2

This is to certify that the

dissertation entitled

presented by

Lily Cheng

has been accepted towards fulfillment
of the requirements for

Ph.D. degree in Computer Science

Herman D.
Major professor

Date 4-1-96

PLACE IN RETURN BOX to remove this checkout from your record.
TO AVOID FINES return on or before date due.

DATE DUE	DATE DUE	DATE DUE
JUN 0 9 1998		
OCT 1 9 1998		

MSU Is An Affirmative Action/Equal Opportunity Institution

A Call Admission Control Framework Based on Empirical Traffic Measurements

By

Lily Cheng

AN ABSTRACT OF A DISSERTATION

Submitted to
Michigan State University
in partial fulfillment of the requirements
for the degree of

DOCTOR OF PHILOSOPHY

Department of Computer Science

Abstract

A Call Admission Control Framework Based on Empirical Traffic Measurements

By

Lily Cheng

Asynchronous Transfer Mode (ATM) has been accepted as the transport layer protocol for supporting Broadband-ISDN (B-ISDN) services. Call admission control in ATM-based networks is an important issue because it enables the network to achieve efficient use of network resources while maintaining good quality of services (QoS) to all network users. We present a call admission control framework based on the following crucial components: traffic measurements, a call admission control algorithm, and a scheduling policy. Our work is empirically based, as we have identified important traffic characteristics from a large amount of data collected in an ATM testbed. The call admission control algorithm accepts a call based on the **`traffic descriptors`** and **`QoS class`** requested by the given call. A scheduling algorithm is used by an ATM model to preserve different QoS when queuing occurs. Our algorithm can be used to provide different QoS for different classes of traffic.

To Mi, without whom this work would not have been possible.

Acknowledgments

I would like to thank the members of my Ph.D. committee for their guidance and support during my Ph.D. study. I would particularly like to express my appreciation to Dr. Herman D. Hughes, my advisor, for his patience of listening to my idea and giving me freedom to define my own research interests. It was a privilege to work with Professor Hughes, who gave financial support, provided a state-of-art research lab, and supported me numerous conference trips. I am also very grateful to Dr. Abdol H. Esfahanian for his encouragement; to Dr. Phlip K. McKinley, who could always come up with challenging questions; to Dr. Rauol D. LePage for his possitive comments.

A lot of friends made my stay at MSU enjoyable. To name a few, Cosine Lai, Jennifer Lai, Jennice Chao, Joe Chao, Tina Chang, I-Wen Huang, Ling-Na Tsai, etc. I do wish to acknowledge the Lees, who had been my precious friends along my graduate school years; Frank Northrup, who showed me the art of being a system administrator; Yih-Jia Tsai, with whom I had a lot of brain-storming and fruitful disscussion. Special thanks to Hui-Chia, who was always by my side, and delicated her love to Jo-Jo.

To my parents, who gave me my wings.

To Ting-Pang, for his endless love, patience and support.

To Jo-Jo, who completely changed my life.

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 Overview

Broadband ISDN (B-ISDN) [1, 2] is a powerful communication architecture that can support a large variety of traffic such as video-telephony, multimedia conferencing, high-definition TV (HDTV), distributed computing, client server traffic, medical imaging, and data transfer. Traffic propagating through B-ISDN can be grouped into two categories [3, 4]: variable bit rate (VBR) traffic and constant bit rate (CBR) traffic.

The ability to satisfy different service requirements for B-ISDN makes Asynchronous Transfer Mode (ATM) the most popular transport protocol for emerging high-speed networks, for both wide-area networks and local-area networks. In an ATM network, the traffic stream is packetized into fixed-size blocks called *cells*, each containing a header field and an information field. Cells are asynchronously multiplexed on transmission links and switched without window flow control.

Applications communicating via B-ISDN require that the network guarantee a certain level of quality of service (QoS). The QoS requirements for different applications may vary. Parameters associated with QoS include cell loss rate, cell delay, and cell delay variation. Cell delay occurs because of propagation delay, switching delay, and queuing delay, whereas cell loss is caused by transmission error, lack of buffer space, and cell delay violation. These parameters determine whether the ATM network fulfills its service contract for a given connection.

Service principles regarding traffic management, buffer management, and scheduling policies vary according to different classes of traffic. For example, real-time traffic such as voice/video is delay sensitive and therefore requires rapid transfer. Even within delay-sensitive traffic (e.g., voice or video), different traffic streams may have different delay requirements. If packets from real-time traffic cannot be delivered within their delay requirements, they are considered lost. File/data transfer is loss-sensitive traffic that requires error-free transmission. While real-time traffic only needs a small buffer size to limit the delay, data traffic needs a large buffer size to prevent data losses.

Generally speaking, the delay for connection-oriented services is smaller than that for connectionless services. For automatic generated data transfer such as daily database exchanges, QoS parameters can easily be defined, and this enables cell loss rate to be guaranteed. Both peak cell rate (PCR[1]) and sustainable cell rate (SCR[2]) can be controlled by a traffic shaping and flow control mechanism.

[1] PCR is also referred to as peak rate.

[2] SCR is also referred to as mean rate.

A fundamental characteristic of high-speed networks is that the ratio of propagation delay to cell transmission time and the ratio of processing time to transmission delay are both very high [5, 6]. This high propagation delay ratio will affect QoS. Due to the high speed of ATM networks and the provision of real-time services within the network, conventional reactive flow control mechanisms are not fast enough to prevent misbehaved users from causing congestion.

Ideally, the network should provide the QoS requirements for communication sessions while simultaneously taking advantage of the benefits gained by statistical multiplexing. Such a goal poses difficult challenges covering several QoS issues (e.g., traffic shaping, call admission, congestion control, buffer management, and cell scheduling) that need to be implemented at network nodes. Since the problem of providing different classes of QoS requirements cannot be solved by a single technique, a practical strategy is to decompose this problem into several subproblems and use a combination of approaches to address this single problem.

The decision of which call admission control scheme to use is affected by many factors, including the characteristics of the traffic and the types of QoS desired (e.g., deterministic or statistical guarantees). One of the objectives of this research is to study traffic characteristics and investigate call admission control schemes that provide different QoS requirements for various classes of traffic. Network QoS control algorithms heavily depend on the characteristics of measured ATM traffic [7]. Traffic models used in simulation or analytic studies require users to provide traffic descriptors to characterize their applications.

Traffic descriptors associated with each individual application include a set of parameters such as *peak rate*, *mean rate*, *maximum burst length*, and *burstiness*, that are used to define network traffic characteristics. The traffic descriptors are the key parameters in traffic shaping, call admission control [8], and usage parameter control (UPC) [9] of an ATM network. The traffic descriptors mentioned above can also be used to develop ATM traffic models. Although the emphasis of this research will be on call admission control schemes, some work will also be done on traffic characteristics that impacts admission control schemes.

In this research we present a call admission control framework based on empirical traffic measurements and study the relationship between call level and cell level control. Routing, flow control, and buffer management, although important issues in the management and control of ATM networks, are beyond the scope of this research.

1.2 Call Admission Control

When a new call requests admission to the network, it provides the network with a tuple: (*traffic descriptors*, *QoS parameters*). The ATM traffic descriptor is the generic list of traffic parameters that can be used to capture the traffic characteristics of an ATM connection [10]. QoS parameters are the set of conditions that a network must satisfy traffic when connections are made. Typical network users are not very knowledgeable about the actual value of the call set-up parameters associated with the previously described tuple. This call admission problem involves evaluating the network performance (taking into account the new call and those already ongoing), and then determining if the network can guarantee the QoS of the new call without affecting the QoS promised to ongoing calls. If the QoS requirements for both the new call and the existing ones can be satisfied, the new request is accepted; otherwise it is rejected.

The task of the admission control is to accept or reject arriving calls so as to maximize the output link utilization. An admission control scheme is constrained by (1) the need to guarantee the required QoS at the cell level to all calls admitted, (2) the need to limit the call blocking probabilities, and (3) the need to maximize the network link utilization. The objective of the call admission control scheme is to maximize network link bandwidth utilization while at the same time guaranteeing cell level QoS. In other words, from the user's point of view, the QoS requirements must be guaranteed; from the network's point of view, the goal is to maximize bandwidth utilization. Additionally, optimal call admission schemes may be very difficult because they involve several functions, such as bandwidth allocation, traffic shaping, and cell scheduling.

Users located at the edge of the network may want to make a connection. At the connection setup time, the tuple (*traffic descriptors*, *QoS parameters*) must be specified. Generally speaking, users do not know very much about traffic characteristics, so it is very difficult for them to provide meaningful traffic descriptors and QoS parameters. Thus, the problem encountered is often in the traffic characteristics.

After the connection request, the network requires a call admission control algorithm to decide whether to accept the call. Given the speed of the ATM networks, 155 *Mbps*, the algorithm has to be simple, effective, and implementable.

Once a connection is established, users must conform to the traffic descriptors and the network is committed to provide the services specified in the tuple (*traffic descriptors*, *QoS parameters*). But the users may not be able to fully comply with the traffic descriptors. Users may send more traffic to the network than they are allowed and cause congestion. To ensure that users conform to their contracted traffic parameters, a control function has been defined [10] and is known as a *traffic policing*, a *traffic shaping*, or a *usage parameter control* function.

The policing function is needed to protect well-behaving channels from the misbehavior of faulty or malicious sources and from the occasional traffic fluctuation of a channel. In principle, the policing function observes the cell stream for the duration of the call and restricts its behavior to the contracted traffic parameters. It is used in conjunction with the *call admission control* (CAC) function to provide cell level QoS requirements. The leaky bucket algorithm [11] is the most popular traffic-shaping algorithm. In general, the traffic generated by the clients in an ATM network will not be well behaved and the clients will need a mechanism (e.g., the leaky bucket) to convert the traffic to some regulated pattern.

So the call admission control problem not only involves designing an efficient CAC algorithm, it also involves such functions as traffic characteristics, traffic shaping, and cell scheduling. Our study focuses on the entrance point to an ATM network where connections are requested by a local host. Figure 1.1 gives a view of the entrance point to an ATM network. Users at hosts residing at the edge of an ATM network must provide a tuple (*traffic descriptors*, *QoS parameters*) when making the connection request. This tuple will be used by the network for call admission. It can be observed from Figure 1.1 that call admission control issues encompass other research problems, such as traffic shaping and cell scheduling.

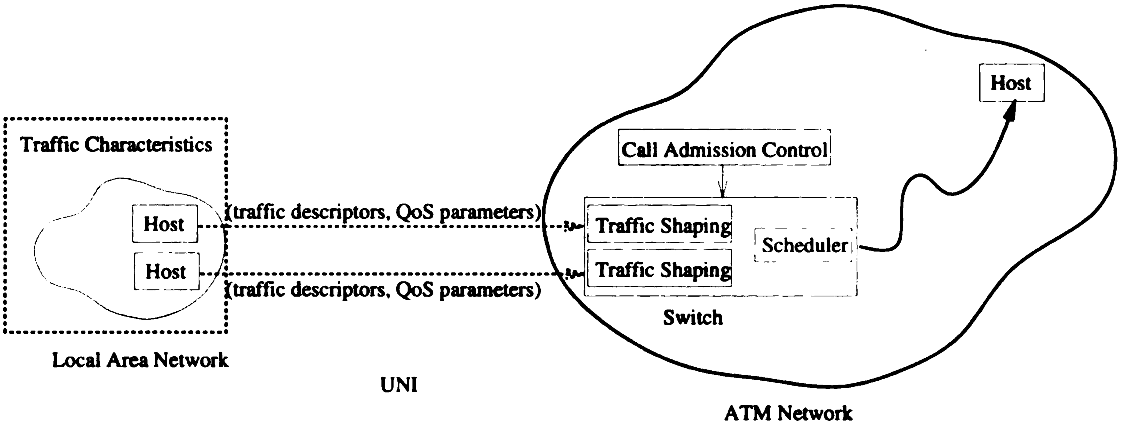


Figure 1.1: Network model at the entrance point of an ATM network.

After the connection is established, the network monitors each individual virtual channel connection to ensure that its traffic entering the network conforms to the traffic parameters (e.g., mean rate and peak rate) that users provide during call setup time. If the connection does not conform to the traffic descriptors, a traffic-shaping function is used to regulate the traffic. The objective of such a function is to protect other connections by assuring that network resources are not dominated by traffic that violates the negotiated parameters. After traffic shaping takes place, a scheduling policy is needed to assure QoS for different classes of traffic when queuing occurs.

Our focus will be on the actual application traffic behavior of an ATM network using the High-Speed Networking and Performance Laboratory (HSNP) available at Michigan State University. Traffic measurements are obtained and are used for evaluating the effectiveness of a proposed call admission control algorithm in providing different QoS requirements to diverse traffic classes.

In this research, we focus on a scheme that makes use of our ATM testbed where possible, and simultaneously relaxes users from providing a lot of parameters. We provide services for various QoS classes in the call admission control algorithm. Traffic class information can be carried in the Virtual Channel Identifier (VCI) field of the cell header as specified at the call setup time.

To this end, this doctoral dissertation focuses on the following objectives relative to call admission control schemes in ATM networks:

- Collecting traffic statistics for a diverse range of applications on an ATM LAN. From these statistics, identifying traffic descriptors and investigating the impact of bandwidth requirements for our call admission algorithm. Developing a workload model for ATM traffic based on the traffic characteristics collected.

- Studying the multiplexing effects of homogeneous/heterogeneous traffic mixes, which are essential for evaluating the performance metrics of our admission control scheme.

- Measuring the performance (e.g., call-blocking probability, bandwidth utilization, and efficiency ratio) of the proposed call admission scheme versus other admission control mechanism (i.e., peak rate allocation).

- Investigating the performance of the proposed cell scheduling scheme in terms of cell delay bounds.

1.3 Organization of the Thesis

The remainder of this dissertation research is organized as follows. *Chapter 2* starts with an introduction of the call admission control problem, together with a discussion of existing approaches toward solving this problem. It includes a review of background materials related to our call admission control framework, including diverse call admission control schemes, traffic characteristics collected from existing networks, and for a large variety of applications, an empirical workload simulator constructed from the traffic measurement studies, traffic-shaping schemes for confining to connection parameters, and the growing trend of combining scheduling and the call admission control mechanism.

Chapter 3 introduces the call admission control framework. A call admission control algorithm and a cell-scheduling algorithm are presented, along with a time-out mechanism that is used to integrate call-level control and cell-level control. This class-based call admission control algorithm can maximize the output bandwidth usage by dynamically measuring network load via simple network management protocol (SNMP).

Traffic measurements are performed that yield large amounts of data collected from a diverse range of applications. *Chapter 4* discusses experimental setups, measurement methodology, and traffic characteristics for various applications such as **Ftp**, **Telnet/Rlogin**, **Video**, **Audio**, and **Mosaic**. Traffic descriptors determined from the existing data show the pitfalls of the definitions of traffic descriptors.

To help facilitate the performance of the call admission control framework, *Chapter*

5 addresses the process of building an ATM workload model where various applications can be generated to better fit the traffic characteristics recognized from the existing testbed.

The performance results of the call admission control framework are studied using both homogeneous and heterogeneous traffic streams (*Chapter 6*). Homogeneous traffic streams provide insight into individual traffic behavior, while heterogeneous traffic streams investigate the network performance under the control of the proposed framework. *Chapter 7* provides a summary of this dissertation research and also address possible future directions.

Chapter 2

Problem Statement and Related Work

2.1 Problem Statement

In order to achieve different QoS requirements at the cell level for diverse traffic classes while simultaneously maximizing output link utilization and reducing call-blocking probability, we present a call admission control algorithm. An optimal call admission control scheme guarantees the QoS requirements promised to all on-going calls while at the same time preventing newly accepted calls from violating the QoS of existing ones. It also maximizes the output link utilization and the number of admitted calls.

The cell-level QoS may be trivially guaranteed by any scheduling mechanism if a conservative admission control policy is used to limit the utilization levels. Guaranteed cell-level QoS to admitted calls is not sufficient, however, if it comes at the cost of unreasonably high call-blocking probability and very low network resource utilization. A call admission algorithm cannot provide cell-level QoS requirements if it allows too many connections.

The task of admission control is now described in a more formal manner. Consider a traffic flow α, providing a tuple (*traffic descriptors, QoS parameters*) and requesting a connection to a network that is currently in state x, where $x = (x^1, x^2, \ldots, x^n)$. The number of connections of class i traffic is x^i, and $x^i \in \{0, 1, \ldots, N\}$. Can a connection request of class i traffic be accepted? The answer to this question will depend on values of the parameters associated with this tuple.

Deciding on the most appropriate set of traffic descriptors to fully describe traffic characteristics remains difficult. The parameters typically involved in this discussion are: peak rate, mean rate, mean burst length, maximum burst length, burstiness, peak-to-mean ratio, and so forth. QoS parameters discussed in the literature usually include cell loss rate, cell delay time, cell delay variance, and delay jitter. In other words, a call admission algorithm needs an input of ({peak rate, mean rate, mean burst length, maximum burst length}, {cell loss rate, cell delay time, cell delay variance, delay jitter}) in order to decide if a call is accepted and if the QoS is satisfied.

A manageable number of parameters is highly desirable in order to reduce the complexity of characterizing traffic. Commonly used traffic descriptors are peak rate, mean rate, and burstiness. In [12], we have shown the difficulties in finding an appropriate way of defining burstiness and applying burstiness to predict network behavior. We have only used two traffic descriptors in our call admission algorithm, (i.e., peak rate and mean rate), which are sufficiently effective to handle a wide range of traffic types. A rationale for choosing these two parameters is that: (1) peak rate reflects the worst-case scenario, which provides guarantee services to real-time traffic, and (2) mean rate represents the average bandwidth requirement for non-real-time services.

Several researchers [13, 14] investigating admission control schemes have made "accept/reject" decisions based on non-real-time parameter values, where a large number of parameters are involved. A problem with this approach is that a network may be underutilized if call admission is not based on real-time measurements of the network load. Some researches [15, 7] have presented optimal solutions based on theoretical approaches. This is evidenced by Guérin et al. [7], who consider the possibility of measuring the current network load for an admission control algorithm. However, the actual implementation is not addressed in their work.

There are also call admission control algorithms [16] based on the tuple (*traffic class*, *QoS parameters*). In such an algorithm, different types of traffic are grouped into the same class based on the traffic characteristics. This algorithm does not require users to provide actual traffic descriptors; neither does it yield an effective way of allocating bandwidth, because of different bandwidth requirements within a class.

In order to accommodate the fact that the same type of traffic (e.g., CBR traffic) has different traffic descriptors (e.g., peak rate), our call admission algorithm is based on the tuple (*traffic descriptors*, *QoS class*). A default set of traffic descriptors, which are obtained from traffic measurements, for several applications is maintained in our algorithm that prevents users from having to provide traffic descriptors. In addition, QoS class is a lot easier for users to select than to provide actual QoS parameters, which includes a set of parameters such as cell loss rate, cell delay time, and cell delay variation.

It should be noted that burstiness is not included in our traffic descriptors. The decision to accept or reject a call is made by conducting real-time measurements of the network load. Our approach is to dynamically measure current network load by using the Simple Network Management Protocol (SNMP). This technique differs from traditional methods that use burstiness to predict bandwidth utilization.

The simplified QoS parameters needed for our call admission algorithm include four different classes: Constant Bit Rate (CBR), Variable Bit Rate (VBR), Available Bit Rate (ABR), and Best Effort services (BE). CBR and VBR connections are allocated a full bandwidth equal to its peak rate, which serves as a guaranteed service. ABR connections are based on the mean rate requirements of the traffic and network bandwidth utilization. Best-effort traffic is intended to use the rest of the network link capacity, which is good for non-delay-sensitive data traffic. The credibility of our call admission algorithm is demonstrated by comparing its performance to that of other schemes. The performance metrics are call-blocking probability, bandwidth efficiency, and efficiency ratio for the call level, and cell delay time for the cell level.

2.2 Problem Discussion and Background

From our traffic measurement observations, networks are underutilized for bursty traffic, which has a high peak-to-mean ratio and small peak durations. In order to study this problem, we use the Simple Network Management Protocol (SNMP) to measure the network load before a call is accepted for less demanding services such as ABR traffic. Our work is based on experiments from real traffic measurements, with the goal of maximizing the bandwidth utilization and the number of admitted calls, while simultaneously providing QoS to different classes of traffic.

This call admission algorithm requires input of the tuple (*traffic descriptors*, *QoS class*) and real-time measurements of the network loads via SNMP. It is especially designed for bursty traffic, whose traffic behavior is difficult to predict. Call requests from such classes of traffic are accepted by dynamically measuring the network load, instead of using a statistical model to predict the network bandwidth utilization. This measurement can be obtained at the call-request time by using SNMP. Since this protocol is very complex, we only request information necessary to realize the output link utilization.

We have also investigated the traffic characteristics in an ATM network and identified traffic descriptors for several applications that dominate wide area network (WAN) traffic. A default set of traffic descriptors {*peak rate*, *mean rate*} associated with each application is maintained for our algorithm in order to prevent users from having to specify these parameters.

As a way of controlling queuing of cells, a scheduling algorithm is used to preserve QoS for different classes. In our model, bandwidth can be allocated to various connections at different levels (call level and cell level). Bandwidth will be allocated to a call if the network decides to accept the call. Bursty sources may not be able to fully utilize the allocated bandwidth. Full utilization of the total link capacity is achieved if the rest of the bandwidth is allocated to best-effort traffic. For connections that request best-effort services, calls are accepted. Cell-by-cell scheduling is needed in order to decide which cell to transmit next. All cells in the best-effort class are marked in order to avoid degrading the QoS of other service classes, once the network bandwidth becomes insufficient for the current connections. In this case, cells belonging to best-effort traffic are dropped.

The following problems must be addressed in order to refine our algorithm:

- We must determine the different types of traffic sources that we are likely to encounter. For example, we need to investigate the impact of different traffic types (e.g., the impact of CBR and VBR traffic).

- We must study the multiplexing effects of homogeneous (i.e., all traffic streams belong to the same application) and heterogeneous (i.e., traffic streams belong to more than one application) traffic streams. Given a certain mix of different traffic sources, the total traffic generated by them must be characterized, in terms of multiplexing effects.

- We must determine an appropriate scheduling scheme for our call admission algorithm.

2.3 Related Work

There has been extensive research on admission control policies for ATM-based networks [17, 18, 16, 19, 20, 21, 15, 22, 14]. Different admission control strategies allow for different treatments of call requests. Calls may be either blocked or queued, or some combination of the two. In our work, the focus on admission control is on loss strategies, which means calls are either admitted or rejected. Some topics that are related to admission control scheme will be discussed in the rest of this chapter.

The rest of this section is intended to describe call admission control as it relates to the issues discussed in this dissertation. The following topics are covered: the call admission control algorithm, traffic characteristics and their relation to network behavior, traffic shaping for call setup parameters, and cell scheduling to provide different QoS.

2.3.1 Call Admission Control Algorithms

An admission control scheme decides whether to accept or reject a connection request based on the required QoS. Intuitively, connections can be admitted as long as the peak bandwidth requirement of a new connection is less than the available link bandwidth. This technique, peak rate allocation scheme, is simple but exhibits a fairness problem. The admission of a few connections requiring large amounts of bandwidth may preclude several smaller connections, thus resulting in a high call-blocking probability.

Several threshold-based schemes [23, 24] address this problem and adapt a threshold allocation mechanism. A connection is thus admitted if its bandwidth requirement does not exceed a predetermined threshold of available bandwidth. This scheme exhibits some disadvantage. Consider a network where connections tend to request big calls. Rejecting a lot of big calls does not result in accepting small calls. This will waste a lot of bandwidth by preserving a predefined amount of the network bandwidth.

There are call admission control schemes based on analytical approaches. Ross and Tseng [15] consider a pure loss system for two classes of traffic with different bandwidth requirements. They formulate the optimal call admission control problem as an unconstrained maximization of expected throughput using dynamic programming and solve it by the method of policy iteration. It is shown that more efficient solutions to this problem can be obtained by value iteration. The dynamic programming problem is reformulated as a linear problem that allows the optimal control policy as a maximization of expected throughput. A call-blocking constraint is applied to enforce fairness.

Ferrandiz and Lazar [14] address the admission control problem for sessions of real-time traffic. They use analytical methods to find the optimal admission control policy, subject to constraints on end-to-end cell delay, average cell loss rate, and average gap length (number of consecutive lost cells). The real-time cell arrivals are modeled by a Markov-Modulated Poisson Process (MMPP), and the scheduling is assumed to be first-come first-served (FCFS). For more realistic cases and more complex source models, analytical approaches become mathematically intractable. The aforementioned analysis [15, 14] are confined to a subset of applications (e.g., real-time traffic).

Several call admission control mechanisms are based on traffic measurements. Tse and Zukerman [25] propose a scheme based on a Gaussian traffic model to predict loss probability. This scheme requires each call to declare only its peak rate and it assumes that the traffic characteristics are measured during its holding time. Knightly and Zhang [26] propose a deterministic bounding interval dependent (D-BIND) model to capture video source characteristics from their traffic measurements in [27, 28]. This model provides a deterministic QoS on the traffic sources if the source can be described by the D-BIND model:

$$\frac{1}{I}\int_{t}^{t+I} r(\tau)d\tau \leq B(I)\,\forall t,\; I > 0,$$

where $B(I)$ is the bounding rate over an interval of length I, and $r(\tau)$ is the source's instantaneous rate at time τ.

Jamin et al. [17] define an admission control scheme for the Clark, Shenker, Zhang (CSZ) service model [18] in a single Integrated Services Packet Network. The CSZ model offers both guaranteed and predictive real-time services. Since the guaranteed flows are isolated from one another, call admission is only required for the predictive service. When making an admission control decision, the current traffic load is measured. This measurement-based admission control scheme assumes that time-averaged measurement of the actual traffic can be obtained and is a valid predictor of future behavior. However, their work does not address how to measure the actual traffic load and the impact of time resolution.

Ramamurthy [16, 29, 20, 21] proposes a UPC call admission control framework based on three level controls: call-level, burst-level, and cell-level controls. The call admission algorithm needs an input tuple (*traffic class*, *QoS parameters*). Central to this framework is a traffic classification scheme that maps applications to specific traffic classes based on their QoS requirements and the statistical traffic characteristics. This work does not provide a systematic way of mapping from applications to traffic class.

Verma [30] identifies four crucial components of a resource management architecture: traffic specification, rate control mechanism, scheduling policy, and admission control scheme. He proposes a call admission scheme for three classes of traffic sources. This scheme can be used to reserve resources for bursty and smooth traffic sources and leads to a network link bandwidth usage of 40%. Taking into account the statistical multiplexing and best-effort traffic, a bandwidth usage higher than 40% is desired.

Ferrari and Verma [22] present a joint scheduling and admission control algorithm for a system with two classes of traffic. They use a version of earliest due date (EDD) scheduling along with a priority mechanism. This algorithm guarantees QoS for a "deterministic" class of traffic that cannot tolerate packet loss. It also reserves enough bandwidth for each admitted call to transmit continuously at peak rate without taking advantage of statistical multiplexing. The computation of the schedulable region in their scheme can take up to an order of $O(m \times n)$ memory space.

There are several call admission control policies based on cell-level and call-level QoS requirements. In [31, 13] a modular approach is pursued, involving cooperation between the processors responsible for scheduling and call admission control. It is shown that the schedulable region can be used at the admission control level to choose the optimal admission policy that guarantees QoS at both call and cell level. The *admissible load region*, which defines the loading conditions under which QoS can be guaranteed on both levels, is also presented. The authors [31, 13] consider scheduling algorithms for use in networks carrying three classes of traffic. Simulation experiments with call-level traffic are used to define a *schedulable region* (a subset of the space of the number of calls for which the cell-level QoS is met). Our work differs from Jamin's work [17] by having one service class between the guaranteed and predicted services and one best-effort class that makes use of the remaining network bandwidth.

There are several call admission control schemes that involve routing a call. Garary and Gopal [32] address the call preemption selection problem when the network encounters node failures. They show that the problem of selecting which calls to preempt in order to minimize the number of calls to be preempted or to minimize the amount of bandwidth to be preempted is NP-complete. Peyravian [33] presents a preemption algorithm based on different priorities of calls where lower-priority calls are preempted.

Such a wide variety of call admission control schemes illustrates the difficulty of providing different QoS for a diverse range of applications. The proposed research will focus on an admission control providing four classes of QoS requirements. In [34], it is observed that interactive applications, such as **ftp** and **rlogin/telnet**, only use bandwidth less than their mean rate during a large percentage of time. Hence, measured link bandwidth usage is less than (e.g., when traffic is equal to its mean rate) or equal to (e.g., when traffic is equal to its peak rate) the assigned bandwidth. The specific issues to be studied in this research include bandwidth requirements for individual applications and multiplexing effects for homogeneous and heterogeneous traffic.

2.3.2 Traffic Characteristics

As indicated in Figure 1.1, traffic characteristics play a vital role in call admission control schemes [34]. A call admission algorithm needs to provide various QoS requirements for diverse types of applications in which traffic characteristics vary substantially.

Traffic measurements taken from real data are highly desirable for estimating traffic descriptors, validating traffic models, designing congestion control algorithms [35, 36], implementing call admission control schemes [34], and developing switch architectures [37, 38]. In addition, network Quality of Service (QoS) control algorithms such as call admission mechanisms [34], routing algorithms, and bandwidth allocation policies depend heavily on the characteristics of ATM traffic [7]. There have been many local-area network (LAN) and wide-area network (WAN) measurement studies [38, 39, 40, 41, 42].

Many studies reported in the literature [17, 18, 16, 19, 20, 21, 15, 22, 14, 43] relative to call admission control are based on simulation. These studies involve the use of either a Poisson process, an on-off model, a Markov Modulated Poisson Process (*MMPP*) [44], or a Packet Train model [45] to generate the network traffic. While these traffic models vary in many fundamental aspects, they rely on different assumptions for determining essential parameters.

Leland et al. [46, 42, 47] trace Ethernet LAN traffic collected between 1989 and 1992, and propose a fractal traffic model. The traces demonstrate that the aggregated LAN traffic is statistically *self-similar*, and there is no natural length of a burst. *Self-*

similar phenomena display structural similarities across a wide range of time scales ranging from milliseconds to minutes.

The notion of self-similarity is defined as follows. Let $X = (X_t : t = 0, 1, 2, \cdots)$ be a *covariance stationary* stochastic process with an autocorrelation function $r(k) = E[(X_t - \mu)(X_{t+k} - \mu)]/E[(X_t - \mu)^2]$, $(k = 0, 1, 2, \cdots)$ that depends only on k. The autocorrelation function of X can be assumed to be of the form

$$r(k) \sim a_1 k^{-\beta}, as\ k \to \infty,$$

where $0 < \beta < 1$. For each $m = 1, 2, 3, \cdots$, let $X^{(m)} = (X_k^{(m)} : k = 1, 2, 3, \cdots)$ denote a new time series obtained via $X_k^{(m)} = 1/m(X_{km-m+1} + \cdots + X_{km}), (k \geq 1)$. Let $r^{(m)}$ denote the corresponding autocorrelation function for $X^{(m)}$. The process X is called *self-similar* with self-similar parameter $H = 1 - \frac{\beta}{2}$, if $r^{(m)}(k) = r(k)$, for all $m = 1, 2, \cdots (k = 1, 2, 3, \cdots)$.

