. If the link bandwidth was any value less than 10Mbps all of these sources would be rejected under peak rate and efficiency would be zero. Thus, as is well known, peak rate allocation does not yield an efficient classifier. The stationary approach is asymptotically (in the number of sources) optimal. While asymptotically it is optimal, for small numbers of sources it is either equivalent to peak rate (for high utilization sources) with its low efficiency, or the approximation assumption on which it is based is violated and it accepts too many sources (for low utilization sources) resulting in low robustness.

The traffic descriptors specified by the ATM Forum include peak and average rate as well as a measure of burstiness. Unfortunately these equally represent a wide range of traffic types that have varying effect on network performance [Gal95]. Earlier work to include burstiness assumed traffic from the ON/OFF model of Figure 3.4 assuming exponential holding times (so the model is Markovian) [Gue91][Elw93][Cho94].

It has been well documented that the ON/OFF model with exponential

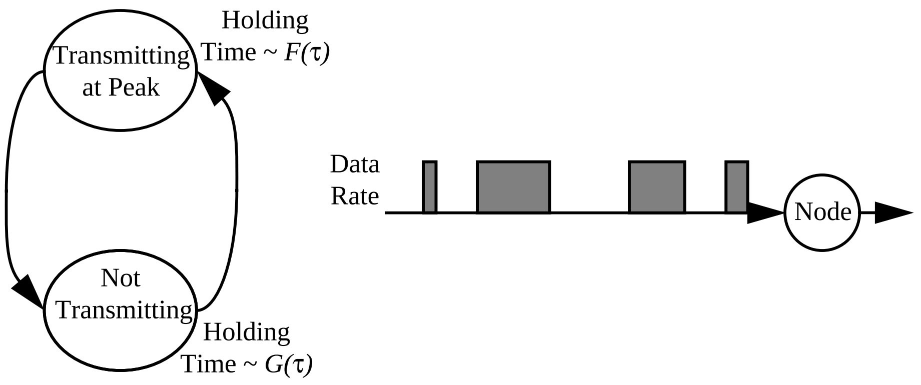
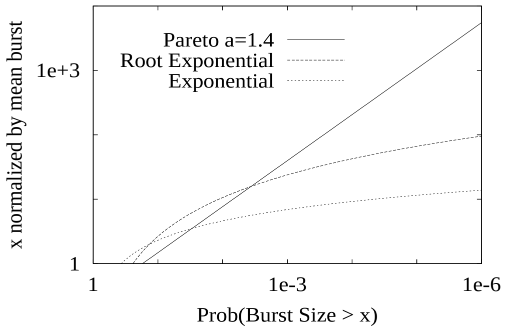


Figure 3.4 Two-State Source Traffic Model

holding times does not reflect the fundamental characteristics of traffic. Analysis of ethernet traffic [Lel93][Nor94] and variable rate video [Gar94] indicate such traffic types are decidedly not simple Poisson processes. Nor are they Poisson burst processes with geometrically distributed burst size. Typically the ON or OFF holding times are characterized by heavy tails with respect to the exponential distribution, i.e., outlier events occur with greater frequency than predicted by the exponential. TCP/IP traffic has also been analyzed and although the session arrivals (telnet, ftp, etc.) are well modeled as a Poisson Process, the traffic within the sessions also exhibits heavy tailed properties [Pax94]. The ethernet data, for instance, has been shown to have interarrival times with finite means and infinite higher moments. Even bounded packet sizes lead to heavy tailed distribution on the number of arrivals in a given period [Kri95]. A heavy tailed interarrival time implies a few very long periods offset by many short periods. So, even though the individual packets are bounded they tend to come in “trains” of packets one after the other followed by long “inter-train” periods.

The model of Figure 3.4 with exponential (or its discrete equivalent, geometric) holding times is completely represented by three components; the ON rate, the mean ON time, and the mean OFF time. Models relying on this representation fail since it is easy to construct distributions that have the same representation, but yet fail to meet the QoS requirements. For instance, as shown in Figure 3.5, an ON/OFF source with exponential holding times will rarely have a burst greater than 13 times the mean bursts (i.e. with probability ~1 in a million), but using Pareto holding times with parameters determined in [Wil95] to match ethernet data, bursts 100’s of times longer than the mean occur often. More standard distributions such as the root exponential[2] also have much longer bursts in the tails.

The effect on traffic can be seen in Table 3.1 which uses the Markov model based technique in [Gue91] to calculate the highest load of four



Distribution	P{period length > *x*}	Variance
Exponential	$e^{-x/m}$	m^2
Root Exponential	$e^{-\sqrt{(2x)/m}}$	$5m^2$
Pareto	$\left(\frac{x\alpha}{m(\alpha-1)}\right)^{-\alpha}, x > \frac{m(\alpha-1)}{\alpha}$	∞ $(\alpha < 2)$

Figure 3.5 Comparing exponential, root exponential, and Pareto holding times (average burst size is *m*).

sources that can be carried on a given link.[3] Using this load in the model in Figure 3.4 and the node model in Appendix B, $2\text{x}10^{10}$ packet time periods are simulated with different holding time distributions. The statistical multiplexing gain (carried load over the greatest load carried with peak rate) and the net loss rate are shown. Since the ON and OFF periods have the same mean period, the utilization is 50% and the greatest multiplexing gain is 2. With short bursts compared to the buffer size, a large multiplexing gain (out of a possible gain of 2) is possible with the geometric source. With long bursts, only 3% more traffic than allowed by peak rate is accepted. Even with this small deviation

2. The root exponential is just a special case of the Weibull distribution with Prob{period length > *x*} = $\exp\left[-\left(\frac{(\Gamma(1+b))}{m}\right)^{1/b}\right]$ and variance $m^2\left(\frac{\Gamma(1+2b)}{\Gamma^2(1+b)} - 1\right)$ for parameter $b > 0$. Figure 3.5 would plot $\frac{-(\log(z))^b}{\Gamma(1+b)}$ vs. *z*. The exponential and root exponential are $b = 1$ and $b = 2$. By choosing large enough *b* the variance and tail can be made arbitrarily large although unlike the Pareto, the tail plot of Figure 3.5 would always be sub-linear.

3. This experiment could be repeated with many more than four sources. Four were used for simplicity.

from peak rate allocation, the Pareto distribution packet loss rate is still many orders of magnitude higher than the 10^{-6} target loss rate.

Section 3.1 argues a better representation is needed to perform better. For instance, the Hurst parameter is one candidate that would differentiate traffic models with infinite holding time variance (e.g. the Pareto) from finite variance distributions [Err96]. As seen in Table 3.1, the geometric and root exponential (both producing Hurst parameter 0.5) have dramatically different loss rates. Another approach taken in [Hey96] attempts to characterize video traffic via a general Weibull distribution and concludes a single model based on a few physically meaningful parameters that applies to all video sequences does not seem possible.

Simulation studies in [Kni99] on a range of fixed decision techniques concludes that simple representations will yield low utilizations for bursty traffic flows, while more detailed representations put undo burden on network clients and policing.

It should be clear from these results that given any simple traffic statistics, a wide range of traffic streams can be generated having these statistics. Conversely and more significantly, given a wide range of realistic traffic it is unlikely a representation consisting of a single set of simple statistics will be separable. Therefore, low efficiency or robustness can be expected when fixed decision functions based on such representations are used either across many traffic streams or for extended periods when new usage and new applications can alter the fundamental structure and distribution of traffic.

Table 3.1: $2\text{x}10^{10}$ time slot simulation using Markov-based equivalent bandwidth allocation

Mean Period On/ Off	Multiplexing Gain	Source Distribution	Loss Rate (10^{-6} target)
100/100	1.75	Geometric	$5\text{x}10^{-8}$
		Root Exponential	$4\text{x}10^{-3}$
		Pareto ($\alpha = 1.4$)	$2\text{x}10^{-2}$
10000/10000	1.03	Geometric	$4\text{x}10^{-9}$
		Root Exponential	$3\text{x}10^{-5}$
		Pareto ($\alpha = 1.4$)	$2\text{x}10^{-5}$

3.3 DIFFICULTIES IN REAL NETWORKS

None of the statistical multiplexing methods known to this author specifically addresses the problem of correlated sources. Instead all sources are assumed independent. Two high-rate sources could be synchronized identical outputs violating this assumption as in a three-way video conference where the traffic from one participant to the other two may be sent as two identical streams that have partially overlapping paths. As a trivial extreme, even peak rate allocation can produce losses if the buffer size is less than the number of sources and the sources generate their packets simultaneously (despite the "Asynchronous" in ATM, sources can potentially be highly correlated).

Most methods assume the traffic descriptors are accurate. Due to traffic shaping by the network, bursts from independent sources can become coupled [Lau93], and even CBR sources may enter a node in bursts [Lee96]. For sources with long-range dependencies, such as ethernet traffic, the measured average traffic rate varies widely from one averaging period to the next on averaging periods ranging up to 100's of seconds [Lel93]. This indicates accurate traffic measurements are not possible. Similarly, many sources may not have any traditional measure of how to describe the traffic other than crude limits and broad classes despite having useful information e.g. "residential world-wide-web surfer from 28.8 baud modem." Within the framework of Figure 3.3 these practical difficulties are reduced in (3.1) to asking whether the average over all source types with representation $\overline{\phi}$ will meet QoS.

These results suggest the solution to good statistical multiplexing decision functions is not strictly better modeling or better representations, but rather a method that is optimal for the given representation and robust to deviations from the model on which the representation is based.

4. ADAPTIVE METHODS

Adaptive schemes allow the decision function to depend on the results of carried traffic performance. They have the potential to make decisions that vary according to the source distribution, *f*. For example, if only one source combination is possible (as in the proof in Appendix A), a reasonable adaptive method would learn whether the combination should be rejected or accepted. This section describes the steps and elements to this adaptation. It is derived from the formalism of statistical function approximation [Dud73][Bis95] rather than adaptive control.

4.1 OVERVIEW AND EVALUATION CRITERIA

The adaptive method collects a performance data set, $X = \{(\overline{\phi}_i, \overline{\mu}_i)\}$, where $\overline{\phi}_i$

is the feature vector, $\bar{\mu}_i$ is a real vector of monitoring information, and $1 \le i \le |X|$ ($|X|$ is the number of elements in X). A sample in this data set represents the output of monitoring hardware with the measured performance, $\bar{\mu}_i$, from a source combination with feature vector, $\bar{\phi}_i$. As before, the source combinations are distributed according to f. The monitoring information, $\bar{\mu}_i$, might be as simple as 1 or –1 depending on whether or not the source combination met its QoS. Or, it might be more detailed, like the number of packets sent and the number of packets lost, delay statistics, etc. The classification function derived from the data set X is denoted as $C_X(\bar{\phi})$.

We specify two criteria for $C_X(\bar{\phi})$. The first, *consistency*, is an asymptotic property:

$$\lim_{|X| \to \infty} \text{Prob}\{C_X(\bar{\phi}) \neq C_f(\bar{\phi})\} = 0. \tag{3.8}$$

The probability is over source combinations chosen from f. This says that as we collect more data the probability of decision error goes to zero. The second is a finite sample property that requires the fraction of correct accepts to be greater than a confidence level Λ:

$$R_f(C_X) > \Lambda \tag{3.9}$$

An easy way to satisfy (3.9) is to simply not accept any source combination so by definition $R_f = 1$. When X has few samples, this may be a viable strategy. The requirement in (3.8) ensures that with more data, the classifier converges in probability to the true classifier and $R_f(C_X) = E_f(C_X) = 1$. Since this applies for any f, we conclude a classifier satisfying (3.8) will yield $R(C_X) = E(C_X) = 1$ as more data is collected. With limited data, (3.9) bounds QoS violations to probability less than $1 - \Lambda$.

Is any adaptive decision function consistent? Can any adaptive decision function give confident estimates with finite samples? In general, the answer to both questions is yes. Appendix C discusses these questions further.

4.2 PRIOR WORK

A variety of researchers have examined the adaptive approach. The methods in [Che92], [Nev93], and [Nor93] choose a particular traffic model and then refine parameters based on the controller’s performance. These are based on Markov source models and thus suffer the same deficiencies as noted in the previous section when traffic is non-Markovian. The method in [Jam92], while adaptive, controls only for delay. As noted in [Lev97], delay is a much more stable parameter than packet loss rate. Also, as is the method in [Kaw95], it is based on short term traffic measurements which can be misleading for data with long-range dependencies [Pax94]. The methods in [Hir90],[Tra92], and [Est94] are closest to the method presented here. They do not assume a particular traffic model in developing the decision function and can choose long time-scales to adapt over (e.g. days or weeks). While promising, they are applied to very simple models (e.g. combinations of different numbers of only one or two source types). It is not clear how the methods would scale to many heterogeneous copies of source types. Further, as will be elaborated shortly, these approaches have a distinct bias that under real-world conditions leads to accepting source combinations that miss QoS targets by orders of magnitude. Incorporating preprocessing methods to eliminate this bias is critical and two methods from prior work by the author will be described. Unlike this previous work, the methods are applied to a range of source models, difficult-to-model ethernet traffic, sources with intra-source correlations, and sources with misspecified parameters.