Their work also shows that the degree of self-similarity (defined via *Hurst parameter H*) typically depends on the utilization level of the Ethernet and can be used to measure *burstiness* of LAN traffic. Estimated value of Hurst parameter directly from the corresponding plot with a simple least-squares fit results in $H \approx .80$.

Cáceres et al. [48, 49, 50, 40, 51, 52] trace individual TCP conversations from wide-area Internet traffic at three sites: UC Berkeley, University of Southern California, and Bell Communications Research. Their traffic trace consists of a total of 5,891,622 TCP packets from the above three sites. The study shows that the breakdown of traffic varies greatly from site to site. However, the characteristics of conversations are essentially identical between the three sites. Also, TCP packets make up roughly 80% of all wide-area network traffic, and hence a model based on TCP traffic is necessary for studying the network behavior.

Table 2.1[1] shows the breakdown of unidirectional TCP traffic in units of packets, or bytes by conversations from [48]. Although this study analyzes a significant portion of existing Internet traffic, it does not consider future network traffic such as multimedia applications.

Traffic Type	%Packets			%Bytes			%Conversions		
	UCB	USC	BELL	UCB	USC	BELL	UCB	USC	BELL
ftp	12.0	5.0	18.7	36.2	10.6	54.9	2.2	1.8	4.7
smtp	11.6	3.1	12.6	11.0	1.9	10.6	54.0	29.3	65.2
nntp	11.6	36.3	9.2	15.8	44.5	15.6	22.5	44.8	4.7
telnet	28.0	16.6	36.3	5.5	2.3	6.5	3.2	4.9	8.4
rlogin	15.5	5.8	18.5	2.8	0.7	3.1	1.6	1.5	4.1
Total	78.7	66.8	95.3	71.3	60.0	90.7	83.5	82.3	87.1

Table 2.1: TCP traffic breakdown (**ftp**: file transfer protocol; **smtp**: send mail transfer protocol; **nntp**: network news transfer protocol; **telnet**: remote terminal protocol; **rlogin**: remote login protocol).

Traffic descriptors need to be computed from the measured data for different applications. Previous experiments dealing with network traffic measurements have focused on the statistics regarding host reference patterns, the distribution of packet sizes, and packet interarrival times for WANs [39, 41, 48, 49, 53, 50] and LANs [38, 42]. While packet-level statistics play an important role in traditional networks, cell-level statistics are critical for ATM networks.

[1] The applications that appear in boldface are studied in our work. SNMP and NNTP will be considered in our future study. This restricted set was chosen due to our local ATM configuration.

In order to effectively study the traffic characteristics of ATM networks, we have conducted experiments to obtain measurements that facilitate the identification of traffic descriptors. We have performed traffic measurements from our ATM testbed, which concentrated on cell-level information. Cell-level statistics reflect the network link speed and the hardware/software speed limitations of cell segmentation and reassembly. An emphasis of our work is on investigating traffic characteristics that impact admission control schemes.

2.3.3 Empirical Workload Simulator

Simulation tools allow us to evaluate the performance of the proposed call admission algorithm by changing certain parameters to study the network behavior in a controlled environment. In order to drive our network simulations to evaluate the call admission algorithm with realistic models of existing traffic sources, we have developed a traffic source library, which will be addressed in Chapter 5.

Danzig and Jamin [52, 49] create a traffic source library, *tcplib*, for network simulation based on the work of Cáceres et al. [48, 50, 48, 51, 52]. Tcplib is a library to help generate TCP/IP network traffic. This library consists of a *stub-dependent* component (i.e., the load of individual TCP traffic differs by individual network) and a *stub-independent* component (their data shows that characteristics of a given TCP application is independent of the network). In the stub-independent component, tcplib models five of the following six applications: FTP, SMTP, NNTP, VMNET[2],

[2] VMNET, an IBM mail exchange application, is excluded.

TELNET, and RLOGIN. These applications are currently responsible for 96% of wide-area TCP/IP bytes. However, with increasing usage of Internet services, this percentage will change.

This experimentally-based library has several limitations, as identified in [52, 49]. For example, tcplib lacks several application-specific details and does not model either the FTP control packets or handshakes for SMTP or NNTP conversations. Neither does it model other applications such as voice and video that may eventually become an important component of wide-area network traffic.

We present traffic measurements that emphasize the development of a workload model. The construction of such a model can be regarded as a primary step in defining traffic descriptors from experimental data. Such descriptors will enable researchers to better formulate and validate both simulation and analytical models. These descriptors are needed for our call admission algorithm. The workload model presented consists of a collection of functions that can be used to simulate application programs. This model relies on measurements from a WAN environment (i.e., *tcplib* [49, 52]) and measurements from our ATM testbed. Such an empirically based model can be used to simulate LAN/WAN ATM traffic for studying communication protocols, scheduling algorithms, and congestion control schemes.

2.3.4 Traffic Shaper: Conformance to Connection Parameters

After a connection is established, users may not be able to fully comply with the traffic descriptors. Misbehaving channels can be caused by faulty or malicious sources. User traffic needs to be monitored and enforced to comply with traffic parameters established at the connection setup time. Therefore, several *traffic shaping functions* have been proposed as a means of regulating traffic according to traffic descriptors provided at call setup time. A traffic-shaping function should be simple, fast, and cost effective to implement in hardware [9]. Additionally, it needs to be located at the user-network interface (UNI). Traffic-shaping functions, also called *usage parameter control* functions, can be grouped into two categories: jumping window mechanisms [9, 54, 55], and the leaky bucket algorithm and its variants [3, 56, 57, 58, 59, 60].

In the *jumping window* (JW) mechanism, the time axis consists of consecutive windows of a fixed time interval T. This mechanism accepts a maximum number of cells N_{max} from a given source within a window. The *triggered jumping window* (TJW) mechanism works similarly to the JW mechanism, except that the new window is triggered by the first arriving cell. The *moving window* (MW) mechanism works similarly to the JW mechanism, except that the window slides along the time axis in the moving window mechanism. Figure 2.1 shows the JW, TJW, and MW mechanisms.

In the *exponentially weighted moving average* (EWMA) mechanism, windows are consecutive and have a fixed time interval T. The maximum number of cells allowed

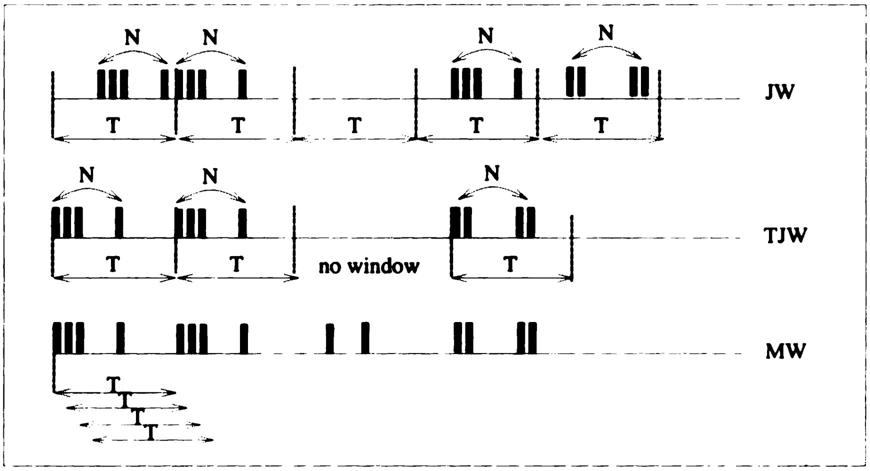


Figure 2.1: Different window mechanisms for peak rate regulations.

to transmit in the ith window, N^i_{max}, is

$$N^i_{max} = \frac{N-\gamma S_{i-1}}{1-\gamma}, 0 \leq \gamma < 1, \tag{2.1}$$

where

$$S_{i-1} = (1-\gamma)X_{i-1} + \gamma S_{i-2}, \tag{2.2}$$

with X_i being the average number of cells per window that the network is willing to accept from the connection. Solving the above two equations, N^i_{max} is given by

$$N^i_{max} = \frac{N-(1-\gamma)(\gamma X_{i-1}+\cdots+\gamma^{i-1}X_1)-\gamma^{i-1}S_0}{1-\gamma}, \tag{2.3}$$

where S_0 is the initial value of the EWMA mechanism. The constant γ represents the flexibility of the algorithm. If $\gamma = 0$, the EWMA mechanism becomes identical to the JW mechanism. The greater the value of γ, the more the algorithm allows for short-term fluctuations.

In [54], it is shown that the following inequality holds:

$$P^{MW}viol \geq P^{TJW}viol \geq P^{JW}viol \geq P^{EWMA}viol, \tag{2.4}$$

where $P^{X}viol$ denotes the violation probability for the mechanism X. The violation probability is the probability that cells violate their admission parameters. Therefore, a lower probability indicates more tolerance for short-term fluctuations.

Among the most popular traffic shapers proposed is the leaky bucket algorithm [11]. Such a traffic enforcement method involves only local information. While these algorithms are primarily designed to shape traffic, they also serve to prevent congestion within the network. Chao [61] designs a VLSI chip (called a sequencer) to implement this algorithm in hardware that contains about 200K CMOS transistors. Figure 2.2 shows the various leaky bucket models. Each cell entering the node has to get a token from the token pool before it can enter the network. If the token pool is empty, the cell is dropped and will be considered lost.

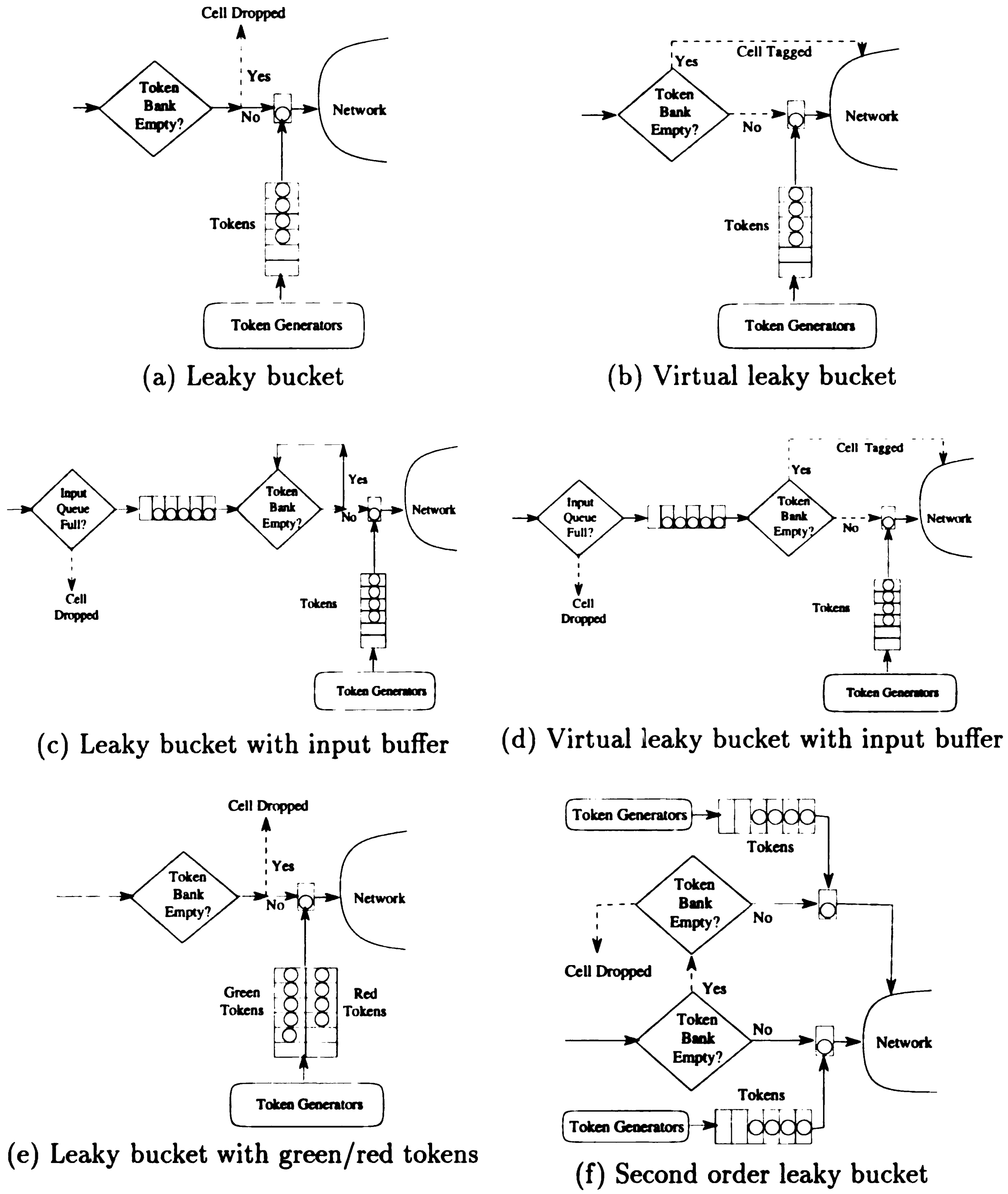
The leaky bucket algorithm is characterized by two parameters: the token generation rate ρ and the pool size σ. The size of the token pool is the maximum number of tokens that can be accumulated. This will allow bursty traffic of up to σ cells to be transmitted. The size of the token pool determines the maximal burst length of cells that can enter the network. The token generating rate ρ is the mean rate that the node is allowed to transmit cells to the virtual channel of a network. A source is said to satisfy (ρ, σ) if, during any time interval μ, the number of cells that the source transmits is less than $(\sigma+\rho\mu)$.

If a token is generated when the pool is still full (i.e., very light traffic), the newly generated token is discarded. In an ATM environment, it is assumed that cells belonging to the same packet can be transmitted through the network independently. This means that the first cell can be transmitted immediately if a token is available without having to wait for other cells from the same packet to get their own tokens. A newly generated token can only be used starting from the next slot. Each cell is transmitted in a single time slot. The idea is to use tokens to enforce traffic descriptors provided during the connection request for the entire connection period.

The difference between Figure 2.2(a) and Figure 2.2(c) is that the second model has an input queue of size N for the incoming cells. If a cell entering the node finds an empty token pool, it then waits in the input queue for the token generator to generate available tokens. If the waiting time for the cell to receive the token exceeds the cell's maximum delay, the cell will be discarded. If a cell arrives and the input queue is full, it will be dropped immediately and considered lost.

One disadvantage of the leaky bucket mechanism is that it may mistake non-violating cells for violating cells. For example, when traffic is bursty, a large number of cells may be generated in a short time period, even when the long-term mean rate conforms to the traffic descriptors. To overcome such a disadvantage, a virtual leaky bucket was proposed [62, 63, 64]. In this mechanism, violating cells are tagged, rather than discarded, and sent to the network. These violating cells are discarded only when the network is congested.

Hluchyj [65] distinguishes bursty traffic from violating traffic by using a second-



(a) Leaky bucket

(b) Virtual leaky bucket

(c) Leaky bucket with input buffer

(d) Virtual leaky bucket with input buffer

(e) Leaky bucket with green/red tokens

(f) Second order leaky bucket

Figure 2.2: The leaky bucket algorithm and its variants.

order leaky bucket algorithm. The algorithm is consisted of two leaky buckets to police the sustainable cell rate and burst tolerance for each connection. Cells conforming to the first leaky bucket are allowed to enter the ATM network with the cell loss priority (CLP) field of the ATM cell header set to "0". Any cell non-conforming to the first leaky bucket is sent to a second leaky bucket, and will be tagged CLP=1 if found to be non-conforming to the second leaky bucket. Cells conforming to the second leaky bucket are allowed to enter the ATM network with CLP=0. Hence, a cell is tagged CLP=1 only if found to be non-conforming by the first and the second buckets.

We have studied the leaky bucket algorithm and its variants, and have presented our work in [35]. Our results show that cell loss rate increases rapidly as the maximum burst length increases. This happens because if many cells arrive at the same time some of them will not get tokens, and without available input buffers to save them, they will be dropped. For CBR traffic, if the token-generating rate is the same as the cell arrival rate, the cell loss rate is approximately zero. This is true for different mean burst lengths. In the case where the offered load approaches 1 (i.e., the network is bursty 100% of the time), the cell loss rate remains about steady. This is the characteristic of the leaky bucket algorithm.

Since the leaky bucket algorithm is a short-term congestion control scheme, it can smooth short-term congestion but cannot handle long bursty traffic. If the traffic has long burst length and the allocated bandwidth is not sufficient, some cells will be lost. Increasing the size of input buffers is unnecessary under light load conditions. Our study also shows that the increment and decrement of token rates have a greater impact on light load traffic than on high load traffic. The leaky bucket model is sufficient for providing deterministic guarantees. In our work, traffic measurements are done after traffic-shaping, since every ATM interface card has its own traffic-shaping scheme based on its buffer space.

2.3.5 Scheduling Policies

Two prerequisites for providing performance guarantees in a connection-oriented network are connection admission control and priority scheduling [26]. A scheduling policy is needed to maintain QoS for different classes of traffic when queuing occurs. Figure 2.3 shows the relation between the number of admitted calls and the scheduler in providing cell-level QoS. The cell-level QoS may be trivially guaranteed by any scheduling mechanism if a conservative admission control policy is used. The disadvantage of a conservative scheme is the cost of high call-blocking probability and low network link utilization. On the contrary, if too many calls are admitted to the network, no scheduling algorithm is able to satisfy the cell-level QoS for all traffic classes. In this case, call preemption may be neded [32]. This *schedulable region*,

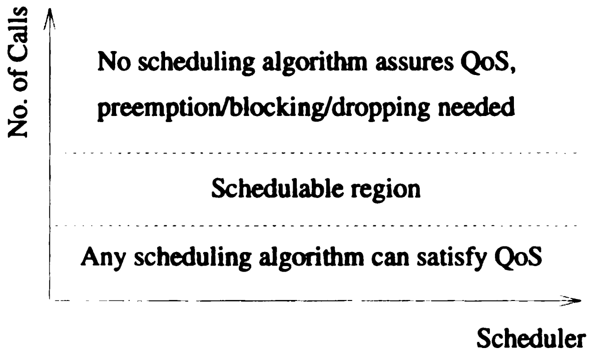

No. of Calls
No scheduling algorithm assures QoS,
preemption/blocking/dropping needed
Schedulable region
Any scheduling algorithm can satisfy QoS
Scheduler

Figure 2.3: The relation of the number of admitted calls and the scheduler.

denoted by S, has been defined as follows [13]: Let

$$S = \{x \in N^n | Scheduler\ guarantees\ the\ cell\ level\ QoS\ for\ all\ classes\},$$

where N is the set of natural numbers. The maximum number, N^i, of class i calls allowed to the system is defined from the limits of the schedulable region by

$$N^i = \max_{x \in S} X^i.$$

The region S represents the limits on the admission control policy imposed by the QoS at the cell level, which depend on the traffic characteristics and QoS constraints of each traffic class. Several important scheduling algorithms are discussed below.

The Head-Of-Line (HOL) mechanism [66] divides the cells according to different classes. Cells enter different input queues according to priority levels. They are serviced only when higher priority queues are empty. This scheduling algorithm is also used as static priority scheduling (SPS) [67] for the ACORN project in TeraNet [68]. This technique is class-dependent and might starve low priority cells indefinitely. In the case where high-priority cell delays are far from their allowed limits, performance can be improved by delaying high-priority cells within their QoS bounds. The MAGNET II Real-time Scheduling (MARS) algorithm [13] adaptively sets the parameters that govern a cycle, based on observation of cell arrivals/departures. Each cycle serves only those cells whose transmission cannot be further delayed and still conform to the QoS parameters.

The Push-Out mechanism [69] allows all cells to share a single buffer. When a high-priority cell arrives at a full queue, the lowest priority cell is forced to be dropped from that queue; otherwise it is lost. This scheme can fully utilize the buffer. Bursty traffic can take advantage of this mechanism whenever the load is low. When the load is high, this system favors high-priority traffic, and thus it may not satisfy the QoS for low-priority cells.

The Nested Threshold Cell Discarding (NTCD) mechanism [70] partitions a single buffer into different classes. Each class owns a portion of the buffer. A cell can enter the buffer if its priority is greater than or equal to that of the class under consideration. This scheme cannot fully utilize the buffer when a large number of low-priority cells arrive in a short time interval and only a few high-priority cells are in the buffer.

The Partial-Buffer-Sharing (PBS) mechanism [71] implements space priority on input buffers. Within a given class, traffic requiring a large buffer can share the buffer with traffic requiring a small buffer. The buffer management is relatively simple for such a scheme, but the delay can be very high.

Two versions of multiclass Earliest Due Date (EDD) schedulers are proposed for the Tenet scheme [72]: the Delay-EDD (or D-EDD) [22, 73] and the Jitter-EDD (or J-EDD) [73]. The former scheme eliminates faulty or malicious sources by postponing the misbehaving packets' deadlines. The latter scheme cures traffic fluctuation by restoring the original traffic pattern.

The Head-Of-Line with Priority-Jump (HOL/PJ) mechanism [66] enforces a certain level of fairness by placing a timer at each queue. Cells can jump to a higher priority queue when their timers expire. This mechanism is better than HOL, but the cost of implementing such a complex queuing system is excessive. The HOL/PJ mechanism involves cells jumping to a higher level buffer upon time-outs. For the case when cells in the lowest priority buffer are waiting for service and the highest priority buffer is highly loaded (i.e., the server is busy) while other queues are empty, the cells from the lowest priority buffer must go through unnecessary jumps in order to move to the highest priority buffer. Also, when several cells sense the time-out simultaneously, the complexity of handling the associated jumps increases.

In order to make use of a cell's priority bit and provide a certain level of fairness between different priority classes of traffic, we propose scheduling algorithms in [74]. These algorithms employ a simple queuing management scheme, while at the same time guaranteeing services to each class by means of a time-out mechanism.

The aforementioned scheduling algorithms are expected to cooperate with our CAC algorithm in order to give different priorities to different classes of traffic. Because of the high-speed switching associated with an ATM network, the scheduling algorithm must be simple. The proposed algorithm is a priority-scheduling mechanism that gives priority to delay-sensitive traffic and is feasible at the high speeds required for B-ISDN networks.

Chapter 3

Call Admission Control Framework

3.1 Introduction

In order to maximize the link bandwidth utilization while at the same time guaranteeing cell-level QoS for different classes of traffic, we propose a call admission control framework based on three important components: traffic measurements, call admission control, and cell scheduling. Traffic measurements, which lead to the traffic characteristics used by the call admission control framework, will be described in chapter 4. In this chapter, we focus on the call admission control algorithm, the cell-scheduling algorithm, and the relation between these two algorithms.

Bandwidth can be allocated to various connections at different levels (i.e., call level and cell level in our work). The task of admission control is to decide whether to accept or reject a new connection based on the traffic descriptors and QoS parameters provided at the call setup time. Traffic descriptors, at minimum, must include the peak rate [75]. As we have mentioned, the QoS parameters are the set of parameters that a network must satisfy when connections are made.

In order to achieve different QoS requirements for diverse traffic classes and to minimize the parameters used for negotiating a connection, we propose integrating a connection admission control algorithm and a scheduling algorithm by using a time-out mechanism. The task of the scheduler is to resolve contention between cells of different classes at a switching node and to carry out bandwidth allocated from a call controller. Higher utilization of the total link capacity is achieved if bandwidth is allocated on a cell level. Cell level QoS requirements for all classes of traffic must be satisfied as specified in the tuple (*traffic descriptors*, *QoS parameters*) at the connection setup time.

A time-out value is computed based on the equivalent bandwidth allocated by the call admission, which in turn is used by the scheduling algorithm. Such a time-out value serves as an essential parameter for combining the call admission and scheduling algorithms, which is necessary in order to enforce different cell-level QoS for various classes of ATM traffic.

We believe that these control functions are significant and challenging. Because control must be exerted at the level of each virtual connection and individual cells, the processing involved is very intensive. The high speeds at which future networks will operate impose real-time constraints on scheduling decisions that suggest the use of a simple algorithm.

3.2 Quality of Service (QoS) Requirements

Services can be either connection-oriented (CO) or connectionless (CL) in B-ISDN's integrated ATM network. Connection-oriented services include constant bit-rate (CBR) and variable bit-rate (VBR) services. Connectionless services include *best-effort* or available bit-rate (ABR) services. Table 3.1 shows typical QoS requirements for diverse types of multimedia applications [76, 77, 78]. All parameters, traffic descriptors and QoS parameters, must be simple enough to be understood and calculable by users, useful to CAC, and enforceable by the network.

3.2.1 Specified and Unspecified QoS Class

According to service specifications, there are two QoS classes, namely *specified QoS class* and *unspecified QoS class* [10]. *Specified QoS class* has been standardized [10] and is expected to support services for constant bit rate video, variable bit rate audio and video, connection-oriented data transfer, and connectionless data transfer. Constant bit rate (CBR) service is considered as an alternative to leased line services. The burstiness of this type of traffic in an ATM network is minimized and the cell loss rate is strictly avoided.

Variable bit rate video service will probably generate the largest revenue for future ATM networks. The burstiness of this application comes from video compression techniques. Due to the high data rate, traffic shaping requires a large buffer at a customer premise equipment. Because of the periodic nature of this traffic pattern, limited control of QoS with reasonable cost through CAC can be achieved.

Applications	Connection Mode	Service Type	Symmetry	QoS	Bandwidth Ranges
Video telephony	CO	CBR	bidirectional	Call blocking permitted Low-med cell loss OK Isochronous	2.4–32 Kbps
Digital audio	CO	CBR	bidirectional	Call blocking permitted Low cell loss required Low delay jitter	128–512 Kbps
Teleconference multimedia communication	CO	CBR/VBR	bidirectional	Statistical multiplex (VBR) Call blocking permitted Low-med cell loss OK Low delay jitter	64–384 Kbps
Digital video	CO	CBR/VBR	unidirectional	Statistical multiplex (VBR) Call blocking permitted Low-med cell loss OK Low delay jitter	1–6 Mbps
Digital HDTV	CO	CBR/VBR	unidirectional	Call blocking permitted Low-med cell loss OK Low delay jitter	15–20 Mbps
TV distribution	CO	CBR/VBR	unidirectional	Low-med cell loss OK Low delay jitter	≤10 Mbps
Computer data	CL	Best effort ABR	bidirectional	NonCall blocking Low cell loss required Low-med delay OK	0.1–1 Mbps
E-mail	CL	Best effort ABR	unidirectional	NonCall blocking Low cell loss OK High delay OK	9.6–128 Kbps
High-speed data (multimedia)	CL	VBR ABR	bidirectional	High transfer rate Very low cell loss required Med delay OK	1–10 Mbps
Supercomputer connection	CO/CL	Bursty	bidirectional	High transfer rate Very low cell loss required low delay jitter	≤100 Mbps
LAN internetting	CO/CL	Bursty	bidirectional	Call blocking permitted Low cell loss required	10–30 Mbps
Image transfer	CO/CL	Bursty	unidirectional	Call blocking permitted low-med cell loss OK low delay jitter	10–30 Mbps

Table 3.1: Typical QoS requirements for broadband and multimedia applications.

For the services that support connection-oriented data transfer, short end-to-end data delay is an advantage over connectionless data service. For automatic generated data transfer such as daily database exchange, QoS parameters can easily be defined and cell loss rate can be guaranteed. Services not included in *specified QoS class* fall into *unspecified QoS class. Best-effort* service [10] is considered a cost-effective approach for ultra-bursty traffic in this class. Traffic requesting *best-effort* service consumes only memory that is used for the connection description table at an ATM switch. The cell loss of this class is not guaranteed.

3.2.2 Available Bit Rate (ABR) QoS Class

In this section, we identify application traffic that can virtually form an unique service class. This traffic belongs to the Available Bit Rate (ABR) service class. The ATM standards committee has recognized the fact that data traffic often requires no guarantee of bandwidth, and can be sent at whatever bandwidth is available for the network. ABR traffic gives the network the opportunity to offer guaranteed traffic, and makes the most use of the remaining capacities.

The bandwidth available in ATM networks makes sophisticated graphic interface applications feasible (e.g., VLSI design tool). Users can execute applications from a remote site. Such interactive application requires fast response between the server and the client. Unlike video services, cell loss is undesirable for this type of application. Accordingly, the mean rate of interactive traffic, submitted under the traffic contract agreement, is meaningless. As shown in Figure 3.1, traffic transferred by the same application with different durations results in different mean rates. Interactive application traffic is extremely bursty, requiring high reliability and transmission speed. It may be silent for a long time followed by a bursty period. Such a traffic pattern is unpredictable. In most cases, even the user cannot predict the traffic load.

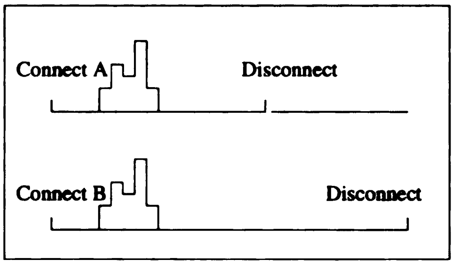


Figure 3.1: Traffic resulting in different mean rates due to different connection periods.

Two flow control proposals for providing ABR services in ATM networks have been studied: the *rate-based* and the *credit-based.* Both proposals aim at achieving high network utilization by allowing ABR traffic to utilize the remaining bandwidth after the guaranteed traffic, Constant Bit Rate and Variable Bit Rate, has been served.