4.3 STATISTICAL CLASSIFICATION

The adaptive methods in this chapter generate a decision function using statistical classification methods. The reader is directed to any of a number of books in the statistical classification area often under the label of “pattern recognition” or “neural networks”. Two good examples are [Dud73] and [Bis95]. Using statistical classification, the decision function is derived from examples of previously carried sources and their received QoS. A statistical classifier is given a *training set*, $Y = \{(\bar{\phi}_i, d_i, w_i)\}$, consisting of feature vectors, $\bar{\phi}$, with corresponding desired output classification, $d_i \in \{+1, -1\}$ and positive real-valued sample weight, w_i. For many applications all samples are weighted equally and the weight is disregarded. A classification function, $C(\bar{\phi};\bar{\nu})$, parameterized by a real-valued vector $\bar{\nu}$, divides the feature space into positive and negative regions separated by a *decision boundary* and can be used to classify future feature vectors. Based on the training set, a classifier (i.e., $\bar{\nu}$) is selected that minimizes some criteria (so-called *training*). We describe in turn the data collection, decision function model, and objective criteria.

4.4 GENERAL SCENARIO AND COLLECTING DATA

This chapter focuses on the scenario in Figure 3.1 where a node is multiplexing multiple sources. The goal is to collect a data set about the node in the form $X = \{(\bar{\phi}_i, \bar{\mu}_i)\}$ consisting of monitoring information, $\bar{\mu}_i$, for source combinations represented by $\bar{\phi}_i$. The node architecture, feature vector representation, and source distributions are assumed fixed but not necessarily known. In a running network sources dynamically arrive/depart and at transitions network monitors record information about the traffic carried and QoS since the last transition. Alternatively, off-line network simulations of different traffic combinations could be used. These are not equivalent since, off-line, any combination can be simulated without regard to the QoS given, while in an operating network customers care about received QoS. Also in an on-line admission control scenario, the observed distribution of source combinations is a function of what connection requests are accepted or rejected. This issue is discussed in [Hir95] and work in [Bro99a] shows the interaction is stable and does not fundamentally change the problem. All of the results presented in this chapter are based on simulated source combinations.

For simplicity the rest of this chapter focuses on a single QoS criterion. Since, as noted earlier, packet loss is the most difficult parameter to control, we focus on a system where the QoS criterion is in terms of a maximum packet loss rate, $p*$. It should be clear the general technique applies to any QoS metric. With these disclaimers we assume the samples contain the number of packet arrivals, T, the number of lost packets, s, and the feature vector, $\bar{\phi}$. The underlying QoS, $q(\bar{\phi})$, is the average loss rate of source combinations with representation $\bar{\phi}$. The data given is $X = \{(\bar{\phi}_i, s_i, T_i)\}$ and the training set is $Y = \{(\bar{\phi}_i, d_i, w_i)\}$ where $|X| = |Y|$. How is Y computed from X, what is the form of the objective function that will be minimized, and what decision function model will be used? We look at the latter question first.

4.5 DECISION FUNCTION MODEL

Given a training set, Y, the decision function, $C(\bar{\phi};\bar{\nu})$, can use many models [Bis95]. The basic consideration is the so-called bias-variance trade off. To illustrate this trade-off we consider two extremes. At one extreme the decision function is unconstrained; for instance, $C(\bar{\phi};\bar{\nu})$ is an arbitrarily large look-up table. This table stores a value for every unique $\bar{\phi}$ in Y and $C(\bar{\phi};\bar{\nu})$ is simply the value stored at $\bar{\phi}$. While this can always produce an unbiased estimate for each $\bar{\phi}$ in Y, it is not defined for other $\bar{\phi}$ and the output will have a high variability from one Y to the next. At the other extreme the decision function is highly constrained; for instance, $C(\bar{\phi};\bar{\nu})$ is a constant. If the output is always accept or always reject then this is a sufficient model. In more interesting cases, the output will not be a constant. On the other hand the output will vary little from data set to data set and have low variance.

Since the decision errors can be decomposed into bias and variance components selecting a decision function is a trade-off between models that can capture the correct decision function, and models with few parameters that can be trained quickly with small data sets. We highlight this issue to show that selecting a model requires some care and including prior knowledge about the type of decision function we expect is more likely to produce simple efficient and robust decision functions.

In order to focus on the central issues of this chapter, we use a simple model; the linear discriminant:

$$C(\bar{\phi};\bar{\nu}) = \operatorname{sign}\left(\sum_{i=1}^{n} \nu_i \phi_i + \nu_0\right). \tag{3.10}$$

The parameters, $\bar{\nu}$, are determined by minimizing an objective (next section) with respect to the weights. For the experiments in this chapter the features are loads. The optimal classifier is monotonic in that if $\bar{\phi}$ is rejected, then any feature with greater load is rejected. The linear decision function also has this property. If there is only one feature in the feature vector, then the linear decision function reduces to a threshold on the feature; which is optimal if the feature is a load. By Appendix C, the linear discriminant can form a consistent estimator in this case with the correct objective function. Therefore, although (3.10) is a simple model, it is sufficient for the experiments in this chapter.

4.6 OBJECTIVE FUNCTION

Given a training set, $Y = \{(\bar{\phi}_i, d_i, w_i)\}$, and a classifier, $C(\bar{\phi};\bar{\nu})$, a set of parameters is chosen that minimizes an objective function. This chapter minimizes a weighted sum squared error. More general criteria also work well [Bro99b]:

$$E(\bar{\nu}) = \sum_i [w_i (C(\bar{\phi}_i;\bar{\nu}) - d_i)^2]. \tag{3.11}$$

An unconstrained classifier, will set $C(\bar{\phi};\bar{\nu}) = d_i$ if all the $\bar{\phi}_i$ are different. With multiple samples at the same $\bar{\phi}$, the error in (3.11) is minimized when

$$C(\bar{\phi};\bar{\nu}) = \operatorname{sign}\left(\sum_{\{i \mid \bar{\phi}_i = \bar{\phi}\}} w_i d_i\right). \tag{3.12}$$

If the classifier is more constrained (e.g. a low dimension linear classifier) or no data is precisely at $\bar{\phi}$, $C(\bar{\phi};\bar{\nu})$ will be the weighted average of the d_i in the neighborhood of $\bar{\phi}$, where the neighborhood is, in general, an unspecified function of the classifier. A more direct form of averaging would be to choose a specific neighborhood around $\bar{\phi}$ and average over samples in this neighborhood. This suffers from having to store all the samples in the decision mechanism, and incurs a significant computational burden to find the samples in the neighborhood. More significant is how to decide the size of the neighborhood. If it is fixed, in sparse regions no samples may be in the neighborhood. In dense regions near decision boundaries, it may average over too wide a range for accurate estimates. Dynamically setting the neighborhood so that it always contains the *k* nearest neighbors solves this problem, but does not account for the size of the samples. We will return to this in Section 4.6.2.

4.6.1 The Small Sample Problem

If sample sizes are large ($Tp^* >> 1$), then $d_i = \text{sign}(p^* - s_i/T_i)$ accurately estimates $\text{sign}(p^* - q(\bar{\phi}_i))$ and the problem reduces to fitting a function to $Y = \{(\bar{\phi}_i, d_i)\}$ using standard statistical classification techniques. For example, this approach has been used in [Hir90][Tra92][Est94] and is denoted the *normal* method in Table 3.2. When sample sizes are small, ($Tp^* << 1$), the number of losses, s, will be zero with high probability while even one loss yields a sample estimate, $s/T >> p^*$, so that individual samples are poor estimates of the underlying rate. As will be shown, the above procedure when applied to small samples can accept a $\bar{\phi}$ despite $q(\bar{\phi})$ being orders of magnitude larger than $p*$.