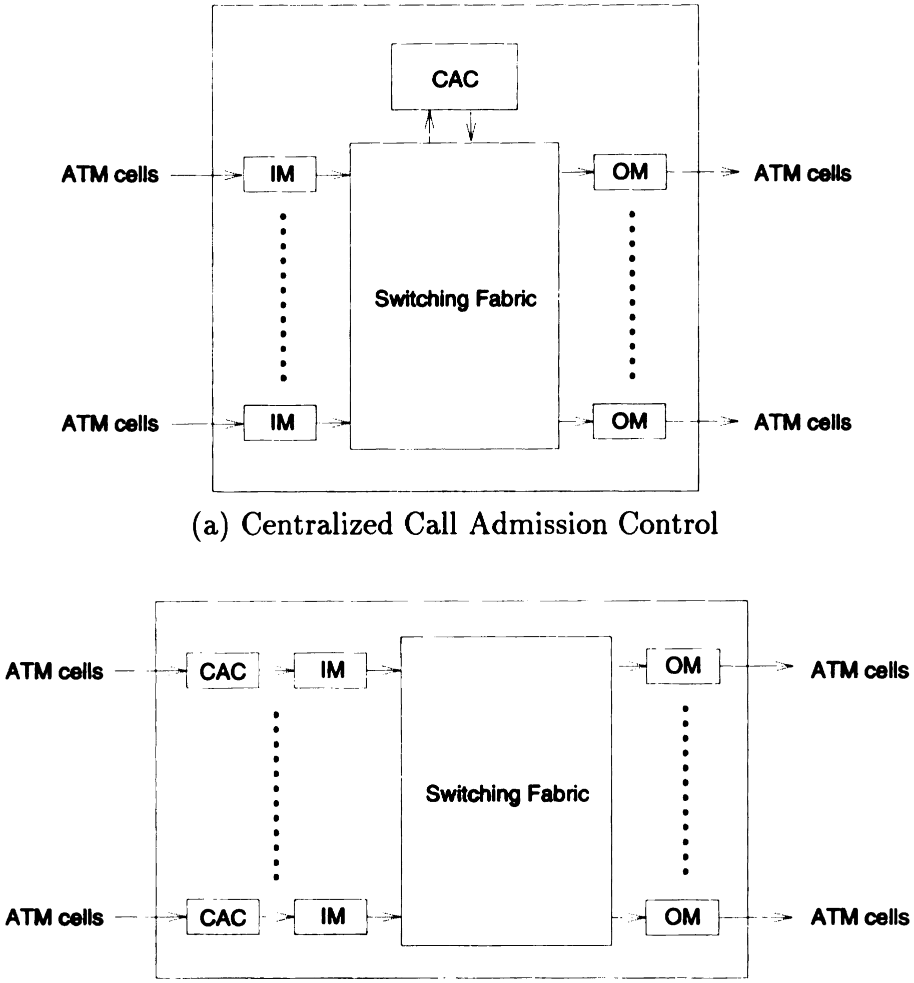
The first approach is a hop-by-hop credit-based flow control scheme. Kung [79, 80, 81] proposed the N23 scheme, which works as follows. For each link in the path of a virtual circuit (VC), the sending host needs to receive credit cells sent by the receiving host before it can forward any cell to the VC. The receiver sends credit cells to the sender each time after forwarding $N2$ cells to the VC. After receiving the credit cells, the sender updates its credit information. The sender decrements its credit balance by one each time it sends a data cell. This credit-based scheme is designed in such a way as not to overflow the buffer corresponding to this VC at the receiver, which has a buffer capacity of $(N2 + N3)$ cells. The value of $N2$ can be set statically or adaptively and is chosen to be small enough to limit cells. $N3$ is chosen to be large enough to allow the VC to sustain its peak rate. This credit-based flow control has been implemented in an experimental ATM switch by BNR and Harvard University.

The second approach is an end-to-end rate-based flow control scheme. Yin and Hluchyj [82] considers a Forward Explicit Congestion Notification (FECN) mechanism. Whenever a resource along the path of a VC is congested, FECN is sent to the destination. A Resource Management (RM) cell is then periodically sent by the destination node to the source node while congestion remains. RM cells may also be generated by intermediate nodes. Upon receiving the RM cells, the source regulates the flow of its traffic entering the ATM network.

ABR services in our call admission control are used to maximize the network link bandwidth utilization. Our algorithm achieves different QoS requirements for various classes of traffic and minimizes the parameters used for negotiating a connection. The call admission algorithm allocates sufficient bandwidth to different classes of traffic. It admits a call if the current network bandwidth can satisfy the ordered pair (*traffic descriptors*, *QoS parameters*) provided by a call request.

3.3 Functional Architecture Model

In this section, we discuss the architectural view of the call admission control (CAC). The flow of user data, that is, ATM cells, through three functional blocks is shown in Figure 3.2. First, arriving cells are delineated from the fiber links and put in the input modules (IMs). Switching fabric routes the cells from the input modules to the destinated output modules (OMs) according to the VPI/VCI fields in the cell header. Finally, the cells are sent to the physical links by output modules. The CAC performs connection admission or rejection for each connection request before the other three functional modules (i.e., input modules, switching fabric, output modules) can perform their functionalities. The partitioning of these functions does not have defined boundaries between functional blocks.



(a) Centralized Call Admission Control

(b) Distributed Call Admission Control

Figure 3.2: Different switching architectures for call admission control, where IM is input module and OM is output module.

The call-processing functions (admission decision and bandwidth allocation) for the entire switch can be centralized in a single CAC unit, as shown in Figure 3.2(a), or distributed to different input ports, as shown in Figure 3.2(b). The centralized approach is simple to implement; however, it becomes a bottleneck for large-size switches upon handling all the call admission control functions for all input ports. In the decentralized approach, each CAC block processes connection requests for each input port. Hence large switches can be constructed without the bottleneck caused by centralized CAC, but this approach is more complex. Each distributed CAC functional block also needs global state information (e.g., signaling and output port states) in order to coordinate.

Our work on call admission control does not distinguish between the centralized approach and the distributed approach. The proposed call admission control algorithm can be applied to both architectures. A B-ISDN connection request may involve one or more virtual channel connections (e.g., for multimedia traffic or multi-party connections). If multiple connections are involved, connection admission is determined for each virtual channel connection.

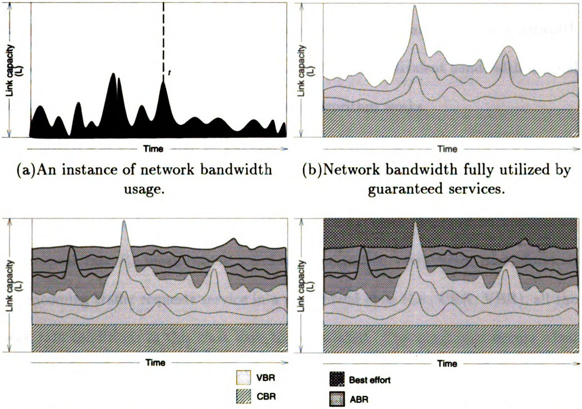
3.4 Call Admission Control Algorithm

The call admission algorithm allocates sufficient bandwidth to different classes of traffic. It admits a call if the current network bandwidth can satisfy the ordered pair (*traffic descriptors*, *QoS class*) provided by a call request. Equivalent bandwidth (EB), denoting the amount of bandwidth assigned to that service class, is then calculated based on this tuple.

Parameters associated with our CAC algorithms are traffic descriptors (peak rate, mean rate) and QoS classes. Figure 3.3(d) shows a graphical view of our CAC algorithm that maximizes the link bandwidth utilization. The proposed algorithm provides QoS for the following classes: constant bit rate, variable bit rate, available bit rate, and best-effort. Services provided for constant bit rate and variable bit rate are guaranteed services, and this means that the network bandwidth can support an amount of bandwidth equal to a ratio of the peak rate for all connections, even if such connections generate their peak rates at the same time (refer to Figure 3.3(b)). The available network bandwidth is greater than or equal to the peak rate of such traffic during the life of the connection. ABR and best-effort services can only use the remaining bandwidth of the guaranteed services.

Available bit rate and best-effort services intend to provide different QoS to data traffic and make the best use of the network bandwidth. To support ABR traffic, the network link bandwidth must be greater than the mean rate of such traffic (refer to Figure 3.3(c)). Best-effort services can fully utilize the rest of the network bandwidth. Figure 3.3 depicts the above ideas. We illustrate how our algorithm works in the rest of this section.

The algorithm under consideration maintains a list of traffic descriptors associated with each connection. It then computes *equivalent bandwidth* for the connection from traffic descriptors based on services requested. The equivalent bandwidth for a connection represents the maximum bandwidth allocated to this connection. Hence the remaining capacity of the network for each physical link can be calculated. The CAC algorithm accepts a connection based on the remaining capacity and the equiv-



(a)An instance of network bandwidth usage.

(b)Network bandwidth fully utilized by guaranteed services.

(c)Network bandwidth fully utilized by guaranteed services and available bit rate services.

(d)Full bandwidth usage by different QoS classes.

Figure 3.3: Connection admission control based on QoS (CBR, VBR, ABR, $BestEffort$) and available bandwidth.

alent bandwidth requested by the connection. Usually, the equivalent bandwidth of a connection falls between the peak rate and the mean rate. The equivalent bandwidth of a CBR connection is equal to its $\theta_\alpha P_\alpha$, where $0 \leq \theta_\alpha \leq 1$, and P_α is its peak rate. This is also true for a VBR connection. The actual value of θ_α is application dependent. It is adjustable in our algorithm. The smaller the value of θ_α, the higher the network bandwidth efficiency. In the case where $\theta_\alpha = 1$, an amount of bandwidth equal to its peak rate is allocated. For ABR traffic, the equivalent bandwidth is equal to the mean rate. We now illustrate how our algorithm works.

Consider a new traffic flow α (P_α, M_α, $class_\alpha$) requesting a connection to a network where n connections have been established. The set of parameters (P_α, M_α, $class_\alpha$) represents a connection requesting a QoS service for class $class_\alpha$ of an application with a peak rate P_α, mean rate M_α, and $class_\alpha \in$ (CBR, VBR, ABR, BestEffort). If the incoming traffic requests service in a guaranteed class (i.e., CBR, VBR), all the peak rate conditions in this class have to be satisfied. The worst case occurs when all traffic flows in the guaranteed class generate peak rates at the same time (refer to Figure 3.3(b)). When there is a connection request, the call admission control algorithm performs the following test:

1. Determines the equivalent bandwidth for flow α:

$$EB_\alpha = F\ (P_\alpha, M_\alpha, class_\alpha) \tag{3.1}$$

$$F(P_\alpha, M_\alpha, class_\alpha) = \begin{cases} \theta_\alpha P_\alpha & \text{if } class_\alpha \in CBR \\ \theta_\alpha P_\alpha & \text{if } class_\alpha \in VBR \\ M_\alpha & \text{if } class_\alpha \in ABR \\ 0 & \text{if } class_\alpha \in BestEffort, \end{cases} \tag{3.2}$$

where F is the equivalent bandwidth evaluator.

2. Determines whether there is enough link capacity, after adding the new flow α, assuming that all flow i, $i \in VBR$, will generate peak rate at the same time.

$$L > \begin{cases} \sum_{i \in CBR} EB_i + \sum_{i \in VBR} EB_i & \text{if } class_\alpha \in CBR \\ \sum_{i \in CBR} EB_i + \sum_{i \in VBR} EB_i & \text{if } class_\alpha \in VBR \\ \min(\hat{R}, \sum_{i \in CBR} EB_i + \sum_{i \in VBR} EB_i) + M_\alpha & \text{if } class_\alpha \in ABR \\ \hat{R} & \text{if } class_\alpha \in BestEffort \end{cases} \tag{3.3}$$

3. Accepts the connection if the condition is true; otherwise, rejects it.

From data obtained via experience, the measured link bandwidth usage ($\hat{R}$) is always less than (e.g., when the traffic level equals its mean rate) or equal to (e.g., when the traffic level equals its peak rate) the bandwidth assigned for CBR and VBR traffic (refer to Figure 3.3(a)). In order to fully utilize the network bandwidth, call connection requests are accepted for ABR traffic based on the current bandwidth utilization ($\hat{R}$). The measurement of $\hat{R}$ is presented in the next section. The value of $\hat{R}$ affects the acceptance of calls requesting ABR service, which also depends on the time instance that the measurement is taken. Consider a new flow α, requesting a connection to the network at time t in Figure 3.3(b), where $\hat{R}$ has a large value. This reduces the probability that the flow α can be accepted in order to allow maximum admission of calls. We consider the fact that traffic will pass its peak rate at point t_1, ($t_1 > t$), when the connection is established.

Finally, if the incoming traffic requests *best-effort* services, a connection is established, since the best-effort class intends to make use of the remaining bandwidth. Equivalent bandwidth is 0 for this service. Note that in our algorithm, a call requesting best-effort services is always accepted at the call level. Cells belong to this class will be dropped at the cell level if the amount of bandwidth is not enough. In order to have greater insight into diverse QoS requirements for different types of applications, we have conducted experiments that collect traffic statistics from an ATM testbed.

The above mechanism is a resource over-allocation scheme (e.g., when $\theta_\alpha < 1$) to achieve high bandwidth utilization. To better utilize the bandwidth, ABR and best-effort traffic are admitted if there is bandwidth available, regardless of the fact that total bandwidth might have been allocated to guaranteed services. In the case where guaranteed traffic needs to compete with the bandwidth allocated to ABR or best-effort traffic, guaranteed services have higher priority. A discussion which focuses on the performance of our CAC scheme is presented in chapter 6.

3.5 Measuring Network Link Utilization

As we have stated, the objective of the call admission control is to allow as many calls as possible. An important task is to monitor the current link usage in order to maximize bandwidth utilization. Our call admission control algorithm uses the Simple Network Management Protocol (SNMP) [83] to measure the dynamic network loads. The performance matrix of a CAC algorithm will be call-blocking probability and network link bandwidth utilization.

SNMP is becoming the most popular and important standard protocol for network management [34]. Figure 3.4 shows the SNMP architecture. This architecture consists of an SNMP manager, an SNMP agent, and the Management Information Base (MIB) [84, 85, 86, 87, 88]. The MIB is a collection of objects defined to contain a lot of manageable network information. SNMP can issue a *GetRequest* to access the network information by receiving a *Response* from the SNMP agent.

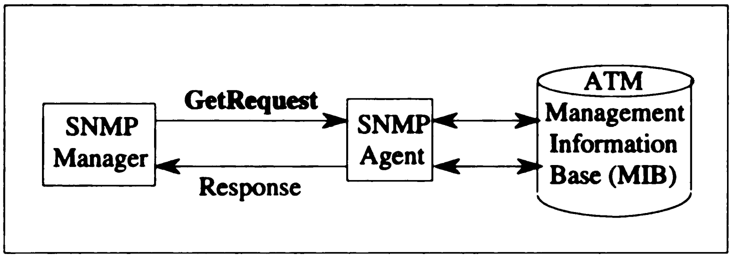


Figure 3.4: The SNMP architecture.

Currently, a lot of management groups have been defined in MIB–II, a de facto network management standard. Figure 3.5 shows the information in MIB–II [88], where information is related to our traffic measurements. The group of MIB–II is defined as a mandatory object in the SNMP architecture [85]. This implies that we can gather the appropriate information if an SNMP daemon is running in a network.

The follows show how to compute network link speed, L, and current bandwidth utilization, $\hat{R}$ from SNMP:

1. Determine the current link speed, L:

$$L = snmpget \;\; 1.3.6.1.2.1.2.2.1.ifSpeed$$

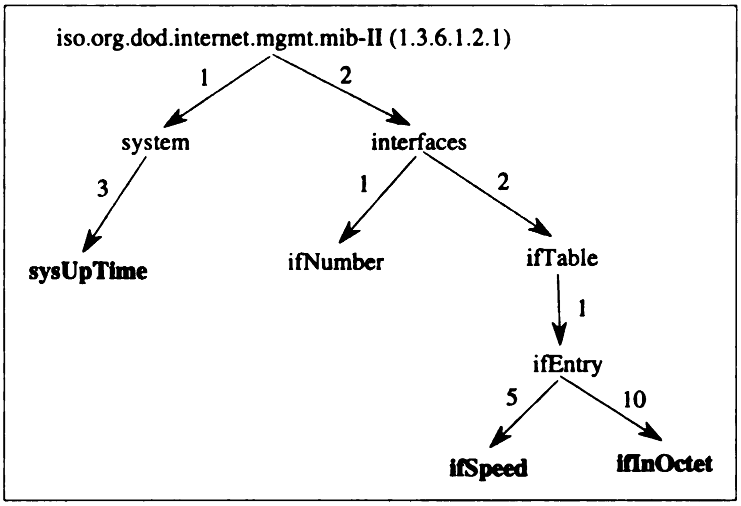


Figure 3.5: MIB–II information related to network link speed L and current bandwidth utilization $\hat{R}$.

2. Determine the current bandwidth utilization:

$$\begin{aligned} T_1 &= \textit{snmpget}\ 1.3.6.1.2.1.1.3.sysUpTime \\ B_1 &= \textit{snmpget}\ 1.3.6.1.2.1.2.2.1.ifInOctet \\ T_2 &= \textit{snmpget}\ 1.3.6.1.2.1.1.3.sysUpTime \\ B_2 &= \textit{snmpget}\ 1.3.6.1.2.1.2.2.1.ifInOctet \\ \hat{R} &= \frac{B_2 - B_1}{T_2 - T_1}, \end{aligned}$$

where $T - 2 > T_1$, and L and $\hat{R}$ are in units of $Mbps$.

The computed bandwidth utilization uses two tunable constants, T_1 and T_2. The time between T_1 and T_2, ΔT, is the measured period in which the number of cells occuring in an output link is recorded, and this in turn, determines the output link utilization. The measurement period ΔT controls how conservative the measurements of link utilization are; increasing ΔT will mean that more history is retained, thus making the admission control more conservative.

Immediately after a new connection is set up, the measured utilization does not represent the current network load because the measurements were done before adding the new flow. Hence, the measurements have to be performed each time a new connection request occurs.

This algorithm is connection-oriented and reservation-based. Before a channel can be used by its caller, it must be established. Channel establishment is a distributed process. A call setup request is first issued by the source node, and it then passes each node along the path. This request message causes admission tests to be conducted at each node. If the new channel passes the tests in a node, the message is then forwarded to the next node. The final tests are performed by the destination node. If all tests at each node are passed, a call-accepted message is sent by the destination node to the source node along the reverse route. If any node in the path fails the admission test, the channel cannot be established and a cell-rejected message is sent back to the source node.

3.6 Class-Based Scheduling Algorithm

The switch hardware needs to ensure that at no time will the quality of services of one class of traffic be affected by other classes of traffic. The simplest approach to ensure this is to separate the cell buffering in the ATM switch into different traffic classes, as they are at the call level. These different buffers can be implemented in separate FIFO queues.

The proposed scheduling algorithm is the Round-Robin with Time-out (RR/T) system[89], which ensures that all queues receive a service time equal to the time-out value during overload. Cells are placed at the end of selected input buffers based on their associated classes, and the priority level is defined according to the traffic classes. Guaranteed classes of cells have higher priority than non-guaranteed classes of cells.

A time-out mechanism is employed to carry out the bandwidth allocated from the call admission. The delay time slot, D_i, is defined as the interval from the time the cell enters the queue to the time the transmission begins. This value is used for subsequent simulation and analysis. Section 6.5 shows that the delay of real-time traffic regulated by RR/T is bounded.

Figure 3.6 shows how the scheduling algorithm provides bandwidth requirements for different QoS classes. In such a model, an ATM switching node consisting of four separate buffers provides different QoS requirements to four input classes (i.e., CBR, VBR, ABR, BE) of traffic from the call admission control. The scheduling model corresponds to an output module in Figure 3.2, or to the scheduler in Figure 1.1.

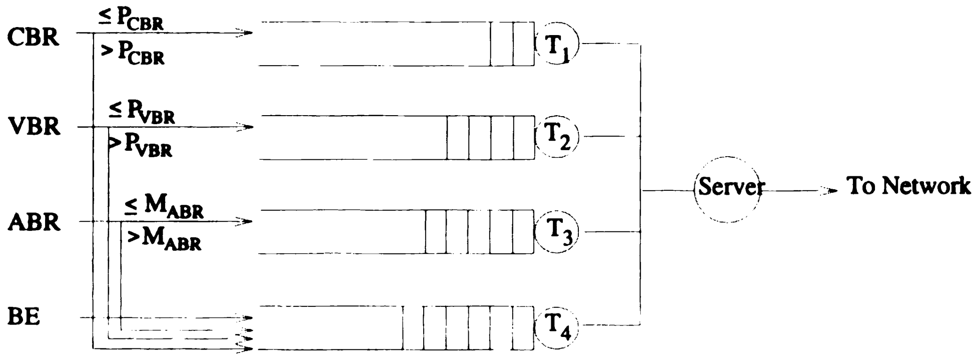


Figure 3.6: Scheduling model for different QoS classes.

If a cell within a given class arrives at an ATM node and its corresponding queue is full, the cell is placed in the best-effort queue. If a cell comes to a full best-effort queue, the cell is dropped. Cells in higher priority queues (i.e., CBR and VBR) are processed first, whereas those in the lower priority buffers (i.e., ABR and BE) receive service only when higher priority queues encounter a time-out. Note that figure 3.6 depicts a scheduling model whose input queues can accept only arrival cells less than or equal to its equivalent bandwidth. The scheduler uses a time-out mechanism to enforce bandwidth allocated to different service classes by the call admission algorithm. A time-out value, T_i, is associated with each queue. Each queue, q_i, can receive service in a round-robin fashion for a maximum service time up to T_i. In other words, a queue, q_i, can receive service until the queue is empty or the timer, T_i, expires, whichever occurs first. In the case where the server serves each queue until the timer expires, this is also called *weighted round robin* (WRR). If the queue is empty before the timer expires, the actual service time is $\tilde{T}_i$. The remaining extra time $(T_i - \tilde{T}_i)$ from the guaranteed classes (i.e., CBR and VBR) is used to service other classes of applications (i.e., ABR and BE). Refer to Figure 3.8 for an algorithmic presentation of the scheduler for our CAC scheme.

The CAC algorithm, as described in this chapter, determines the admission or rejection of a new call, α, and computes the equivalent bandwidth, EB_α, for the call based on equations (3.1) - (3.3). The value of EB_α can be used to calculate the time-out value T_i for the queue. Assume T_L is a cycle time according to the link speed L in the unit of one cell time, then

$$\begin{aligned}
T_1 &= (\frac{EB_{CBR}}{L})T_L && (3.4)\\
T_2 &= (\frac{EB_{VBR}}{L})T_L && (3.5)\\
T_3 &= \min\{(T_L - \tilde{T}_1 - \tilde{T}_2), (\frac{EB_{ABR}}{L})T_L\} && (3.6)\\
T_4 &= (T_L - \tilde{T}_1 - \tilde{T}_2 - \tilde{T}_3) && (3.7)
\end{aligned}$$

The value of the time-out, T_i, where $i = 1$ to 4, is computed according to the above equations. $\tilde{T}_i$ is the actual time used to process the queue, q_i, and $\max(\tilde{T}_i) = T_i$, where T_i is an indicator of the amount of bandwidth allocated to the particular queue. The values of T_1 and T_2 have to be recomputed when there is a new connection for CBR and VBR traffic, respectively, whereas T_3 and T_4 are calculated each time all T_i expire. Figure 3.7 shows the pictorial view of the relations between T_i and $\tilde{T}_i$.

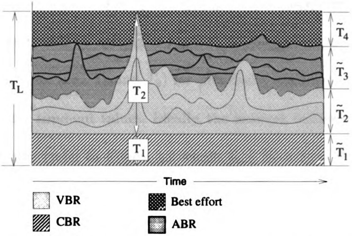


Figure 3.7: Pictorial view of T_i and $\tilde{T}_i$. Note that $T_L = \sum_{i=1..4} T_i = \sum_{i=1..4} \tilde{T}_i$.

From equation (3.6), if the scheduler serves T_1 and T_2 until the timer expires, (i.e., $\tilde{T}_1 = T_1$ and $\tilde{T}_2 = T_2$), then

$$T_3 = \min\{(T_L - T_1 - T_2), (\frac{EB_{ABR}}{L})T_L\} \qquad (3.8)$$

In the above equation, if $T_3 = (T_L - T_1 - T_2)$, then $(T_1 + T_2 + T_3) = T_L$. Therefore $T_4 = 0$, which is the case when CBR, VBR, and ABR traffic consume all of the bandwidth, leaving none for the best-effort traffic. On the other hand, if $T_3 = (\frac{EB_{ABR}}{L}T_L) < (T_L - T_1 - T_2)$, then $T_4 = (T_L - T_1 - T_2 - T_3) = ((T_L - T_1 - T_2) - (\frac{EB_{ABR}}{L})T_L) > 0$. This occurs when the best-effort traffic has some available bandwidth.

The service mechanism for the aforementioned scheduling scheme employs a First-Come-First-Serve discipline within a given class. For the guaranteed classes of connections, the time-outs (i.e., T_1 and T_2) are calculated based on the equivalent bandwidth, which is the peak rate allocated by the CAC algorithm. Therefore, T_3 and T_4 can use the remaining bandwidth by setting it equal to the remainder of the available time slots. The actual time (i.e., $\tilde{T}_1$ and $\tilde{T}_2$) for processing queues q_1 and q_2 will be less than or equal to the time-out values. Ideally, the network bandwidth is fully utilized by guaranteed traffic (refer to Figure 3.7 where $T_1 + T_2 = T_L$). When this happens, $T_3 = T_4 = 0$ and cells entering the VBR and best-effort queues are buffered until the next service cycle, when these two queues can usually receive service.

Figure 3.8 shows how the scheduling algorithm works. The scheduling algorithm is similar to the weighted round-robin (WRR), in that the server serves each queue in a round-robin fashion with different amounts of service time (i.e., the time-out values). It is different from the WRR in that the time-out value for each queue is dynamically computed in each service cycle according to the bandwidth allocated to each class. The implementation of this scheme can be done either in hardware or in software. As an example, the queuing mechanism can be implemented as a circular buffer.

```
repeat{

  $T_1 = (\frac{EB_{CBR}}{L})T_L$;
  $T_2 = (\frac{EB_{VBR}}{L})T_L$;
  while $NoTimeOutOccurs$ from $T_1$ and $NotEmpty(q_1)$
      process cells in $q_1$;
  $\tilde{T}_1$ is the actual time for processing $q_1$;
  while $NoTimeOutOccurs$ from $T_2$ and $NotEmpty(q_2)$
      process cells in $q_2$;
  $\tilde{T}_2$ is the actual time for processing $q_2$;
  $T_3 = \min\{(T_L - \tilde{T}_1 - \tilde{T}_2), (\frac{EB_{ABR}}{L})T_L\}$ ;
  while $NoTimeOutOccurs$ from $T_3$ and $NotEmpty(q_3)$
      process cells in $q_3$;
  $\tilde{T}_3$ is the actual time for processing $q_3$;
  $T_4 = (T_L - \tilde{T}_1 - \tilde{T}_2 - \tilde{T}_2)$ ;
  while $NoTimeOutOccurs$ from $T_4$ and $NotEmpty(q_4)$
      process cells in $q_4$;
}
```

Figure 3.8: Scheduling algorithm for the call admission control.

3.7 Summary

In this chapter we have presented a call admission control framework that consists of three components: traffic measurements (will be addressed in the next chapter), call-level control (call admission algorithm), and cell-level control (cell scheduling algorithm).

Various QoS requirements for ATM applications have been addressed. The call admission control algorithm accepts a call based on various QoS classes and provides guaranteed services to CBR and VBR traffic. It also maximizes output link bandwidth utilization by dynamically measuring current network load, that is, the amount of bandwidth consumed, by using SNMP and allocates available bandwidth to ABR traffic. Such a call admission control algorithm can be implemented in an ATM switch to operate in a centralized control or distributed control manner.

A class-based cell-scheduling algorithm is also presented to resolve contentions between different connections. This scheduling algorithm is similar to the weighted round-robin (WRR), in that the server serves each queue in a round-robin fashion with different amounts of service time (i.e., the time-out value). It is different from the WRR in that the time-out value for each queue is dynamically computed in each service cycle according to the bandwidth allocated to each class. Such a scheduling scheme is an approach for integrating call-level and cell-level control to satisfy QoS requirements. This adjustable time-out value is used to carry out bandwidth allocated for different classes of traffic. In chapter 6, we will discuss the performance of this framework, based on both call level and cell level.

Chapter 4

Traffic Measurements and Data Collection

4.1 Introduction

The characterization of ATM traffic is very important in many aspects, such as validating traffic models, designing congestion control algorithms, implementing communication protocols, and developing switch architectures. While traffic models vary in many fundamental aspects, they rely on different assumptions for determining essential parameters. Traffic models generally require users to provide traffic descriptors to characterize their applications. Due to the limited availability of real traffic data gathered from existing ATM networks, researchers have not yet obtained good credible estimates of these descriptors.

In order to study the traffic characteristics of ATM networks effectively, we have conducted experiments to obtain measurements that facilitate the identification of traffic descriptors. We present measurements of applications obtained from the High-Speed Networking and Performance Lab (HSNP) at Michigan State University in this chapter. We investigate the traffic characteristics of several applications in an ATM network with the goal of identifying traffic descriptors and formulating a traffic model designed to generate application traffic for network simulations.

These applications are selected based on studies of Wide-Area TCP/IP traffic measurements conducted by Danzig et al. [48, 49, 50]. Experimental results designed to investigate traffic characteristics from a large amount of data collected are discussed. Traffic descriptors such as *peak rate* and *mean rate* are calculated from the gathered data. These descriptors are used to define parameters for an ATM LAN/WAN workload model. In this dissertation, we concentrate on cell-level information collected from our ATM testbed.

4.2 Measurement Methodology

The HSNP lab supports a testbed that has two independent local area networks, one of which is the traditional Ethernet network, and the other of which is an ATM network. This ATM testbed includes three Fore ASX-100 ATM switches [90], twelve SPARCstation-10's equipped with SBA-200 interface cards [91], and fiber optic links. Figure 4.1 shows both the physical and the logical view of the HSNP network configuration.

Data analysis would greatly benefit from the availability of a high-resolution clock, which can be used to time-stamp each arriving cell. Since the resolution of the

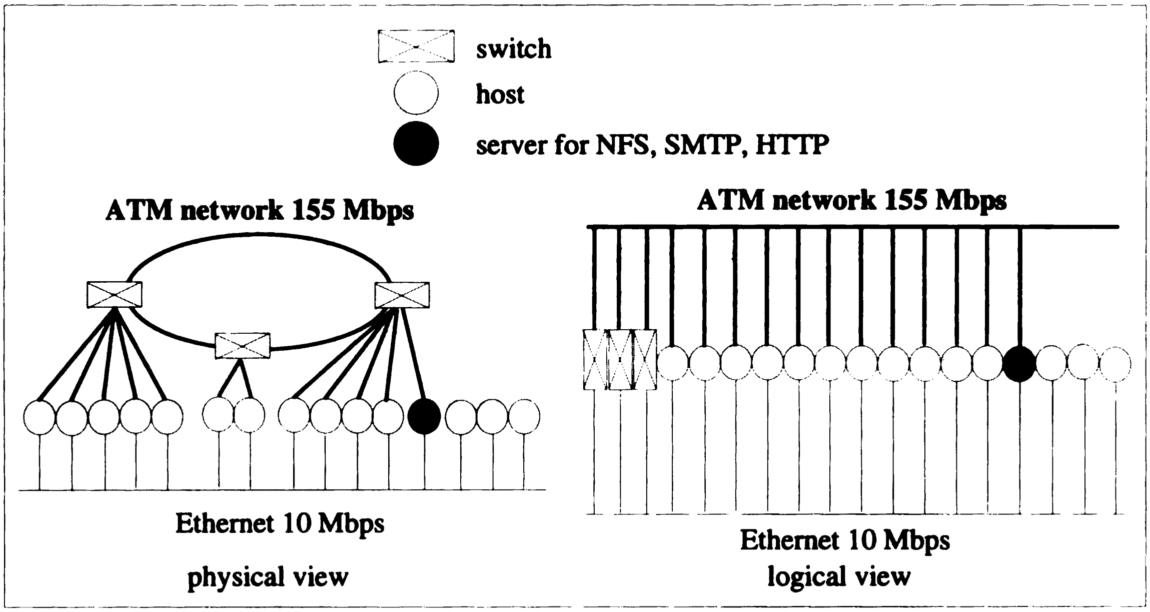


Figure 4.1: Network configuration of HSNP Lab.

machine clock was only 40 μsec, we cannot gather information based on burst-level. That is, in order to gather burst level information, the clock rate has to be less than a cell time, 2.87 μsec. In order to address the fact that machine clock resolution is greater than the cell interarrival time, we collect data at fixed time intervals.