An alternative approach in [Hir95] attempts to estimate $q(\bar{\phi})$ directly using s_i/T_i as the measured loss rate at $\bar{\phi}_i$, and then using regression techniques to make an estimate, $\hat{q}(\bar{\phi})$, and defining $C_X(\bar{\phi}) = \text{sign}(p^* - \hat{q}(\bar{\phi}))$. The probabilities can vary over orders of magnitude making accurate estimates difficult. Estimating the less variable $\log(q(\bar{\phi}))$ is inconsistent for small samples where most of the samples have no losses and $s = 0$, and the logarithm must be artificially defined. Preliminary work in [Ton98] indicates a proper modeling of the regression problem may lead to satisfactory results that are unbiased and are insensitive to intra-sample correlations (c.f. Section 4.6.3).

One obvious solution is to have large samples. In communication networks, such as packet data networks, sample sizes are limited by three effects. First, desired loss rates are often small; typically in the range 10^{-6}–10^{-12}. This implies large samples must be at least 10^7–10^{13} observed packets. For 10^{13}, even a Gbps packet network with short packets requires samples lasting several hours. At typical rates, samples of size 10^7 require samples lasting minutes. Second, in dynamic data networks, while individual connections may last for significant periods, the aggregate flow of connect and disconnect requests prevents traffic combination for lasting the requisite period. Third, in any queueing system, even with uncorrelated arrival traffic, the buffering introduces memory in the system. A typical sample with losses may contain 100 losses,. But, a loss trace would show the losses occurred in a single short overload event. Thus, the number of independent trials can be several orders smaller than the raw sample size indicating the loads must be stable for hours, days, or even years to get samples that lead to unbiased classification.

4.6.2 Consistent and Confident Training Sets

We present without proof two preprocessing methods derived and analyzed in

[Bro95, Bro99b]. The first chooses an appropriate d and w so (3.12) corresponds to a consistent maximum likelihood solution. This is the *weighting* method shown in Table 3.2.

The second preprocessing method assigns uniform weighting, but classifies $d_i = 1$ only if a certain confidence level, Λ, is met that the sample represents a combination where $q(\bar{\phi}) < p*$. Such a confidence was derived in [Bro99b]:

$$\mathrm{Prob}\{q(\bar{\phi}) < p^* | s, T\} = 1 - B(s, T, p^*) \tag{3.13}$$

where

$$B(s, T, p) = \sum_{k=0}^{k \le s} \binom{T}{k} p^k (1-p)^{T-k}. \tag{3.14}$$

For small T (e.g. $Tp* < 1$ and $\Lambda > 1 - 1/e$), even if $s = 0$ (no losses), this level is not met. But, a neighborhood of samples with similar load combinations may all have no losses indicating this sample can be classified as having $q(\bar{\phi}) < p*$. Choosing a neighborhood requires a metric, m, between feature vectors, $\bar{\phi}$. In this chapter we simply use Euclidean distance. Using the above and solving for T when $s = 0$, the smallest meaningful neighborhood size is the smallest k such that the aggregate sample is greater than a critical size,

$$T^* = \ln(1 - \Lambda)/\ln(1 - p*). \tag{3.15}$$

From (3.13), this guarantees that if no packets in the aggregate sample are lost we can classify it as having $p(\bar{\phi}) < p*$ within our confidence level. For larger samples, or where samples are more plentiful and k can afford to be large, (3.13) can be used directly. Table 3.3 summarizes this *aggregate* method.

4.6.3 Generating Samples of Independent Bernoulli Trials

The above preprocessing methods assume the training samples consist of independent samples of Bernoulli trials. Because of memory introduced by the buffer and possible correlations in the arrivals, this is decidedly not true. The methods can still be applied, if samples can be subsampled at every Ith trial where I is large enough so the samples are pseudo-independent, i.e. the dependency is not significant for our application. As indicated in [Gro96],

Table 3.2: Summary of Methods.

Method	d_i	w_i
Normal	$\mathrm{sign}(p^* T_i - s_i)$	1
Weighting	$\mathrm{sign}(p^* T_i - s_i)$	$p^* T_i - s_i$
Aggregate	Table 3.3	1

buffered systems have a finite time horizon beyond which the impact on loss of correlations in the arrival process become nil even for sources with long-range dependencies. We therefore can expect to find a suitable *I* for most traffic types. An explicit bound on this time horizon appears in [Gro96], which would be a suitable *I*. This bound has not yet been tried for this chapter. It is expected the bound will be larger than necessary for our purposes. Further, it depends on knowing explicit characteristics of the node and source arrival process that may not be known to the statistical multiplexer.

A simple graphical method for determining *I* is given in [Bro99b]. The weighting and desired output:

$$\begin{aligned} w_i &= 1 - B(T_i p^*, T_i, p^*) \text{ if } d_i = 1;\ w_i = B(T_i p^*, T_i, p^*) \text{ if } d_i = -1 \\ d_i &= \operatorname{sign}(T_i p^* - s_i) \end{aligned} \quad .(3.16)$$

has the property that with uncorrelated trials it produces a consistent estimator. Alternatively, if *T* overstates the true sample size by a large factor, $w_i = 0.5$ and (3.16) is the same as the normal scheme. This is the case with correlated samples. The sample size, *T*, overstates the number of independent trials. As will be shown, this implies the decision boundary is biased to orders of magnitude above the true boundary. As the subsample factor is increased, the subsample size becomes smaller, the trials become increasingly independent, the weighting becomes more appropriate, and the decision boundary moves closer to the true decision boundary. At some point, the samples are sufficiently independent so that sparser subsampling does not change the decision boundary. By plotting the decision boundary of the classifier as a function of *I*, the point where the boundary is independent of the subsample factor indicates a suitable choice for *I*.

If the subsample factor is known, the packets can be subsampled explicitly as the data is collected. As will be seen, the subsample factors are large, implying an easy-to-implement sparse monitoring is possible in an on-line system. If the raw $(\bar{\phi}, s, T)$ samples are given, then as a worst case they are

Table 3.3: Aggregate Classification Algorithm

1. Given sample (ϕ_i, s_i, T_i) from training set $\{(\phi_i, s_i, T_i)\}$, choose confidence level Λ and metric *m* on the feature space.
2. Calculate T^* from (3.15).
3. Find nearest neighbor sequence $n_0, n_1, \ldots$ where $n_0 = i$ and $m(\bar{\phi}_{n_j}, \bar{\phi}_i) \le m(\bar{\phi}_{n_{j+1}}, \bar{\phi}_i)$ for $j \ge 0$.
4. Choose smallest *k* s.t. $T' = \sum_{j=0}^{k} T_{n_j} \ge T^*$. Let $s' = \sum_{j=0}^{k} s_{n_j}$.
5. $d_i = \operatorname{sign}(1 - \Lambda - B(s', T', p^*))$.