The data was gathered from different ATM layers using a monitor program. Figure 4.2 shows the protocol stack hierarchies. This program runs as a time-driven process and collects traffic statistics at the cell level every Δt time interval, where $\Delta t = t_i - t_j, i = j + 1$, $(0 \leq i, j \leq n)$, and $(t_0, t_1, t_2, ..., t_n)$ is a sequence of statistics gathered time stamps. The Solaris C library function **nanosleep()**, which has an accuracy within nanoseconds, is used. Since there are two connected networks in the HSNP lab, sample data were measured through the ATM side of the network.

Cáceres et al. [48, 49] report that five applications comprise the majority of the

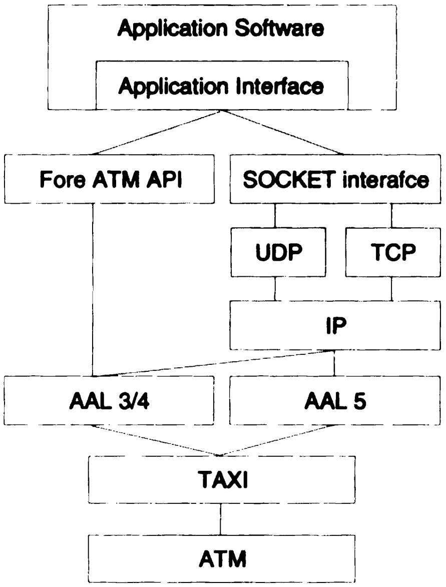


Figure 4.2: Protocol hierarchies.

WAN traffic measured (refer to Table 2.1[1]). Our study, however, only considers three applications[2] (**FTP**, **Rlogin**, and **Telnet**) are considered for generating data traffic. In addition, two other applications (**audio** and **SunVideo**) are also investigated. The application program **audio** is used to transmit voice traffic and the application software **SunVideo** is applied to capture and display video images.

We have conducted the measurements on our local ATM testbed for individual applications, in order to keep the measurement effort manageable. Moreover, the arrival process is more evident in such an environment, where there are fewer queuing interactions and there is less traffic smoothing due to the superposition of different types of application traffic. The next section provides descriptions of the different types of traffic statistics obtained from our ATM testbed.

[1] The applications that appear in boldface are studied in this paper. SNMP and NNTP will be considered in our future study.

[2] This restricted set was chosen due to our local ATM configuration.

4.3 Traffic Characteristics

4.3.1 FTP

The first set of experiments investigates the characteristics of FTP (the Internet standard file transfer protocol) [92]. There may be several file transfers[3] in a single FTP session[4]. Different sized files are transferred from host_1 to host_2 via two intermediate switches connected via fiber-optic links operating at 155 *Mbps* (refer to Figure 4.3).

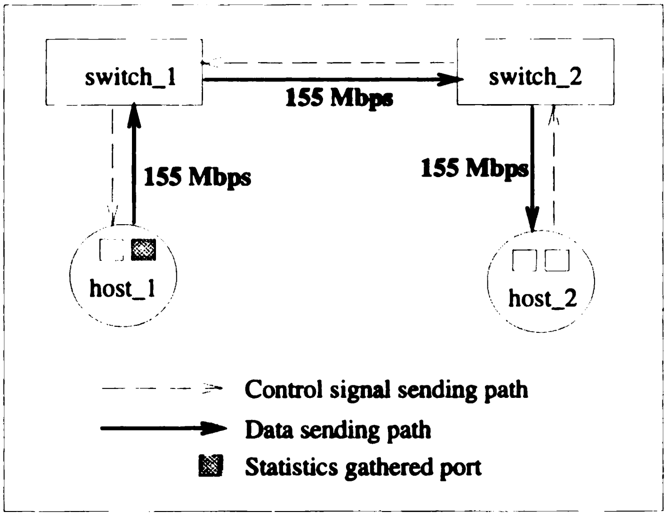
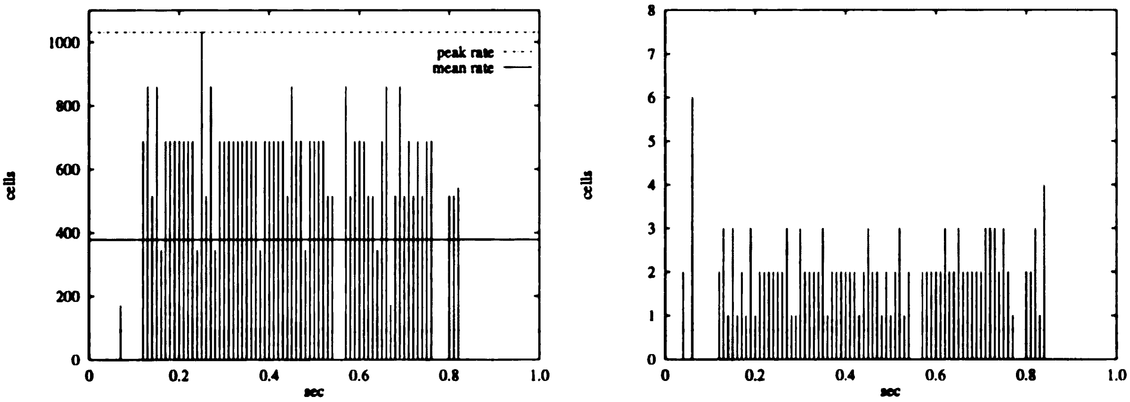


Figure 4.3: Network configuration for transferring different sizes of files using FTP.

Figure 4.4 shows both output and input port statistics of host_1, where a transfer of a 2 *Mbyte* file was monitored. Statistics gathered from the output port of the sending host show that file transfer is a form of Variable Bit Rate (VBR) traffic. Figure 4.4 depicts *peak rate* and *mean rate* for this application. The burst length in a typical FTP session will be the size of the transferred file.

[3] A transfer means a single file transfer in a particular session.

[4] A session refers to the duration of the entire application.



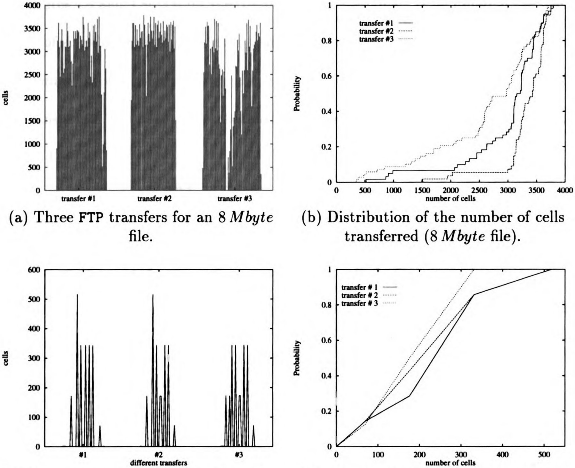
(a) Traffic from the output port: file transfer.

(b) Traffic from the input port: control signals.

Figure 4.4: Comparison of traffic in the input/output port of the sending host (2 *Mbyte* file is transferred).

Statistics gathered from the input port of the sending host show the ACK signaling required for FTP. In Figure 4.4(b), a message size of six cells marks the beginning of the file transfer. ACK messages less than three cells are exchanged throughout the FTP session. Figure 4.4(b) shows periodically exchanged SPANS signals (one-cell messages) [93] in the ATM network while an FTP session is idle. Our data for Figure 4.4(a) also illustrate the SPANS signaling cells; however, their values are too small for easy visibility.

Experiments are also repeated to study if the *mean rate* requirement and the *peak rate* requirements are the same for different FTP sessions. Our study shows that traffic characteristics of this application are affected by the network load. Figure 4.5(a) depicts transferring the same 8*Mbyte* file three times from host_1 to host_2. The first transfer and the second transfer were performed under the same network load. The third transfer was performed when there was an rn process on host_1. The gap where the minimal bandwidth is zero occurs when host_1 starts the rn process.



(a) Three FTP transfers for an 8 *Mbyte* file.

(b) Distribution of the number of cells transferred (8 *Mbyte* file).

(c) Three FTP transfers in a single session.

(d) Distribution of the number of cells transferred (100 *Kbyte* file).

Figure 4.5: Traffic characteristics: replication of FTP.

Figure 4.5(b) shows the cumulative probability of number of cells transferred during each time interval the traffic is monitored. It is clear from the three file transfers in Figure 4.5(b) that the VBR traffic is very dissimilar. The distribution shows that traffic behavior for transferring large files is hard to predict.

Figures 4.6 and 4.8(a) show the comparison for applying FTP on different file sizes. If files of sizes less than 1 *Mbyte* are transferred (refer to Figures 4.6), the peak rate requirement grows as the file size increases. If files of sizes larger than 1 *Mbyte* are transferred, the peak rates obtained for different file sizes are similar (refer to Figure 4.8(a)).

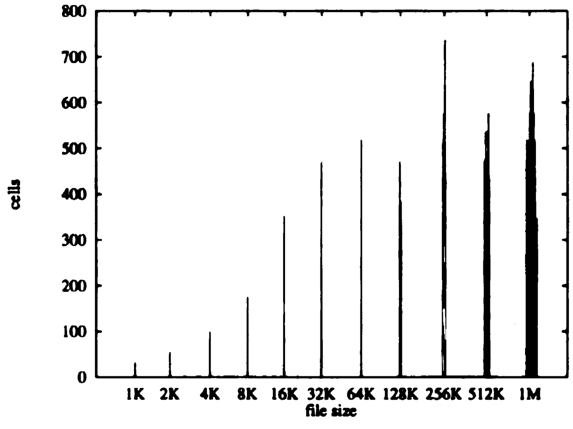
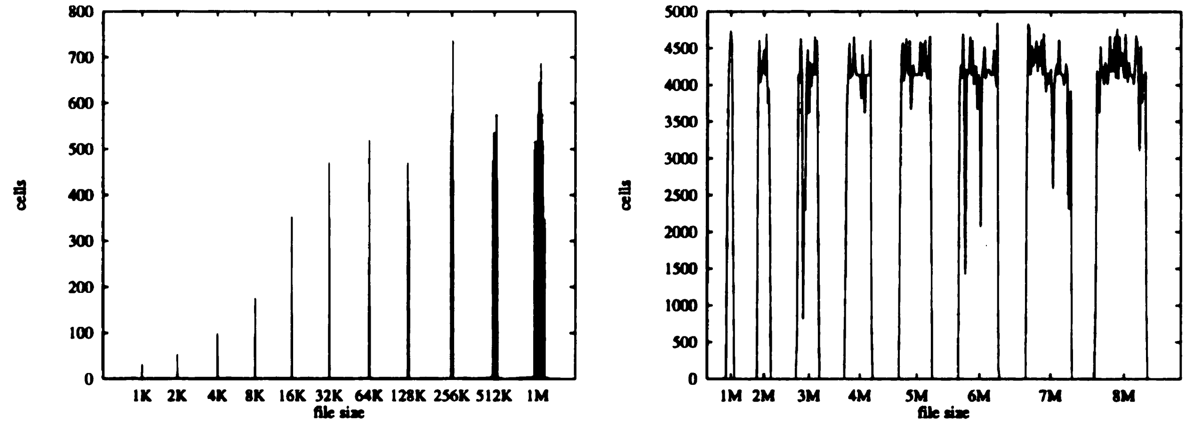


Figure 4.6: Transferring files of sizes less than 1 *Mbyte*.

The fact that the bandwidth requirements are not affected by the file sizes transferred suggests a possible solution to the problem that a user may not be able to provide the parameters needed for their applications. One approach toward addressing this problem is to recognize that all available user services can be categorized into different service classes. Each class will contain a unique set of traffic descriptors. Hence, the user would only have to request a certain class of service, which would invoke the appropriate parameters. Note that this can be implemented by establishing



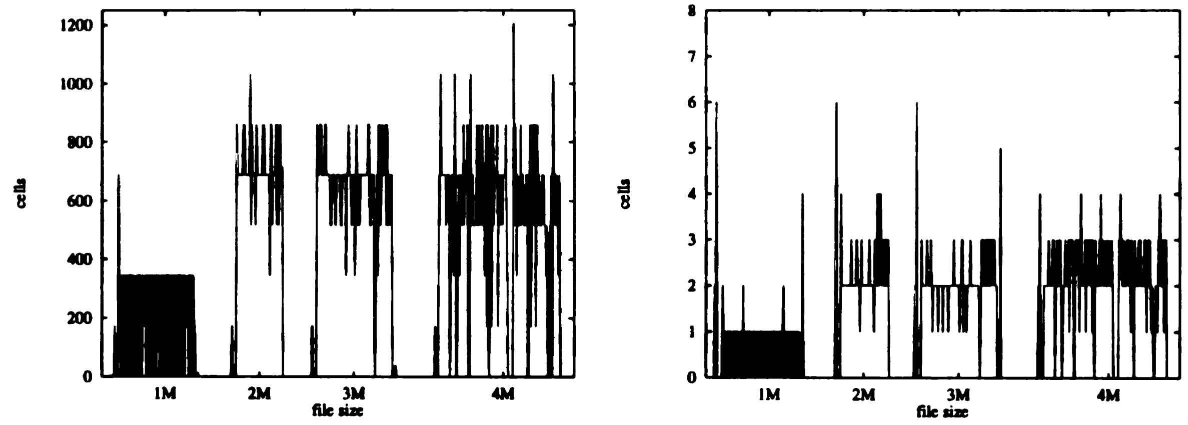
(a) Transferring files of sizes less than 1 *Mbyte*.

(b) Transferring files of sizes greater than 1 *Mbyte*.

Figure 4.7: FTP traffic characteristics for different size files.

a table of classes in a host.

Figure 4.8(b) also depicts the control signals captured for transferring different sizes of files and the periodically exchanged SPANS signaling cells. The bandwidth consumed by an FTP session is fixed once a connection is established. Also, the amount of bandwidth used for signaling by an FTP session is minimum and is not related to the file size.



(a) Traffic for file transfer.

(b) Traffic for control signals.

Figure 4.8: Transferring files of sizes greater than 1 Mbyte: input/output traffic.

Figure 4.9 shows the empirical distribution of the number of cells for each traffic monitored interval. The cumulative distributions for transferring different file sizes are shown. Note that these distributions deviate substantially from each other and are mostly affected by traffic loads.

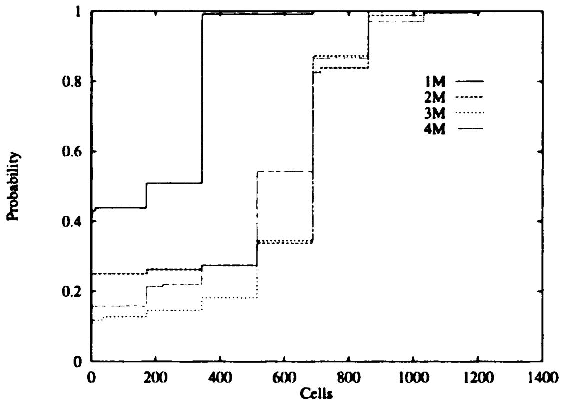


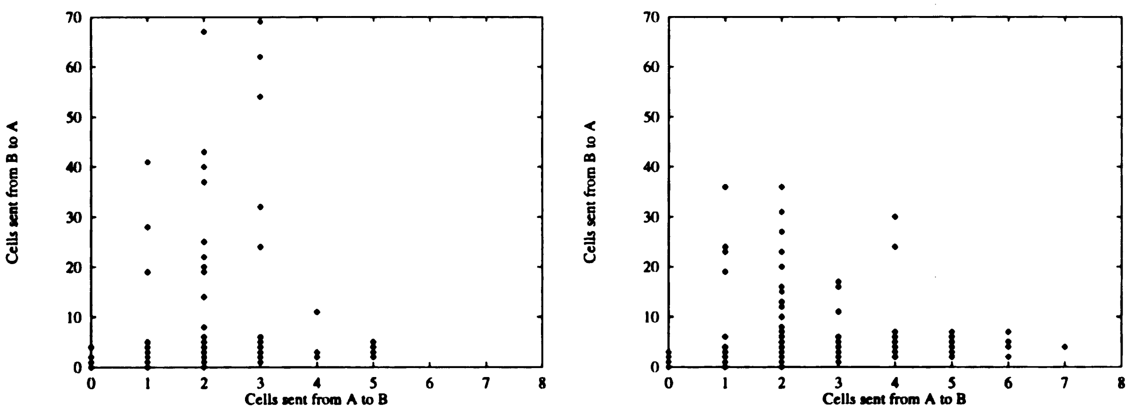
Figure 4.9: Cumulative probability for transferring different size files.

4.3.2 Telnet/Rlogin

Figures 4.10 and 4.11 show the traffic characteristics of **Telnet** and **Rlogin**. Most simulation studies have assumed unidirectional traffic flows. Figure 4.10 shows that **Rlogin** and **Telnet** traffic is strongly bidirectional. Statistics from both input ports and output ports are plotted. Output port traffic (Figures 4.11(b), (d)) is caused by the commands used in a **Telnet/Rlogin** session, while input port statistics (Figures 4.11(a), (c)) are the responses of these commands.

Cumulative distributions for **Telnet** and **Rlogin** show that these two types of traffic have similar characteristics. Danzig et al. [48, 49] have identified that **Telnet** and **Rlogin** in a WAN have the same statistics in terms of the number of bytes sent and the number of conversations[5] that took place. We have also identified that

[5] A conversation is defined as a stream of packets traveling between the end points in [49].



(a) **Telnet**: 130 *sec* trace. (b) **Rlogin**: 200 *sec* trace.

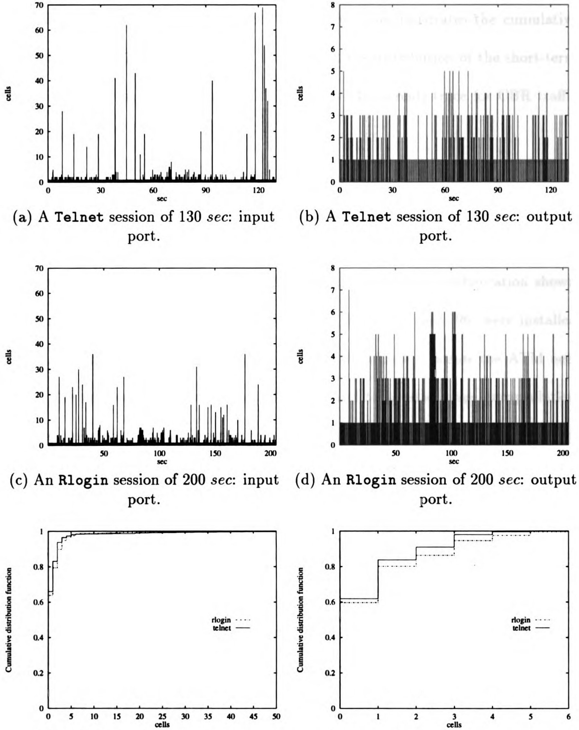
Figure 4.10: Bidirectional **Telnet** and **Rlogin** traffic.

Telnet and **Rlogin** have similar cell-level statistics, regardless of the different time durations that are used to monitor traffic.

4.3.3 Audio

Voice traffic (a form of CBR traffic) was gathered using the same network configuration employed for collecting data traffic (see Figure 4.3). A SunMicrophone [94] was attached to host_1 and used to transmit talk spurts from host_1 to host_2. Also, a SpeakerBox [94] was attached to host_2 to display the talk spurts. The application software **audio** was run on both host_1 and host_2 to handle voice transmission. Voice traffic was generated with alternating talk spurts and silence periods.

Figure 4.12 shows the statistics for voice traffic transmitted by **audio**. It depicts the number of cells transmitted during every time interval Δt ($\Delta t = 0.1$ *sec*). Experimental results indicate that the characteristics of voice traffic are application-dependent. Voice traffic can be either CBR (i.e., when silence periods are transmitted) or VBR (i.e., when silence periods are not transmitted). Figure 4.13(a) shows



(a) A **Telnet** session of 130 *sec*: input port.

(b) A **Telnet** session of 130 *sec*: output port.

(c) An **Rlogin** session of 200 *sec*: input port.

(d) An **Rlogin** session of 200 *sec*: output port.

(e) Cumulative distributions for **Telnet/Rlogin**: input port.

(f) Cumulative distributions for **Telnet/Rlogin**: output port.

Figure 4.11: **Telnet** and **Rlogin** traffic statistics.

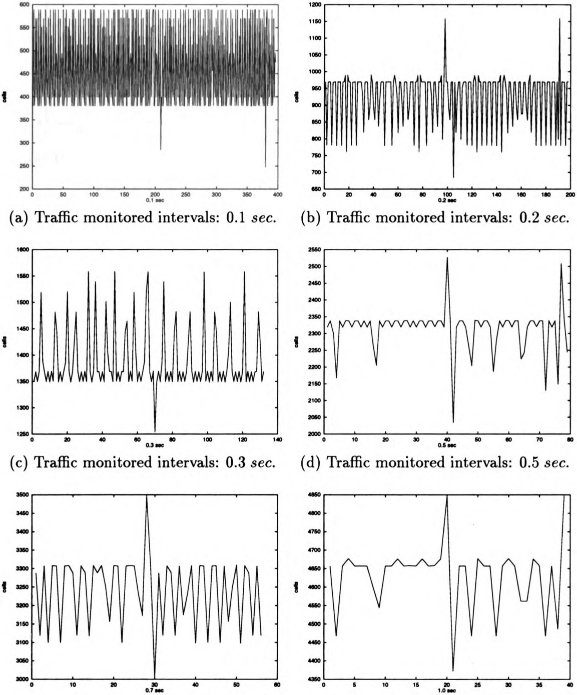
the distribution of the number of cells for **audio** traffic. While one curve shows the cumulative distribution for a 40-second trace, the other illustrates the cumulative distribution for a 10-second trace. It is clear that the distribution of the short-term (10 second) trace resembles that of the long-term (40 second) trace for CBR traffic such as **audio**.

4.3.4 Video

In order to collect data representing video traffic, the network configuration shown in Figure 4.3 was used. A SunVideo [95] and a SunVideo card [96] were installed in host_1 to capture images and display the signals to host_2 across the ATM network. The application software *SunVideo* was used to handle image encapsulation and transmission, with an image display window size of 160 × 120, using a direct mode playing 30 frames per second.

In figure 4.13(b), the distribution of the number of cells transmitted by **SunVideo** traffic is given. Cumulative distributions for an 8-second trace and a 28-second trace are plotted. For traffic generated by this application, the distributions of the short-term trace and the long-term trace are similar.

Figures 4.12 and 4.14 show the traffic statistics for **audio** and **SunVideo**. Statistics are plotted for the number of arrival cells during each time interval (e.g., 0.04 *sec*, 0.1 *sec*, 0.2 *sec*, 0.3 *sec*, 0.5 *sec*, 0.7 *sec*, and 1.0 *sec*). These figures illustrate the difficulties of using a single model to characterize traffic arrival patterns, since traffic patterns depend on traffic monitored time intervals ($\triangle t$). The *peak rate* computed



(a) Traffic monitored intervals: 0.1 *sec.*

(b) Traffic monitored intervals: 0.2 *sec.*

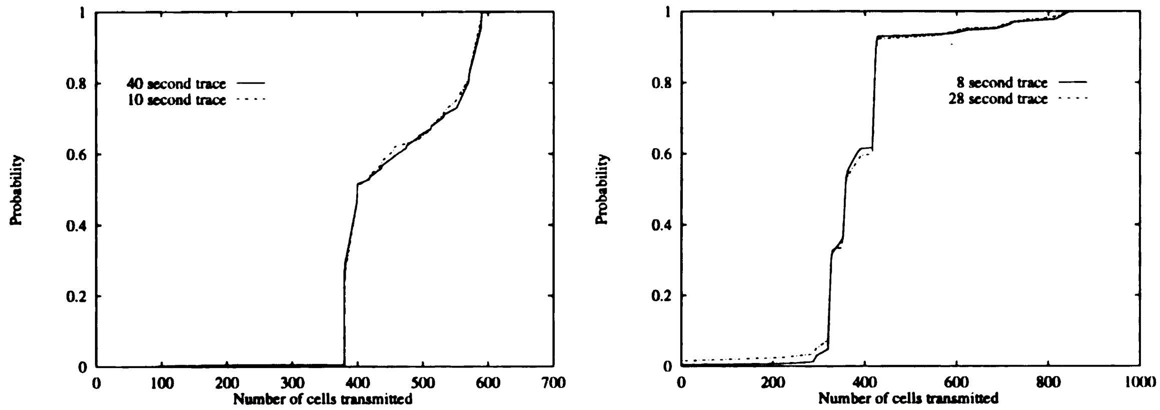
(c) Traffic monitored intervals: 0.3 *sec.*

(d) Traffic monitored intervals: 0.5 *sec.*

(e) Traffic monitored intervals: 0.7 *sec.*

(f) Traffic monitored intervals: 1.0 *sec.*

Figure 4.12: Audio traffic characteristics: traffic gathered period 40 *sec.*



(a) Distribution for **audio** traffic. (b) Distribution for **SunVideo** traffic.

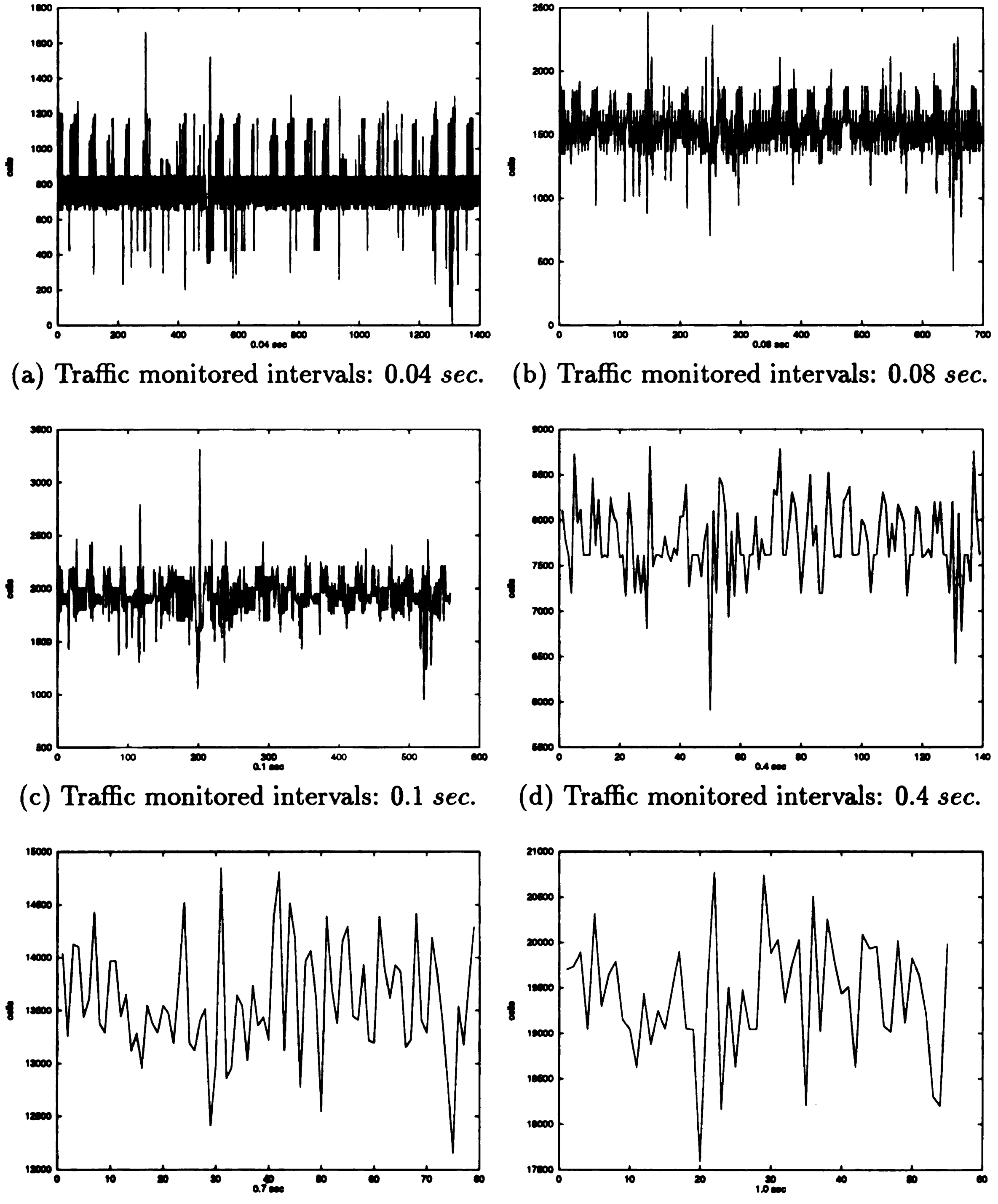
Figure 4.13: Traffic distributions for different durations.

for each application depends on $\triangle t$; that is, the smaller the interval, the higher the peak rate. This issue will be discussed in more detail in the next section.

4.3.5 World-Wide Web

Internet resource discovery tools have become available and very popular over the past few years. A number of Internet resource discovery systems are available (e.g., WHOIS[97], X.500[98], Archie[99], WWW[100, 101], WAIS[102], Knowbots[103], Netfind[104], Gopher[105]). Among these tools, the world-wide web is the most popular. Its purpose is to provide users on computer networks with a consistent way of accessing a variety of media in a simplified fashion. A number of existing protocols (e.g., FTP, NNTP, WAIS, Gopher) and one new protocol (i.e., HTTP) are implemented for the web browser. The power of the web system lies not in a complex protocol, but in the universal document identifier (UDI).

In this section, we first analyze the user access behavior from the log file, with particular attention to access patterns and the file size distributions. We trace the



(a) Traffic monitored intervals: 0.04 *sec.*

(b) Traffic monitored intervals: 0.08 *sec.*

(c) Traffic monitored intervals: 0.1 *sec.*

(d) Traffic monitored intervals: 0.4 *sec.*

(e) Traffic monitored intervals: 0.7 *sec.*

(f) Traffic monitored intervals: 1.0 *sec.*

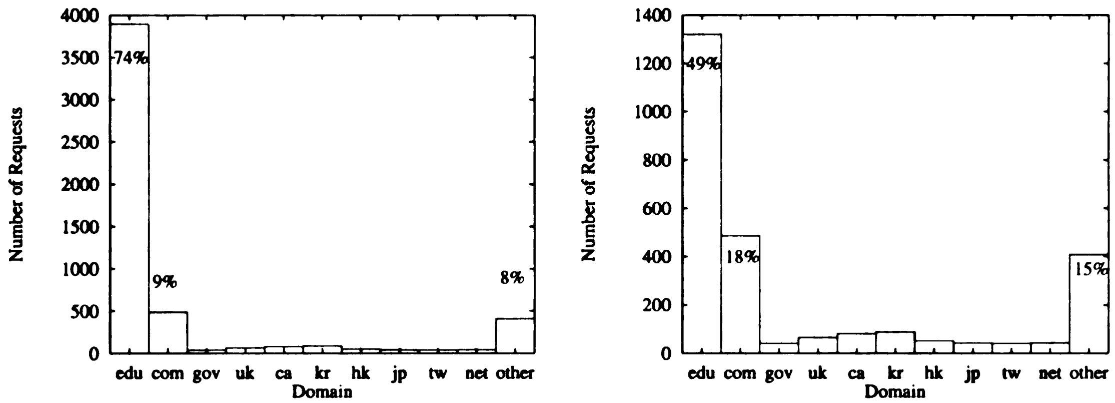
Figure 4.14: Video traffic characteristics: traffic gathered period 56 *sec.*

WWW activities at the HSNP lab for the period, from September 1994 through November 1995. Although the number of requests made was relatively small compared to other servers, the request patterns were similar for short-term traces. This fact makes the examination of the HSNP lab server valuable.

First we try to construct a profile of typical web usage from the access-log file obtained from the server. The file contains $5,243$ requests to the server during that time period. Each request record contains the following fields: source machine, accessed time, type of request, accessed file, and size of accessed file. Among the total of $5,243$ requests, $2,568$ requests are internal accesses, originated from **cps.msu.edu**. Figure 4.15 shows the number of requests for the different domains. In both cases (i.e., including and excluding internal accesses), the **edu** domain accounts for the majority of the requests, which is consistent with the Internet statistics of other Web servers [106, 107]. This implies that most people who have access to the web are members of the educational domain.

Note that in Figure 4.15 requests from the Internet provider (i.e., traffic originated from domain **net**) are visible. These requests comprise approximately 3 percent of the total traffic volume. Excluding the domain **other**, which consists of 8% of miscellaneous small accesses from various domains, five of the top ten domains represent requests from outside of North America. This implies that there is no space locality in Web accesses. The **edu**, **com**, **gov**, **ca**, **uk**, **net**, and **internal access** domains, are usually classified in the top ten accessed domains for Mosaic traffic for well-known servers, as well as for private lab servers such as ours.

Figure 4.16 depicts the access statistics by month, week, and day. The statistics



(a) Including internal accesses

(b) Excluding internal accesses

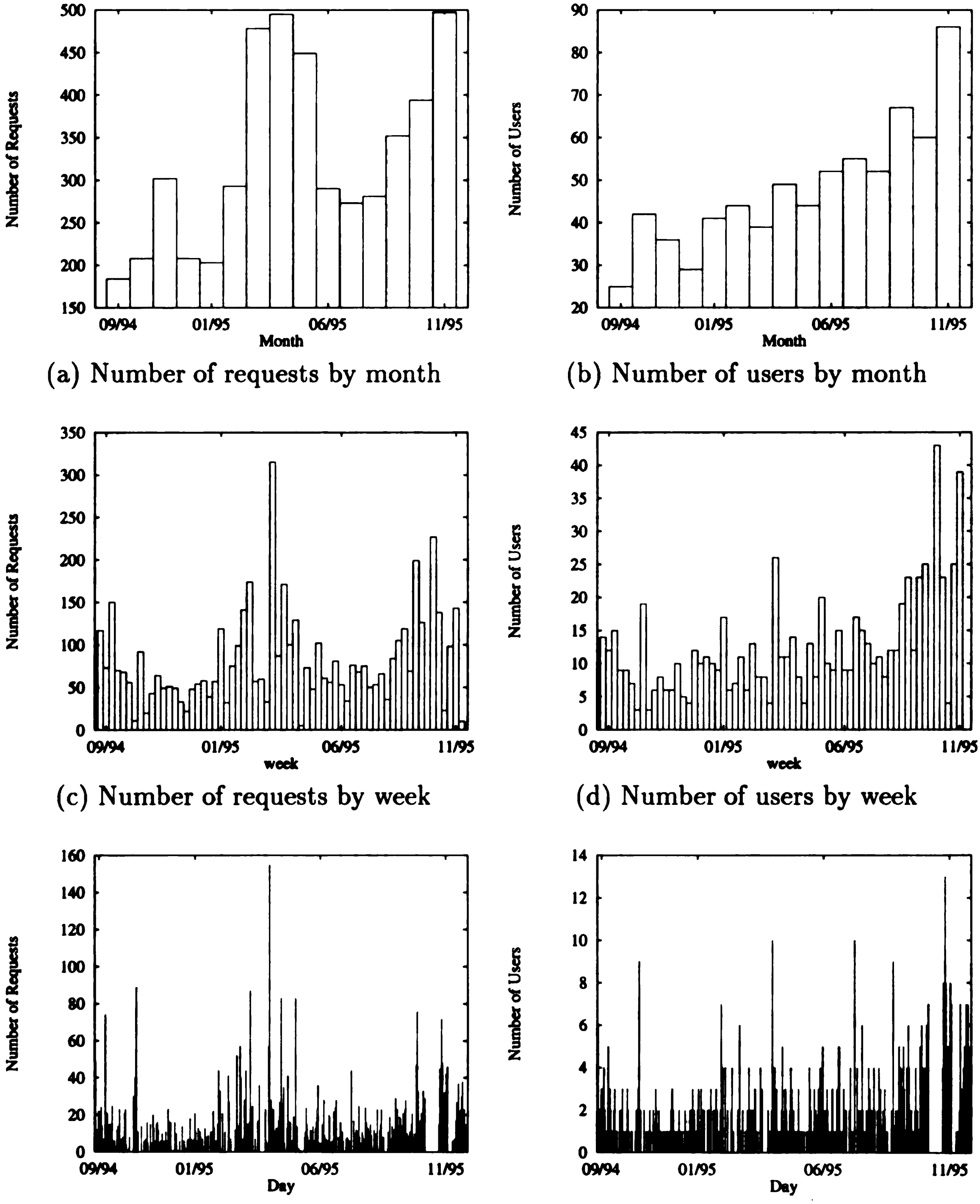
Figure 4.15: Domain distributions for WWW requests

here are computed based on the number of requests and the number of users. Note that a user can have more than one request when accessing the web services. Since the users at the HSNP lab are using SPARC10 stations, one user per client is an underlying assumption in our log-file analysis. It is shown in Figure 4.16(a) that the peak request times for the periods observed are during the months of March, April, May, September, October, and November of 1995. This trend exists due to the fact that most requests originate from the **edu** domain, and frequent accesses during the fall and spring semesters are expected. Regardless of the number of requests per user, Figure 4.16(b) shows that the number of users of the web services is an increasing trend. Figures 4.16(c), (d), (e), and (f) give higher resolutions of the number of requests than do Figures 4.16(a) and (b) (i.e., the total number of requests per week and per day). However, traffic volume cannot be predicted better by using small time-scales (e.g., by week or by hour) for a server without a huge amount of traffic. The total number of requests (e.g., ranging from 0 to 160) and the total number of users vary significantly for each day. It is interesting to note that the maximum number of users, as shown on 11/95 in Figure 4.16(d), does not correspond to the maximum number of requests, as shown on 03/95 in Figure 4.16(c). This is because that one user can generate multiple requests.

Figure 4.17 shows the access activities by the hour. From a long-term view, peak request time is at 3:00 p.m., as shown in Figures 4.17(a) and (b), which summarize the statistics for the period between September 1994 and November 1995. As we look at the busiest month, November 1995, from the access-log file as shown in Figures 4.17(c) and (d), heavy traffic occurs during a time period between 11:00 a.m. and 7:00 p.m. (by requests) and between 11:00 a.m. and 4:00 p.m. (by users). It is interesting as we look at both the long-term statistics (i.e., Figures 4.17(a) and (b)) and the short-term statistics (i.e., Figures 4.17(c) and (d)) that light traffic occurs during the time period 5:00 a.m. to 7:00 a.m..

Figure 4.18 shows the statistics by weekdays. Although the maximum number of requests and the minimum number of requests differ substantially, the average number of requests does not vary much between different weekdays (refer to Figure 4.18(a)). However, the total number of requests and the total number of users for each weekday, represent the period between September 1994 and November 1995. As can be seen, these total times vary tremendously. Almost 74 percent of the activity occurs between Tuesday and Friday. Note that Saturday has a total number of requests that is even greater than the summation of Sunday and Monday. This is also true for the total numbers of users. Although there is a large number of users accessing web services on Saturday, yet these users generate only a small number of requests.

We also studied the access pattern for files. `Gif` files are the most frequently



(a) Number of requests by month

(b) Number of users by month

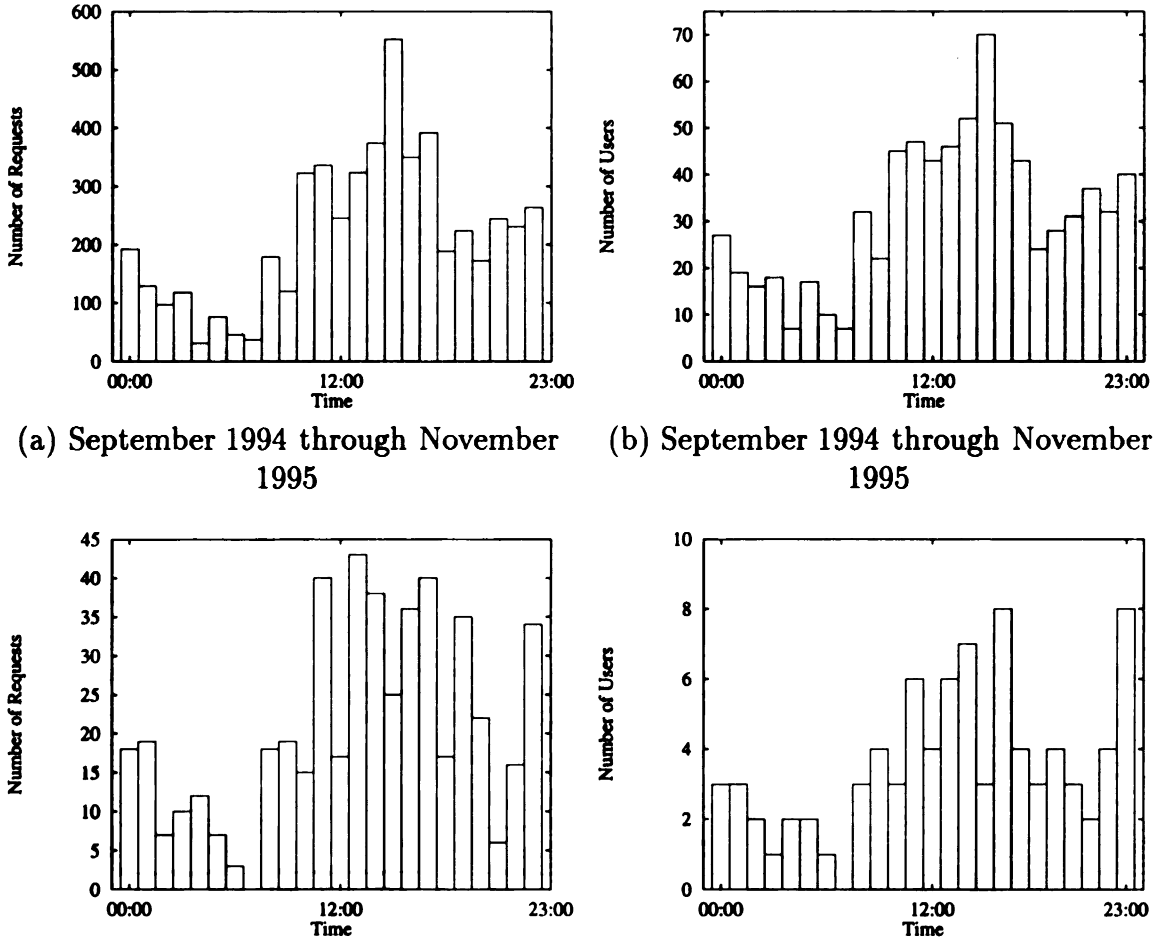
(c) Number of requests by week

(d) Number of users by week

(e) Number of requests by day

(f) Number of users by day

Figure 4.16: Access statistics for WWW requests



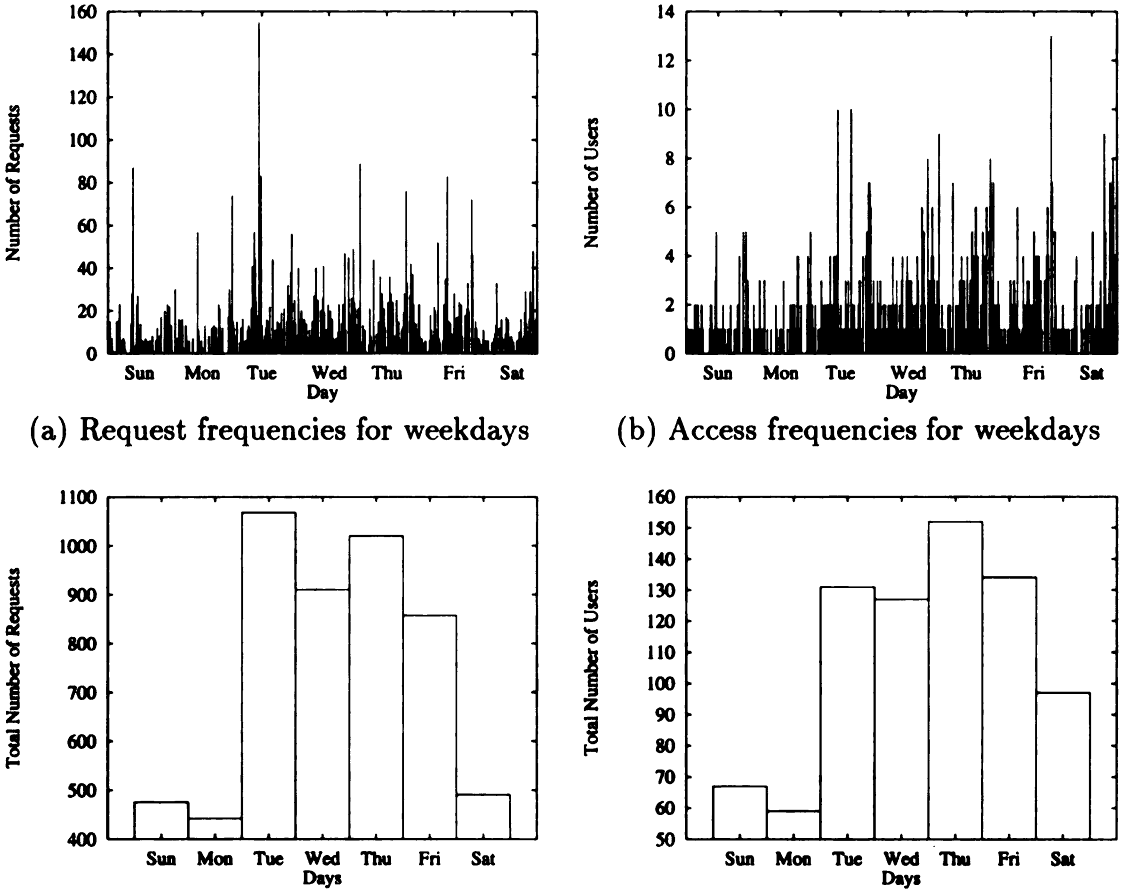
(a) September 1994 through November 1995

(b) September 1994 through November 1995

(c) November 1995

(d) November 1995

Figure 4.17: Access frequency distributions by the hour



(a) Request frequencies for weekdays

(b) Access frequencies for weekdays

(c) Request summaries for weekdays

(d) Access summaries for weekdays

Figure 4.18: Access statistics for weekdays

accessed, followed by **html** files. Both consist of 80 percent of total accesses. One thing notable in the accessed files is that 23 percent of accesses in **gif** files are to **ball.gif**, which is the most used image. This means that caching this ball image will improve performance, reduce network traffic, and minimize the impact of this image. The file size distributions do not fit into any of the existing models (refer to Figure 4.19). This figure shows a noticeable amount of **ball.gif**, and large files are mostly **gif** files.

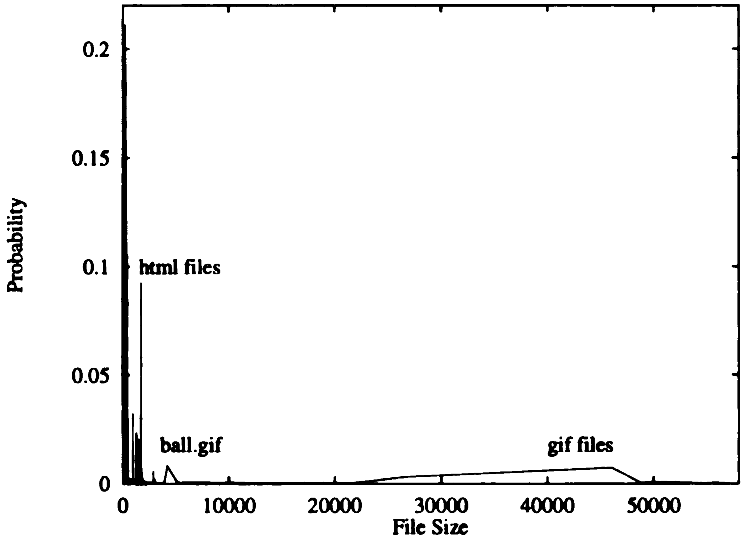


Figure 4.19: File size distributions for Mosaic traffic

4.4 Traffic Descriptors

The ATM traffic descriptor is the generic list of traffic parameters that can be used to capture the traffic characteristics of an ATM connection [10]. In this section we present the computation of the set of traffic descriptors from our traffic measurements. Note that in our call admission control algorithm, traffic descriptors such as peak rate and mean rate are necessary in order for the network to decide whether a call can be admitted.

4.4.1 Peak Cell Rate and Sustainable Cell Rate

As we have shown, traffic characteristics depend on the time interval (Δt) in which the statistics were gathered. Figure 4.20 shows the same FTP transfer that was monitored using different time-scales. Experiments $1-6$ use different time-scales ($\Delta t = 0.01$ *sec*, 0.02 *sec*, 0.03 *sec*, 0.04 *sec*, 0.05 *sec*, and 0.1 *sec*, respectively) to monitor the same FTP session. In this figure, the number associated with each file transfer indicates the peak number of cells observed during every monitored time interval.

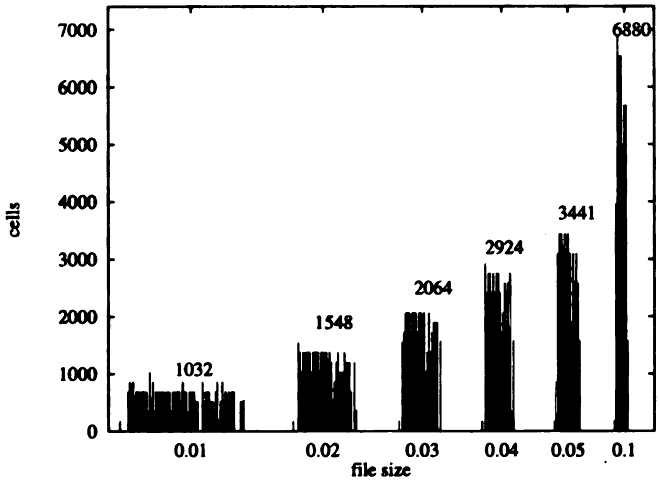
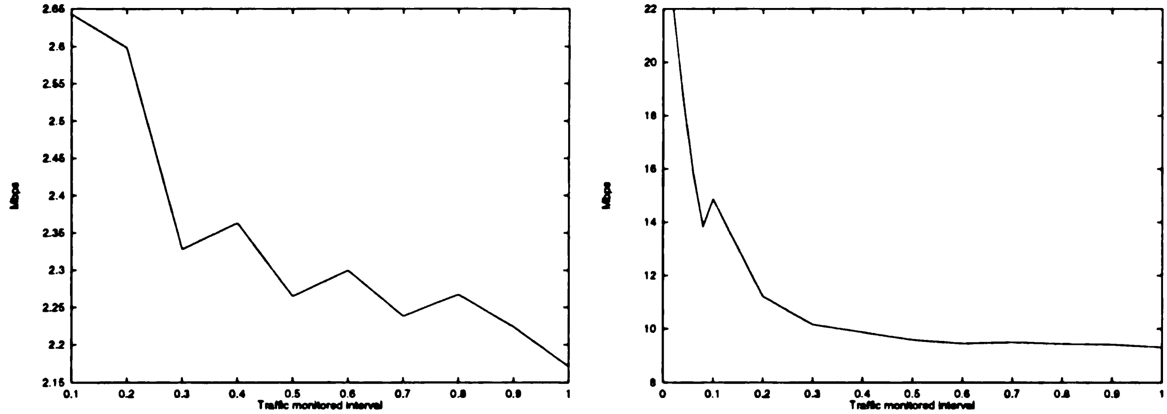


Figure 4.20: Transferring the same $2M$ file monitored by different time scales.

The *peak rate* computed for each application depends on Δt; that is, the smaller the interval, the higher the peak rate. Figure 4.21 shows the *peak rate* for **audio** and **SunVideo** traffic for different values of Δt. Note that the relation between the *peak rate* and the traffic monitored interval is not linear, due to the bursty nature of voice and video traffic. Figure 4.21 illustrates that video traffic requires a much higher peak bandwidth (> 22 *Mbps*) than audio traffic does (3 *Mbps*).



(a) `Audio` traffic.

(b) `SunVideo` traffic.

Figure 4.21: Peak rates computed from different traffic monitored intervals.

For both sources, the peak rates are bounded in intervals of different lengths. Generally speaking, over shorter intervals, a source's rate may be bounded by a random variate that is weighted toward its peak rate; whereas for longer intervals, the source's bounding rate approximates its long-term average rate. By capturing the characteristics of sources, higher network utilization is achievable.

The long-term average rate can be defined as $\lim_{t\to\infty} \frac{1}{t} \int_0^t r(\tau)d\tau$, where $r(\tau)$ is the source's instantaneous rate at time τ (e.g., in cells per second). The curve is not necessarily convex or monotonically decreasing, as shown by the real traffic trace in Figure 4.21. Without an existing mathematical model, the general trend is that peak rate decreases with increasing interval length.

A maximum number of 1032 *cells* was observed from the statistics gathered via the time interval 0.01 *sec*, which results in 103200 *cells/second.* Hence, a peak rate of (103200 *cells/second*) $*$ (56[6] *bytes/cell*) $*$ (8 *bits/byte*) $=$ 46.233 *Mbps* is obtained.

[6] The cell size for FORE System is 56 bytes.

Similar calculations can be applied to find peak rates for statistics monitored via different time intervals (refer to Table 4.1). Results from Table 4.1 suggest that the peak rates calculated are affected by the time intervals for which traffic is monitored. Applying the same calculations for other application traffic, the peak rates and mean rates can be obtained (refer to Table 4.2[7]).

Time slot monitored (Δt)	0.01 *sec*	0.02 *sec*	0.03 *sec*	0.04 *sec*	0.05 *sec*	0.1 *sec*
$\frac{\text{cells}}{\text{slot}}$	1032	1548	2064	2924	3441	6880
$\frac{\text{cells}}{\text{second}}$	103200	77400	68800	73100	68820	68800
Mbps	46.23	34.68	30.82	32.75	30.83	30.82

Table 4.1: Peak rate measurements for FTP traffic.

	FTP	Audio	Video	Telnet	Rlogin
Peak rate	46.23 *Mbps*	2.7 *Mbps*	22 *Mbps*	310 *Kbps*	160 *Kbps*
Mean rate	–	1.8 *Mbps*	9.5 *Mbps*	–	–

Table 4.2: Bandwidth consumed by different applications.

4.4.2 Burstiness

Because the burstiness of traffic can have a significant effect on queuing delays and network congestion, traffic measurements and burstiness characterizations are essential for the analysis and evaluation of network performance and for call admission

[7] Mean rates for VBR traffic are not applicable since different conversation durations for the same application result in different mean rates.

control. The open issues in burstiness characterizations are how to define burstiness and how to quantify it.

Burstiness metrics fall into two categories: those that measure interarrival processes (e.g., time between packet arrivals) and those that measure arrival processes (e.g., number of packets per time interval). Figure 4.22 gives a summary of different definitions of burstiness ($b_1, \ldots, b_6$).

The simplest way to define burstiness is the ratio of peak rate to mean rate (b_1). This definition is widely used [108, 109, 27, 110]. However, it has several pitfalls. We have shown that the peak rate calculated is highly dependent on the time interval that is used to monitor the traffic [74], as also illustrated in Figure 4.21. Furthermore, the mean rate is not only application-dependent, but also user-dependent. Consider calculating the mean rate for an ftp session[8] where $mean\_rate = \frac{transferred\_file\_size}{ftp\_duration}$. $ftp\_duration$ may vary for different users even when the same size files are transferred. All of these reasons make the mean rate of FTP difficult to predict. Therefore, the definition of burstiness (b_1) is not unique, since it depends on both peak rate and mean rate.

Hirano et. al. [63] defines burstiness as the average time interval for a burst at an active state in his two-state cell arrival model for an input line (b_2). Although bursty traffic can be characterized by *maximum burst length* and *mean burst rate*, it is very difficult to measure bursty traffic. Figure 4.23 illustrates this difficulty.

Input traffic $T1$ has the same *maximum burst length* and *mean burst rate* as $T2$, but we can see in Figure 4.23 that their bursty characteristics are different. It can be

[8] A session refers to the duration of the entire application.

Definition	Expression
b_1	$\frac{\textit{peak rate}}{\textit{mean rate}}$
b_2	average burst length at the unit of time interval
b_3	$\frac{\text{Var[cell interarrival times]}}{\text{E[cell interarrival times]}}$
b_4	$\frac{\text{Var[cell interarrival times]}}{E^2\text{[cell interarrival times]}}$
b_5	$\frac{\text{Var}[\text{N}(\Delta t)]}{\text{E}[\text{N}(\Delta t)]}$, N($\Delta t$) is the counting process associated with an arrival process
b_6	$\frac{\text{Var}[\text{N}(\Delta t)]}{E^2[\text{N}(\Delta t)]}$, N($\Delta t$) is the counting process associated with an arrival process

Figure 4.22: Different definitions of burstiness.

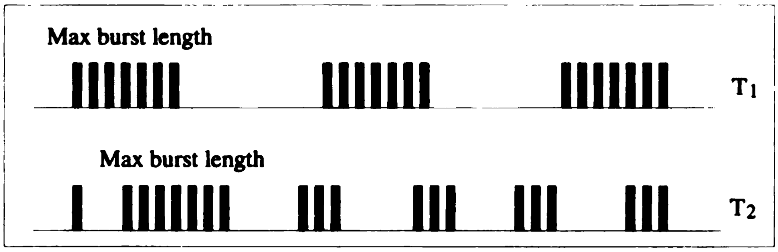


Figure 4.23: Different bursty traffic with the same mean burst rate and maximum burst length.

observed that the bursty traffic depicted by $T1$ has a higher probability of causing congestion than that in $T2$. Of course this is due to the variations of the respective burst lengths. Burst length is sensitive to the input buffer size. If the burst length approximates the buffer size, only one burst can cause buffer overflow, which, in turn, causes cell loss and delays.

It is known that bursty traffic is very difficult to characterize using a two-state model [111]. However, Hirano's work shows that the burst length significantly affects network performance: the longer the burst length, the larger the buffer needed to prevent cell loss. Therefore, b_2 is a good indicator of the buffer requirements for preventing a significant cell loss.

Hui and Arthurs [112] calculate the burstiness as the packet jitter ratio defined as the variance-to-mean ratio of the packet interarrival times (b_3). Sriram and Whitt [113] define burstiness as the squared coefficient of the interarrival time (b_4). In the original proposed definitions (b_3 and b_4), packet interarrival times are calculated instead of cell interarrival time. However, with a link speed of 155 $Mbps$, a cell can have an interarrival time as short as 2.89 μsec, which is only feasible in a mathematical model, but is not possible to measure without excessive hardware supports. Hence, we do not compute b_3 and b_4 from our measured data.

We found no satisfactory definitions of burstiness in the literature. In order to solve the problem, we propose two definitions of burstiness (b_5 and b_6), which are feasible to compute from our traffic measurements. Since burstiness can be related to the relationship among successive arrivals, we propose definitions based on the counting process, $N(\triangle t)$, associated with an arrival process, where $\triangle t$ is the traffic monitored interval in our traffic measurements, as discussed at the beginning of this chapter. We argue that burstiness depends on the lengths of sequences of observed cells smaller than the mean, the lengths of sequences larger than the mean, and the way the two types of sequences alternate.

Tables 4.3 – 4.6 show the calculations of burstiness for different applications. A good definition of burstiness greatly facilitates our ability to predict traffic behavior for different applications. We expect the value of burstiness to be about the same for each application. The index of dispersion for each definition of burstiness is defined as the ratio of mean value ($\bar{s}$) to the standard deviation (σ). The index will serve as the primary criterion for evaluating definitions of burstiness. For example, a better definition of burstiness will have a smaller index of dispersion.

Burstiness	#1	#2	#3	#4	#5	#6	#7	#8	#9	#10	$\bar{s}$	σ	Dispersion
s_1	3.58	3.75	3.87	3.74	5.56	5.65	5.51	5.61	5.85	5.51	4.86	0.99	0.20
s_2	96.1	92.0	89.0	92.0	92.8	91.3	93.7	92.0	88.3	93.6	92.1	2.26	0.02
s_5	1.57	1.62	1.67	1.60	1.63	1.64	1.61	1.65	1.72	1.59	1.63	0.04	0.02
s_6	0.02	0.02	0.02	0.02	0.02	0.02	0.02	0.02	0.02	0.02	0.02	0.00	0.00

Table 4.3: Burstiness for replication of applying FTP.

Tables 4.3 and 4.4 show the burstiness for FTP traffic. Table 4.3 shows the replications of applying FTP for a 500 *Kbyte* file. From this table, it is observed that s_6 $(\frac{\mathrm{Var}[\mathrm{N}(\Delta t)]}{E^2[\mathrm{N}(\Delta t)]})$ is the best definition of burstiness since the value of dispersion is zero. The value of s_6 is the same for any of the replications of FTP. Table 4.4 shows the results of applying FTP traffic for different sized files, where s_5 $(\frac{\mathrm{Var}[\mathrm{N}(\Delta t)]}{\mathrm{E}[\mathrm{N}(\Delta t)]})$ has the smallest dispersion value.