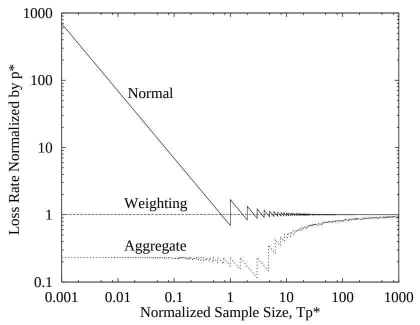


Figure 3.6 Expected Decision Normalized by $p*$. The nominal boundary is $p/p* = 1$. The aggregate method uses $\Lambda = 0.95$.

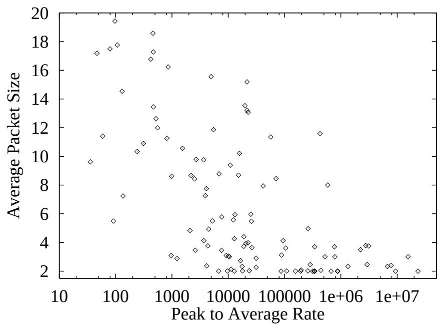


Figure 3.7 Average Packet Size (in 48-byte units) vs. Peak to Average Rate Ratio for Ethernet Data. The peak rate is 10Mbps for all connections.

subsampled by dividing s and T by I. The results are rounded up with probability proportional to the remainder.

In this way, despite the correlations in the data, we can produce independent trials for the statistical classification methods. Looking at Table 3.2, subsampling only scales the weighting by a factor of $1/I$ for all samples. Since this has no essential effect on the minimization of (3.11), the weighting method is independent of these correlations and thus does not need to estimate I.

4.7 SUMMARY AND PRIOR RESULTS

This section has presented a method for using samples of the QoS for different source combinations to be combined into a consistent decision function. The procedure consists of collecting traffic data at different combinations of traffic loads that do and do not meet QoS. These are then subsampled with a factor I determined as in Section 4.6.3. Then one of the methods for computing a training set, summarized in Table 3.2, are applied to the data. This training set is then used in any statistical classification scheme. Analysis in [Bro99b] derives the expected bias (shown in Figure 3.6) of the methods when used with an ideal classifier. The normal method can be arbitrarily biased, the weighting method is unbiased, and the aggregate method chooses a conservative boundary. Simulation experiments in [Bro99b] with a well characterized M/M/1 queueing system to determine acceptable loads showed the weighting method was able to produce unbiased threshold estimates over a range of values; and the aggregate method produced conservative estimates that were always below the desired threshold, although in terms of traffic load were only 5% smaller. Even in this simple system, where the input traffic is uncorrelated (but the losses become correlated due the memory in the queue), the subsample factor was 12, meaning good results required more than 90% of the data be thrown out.

5. EXPERIMENTS

A range of experiments are performed using the node model of Appendix B under different source models. The experiments are not necessarily designed to be realistic, but rather to demonstrate the method under a variety of conditions. Each experiment, used the three methods of Table 3.2 for creating a training set, *Y*, from the monitoring data, *X*, based on a QoS requirement of maximum packet lost rate, $p* = 10^{-6}$, confidence of $\Lambda = 95\%$ for the aggregate method, and the decision function, $C(\overline{\phi};\overline{\nu})$, is the linear discriminant decision function (3.10). A simple representation, total load, is used in each case.

5.1 SIMULATED SOURCES

In this section, the arrival process consisted of 4 identical ON/OFF sources from the model in Figure 3.4 similar to the experiments in Table 3.1 with equal, short mean ON/OFF periods of size 100 time slots. The training set for a given holding time distribution consisted of at least 10,000 simulations at randomly chosen loads for 10^7 timeslots. The load per source are uniformly distributed between 0.25 (accepted by peak rate) and 0.5 (a net load of 1, i.e. 4 sources times 50% duty cycle times a load of 0.5). The representation, ϕ, is simply the total load.

To create pseudo-independent trials necessary for the aggregate methods, we subsampled every *I*th packet. Using the graphical method of Section 4.6.3, the resulting *I* are shown in column 4 of Table 3.4. The median subsample factor is ~200. The sample sizes ranged up to 10^7 plackets, But, after subsampling by a factor of 200, even for the largest samples, $p*T < 0.05 << 1$.

These results for the geometric and Pareto holding time distributions appear in Table 3.4. A third distribution, labeled correlated, introduces a small cross-source correlation to the geometric distribution where in 10% of the samples, 2 of the 4 sources are 100% correlated. To test the results we use the threshold found by the aggregate method and simulate against the other distri-

Table 3.4: Experiments with Simulated Sources.

Mean Period On/Off	Distribution	Training Set Size	Subsample Factor	Threshold Found		
				Normal	Weighting	Aggregate
100/100	Geometric	15112	60	0.909	0.895	0.884
	Pareto ($\alpha = 1.4$)	10492	520	0.659	0.573	0.567
	Correlated	16791	180	0.907	0.874	0.855

butions for 10^{10} time slots as shown in Table 3.5. When the threshold is tested against the same distribution as was trained, the losses are less than the target $p* = 10^{-6}$. Comparing the thresholds, the geometric-based threshold carries 50% greater loads than with the Pareto-based threshold. Conversely, if the geometric-based threshold is given Pareto traffic, then the QoS targets are missed by more than 4 orders of magnitude. The Correlated and geometric thresholds differ by only a few percent. Yet, the resulting loss rates vary by orders of magnitude. Conversely, since the loss rates are all within an order of magnitude of the target value when the same distribution is used for testing and training, these thresholds are within a few percent of the optimal. Thus given different distributions, the adaptive method can find a nearly optimal decision function for each. This is evidence the adaptive method gains significant efficiency and robustness.

5.2 ETHERNET DATA

We now turn from simulated traffic data to ethernet traffic traces. An earlier version of this section appeared in [Bro97]. The ethernet data is described in [Lel93] as the August 89 busy hour containing traffic ranging from busy file-servers/routers to users with just a handful of packets. The detailed data set records every packet's arrival time (to the nearest 100μsec), size, plus source and destination tags. From this, 108 "data traffic" sources are generated, one for each computer that generated traffic on the ethernet link. With ATM in mind[4], each ethernet packet (which ranged from 64 to 1518 bytes) is split into 2 to 32 48-byte packets (partial packets were padded to 48 bytes). Depending on the experiment, a link data rate was fixed at 10, 17.5, 30, or 100Mbps. Dividing this by 48x8 = 384 produced a link packet rate. Each ethernet packet arrival time is mapped into a particular time slot in the node model. Since the

Table 3.5: Aggregate Thresholds in Table 3.4 Tested for 10^{10} Timeslots.