Burstiness	1M	2M	3M	4M	5M	6M	7M	8M	$\bar{s}$	σ	Dispersion
s_1	4.98	4.86	5.55	5.44	5.23	5.33	5.08	4.63	5.14	0.31	0.06
s_2	69.2	70.9	62.2	63.5	66.0	64.7	67.9	74.5	67.5	4.1	0.06
s_5	1.82	1.81	1.96	1.95	1.90	1.88	1.82	1.73	1.86	0.08	0.04
s_6	0.03	0.03	0.03	0.03	0.03	0.03	0.003	0.02	0.03	0.003	0.10

Table 4.4: Burstiness for applying **FTP** for different sized files.

Tables 4.5 and 4.6 show the computed burstiness values for **SunVideo** and **audio** traffic when these two types of traffic are gathered using different time scales. Here s_6 has the least value of dispersion for **SunVideo** and **audio** traffic. In Table 4.6, the value of s_6 is very small (i.e., approximately equal to zero). In summary, the proposed definition of burstiness, s_6, has the lowest dispersion value for different traffic. Hence, it is considered a better definition.

Burstiness	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1.0	$\bar{s}$	σ	Dispersion
s_1	1.71	1.29	1.17	1.14	1.10	1.09	1.09	1.09	1.08	1.07	1.18	0.20	0.17
s_2	1941	3882	5823	7764	9706	11646	13589	15532	17470	19410	10677	5877	0.55
s_5	8.79	13.51	16.14	17.58	20.47	25.17	25.12	24.76	26.93	29.58	20.8	6.65	0.32
s_6	0.04	0.05	0.05	0.04	0.04	0.05	0.05	0.04	0.04	0.05	0.05	0.005	0.10

Table 4.5: Burstiness for **SunVideo** traffic.

Burstiness	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1.0	$\bar{s}$	σ	Dispersion
s_1	1.27	1.25	1.12	1.14	1.09	1.11	1.08	1.09	1.07	1.05	1.13	0.08	0.07
s_2	463	926	1389	1851	2314	2777	3238	3701	4166	4630	2545	1401	0.55
s_5	0.19	0.09	0.04	0.05	0.03	0.03	0.03	0.02	0.02	0.02	0.05	0.05	1.00
s_6	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00

Table 4.6: Burstiness for audio traffic.

4.5 Multiplexing Effects

4.5.1 Multiplexing Issues

Multiplexing effects and network loads are two major factors that affect traffic characteristics. The multiplexing effect of homogeneous and heterogeneous traffic mixes is an important issue in effective bandwidth utilization. By using statistical multiplexing, VBR traffic can fully utilize the network bandwidth. Multiplexing can also change the traffic statistics. Figure 4.24 shows an example of an increase in peak rate for connection C_3 due to multiplexing. Bae et al. [114] have shown that the QoS specification for a set of heterogeneous sources may have to be set more stringently than the most stringent of any of the individual QoS requirements, in order to ensure that all sources receive their required QoS.

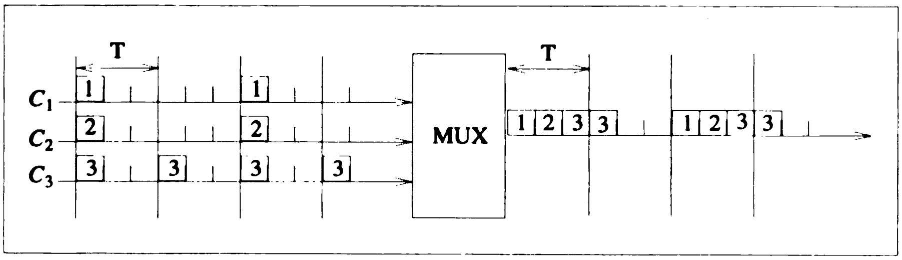


Figure 4.24: Peak rate increases due to multiplexing effects.

Network load is another factor that affects traffic statistics. Traffic behaves differently under different loads. A CAC algorithm will not be able to meet QoS under high load. We focus our study on the ***admissible load region*** [31, 13], the maximum network load for which the network can satisfy QoS, for different types of traffic.

Several studies have shown that different traffic types that vary widely in burstiness do not multiplex very well [114]. The admission control algorithm should make efficient use of its resources. In particular, it should not waste bandwidth. High-speed networks require multiplexing techniques that conserve bandwidth usage for homogeneous/heterogeneous traffic mixes. The goals of studying the multiplexing effects are to understand traffic characteristics, to realize the real load bandwidth requirements, and to refine the call admission control algorithm.

4.5.2 Multiplexing FTP Traffic

Figures 4.25 and 4.26 show the multiplexing effects of applying multiple FTP sessions. Figure 4.25 shows the multiplexing effects for 100 *Kbyte* files; whereas Figure 4.26 is for 1 *Mbyte* files. The figures are plotted when multiple FTP connections exist at the same time. As shown from the figures, the bandwidth requirements for the aggregate traffic do not grow as more ftp connections are simultaneously requested.

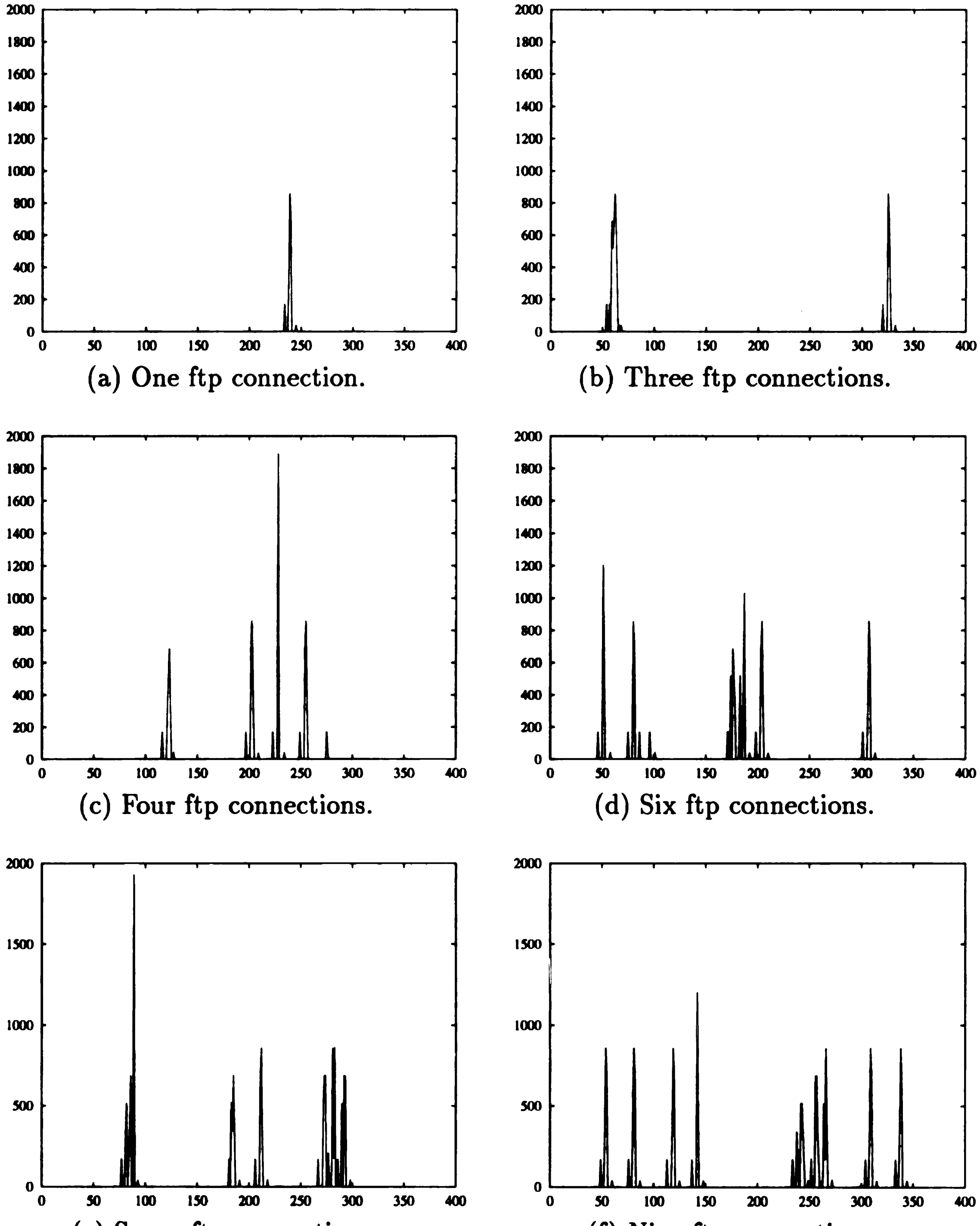
The effect of multiplexing small files (refer to Figure 4.25) is similar to that of having sequential FTP connections. This is caused by the small ratio of the file size to the high link speed available in an ATM network. The effect of multiplexing large files (see Figure 4.26) is similar to that of having a single FTP connection for a concatenated file. For example, multiplexing 10 FTP connections of a 1 *Mbyte* file has a similar result to that of having a single FTP session for a 10 *Mbyte* file. Such characteristics of the multiplexing effects can be applied to develop an efficient call admission algorithm based on the empirical bandwidth requirements. From Figures 4.25 and 4.26, it can be observed that multiplexing traffic from several bursty sources produces less bursty traffic. This implies that the bandwidth requirements of the multiplexed traffic vary more slowly with time than unmultiplexed traffic.

4.6 Summary

In this chapter, we investigated the traffic characteristics of several popular applications that comprise a significant load on WANs. Specifically, five types of traffic (FTP, Telnet, Rlogin, voice, and video) in ATM LANs are considered.

Cell-level traffic statistics are gathered by using the ATM testbed at the HSNP lab. The solaris software library provides a time resolution of nanosecond for our traffic monitored program. From the measured data, traffic descriptors (peak rate and mean rate) are determined. Our results indicate that the peak rate calculated for VBR traffic is affected by the time interval ($\triangle t$) that is used to monitor the traffic. When transferring files smaller than 1 *Mbyte*, the bandwidth requirement increases as the file size increases. On the other hand, when files larger than 1 *Mbyte* are transferred, the peak rates are the same.

It was observed that similarities exist between Telnet and Rlogin measured traffic, regardless of the different time intervals that are used to monitor traffic. This is



(a) One ftp connection.

(b) Three ftp connections.

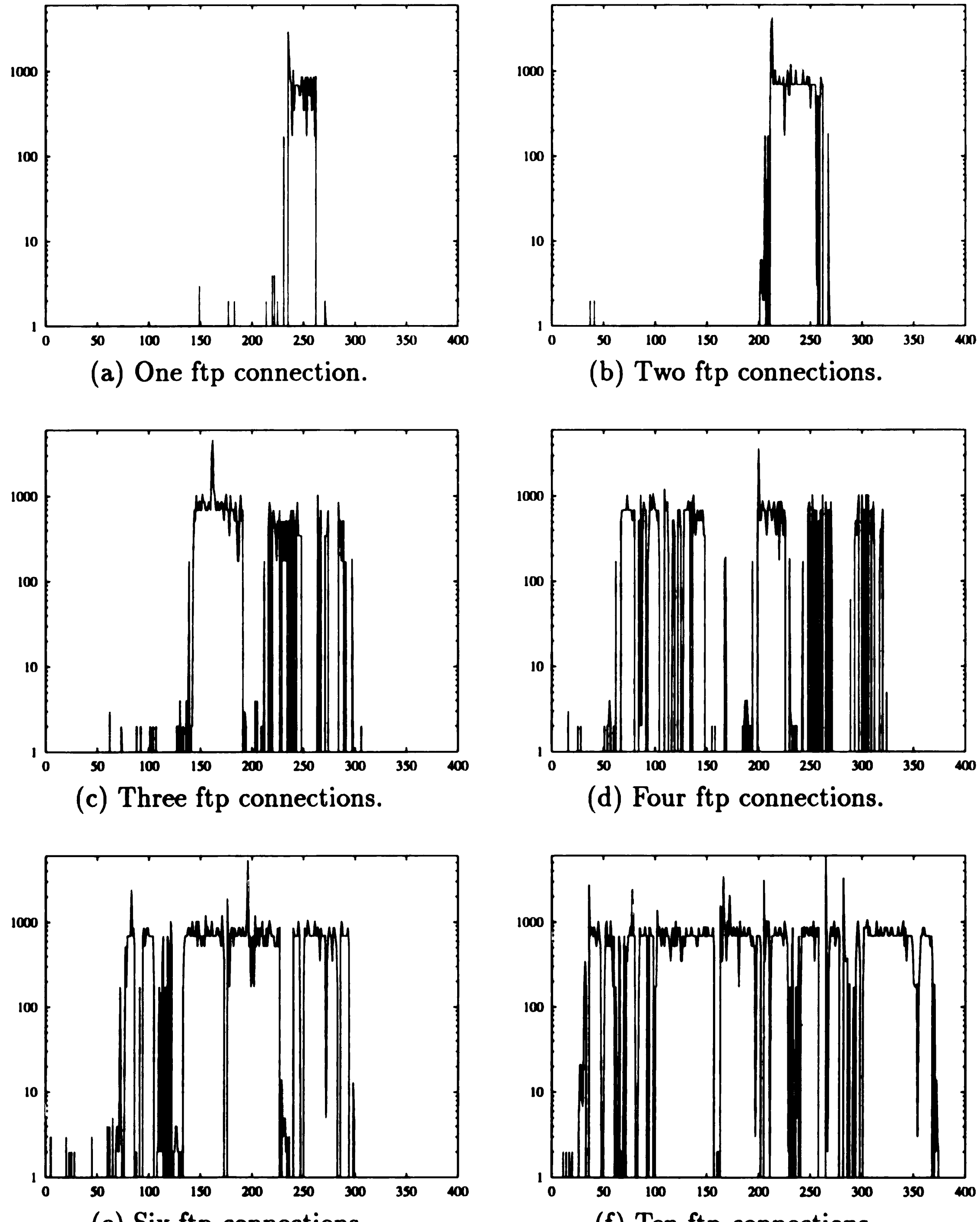
(c) Four ftp connections.

(d) Six ftp connections.

(e) Seven ftp connections.

(f) Nine ftp connections.

Figure 4.25: Multiplexing effects when 100K files are transferred.



(a) One ftp connection.

(b) Two ftp connections.

(c) Three ftp connections.

(d) Four ftp connections.

(e) Six ftp connections.

(f) Ten ftp connections.

Figure 4.26: Comparison of traffic in the output port of the sending host when a 1M file is transferred.

further evidenced by the closeness of their cumulative distributions from our results.

New definitions of burstiness are presented based on the counting process associated with the arrival process. These definitions, along with several existing definitions, are evaluated by using the measured data. The index of dispersion is used to determine a better definition of burstiness. Our results indicate that the proposed definition of burstiness has the smallest dispersion value for different types of traffic, and can be considered a better definition.

The multiplexing effects of applying **FTP** to transfer large and small files are also studied. It can be observed that the bandwidth requirement for multiple **FTP** connections does not scale as the number of connections increases. The effect of multiplexing small files is similar to that of having a sequence of **FTP** connections. This is due to the small ratio of the file size to the high link speed in an ATM network. The effect of multiplexing large files is similar to that of having a single **FTP** connection for a concatenated file.

Chapter 5

ATM Workload Model for Evaluating the Call Admission Control Framework

5.1 Introduction

An accurate workload model is an important component of any simulation study. In the particular realm of our research activity, network simulations need a good model of traffic sources because the performance of our call admission control algorithm is critically dependent on the input traffic mixes. We need the traffic characteristics previously presented to derive a model of application traffic for several of the major types of wide-area traffic, namely FTP, Audio, SunVideo, Telnet/Rlogin, and Mosaic.

This chapter presents results that emphasize the development of a workload model. The workload model presented in this paper consists of a collection of functions that can be used to simulate application programs. This model relies on measurements from a WAN environment (i.e., *tcplib* [49, 52]) and measurements from our ATM testbed. Such an empirically based model can be used to simulate LAN/WAN ATM traffic for studying communication protocols, scheduling algorithms, and congestion control schemes. The purpose of the workload model developed here will be to study the performance of our call admission control framework.

5.2 Workload Model Formulation

Our workload model consists of two parts: a WAN-dependent part and an application-dependent part. For the WAN-dependent part, the model relies on measurements from a WAN environment, which includes the following functions: **ftp_itemsize**, **ftp_duration**, **telnet_duration**, and so on. For the application-dependent part, this model relies on measurements from our ATM testbed. Such measurements are needed for all applications under consideration, since each application has a different cell arrival pattern. Figure 5.1 shows the structure of the workload model. Such a model can be used to simulate ATM LAN traffic when applications are selected randomly. This model can also be used to simulate ATM WAN traffic when parameters such as traffic breakdown, conversation arrivals, conversation durations, and packet sizes are selected based on *tcplib* [48, 49].

The traffic model is invoked by applications such as **FTP_Sim**, **Audio_Sim**, **Video_Sim**, **Telnet_Sim/Rlogin_Sim**, and **Mosaic_Sim**. The corresponding cell-generating function (**audio()**, **ftp()**, **video()**, **telnet()**, and **Mosaic()**) used to

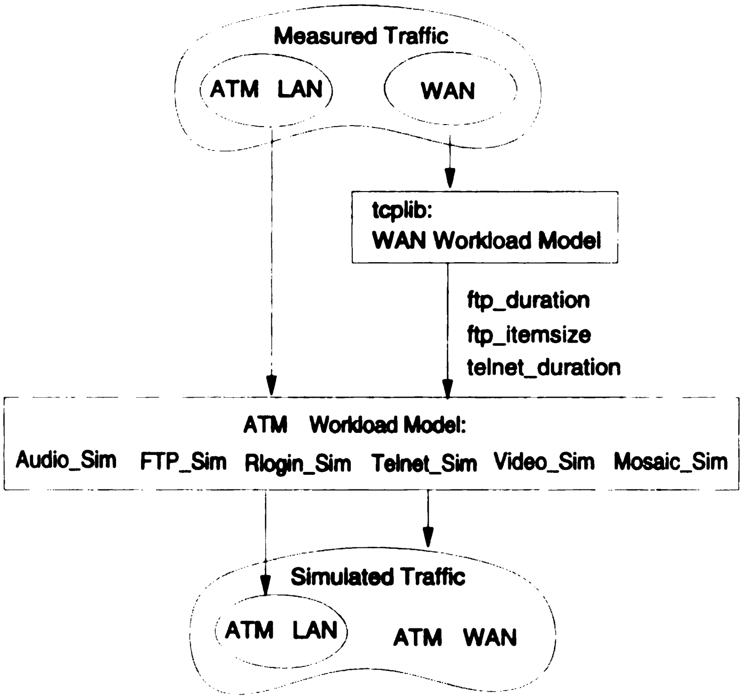


Figure 5.1: The workload model.

simulate source traffic is activated based on the application selected.

Table 5.1 (i.e., the workload model) consists of five applications, along with their associated traffic-generating functions. Within each of these applications, the first function is a cell-generating function and the remaining functions return values for traffic descriptors (e.g., peak rate and mean rate). A cell-generating function is a function that generates the appropriate number of cells in every time interval Δt. More detailed discussions of these cell-generating functions are presented in the next section.

Application	Function Name
Audio_Sim	void audio() float audio_peak_rate() float audio_mean_rate()
FTP_Sim	void ftp() float ftp_peak_rate() int ftp_ctl_cells()
Video_Sim	void video() float video_peak_rate() float video_mean_rate()
Telnet_Sim Rlogin_Sim	void telnet() void telnet_ctl_cells() float telnet_peak_rate() float telnet_mean_rate()
Mosaic_Sim	void mosaic() float mosaic_peak_rate() float mosaic_mean_rate()

Table 5.1: Distribution functions used by the workload model.

5.3 Cell-Generating Functions

5.3.1 Voice Traffic

A heuristic algorithm that uses the Markov Modulated Poisson Process ($MMPP$) model to approximate voice traffic is reported in [36, 74], where the key parameters for the $MMPP$ model were calculated from the real data gathered. In [36, 74], the two states in the $MMPP$ model were used to approximate the states for generating cells at both peak rates and mean rates. Our approach in this paper, however, is to use equation (5.1) for the generating function, $g_a(t)$. This function, $g_a(t)$, is based on an analysis for voice data. The following equation provides more accuracy than the above mentioned method and is used to generate voice traffic based on the data observed in Figure 4.12:

$$g_a(t) = \lambda_1 I_A(t) + \lambda_2 I_B(t) + [(\frac{\lambda_1 + \lambda_2}{2} + (\frac{\lambda_1 - \lambda_2}{2})\sin(\frac{t}{a} + b)]I_C(t) + [(\frac{\lambda_1 + \lambda_2}{2} + (\frac{\lambda_1 - \lambda_2}{2})\sin(\frac{t}{a} + b - \pi)]I_D(t), \tag{5.1}$$

$$\text{for } t = 0, ..., n,\ I_X(t) = \begin{cases} 1 & \text{if } t \in X \\ 0 & \text{if } t \notin X \end{cases},$$

where λ_1 is the mean rate, λ_2 is the peak rate, and $I_x(t)$ is the indicator function. Cell arrivals at mean rate and peak rate are modeled by $(\lambda_1 I_A(t) + \lambda_2 I_B(t))$, where $I_A(t)$ and $I_B(t)$ can be calculated from the measured data. After traffic has been filtered out at the mean rate and at the peak rate, we found that the rest of the traffic can be approximated by two sin() functions. The amplitudes of these sin() functions are adjusted according to λ_1 and λ_2. From Figure 4.12, we can derive $\lambda_1 = 400$, and $\lambda_2 = 600$ ($\triangle t = 0.1\ sec$). Values of a and b are obtained from traffic data: $a = 30, b = 3$,

$$\begin{cases} I_A = 1, & \text{if } \frac{t}{8} = 1, 3, 4, 6 \\ I_B = 1, & \text{if } \frac{t}{8} = 5 \\ I_C = 1, & \text{if } (\sin(\frac{t}{a} + b) > \sin(\frac{t}{a} + b - \pi)) \text{ and } \frac{t}{8} = 7, 0 \\ & \text{if } (\sin(\frac{t}{a} + b) \leq \sin(\frac{t}{a} + b - \pi)) \text{ and } \frac{t}{8} = 2 \\ I_D = 1, & \text{if } (\sin(\frac{t}{a} + b) \leq \sin(\frac{t}{a} + b - \pi)) \text{ and } \frac{t}{8} = 7, 0 \\ & \text{if } (\sin(\frac{t}{a} + b) > \sin(\frac{t}{a} + b - \pi)) \text{ and } \frac{t}{8} = 2. \end{cases}$$

5.3.2 FTP, Rlogin, Telnet, and Mosaic Traffic

Inputs to the function `ftp()` are *file_size*, *ftp_duration*, and $\triangle t$. This function first checks to see if *ftp_duration* is long enough to generate cells at peak rate for a given file size. It then partitions the *ftp_duration* into three intervals: the time period before sending any file when only control signals are sent, the actual file transfer time, and the time until the *ftp_duration* is complete. In order to simulate a transfer of a *file_size* $Kbyte$ file, `ftp()` takes the input parameters and generates cell streams every $\triangle t$ time slot for a total time of *ftp_duration*.

Unlike the CBR traffic (i.e., `audio`), which can be generated using a cell-generating function, VBR traffic (i.e., **FTP**, **video**, **Rlogin**, **Telnet**, and **Mosaic**)) is generated by applying the inverse transformation method [115] to the cumulative distribution functions. Because of the probability mass function (pmf) calculated from the discrete set of data points gathered, linear interpolation was used to improve the accuracy of the random variates.

In our traffic model, we do not distinguish **Rlogin_Sim** from **Telnet_Sim**. **Telnet_Sim** is invoked to simulate both applications. Inputs to the function **telnet()** are *telnet_duration* and $\triangle t$. To simulate **Telnet** traffic, **telnet()** repeatedly generates cell arrivals every $\triangle t$ time slot until the connection is torn down.

5.3.3 Video Traffic

Video traffic in our workload model is generated by using equation (5.2). This function, $g_v(t)$, is based on the data observed from Figure 5.3(a):

$$g_v(t) = \lambda_1 I_A(t) + \lambda_2 I_B(t) + \lambda_3 I_C(t) + \lambda_4 I_D(t), \tag{5.2}$$

$$\text{for } \ t = 0, ..., n, \ I_X(t) = \begin{cases} 1 & \text{if } t \in X \\ 0 & \text{if } t \notin X \end{cases},$$

where λ_1, λ_2, λ_3, and λ_4 are cell arrival rates, and $I_x(t)$ is the indicator function. Traffic generated by `video` can be approximated by using these four different arrival rates associated with their indicator functions. From Figure 5.3(a), $\lambda_1 = 327$, $\lambda_2 = 352$, $\lambda_3 = 423$, and $\lambda_4 = 846$. We have also obtained values for X_A, X_B, X_C, and X_D, for the indicator functions I_A, I_B, I_C, and I_D, respectively.

5.4 Testing the Artificial Workload Model

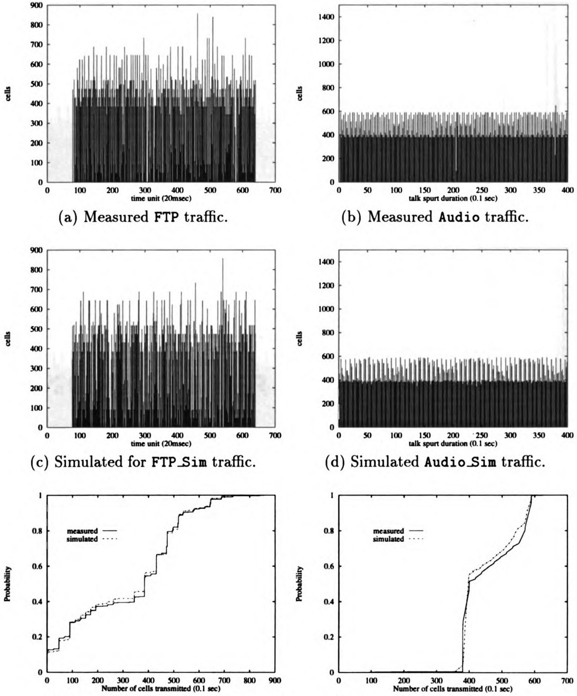
We validate our workload model by graphically comparing measured data with simulated data. A transfer of an 8 *Mbyte* file is simulated to represent **FTP** traffic. **Audio** traffic is generated for a duration of 60 *secs*, **Video** traffic for 16 *secs*, and **Telnet** traffic for 120 *secs*. The duration of the traffic gathered time is arbitrary chosen. Figures 5.2 and 5.3 graphically compare the measured traffic to that generated by the workload model. Each data point represents the number of cell arrivals for each time interval. Figures 5.2(a), and (b) and 5.3(a), and (b) are plots of the measured traffic, whereas Figures 5.2(c), and (d) and 5.3(c), and (d) are plots of the simulated traffic. It is clear from these observations that the measured traffic is similar to that generated by our workload model. Cumulative distributions for both measured traffic and simulated traffic are given in Figures 5.2(e), and (f) and 5.3(e), and (f). The similarity of these two cumulative distributions adds to the credibility of this workload model.

A *t*-**test** [115] of the confidence interval for the difference in mean performance confirms our observation that there is no significant difference between measured and simulated data. Results from the *t*-**test**, including the computation of confidence intervals, are reported in the following table.

t-**test**	`Audio`	`FTP`	`Video`	`Telnet/Rlogin`
$\bar{X}_a = \frac{1}{n_a}\sum_{i=1}^{n_a} X_{ia}$	$\bar{X}_a = 461.9$	$\bar{X}_a = 318.5$	$\bar{X}_a = 388.5$	$\bar{X}_a = 1.0585$
$\bar{X}_b = \frac{1}{n_b}\sum_{i=1}^{n_b} X_{ib}$	$\bar{X}_b = 449.8$	$\bar{X}_b = 313.0$	$\bar{X}_b = 385.7$	$\bar{X}_b = 1.0669$
$S_a = \left\{\frac{\sum_{i=1}^{n_a} X_{ia}^2 - n_a\bar{X}^2}{n_a-1}\right\}^{\frac{1}{2}}$	$S_a = 91.95$	$S_a = 212.42$	$S_a = 113.64$	$S_a = 4.6178$
$S_b = \left\{\frac{\sum_{i=1}^{n_b} X_{ib}^2 - n_b\bar{X}^2}{n_b-1}\right\}^{\frac{1}{2}}$	$S_b = 78.40$	$S_b = 210.05$	$S_b = 111.32$	$S_b = 4.8108$
$\bar{X}_a - \bar{X}_b$	13.55	5.54	2.8	−0.084
$S = \sqrt{\frac{S_a^2}{n_a} + \frac{S_b^2}{n_b}}$	6.04	12.59	3.56	0.1838
$\nu = \frac{(s_a^2/n_a + s_b^2/n_b)^2}{\frac{1}{n_a+1}(\frac{s_a^2}{n_a})^2 + \frac{1}{n_b+1}(\frac{s_b^2}{n_b})^2} - 2$	780	1125	3998	2622
$(\bar{x}_a - \bar{x}_b) \mp t_{[1-\alpha/2;\nu]}s$	$[-6.8, 33.9]$	$[-36.9, 48.0]$	$[-9.2, 14.8]$	[-0.6, 0.6]

Table 5.2: *t*-**test** for measured and simulated data. ($\alpha = 0.001$, n_a: the sample size of measured data, n_b: the sample size of simulated data, ν: degrees of freedom.)

There are limitations to our workload model too. Currently we can only generate ftp and SunVideo traffic with a time interval that is a multiple of $\triangle t$, the accuracy of the traffic monitored interval. For example, as we have explained in section 4.2, the smallest value of $\triangle t$ is 100 *nanoseconds*, hence we can only generate ftp and



(a) Measured FTP traffic.

(b) Measured Audio traffic.

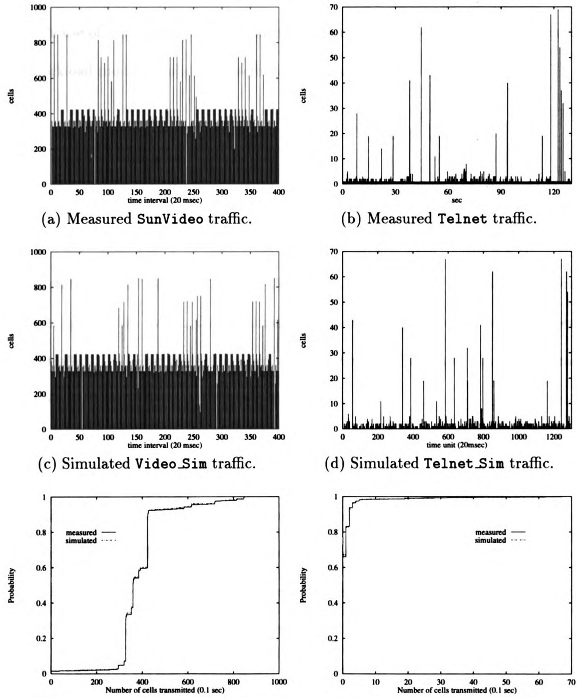
(c) Simulated for FTP_Sim traffic.

(d) Simulated Audio_Sim traffic.

(e) Comparison of FTP traffic.

(f) Comparison of Audio traffic.

Figure 5.2: Comparison of measured traffic to simulated traffic (FTP, Audio).



(a) Measured SunVideo traffic.

(b) Measured Telnet traffic.

(c) Simulated Video_Sim traffic.

(d) Simulated Telnet_Sim traffic.

(e) Comparison of Video traffic.

(f) Comparison of Telnet traffic.

Figure 5.3: Comparison of measured traffic to simulated traffic (Video, Telnet).

SunVideo traffic every 20 μsec, 40 μsec, 60 μsec, ..., and so on. We do not consider correlations and autocorrelations in our traffic model. Our future work will consider a possible use of self-similar model to capture such relations. Despite of its limitations, our workload model is valid and useful for ATM network simulation.

5.5 Summary

In this chapter, we have presented a procedure for building a workload model for an ATM LAN/WAN. This procedure makes use of measured data obtained from WAN traffic statistics and from our ATM LAN testbed. Such a model consists of different types of application traffic. Cell-generating functions are presented to simulate measured traffic. By using these functions, representative ATM LAN/WAN traffic can be generated to develop a workload model.

Traffic is generated from the workload model and the cumulative distributions are obtained to verify the credibility of our workload model. From Figures 5.2 and 5.3, it is clear that the data generated by the workload model is very similar to the simulated data. Credibility of this model can be observed by graphically comparing the traffic characteristics and the cumulative distributions of the measured traffic to that of the simulated traffic. The ***t*-test** illustrates that the difference between the simulated and the measured data is not significant. This model is used to study the performance results of our CAC framework, presented in the next chapter.

Chapter 6

Performance of the Call Admission Control Framework

6.1 Introduction

The main goal of this chapter is to describe how the network performs under the supervision of our call admission control framework. Performance evaluation of such a framework is conducted using a simulation approach. The major advantage of a simulation methodology is its controllable environment (e.g., various adjustable parameters conducted in simulation runs). In the particular case of video traffic, an individual workstation can only support one video connection due to hardware limitation, and this will not generate enough video traffic to investigate network performance. Therefore, traffic is generated by using a traffic model derived from empirical traffic measurements [74, 116], as discussed in chapter 4, to generate data, voice, and video traffic.

Various simulations are performed to study the performance of the framework based on both call and cell level. The traffic model consists of various applications (i.e., `FTP`, `Audio`, `Video`, `Telnet/Rlogin`, and `Mosaic`) where a link is modeled as 155 *Mbps*. Performance of the proposed call admission algorithm is compared with that of the peak rate allocation scheme. We test our call admission control algorithm on both *homogeneous* and *heterogeneous* traffic streams.

A simple FCFS discipline is assumed for each individual queue in our call admission control framework. We will describe how network bandwidth can be efficiently allocated to various classes of traffic streams by dynamically measuring current network loads. Mathematical analysis of the queuing behavior under multimedia traffic streams is not tractable. This is due to the fact that we do not make any assumption about the arrival traffic, since traffic streams are generated according to the models developed in the previous chapter. Nevertheless, efforts are made to provide mathematical bounds on different classes of queues. Hence, both analytical and simulation approaches are used to study the performance of the framework, with emphasis on simulation results.

6.2 Performance Metrics

First we defined a term, *efficiency*, ε_X, of the reserved bandwidth for a call admission control algorithm X as the ratio of the *consumed bandwidth* to the *reserved bandwidth.* A network reserves a certain amount of bandwidth for the current connections (i.e., reserved bandwidth), but the actual bandwidth usage (i.e., consumed bandwidth) is different. In a normal network (e.g., a network without faulty sources or malicious users), the consumed bandwidth is less than or equal to the reserved bandwidth. Hence, the value of efficiency is less than or equal to 1.

Next, we define the *efficiency ratio* of the call admission control algorithm as $\frac{\epsilon_X}{\epsilon_{PRA}}$, where ϵ_X is the efficiency of the reserved bandwidth of call admission control algorithm X, and ε_{PRA} is the efficiency of the reserved bandwidth of the peak rate allocation scheme. Since the proposed call admission control scheme is more efficient than the peak rate allocation scheme, the efficient ratio is greater than or equal to 1. In the rest of the chapter, call-blocking probability, bandwidth utilization, and bandwidth efficiency are performance metrics for the call admission control scheme.

6.3 Homogeneous Traffic Streams

In order to gain more insight into the behavior of individual applications and the call admission control procedure, ***homogeneous*** traffic streams are used to test the algorithm. Such streams belong to the same type of application, α. Hence, the maximum number of connections that the network can support in order to guarantee cell level QoS is N_α. It should be noted that $N_\alpha = L/\theta_\alpha P_\alpha$, where L is the link bandwidth capacity, $\theta_\alpha = 1$, and P_α is the peak rate for connection α. A call α will not be blocked if it makes a call request when the system has x number of connections, where $x < N_\alpha$. This can be observed in Figure 6.1 for both pure video streams and pure voice streams.

The network starts to reject calls when the number of connections x, is greater

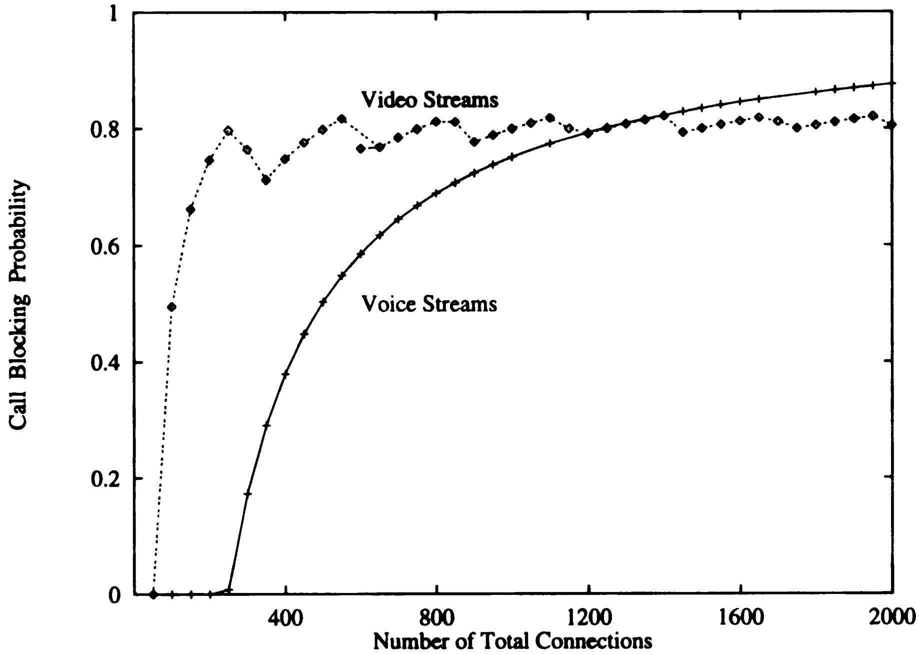


Figure 6.1: Performance results of the proposed call admission control algorithm ($\lambda_{video} = 5\ calls/sec$, $\lambda_{voice} = 50\ calls/sec$).

than or equal to N_α, for homogeneous sources (i.e., audio sources or video sources), because the guaranteed services reserve a bandwidth equal to their peak rates. Hence the call-blocking probability increases rapidly. For homogeneous audio streams, the call-blocking probability grows steadily because of the short connection times and small peak bandwidth requirements. The call-blocking probability for the pure video network oscillates between 0.7 and 0.8. A decrease in the call-blocking rate in this range is caused by the termination of existing connections. The higher blocking probabilities for the homogeneous video streams are caused by the longer connection lifetime and the higher peak bandwidth requirements, whereas the higher blocking rates for the homogeneous audio streams are attributed to the increased arrival rate (i.e., the arrival rate for audio traffic is 50 $calls/sec$ and the arrival rate for video traffic is 5 $calls/sec$).

Experiments were conducted to study the effects of call arrival rates. Let T_α be the connection duration and λ_α be the arrival rate. A call is accepted if λ_α and T_α are under the constraint $\lambda_\alpha T_\alpha \leq N_\alpha$. For the same call arrival rates, a video network has a much higher call-blocking probability than a voice network does, due to a much larger peak rate requirement. The call-blocking probability for voice stream decreases significantly for a light loaded network (i.e., large call arrival intervals), which is caused by a short connection time and a small peak rate for voice traffic.

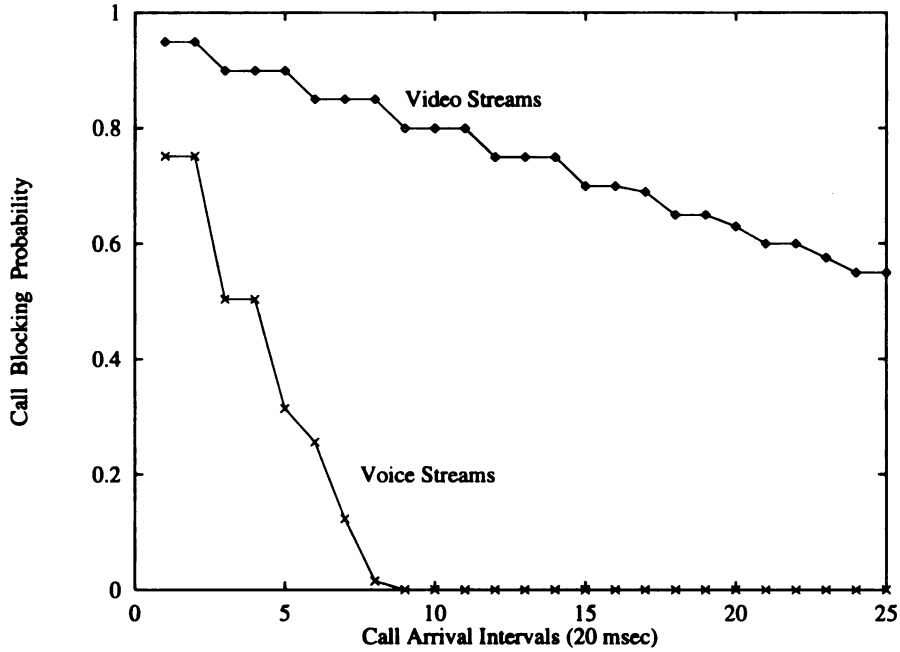
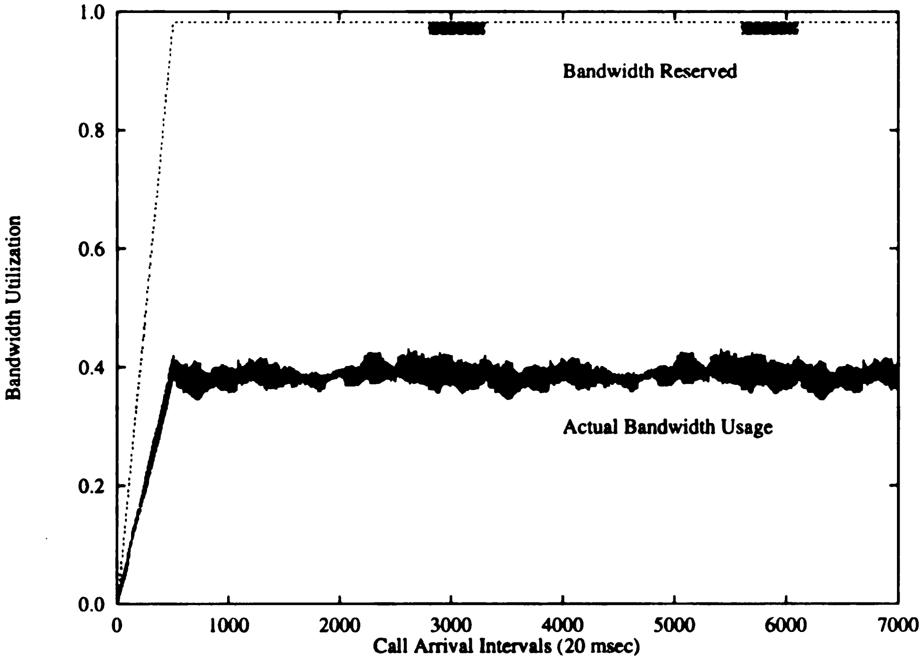
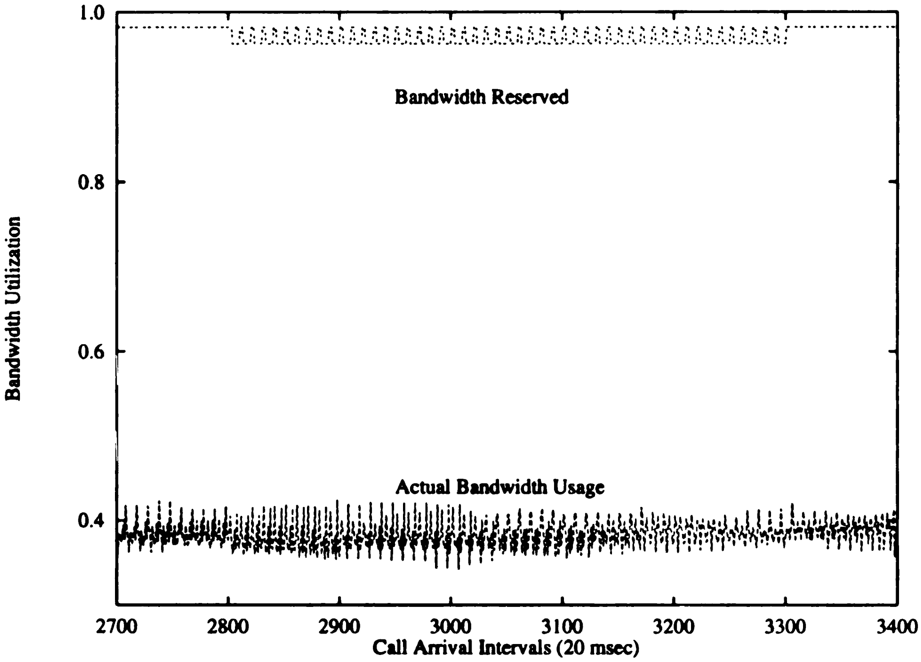


Figure 6.2: Call-blocking probability as a function of the number of arriving calls, where total number of connections is 1,000.

Figures 6.3 and 6.4 trace the bandwidth utilization for video and audio streams, respectively. The reserved bandwidth in Figure 6.3(b) has an oscillation effect because the terminations of previous connections enable the following new call requests. Since the video traffic is also bursty, the result of the aggregated video streams is also bursty (refer to actual bandwidth usage in Figure 6.3(b)).



(a) A longer trace of bandwidth utilization.



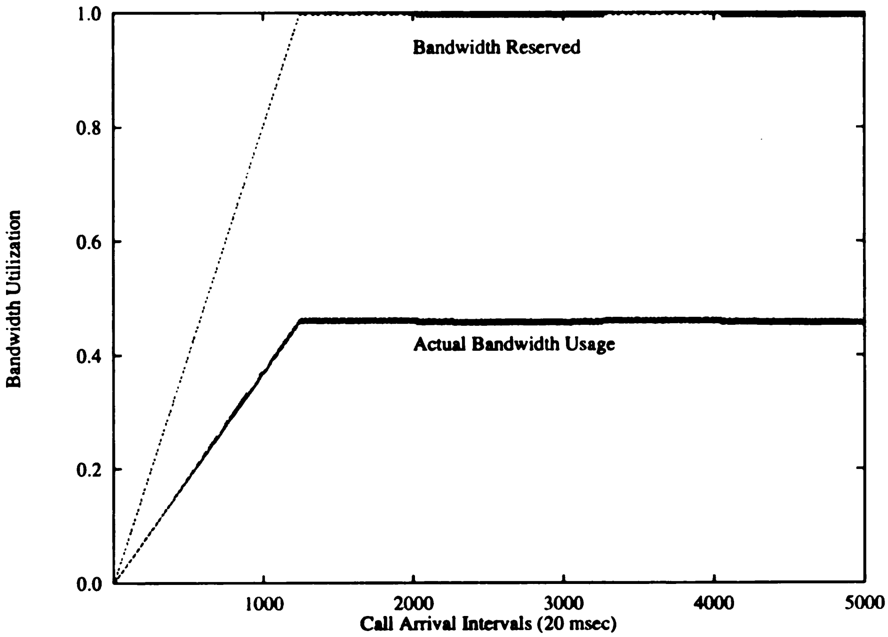
(b) A shorter trace of bandwidth utilization.

Figure 6.3: A trace of bandwidth utilization for pure video streams (total number of connections is 1,000, $\lambda = 5\ calls/sec$).

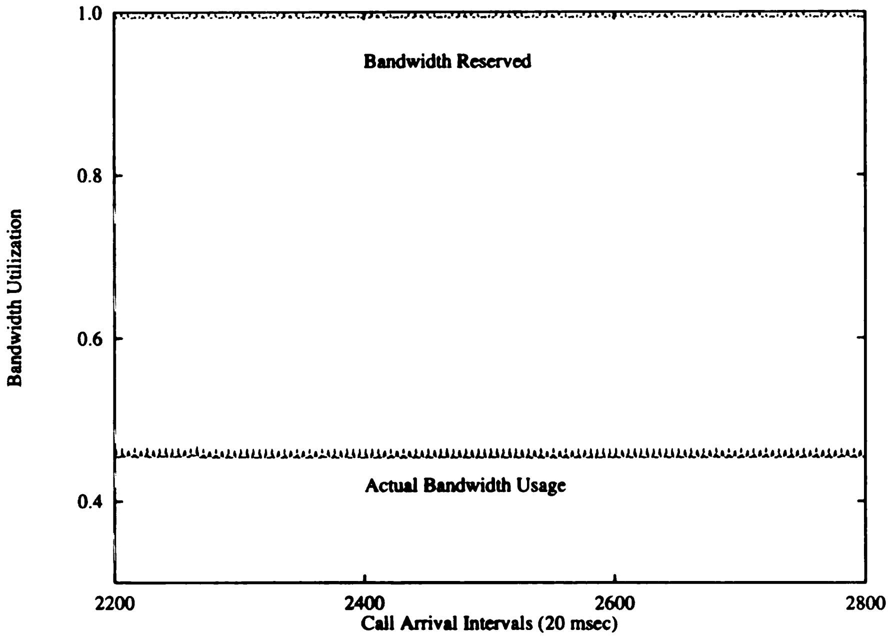
The audio traffic gathered from our ATM testbed is CBR traffic and the actual bandwidth usage approximates a constant value. In other words, the aggregated audio stream is less bursty than the aggregated video streams. As observed from Figures 6.3 and 6.4, in a homogeneous audio and video network, the actual bandwidth utilization is greater than 40% of the network link capacity. Also, an audio network can achieve higher bandwidth usage, compared to a video network.

Bandwidth efficiencies for both networks are shown in Figure 6.5. As discussed in chapter 4, the value of burstiness for audio traffic is the smallest among the three different types of applications (i.e., data, video, and voice), regardless of which definition of burstiness is used. The fact that the degree of burstiness for audio traffic is small gives a fairly stable rate of bandwidth efficiency (refer to Figure 6.5). As shown in Figure 4.12(b), the audio traffic needs a fixed amount of bandwidth, and this is typically less than its peak rate. That is, the efficiency for audio traffic is always less than 50%, regardless of the number of concurrent audio connections. Such characteristics enables the network to allocate bandwidth less than the peak rate, regardless of the number of audio connections, which in turn will increase bandwidth efficiency.

The amount of bandwidth needed by video traffic can not be easily predicted. This can be observed when the number of video connections is small (e.g., at the beginning of the simulation). Figure 6.5 shows an efficiency of more than 80% for a single video connection. When the number of video connections is greater than 5 (i.e., at simulation time t, $t > 50$), the bandwidth efficiency is always less than 50%. This means that to provide cell-level QoS for video traffic, peak rate is needed for a single video connection, and only 50% of the aggregated peak rate is needed for

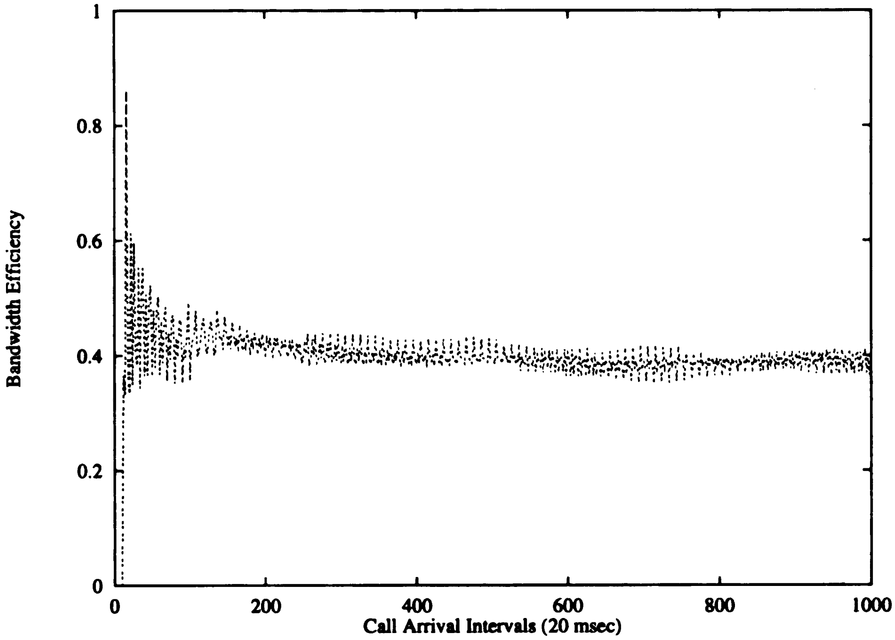


(a) A longer trace of bandwidth utilization.

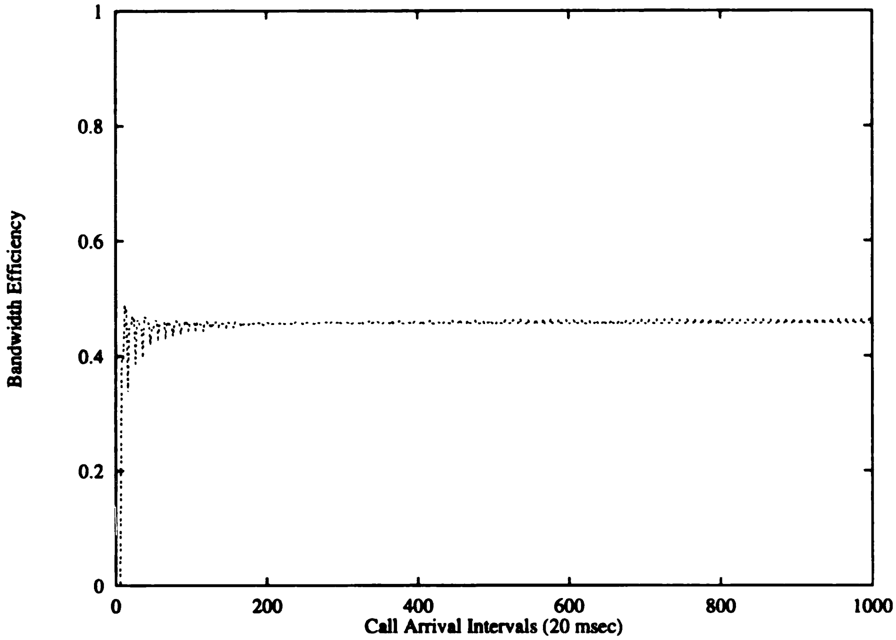


(b) A shorter trace of bandwidth utilization.

Figure 6.4: A trace of bandwidth utilization for homogeneous `audio` streams (total number of connections is 1,000, $\lambda = 10$ *calls/sec*).

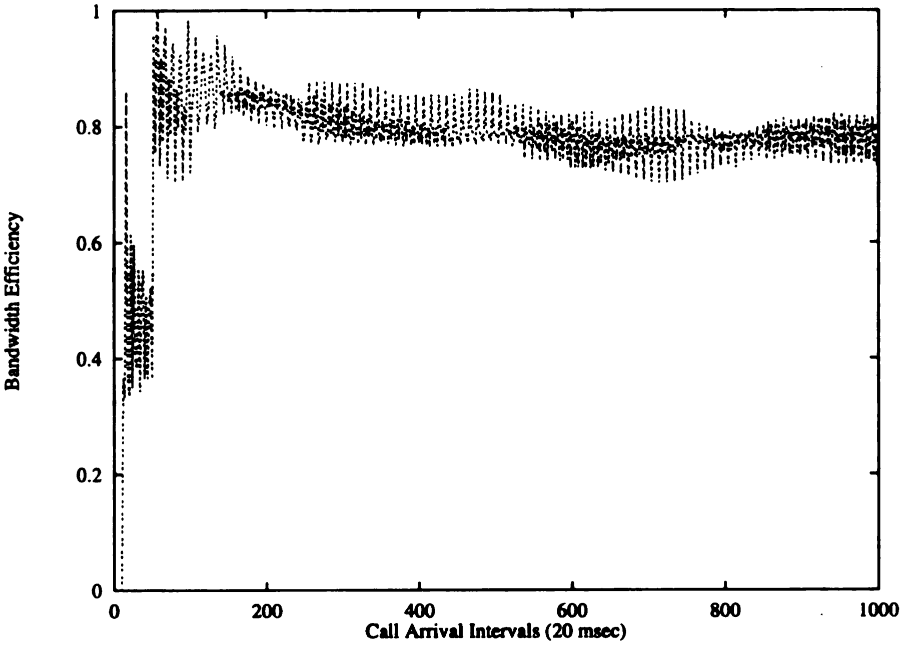


(a) Bandwidth efficiency for video streams ($\lambda = 5$ *calls/sec*, $\theta = 1.0$).

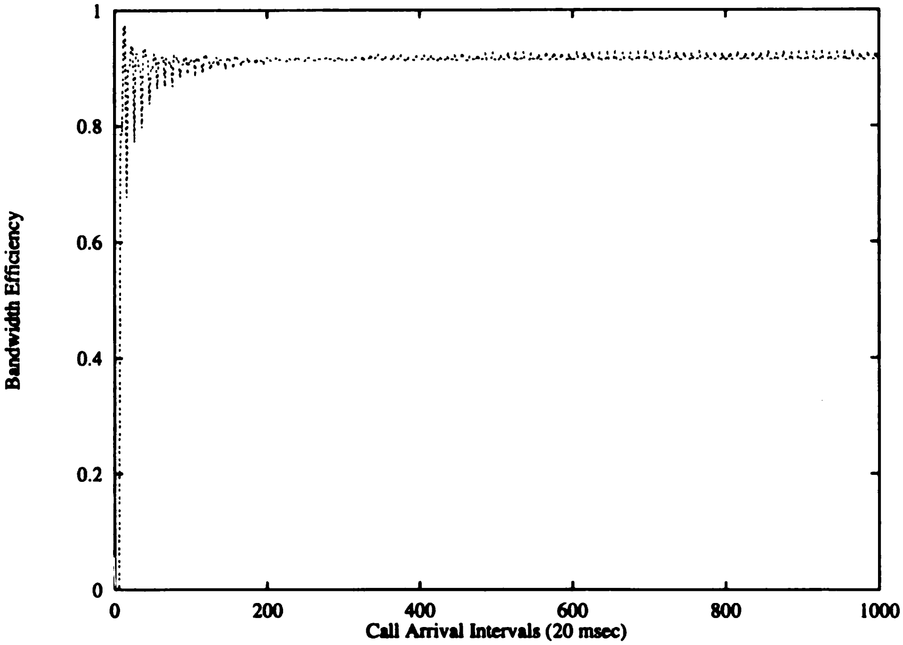


(b) Bandwidth efficiency for audio streams ($\lambda = 10$ *calls/sec*, $\theta = 1.0$).

Figure 6.5: A trace of bandwidth efficiency for homogeneous traffic streams (total number of connections is 1,000).



(a) Bandwidth efficiency for video streams ($\lambda = 5$ *calls/sec*).



(b) Bandwidth efficiency for audio streams ($\lambda = 10$ *calls/sec*, $\theta = 0.5$).

Figure 6.6: A trace of bandwidth efficiency for homogeneous traffic streams (total number of connections is 1,000).

multiplexed video connections, when the number of connections is greater than 5. Figure 6.6 shows an example of higher bandwidth efficiency by having a smaller value of θ.

6.4 Heterogeneous Traffic Streams

For *heterogeneous* traffic streams, different types of traffic streams enter the network to request connections. Figure 6.7 illustrates multiple connections for different types of traffic, where traffic streams consist of a uniform distribution of **FTP**, **Audio**, **Video**, **Telnet/Rlogin**, and **Mosaic**. It further illustrates the better performance from the proposed call admission algorithm compared to the peak rate allocation scheme. Both schemes have call-blocking probabilities associated with an arrival rate ($\lambda = 5$ *calls/sec*), regardless of the total number of connections. With 50% guaranteed traffic mixes, the proposed call admission control algorithm has ≈ 15% call-blocking probability and the peak rate allocation scheme has ≈35% call-blocking probability. With a fixed arrival rate the system is in a steady state, so the call-blocking probability remains stable. Although the algorithm gains more than 20% in network utilization, it should be noted that the algorithm does significantly better (i.e., has a lower call-blocking probability) with a lower guaranteed traffic mix.