Train Distribution	Adaptive On Rate[a]	Multiplexing Gain	Test Distribution[b]:		
			Geometric	Correlated	Pareto
Geometric	0.884	1.77	$3x10^{-7}$	$2x10^{-6}$	$3x10^{-2}$
Correlated	0.855	1.71	0	$1x10^{-7}$	$1x10^{-2}$
Pareto ($\alpha = 1.4$)	0.567	1.13	0	0	$1x10^{-6}$

a. Target Loss Rate = 10^{-6}
b. From 10 Billion Packet Timeslot Simulation

4. Assuming, for simplicity, the cell payload is 48 bytes.

sources came from the same 10Mbps Ethernet link and only packets without collisions were recorded in the data trace, the traces contain at most one packet being sent at any given time. Decorrelating the sources via random starting offsets, produces independent data sources with the potential for multiple packet arrivals and overloads. Multiple copies at different offsets produces sufficient loads even for bandwidths greater than 10Mbps.

The peak data rate with the ethernet data is fixed, while the load (the average rate over the one hour trace normalized by the 10Mbps peak rate) ranges over five orders of magnitude (see Figure 3.7). Also troubling, analysis of this data [Lel93] shows the aggregate traffic exhibits chaotic self-similar properties and suggests that it may be due to the sources' packet inter-arrival time distribution following an extremely heavy tailed distribution with finite mean and infinite higher order moments such as the Pareto distribution in Figure 3.5.

We divided the data into two roughly similar groups of 54 sources each; one for training and one for testing. To create sample combinations we assign a distribution over the different training sources, choose a source combination from this distribution, and choose a random, uniform (over the period of the trace) starting time for each source. Simulations that reach the end of a trace wrap around to the beginning of the trace. The sources are described by a single feature corresponding to the average load of the source over the one hour data trace. A source combination, Σ, is described by a single variable, ϕ, the sum of the average loads. The source distribution, $f(\Sigma)$, was a uniformly chosen 0–M copies of each of the 54 training samples. M was dynamically chosen for each experiment so the link would be sufficiently loaded to cause losses. Each sample combination was processed for $3\text{x}10^7$ time slots, recording the load combination, the number of packet arrivals, T, and the number blocked, s. The experiment was repeated for a range of bandwidths. The bandwidths and number of samples at each bandwidth are shown in Table 3.6

The thresholds found by each method and an estimate of loss rate at this threshold[5] are shown in Table 3.6. The normal scheme is clearly flawed with

Table 3.6: Ethernet Experiments at Different Link Bandwidth.

Band-width (Mbps)	Number of Samples		Sub-sample Factor	Threshold Found & Loss Rate at Threshold on (train,test) Set relative to $p* = 10^{-6}$		
	Train	Test		Normal	Weighting	Aggregate
10	1569	1080	230	0.232 (10, 4)	0.139 (0.8, 1)	0.105 (0.1, 0.08)
17.5	2447	3724	180	0.415 (20, 30)	0.268 (0.5, 0.9)	0.215 (0.003, 0.4)
30	6696	4219	230	0.508 (7, 40)	0.333 (4, 0.05)	0.286 (0.3, 0.02)
100	1862	N.A.	180	0.688 (10, N.A.)	0.566 (0.5, N.A.)	0.494 (0/N.A.)

losses 10 times higher than $p*$, the weighting scheme's loss rate is apparently unbiased with results around $p*$, while the aggregate scheme develops a conservative boundary below $p*$. To test the boundaries, we repeated the experiment generating source combination samples using the 54 sources not used in the training. Table 3.6 also shows the losses on this test set and indicates the training set boundaries produce similar results on the test data.

Applying the Markov model based method in [Gue91] to this ethernet data (treating each packet arrival as an ON period and calculating necessary parameters from there[6]), all but the very highest loads in the training sets are classified as acceptable indicating the loss rate would be orders of magnitude higher than $p*$. If the conservative peak rate is applied, since the original ethernet data rate is 10Mbps, only one source is accepted per 10Mbps of bandwidth. The average source load is 0.0014 and does not increase with increasing bandwidth. The adaptive method takes advantage of better trunking at increasing bandwidths, and carries two orders of magnitude more traffic.

5.3 MIS-SPECIFIED PARAMETERS

The adaptive method as described here does not assign any physical meaning to the representation. Thus any systematic monotonic transformation of the parameters will not change the resulting decisions. For instance, multiplying the feature vector by α and recomputing the classifier results in the parameter vector in (3.10) being divided by α, yielding the same decision boundary.

Random noise added to the parameters would affect the decisions. For instance, the average load could be a measurement over a short period and not be an accurate estimate. To test the method's ability to capture this disturbance[7], the Geometric data from Table 3.4, was randomly divided into two equal sized sets. To the first, a uniform random $[-x, +x]$ was added to the load

5. To get loss rate estimates at these thresholds, the data set, $\{(\phi_i, s_i, T_i)\}$, were ordered by ϕ and the 20% of the data set's samples below each method's threshold were averaged via $\Sigma_i\, s_i/\Sigma_i\, T_i$. Since accepted loads would be below the threshold this is a typical loss rate. 20% was chosen since smaller percentages had a high sensitivity on their loss estimates (in particular they were not monotonic in the threshold).

6. This is an unrealistic use of the method in [Gue91] since the data is known to be non-Markovian and, in ethernet, large packets are broken into blocks of less than 1518 bytes so the average ON period estimate was optimistically small. Several variations of ON and OFF period definitions were tried, and none did any better.
The point is to show what a typical Markov based technique would predict.

7. While in general the adaptive methods would be able to capture this effect, the preprocessing methods in Section 4 do not strictly apply since they assume a single underlying loss rate at $\bar{\phi}$ and not a distribution of loss rates. Despite this good results are obtained except for the largest noise values.

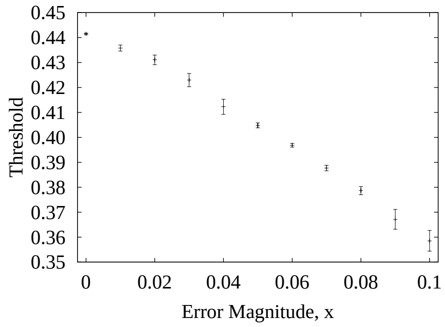


Figure 3.8 Decision threshold as a function of uniform error within [–*x*, +*x*] added to the input feature. Points are mean and one standard deviation of six different splits into test and train sets.

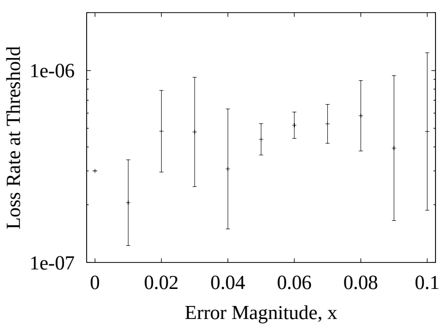


Figure 3.9 Average loss rate of test set samples within [–*x*, +*x*] of threshold ($p* = 10^{-6}$). Points are geometric mean and one standard deviation of six different splits into test and train sets.

feature. The aggregate method was applied. Future samples at the resulting threshold would have true load uniformly distributed within $\pm x$, so the error rate was computed by averaging all samples in the second set within x of the threshold[8]. The results are plotted in Figures 3.8 and 3.9. As expected, the decision threshold decreased with increasing error magnitude to allow for the noise in the measured loads. Except for the largest noise values, the loss rate was kept below the target, $p* = 10^{-6}$.

6. CONCLUSION

This chapter introduces a formal notion of statistical multiplexing decision functions for classifying which combinations of traffic will meet or not meet QoS. These functions form the core of admission control and routing algorithms. Theoretically and experimentally it was shown:

1. Fixed decision functions depend on source and node models that do not universally apply resulting in significantly lower efficiency or robustness than an optimal decision function.

2. Adaptive decision functions make few if any assumptions about the source and node models, and can achieve optimal decision boundaries.

For this reason it is argued adaptive schemes should be considered immediately for existing network applications.

8. The number of underlying simulated timeslots represented by this average was large, ~ $x \cdot 1\text{x}10^{11}$. In the no noise case ($x = 0$), the loss rate from Table 3.5 was used.

The adaptive method is optimal *for a given representation and distribution of source combinations*. Thus, it benefits from additional work on source models and their statistical parameters that better represent traffic. It was shown to work well across varied traffic distributions, and successfully treat small samples, correlations between sources, and mis-specified parameters.

Future work needs to address several practical questions: How the adaptive scheme would be implemented in an operating network so as to avoid excessive QoS violations while adapting; on what time-scale should adaptation take place; and can it address more realistic scenarios, such as multiple QoS classes, and other QoS parameters than packet loss rate. This chapter gives confidence these questions can be successfully answered.

Appendix A: Separable Representations

This appendix proves separable representations are required for fixed decision functions to have non-zero efficiency and robustness.

Theorem 1: Given a representation, $\Phi(\Sigma)$, there exists a fixed $C(\bar{\phi})$ with $E(C) > 0$ and $R(C) > 0$ if and only if $\Phi(\Sigma)$ is separable.

Proof: Suppose there exist two source combinations, Σ_1 and Σ_2, such that $\Phi(\Sigma_1) = \Phi(\Sigma_2)$ and $Q(\Sigma_1)$ meets QoS while $Q(\Sigma_2)$ does not. Let $\bar{\phi} = \Phi(\Sigma_1) = \Phi(\Sigma_2)$. Choose any C. If $C(\bar{\phi}) = -1$ then choose a distribution consisting solely of Σ_1. The optimal classifier accepts this combination so $E(C) = 0$. If $C(\bar{\phi}) = +1$, choose a distribution consisting solely of Σ_2. In this case, the optimal classifier will reject the combination so $R(C) = 0$.

Suppose instead $\Phi(\Sigma)$ is separable. Then, $C(\Phi(\Sigma)) = 1$ if $Q(\Sigma)$ meets QoS requirements, $C(\Phi(\Sigma)) = -1$ otherwise is well defined and will always be optimal regardless of $f(\Sigma)$. Q.E.D.

Appendix B: Simulation Model and Ethernet Data

We describe simple node and traffic models used in this chapter's simulations. The node is modeled as a discrete-time single-server queueing model where in each time slot one packet can be processed and zero or more packets can arrive from different sources. All the packets arriving in a time slot are immediately added to a buffer, any buffer overflows would be discarded (and counted as lost), and if the buffer was non-empty at the start of the timeslot, one packet sent. The server's buffer is fixed at 1000 packets. All rates are normalized by the service rate.

Appendix C: Asymptotic Optimality of the Adaptive Method

Are there any consistent or robust adaptive multiplexing decision functions? The answer is yes. This appendix will not show this rigorously, but instead sketches the proof outline. For more details the reader is directed to [Bro95]

[Bro99b]. For a specific feature, $\bar{\phi}$, we can consider three cases. In the first case, according to the distribution, f, the probability of getting samples at or near $\bar{\phi}$ is zero, i.e. $\bar{\phi}$ is *unsupported* by f. In this case, we don't care, since the value of the classifier at this feature does not affect consistency or robustness. In the second case, $q_i(\bar{\phi}) = \tau_i$ for some i (see eq. (3.2)). In this case, the classifier may or may not agree with the optimal classifier defined by (3.3). But, for continuous valued QoS metrics the measure of $\bar{\phi}$ where $q_i(\bar{\phi}) = \tau_i$ is likely zero.

The third case, is where $q_i(\bar{\phi}) \neq \tau_i$ for all i and $\bar{\phi}$ is supported by f. This is the usual case we are concerned with. We assume that $q_i(\bar{\phi})$ is continuous and there exists some unbiased and consistent point estimator of the $q_i(\bar{\phi})$. For a given data set we can define a neighborhood around $\bar{\phi}$ and use the estimators of the QoS metrics in (3.2) to decide the classifier output. As the sample size grows we can simultaneously shrink the neighborhood size while increasing the number of samples in the neighborhood so the estimates are consistent estimates of the true QoS metric. Thus, a consistent estimator is possible.

While this approach guarantees consistency (an asymptotic result), it says nothing about the confidence of the estimates, (3.9), for finite sample sizes. Unfortunately there are many pitfalls possible depending on f, and q_i. For this reason, Section 4, uses the approach of first deciding where confident estimates can be made and then fitting a function to these estimates, implicitly encapsulating assumptions about the smoothness of f, and q_i.

Acknowledgments

We gratefully acknowledge helpful discussions with Krishnan Komandar, Mark Garrett, Walter Willinger. The Bellcore traces were gathered by D.V. Wilson. This work was supported by NSF CAREER Award NCR-9624791.

References

[Bis95] Bishop, C., *Neural Networks for Pattern Recognition*, Oxford U. Press, Oxford, 1992. 482p.

[Bro95] Brown, T.X, "Classifying loss rates with small samples," in *Proc. of IWANNT*, Erlbaum, Hillsdale, NJ, 1995. pp. 153–161,

[Bro97] Brown, T.X, "Adaptive access control applied to ethernet data," *Advances in Neural Information Processing Systems*, 9, MIT Press, 1997. pp. 932–8.

[Bro99a] Brown, T. X, Tong, H., Singh, S., "Optimizing admission control while ensuring quality of service in multimedia networks via reinforcement learning," in *Advances in Neural Information Processing Systems*, 11, MIT Press, 1999, pp. 982–8.

[Bro99b] Brown, T. X, “Classifying loss rates in broadband networks,” in *INFOCOMM ‘99,* New York, April v. 1, pp. 361–70, 1999.

[Che92] Chen, X., Leslie, I.M., “Neural adaptive congestion control for broadband ATM,” *IEE Proc.-I*, v. 139, n. 3, pp. 233–40, 1992.

[Cho94] Choudhury, G.L., Lucantoni, D.M., Whitt, W., “On the effectiveness of admission control in ATM networks,” in the 1*4th International Teletraffic Congress* in France, June 6-10, 1994. pp. 411–20.

[Dud73] Duda, R.O., Hart, P.E., *Pattern Classification and Scene Analysis*, Wiley & Sons, New York, 1973.

[Elw93] Elwalid, A. I., Mitra, D. “Effective bandwidth of general Markovian traffic sources and admission control of high-speed networks,” *IEEE/ACM Trans. on Networking*, v. 1, n. 3, June 1993.

[Est94] Estrella, A.D., Jurado, A., Sandoval, F., “New training pattern selection method for ATM call admission neural control,” *Elec. Let.*, v. 30, n. 7, pp. 577–9, Mar. 1994.

[Err96] Erramilli, A., Narayan, O., Willinger, W., “Experimental queueing analysis with long-range dependent packet traffic,” *IEEE/ACM T. on Networking*, v. 4, n. 2, pp. 209–3, April 1996.

[Gal95] Galmes, S., et al., “Effectiveness of the ATM forum source traffic description,” in *Local and Metropolitan Communication Systems*, v. 3. ed. Hasegawa, T., et al. Chapman and Hall, 1995. pp. 93–107.

[Gar94] Garrett, M.W., Willinger, W., “Analysis, modeling and generation of self-similar VBR video traffic,” in *Proc. of ACM SIGCOMM*, 1996. pp. 269–80.

[Gro96] Grossglauser, M., Bolot, J-C., “On the relevance of long-range dependence in network traffic,” in *Proc. of ACM SIGCOMM*, 1994. pp. 15–24.

[Gue91] Guerin, R., Ahmadi, H., Naghshineh, M., “Equivalent capacity and its application to bandwidth allocation in high-speed networks,” *IEEE JSAC*, v. 9, n. 7, pp. 968–81, 1991.

[Gue99] Guerin, R., Peris, V., “Quality of service in packet networks: basic mechanisms and directions,” *Computer Networks and ISDN Systems*, v. 31, n. 3, 1999. pp. 169–89

[Hey96] Heyman, D.P., Lakshman, T.V., “Source models for VBR broadcast-video traffic,” *IEEE/ACM T. on Networking*, v. 4, n. 6, pp. 40–8, 1996.

[Hir90] Hiramatsu, A., “ATM communications network control by neural networks,” *IEEE T. on Neural Networks*, v. 1, n. 1, pp. 122–30, 1990.

[Hir95] Hiramatsu, A., “Training techniques for neural network applications in ATM,” *IEEE Comm. Mag.*, October, pp. 58–67, 1995.

[Jam92] Jamin, S., et al., “An admission control algorithm for predictive real-time service,” *Third Int. Workshop Proc. of Network and Operating Systems Support for Digital Audio and Video*, 1992. pp. 349–56.

[Kaw95] Kawamura, Y., Saito, H., “VP bandwidth management with dynamic connection admission control in ATM networks,” in *Local and Metropolitan Communication Systems*, vol. 3. ed. Hasegawa, T., et al. Chapman and Hall, London, 1995. pp. 233–52.

[Kni99] Knightly, E.W., Shroff, N.B., “Admission Control for Statistical QoS: Theory and Practice,” *IEEE Network*, March/April 1999, pp. 20–9.

[Kri95] Krishnan, K.R., “The Hurst parameter of non-Markovian on-off traffic sources,” *Bellcore Technical Memorandum*, Feb., 1995.

[Lau93] Lau, W.C., Li, S.Q., “Traffic analysis in large-scale high-speed integrated networks: validation of nodal decomposition approach” *Proc. of INFOCOMM*, v. 3, 1993. pp. 1320–29.

[Lee96] Lee, D.C., “Worst-case fraction of CBR teletraffic unpunctual due to statistical multiplexing,” *IEEE/ACM Tran. on Networking*, v. 4, n. 1, Feb. 1996. pp. 98–105.

[Lel93] Leland, W.E., et al., “On the self-similar nature of ethernet traffic,” in *Proc. of ACM SIGCOMM* 1993. pp. 183–93, also in *IEEE/ACM T. on Networking*, v. 2, n. 1, pp. 1–15, 1994.

[Lev97] Levin, B., Ericsson Project Report, to appear.

[Mit88] Mitra, D., “Stochastic theory of a fluid model of producers and consumers coupled by a buffer,” *Adv. Appl. Prob.*, v.20, pp.646–76, 1988.

[Mit98] Mitra, D., Reiman, M.I., Wang, J., “Robust dynamic admission control for unified cell and call QoS in statistical multiplexers,” *IEEE JSAC*, v. 16, n. 5, pp. 692–707, 1998.

[Nev93] Neves, J.E., et al., “ATM call control by neural networks,” in *Proc. Inter. Workshop on Applications of Neural Networks to Telecommunication*,” Erlbaum, Hillsdale, NJ, pp. 210–7, 1993.

[Nor93] Nordstrom, E., “A hybrid admission control scheme for broadband ATM traffic,” in *Proc. IWANNT*, Erlbaum, pp. 77–84, 1993.

[Nor94] Norros, I., “A storage model with self-similar input,” *Queueing Systems*, v. 16, pp. 387–96, 1994

[Pax94] Paxson, V., Floyd, S., “Wide-area traffic: The failure of Poisson modeling,” in *Proc. of ACM SIGCOMM*, 1994. pp. 257–68.

[Ton98] Tong, H., Brown, T. X, “Estimating Loss Rates in an Integrated Services Network by Neural Networks,” in *Proc. of Global Telecommunications Conference (GLOBECOM 98)*, v. 1, pp. 19–24, 1998.

[Ton99] Tong, H., Brown, T.X, “Adaptive call admission control under quality of service constraints: a reinforcement learning solution,” to appear in *IEEE JSAC*, Feb. 2000.

[Tra92] Tran-Gia, P., Gropp, O., “Performance of a neural net used as admission controller in ATM systems,” *Proc. GLOBECOM 92*, Orlando, FL, pp. 1303–9.

[Wil95] Willinger, W., Taqqu, M.S., Sherman, R., Wilson, D.V., “Self-similarity through high-variability: statistical analysis of ethernet LAN traffic at the source level,” *Bellcore Internal Memo*, Feb. 7, 1995. Also in *IEEE/ACM T. on Networking*, v. 5, n. 1, pp. 71–86, 1997.

Timothy X Brown received his B.S. in physics from Pennsylvania State University in 1986 and his Ph.D. in electrical engineering from California Institute of Technology in 1991. He has worked at the Jet Propulsion Laboratory and Bell Communications Research. Since 1995 he is an Assistant Professor at the University of Colorado, Boulder. His teaching and research areas include: Telecommunication Systems, Wireless, Switching, ATM, Networking, and Machine Learning. He received the NSF CAREER Award in 1996.