Since the total number of connections does not affect the call-blocking probability, the other simulation results were performed using $1,000$ connections in the simulation. Note that in the default configurations of a FORE ATM network, a host can only have approximately 250 virtual connections open simultaneously. Figure 6.8 depicts

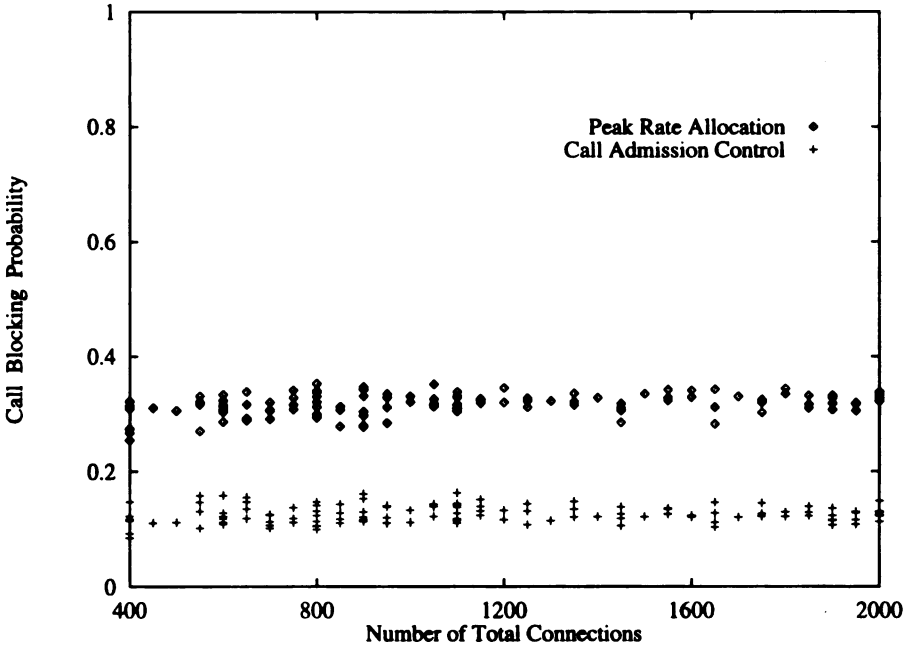


Figure 6.7: Performance results of the proposed call admission control algorithm ($\lambda = 5\ calls/sec$).

the call-blocking probability as a function of call interarrival intervals, given in time units of 20 milliseconds. Again, the proposed algorithm has a lower call-blocking probability. The difference is more significant when the traffic load is high (i.e., for small call arrival intervals). The peak rate allocation scheme causes higher call-blocking probabilities with smaller call interarrival intervals. Reserving peak rates for all existing connections that do not fully utilize the reserved bandwidth will result in a high call rejection rate. This shows that by dynamically measuring network loads at call setup time, the call-blocking probability is significantly reduced and link utilization is increased.

In order to understand the effects of guaranteed services, experiments were also conducted to study different traffic mixes. Figure 6.9(a) plots call-blocking probabilities for different ratios of guaranteed traffic. In these simulation runs, `Audio` and

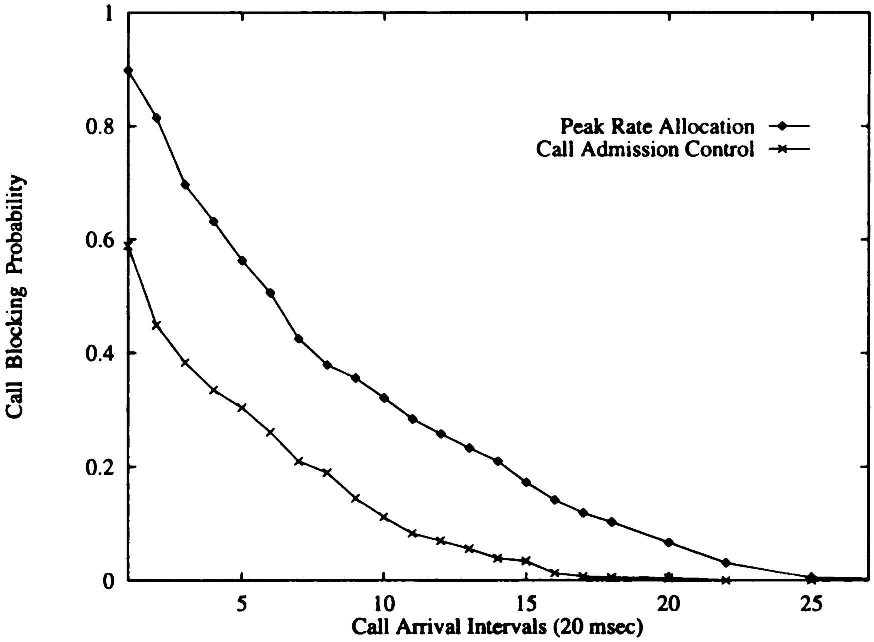


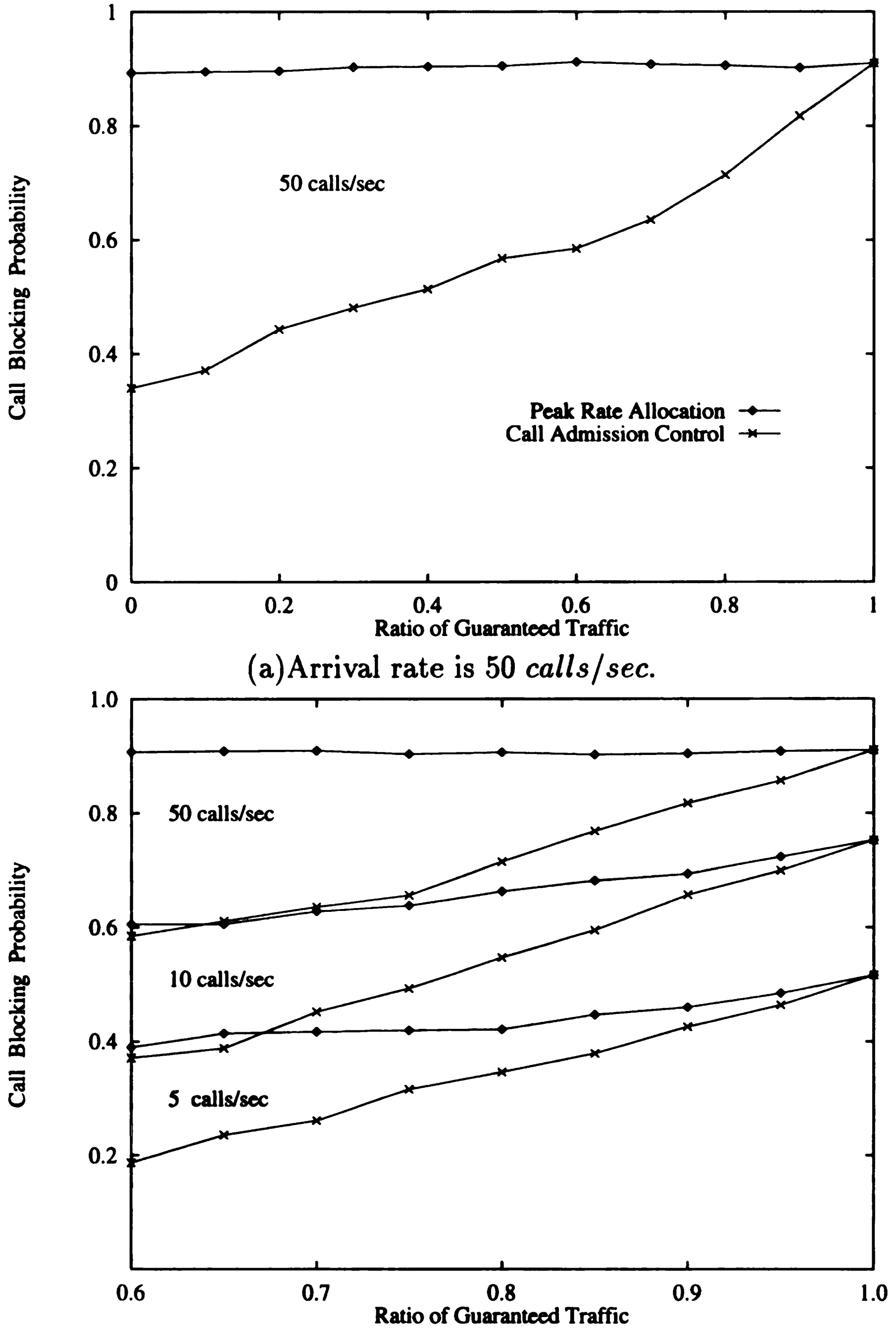
Figure 6.8: Call-blocking probability as a function of the number of arriving calls.

Video require guaranteed services, while **Ftp**, **Telnet/Rlogin**, and **Mosaic** require ABR service. The peak rate allocation scheme, which allocates peak rates to all admitted connections, is not affected by the ratio of guaranteed traffic, because such a scheme does not distinguish between different traffic classes. It allocates peak rates to the admitted connections.

Our algorithm results in a very low call-blocking probability when the fraction of guaranteed traffic is low. Increasing the connections for guaranteed services will reserve more network bandwidth and thus increase the call-blocking probability. Note that when the network consists of all guaranteed traffic (refer to Figure 6.9 where the ratio is 1), the algorithm has the same call-blocking probability as that of the peak rate allocation scheme. If all traffic streams require guaranteed services, the mechanism allocates peak rates to them, thus making it basically a peak rate allocation scheme. For other traffic mixes, the algorithm has better performance.

Figure 6.9(b) shows that similar results hold under different call arrival rates ($5, 10$, and 50 $calls/sec$). For easy visibility, only heavy mixes of guaranteed traffic are plotted. The performance of a peak rate allocation scheme is not affected by the fraction of guaranteed traffic. However, smaller call arrival rates give lower call-blocking probabilities. The performance of the call admission control is better for lower call arrival rates or lower ratios of guaranteed traffic.

A trace of bandwidth utilization is also performed. Figure 6.10 shows the bandwidth reserved (B_PRA and B_CAC) as well as the bandwidth consumed (U_PRA and U_CAC) by the two algorithms, where the simulation time is shown from time $1,000$ to $2,500$ ($1,000\Delta T \leq t \leq 2,500\Delta T$ and $\Delta T = 20msec$). As shown in Figure 6.10(a), 50% of the arriving calls request guaranteed service and 25% of the arriving calls request guaranteed service in Figure 6.10(b). Generally speaking, the peak rate allocation scheme reserves far more bandwidth than does the call admission algorithm presented here, whereas the call admission control algorithm provides more bandwidth usage. In other words, our algorithm has a higher efficiency of bandwidth utilization. The proposed algorithm reserves approximately the same amount of bandwidth as that consumed when $t < 1,000\Delta T$. As more traffic enters the network when $t > 1,000\Delta T$, the proposed algorithm allows for more bandwidth usage. For some time instances ($t = 2,000\Delta T$ in Figure 6.10(a) and $t = 3,000\Delta T$ in Figure 6.10(b)), the peak rate allocation scheme reserves more bandwidth than the proposed algorithm ($B\_PRA > B\_CAC$). This is caused by previously accepting a large call, which then results in rejecting small calls later. Even so, the actual bandwidth usage of the algorithm is still better than that of the peak rate allocation



(a)Arrival rate is 50 *calls/sec.*

(b)Comparison of different arrival rates.

Figure 6.9: Call-blocking probability as a function of the ratio of guaranteed traffic.

sche

ma

Fi

sc

5

i

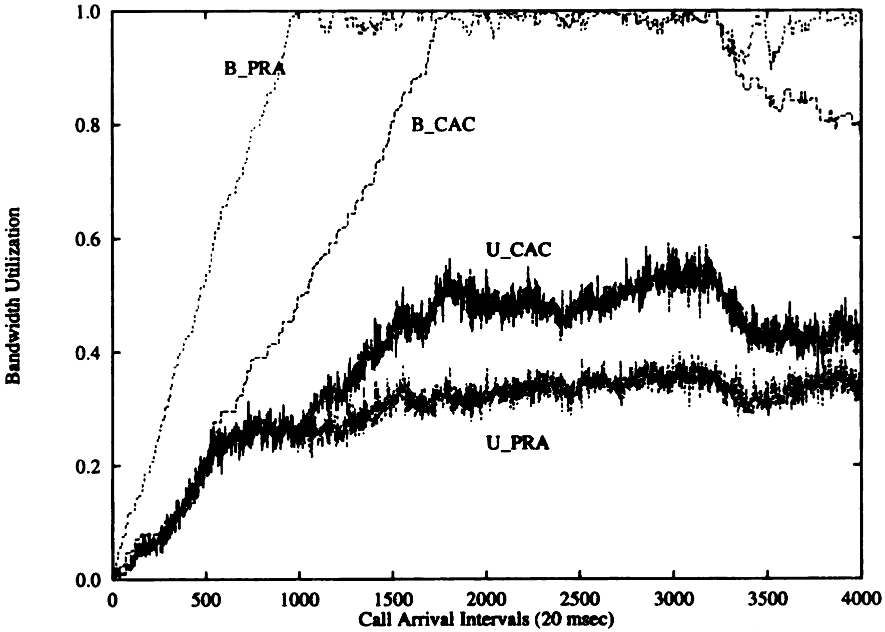
i

scheme (refer to Figure 6.10 where $U\_CAC > U\_PRA$).

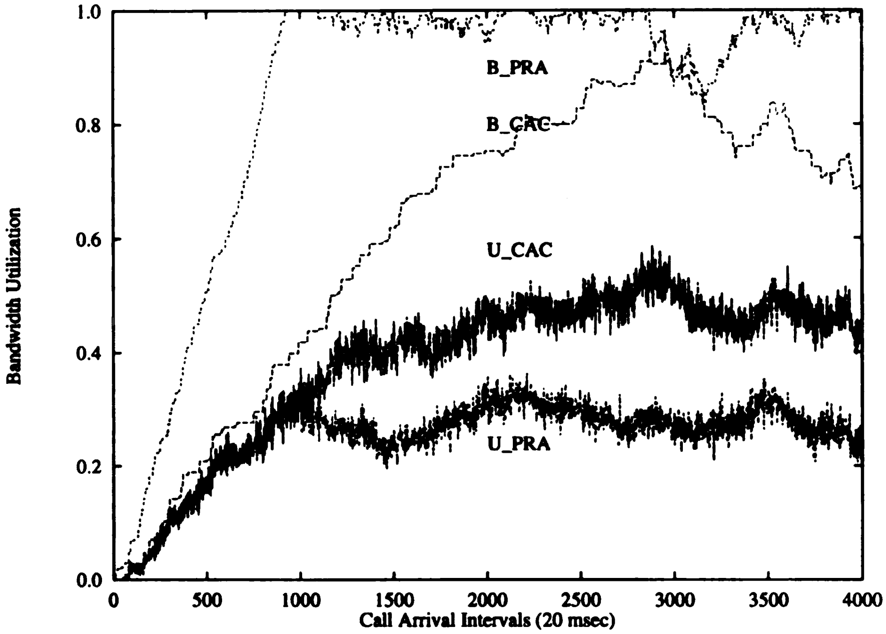
Figure 6.11 shows the efficiencies of the reserved bandwidth. This observation was made by exercising both algorithms (refer to Figures 6.10(a) and (b)). It is clear from Figures 6.11(a) and (b) that the performance of the proposed call admission control scheme is better than that of the peak rate allocation algorithm for both 25% and 50% guaranteed traffic arrivals. With the 25% guaranteed traffic arrival, this increase in performance is more evident. Hence, efficiencies of both algorithms can also be increased by having more traffic that requires ABR and best-effort services. At the beginning of the simulation, the efficiency is close to 100%. This is more noticeable in Figure 6.11(a), where more connections are requesting guaranteed services. As more guaranteed connections occur, the efficiency decreases rapidly. The decrease in the efficiency is caused by the long connection time of the guaranteed traffic, which reserves a peak rate and then only consumes it for a short time period.

Recall that the efficiency ratio is a comparison of the efficiency of the reserved bandwidth utilization of the call admission control algorithm to that of the peak rate allocation scheme. Figure 6.12 shows that the efficiency ratio of the call admission control is approximately $1.5 - 3$ times that of the peak rate allocation. When the traffic load is high (e.g., at the beginning of the simulation), the system has a higher efficiency ratio. When the network consists of less guaranteed traffic, the ratio is higher. This illustrates that non-guaranteed traffic can make use of network bandwidth, and therefore increase utilization.

As shown in Figure 6.5, a network with 100% guaranteed traffic has $\approx 50\%$ bandwidth efficiency. Sources requesting guaranteed services can be multiplexed beyond

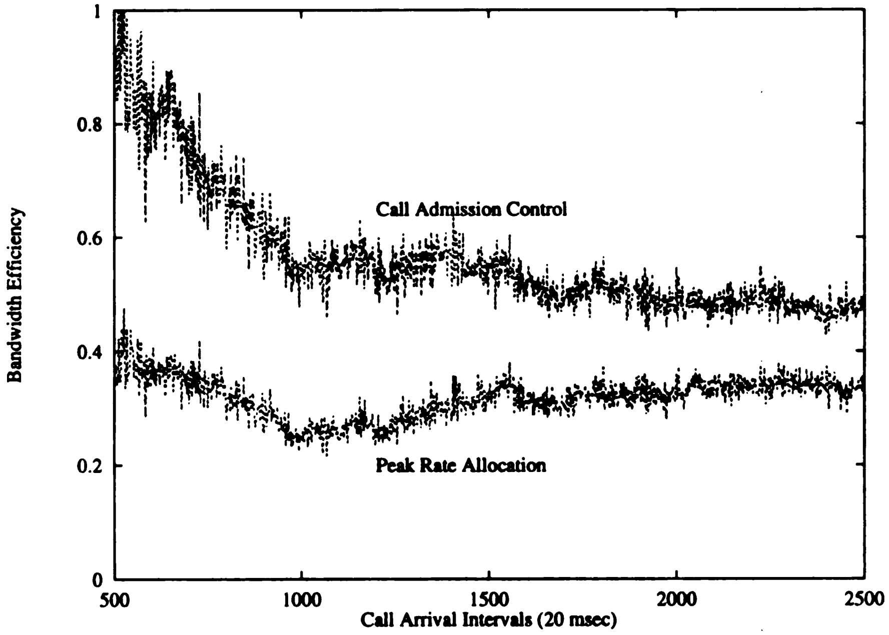


(a)Guaranteed services requested by 50% of arrivals.

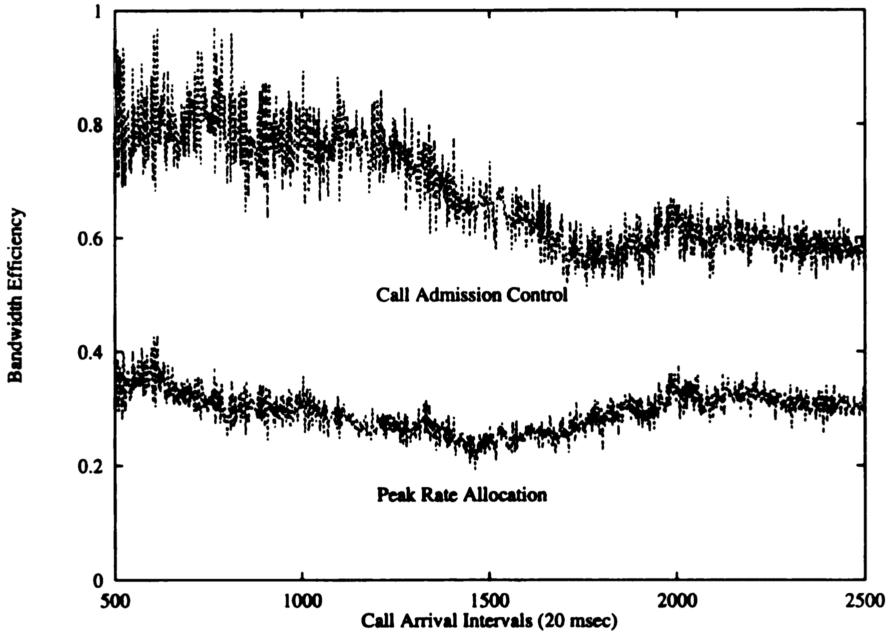


(b)Guaranteed services requested by 25% of arrivals.

Figure 6.10: A trace of bandwidth usage in a simulation run. B_PRA: bandwidth reserved by peak rate allocation. B_CAC: bandwidth reserved by the call admission control. U_PRA: bandwidth utilization for peak rate allocation. U_CAC: bandwidth utilization for the call admission control.



(a)Guaranteed services requested by 50% of arrivals.



(b)Guaranteed services requested by 25% of arrivals.

Figure 6.11: A trace of efficiency of the reserved bandwidth by the call admission algorithms in a simulation run.

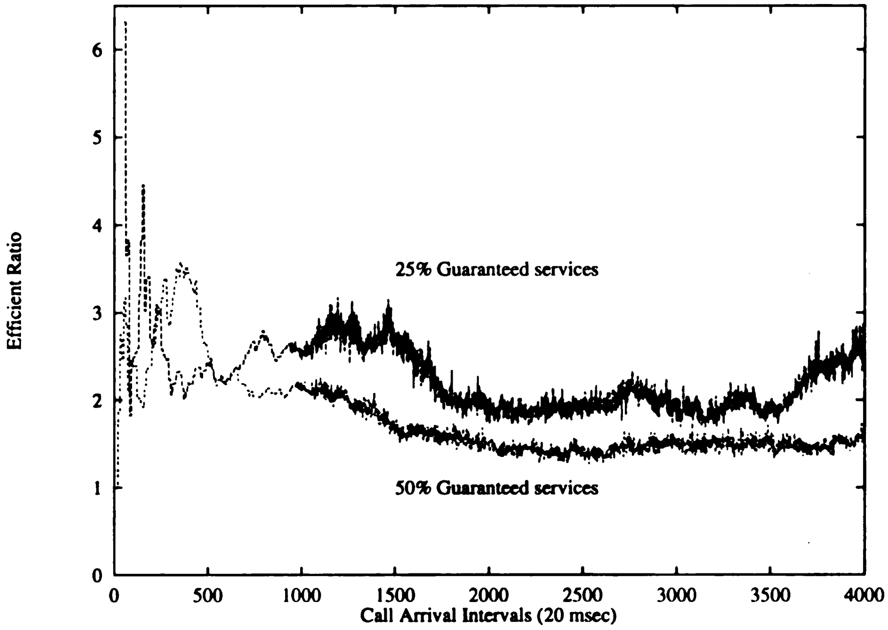


Figure 6.12: Efficiency ratio of the call admission control.

their peak rate to increase network utilization.

6.5 Analysis of the Scheduling Algorithm

For the scheduling algorithm, the maximum delay time slot is bounded for a chosen time-out value. In order to analyze the maximum queuing delay for a cell c joining a particular queue, we always consider the worst case. The delay time slot, D_i, is defined as the interval from the time the cell enters the queue, q_i, to the time the transmission begins. Let T_L be a cycle time for processing all queues according to the link speed L, then $\sum_{all\, i} T_i = T_L$. Here T_i and T_L are in time units for processing a single cell.

General case:

The maximum queuing delay, D_1, for the first queue occurs when a cell c joins the end of the first queue and the other queues are full. Let B_1 be the size of the first queue, and let the time duration that c has to wait in the queue before it is served be B_1, when no time-out from the other queues occurs. During the time B_1, the number of time-outs that occur from the other queues is $(\lfloor \frac{B_1}{T_1} \rfloor + 1)$; hence, the time needed to process other queues is $(\lfloor \frac{B_1 - 1}{T_1} \rfloor + 1)(\sum_{i \neq 1} T_i)$. Finally, the server needs B_1 amount of time to serve the rest of the first queue. From the above description, it follows that the delay for the first queue is:

$$\begin{aligned} D_1 &= (\lfloor \frac{B_1}{T_1} \rfloor + 1)(\sum_{i \neq 1} T_i) + B_1 \qquad (6.1) \\ &\leq (\lfloor \frac{B_1}{T_1} \rfloor + 1)(\sum_{i \neq 1} T_i) + (\lfloor \frac{B_1}{T_1} \rfloor + 1)T_1 \\ &= (\lfloor \frac{B_1}{T_1} \rfloor + 1)(\sum_{i \neq 1} T_i + T_1) \\ &= (\lfloor \frac{B_1}{T_1} \rfloor + 1)T_L, \end{aligned}$$

where T_i is the time-out value of the ith queue.

Similarly, in a general case the maximum queuing delay, D_i, for the ith queue occurs when a cell c joins the end of the ith queue and the other queues are full. Let B_i be the size of the ith queue. The time needed to process other queues due to time-outs is $(\lfloor \frac{B_i}{T_i} \rfloor)(\sum_{j \neq i} T_j)$. It also needs to wait $(\sum_{j \neq i} T_j)$ before it can serve the rest of that queue. Therefore, the delay for the ith queue is:

$$D_i = \lfloor \frac{B_i}{T_i} \rfloor (\sum_{j \neq i} T_j) + (\sum_{j \neq i} T_j) + B_i \qquad (6.2)$$

$$
\begin{aligned}
&\leq (\lfloor\frac{B_i}{T_i}\rfloor + 1)(\sum_{j\neq i} T_j) + (\lfloor\frac{B_i}{T_i}\rfloor + 1)T_i \\
&= (\lfloor\frac{B_i}{T_i}\rfloor + 1)(\sum_{j\neq i} T_j + T_i) \\
&= (\lfloor\frac{B_i}{T_i}\rfloor + 1)T_L
\end{aligned}
$$

From equation (6.1), if $T_i = 1$ for all i, (i.e., cells in different queues are processed in a round-robin fashion where only a single cell gets service for each queue), then $\lfloor\frac{B_1}{T_1}\rfloor = B_1$, and the delay for the first queue is:

$$
\begin{aligned}
D_1 &= (B_1 + 1)(\sum_{i\neq 1} T_i) + B_1 \\
&= (B_1 + 1)(\sum_{i\neq i} T_i + 1) - 1 \\
&= (B_1 + 1)n - 1, \qquad (6.3)
\end{aligned}
$$

so,

$$D_1 < (B_1 + 1)n, \qquad (6.4)$$

where n is the number of different classes of queues. Similarly, for equation (6.2), if $T_i = 1$ for all i, then $D_i < (B_i + 1)n$. Hence, $D_i < (B_i + 1)n$ for all i.

Special case:

For the case $T_1 > B_1$, cells in the first queue are served before time-out occurs, then $\lfloor\frac{B_1}{T_1}\rfloor = 0$. Hence, equation (6.1) becomes:

$$
\begin{aligned}
D_1 &= \sum_{i\neq 1} T_i + B_1 \qquad (6.5) \\
&< \sum_{i\neq 1} T_i + T_1 \\
&= T_L.
\end{aligned}
$$

Similarly, for equation (6.2), if $T_i > B_i$ for all i, then $D_i < T_L$. More precisely, since $T_i > B_i$, the maximum time that is needed to serve the ith queue is $\tilde{T}_i$, which is also equal to B_i, so equation (6.5) becomes:

$$\begin{aligned} D_1 &= (\sum_{i \neq 1} T_i) + B_1 \\ &= (\sum_{i \neq 1} \tilde{T}_i) + B_1 \\ &= (\sum_{i \neq 1} B_i) + B_1 \\ &= \sum_{all\ i} B_i, \end{aligned} \tag{6.6}$$

which occurs when the scheduler processes each queue sequentially. Hence, $D_i = \sum_{all\ i} B_i$. Each queue is thus served until it is empty before the scheduler goes to another queue.

If all T_i are equal, this is the case similar to Time Division Multiplexing (TDM). Equation (6.1) becomes the following, assuming $T_i = T$ for all i:

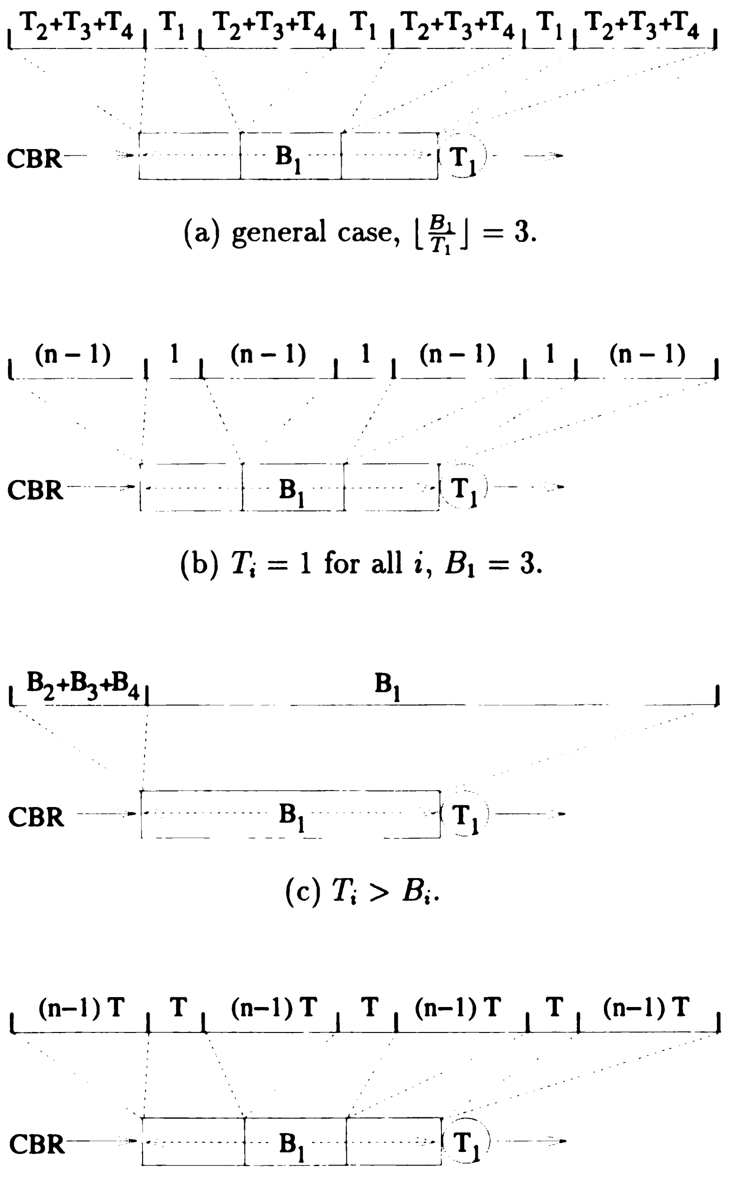
$$\begin{aligned} D_1 &= (\lfloor \frac{B_1}{T} \rfloor + 1)(\sum_{i \neq 1} T_i) + B_1 \\ &= (\lfloor \frac{B_1}{T} \rfloor + 1)(n-1)T + B_1 \\ &\leq (B_1 + 1)(n-1) + B_1 \\ &= (B_1 + 1)n - 1 \\ &\leq (B_1 + 1)n \end{aligned} \tag{6.7}$$

Applying $T_i = T$ for all i to equation (6.2), $D_i \leq (B_i + 1)n$.

It can be observed from the above equations that the maximum queuing delay for a single cell in the first queue is bounded by the queue size. The maximum queuing delay is bounded by equation (6.2). Note that T_L is a constant value determined by the network link speed. For a given (T_i, B_i) pair, D_i is a bounded value. An advantage of the RR/T scheme is that the bound of D_i for a particular queue, q_i, depends on the local time-out value, T_i, which makes it easy to adjust the time-out value for the real-time traffic in order to satisfy its stringent delay requirements. The bounds computed above represent the worst case for a single cell. Table 6.1 summarizes the queuing delay under different conditions. Figure 6.13 depicts examples of the worst-case queuing delays for CBR cells. The queuing delays for VBR cells are similar.

	Queuing Delay for Other Queues
General case	$D_i \leq (\lfloor \frac{B_i}{T_i} \rfloor + 1)T_L$
$T_i = 1$ for all i	$D_i < (B_i + 1)n$
$T_i > B_i$	$D_i = \sum_{all\ i} B_i < T_L$
$T_i = T$ for all i	$D_i \leq (B_i + 1)n$

Table 6.1: Summaries of the queuing delay.



(a) general case, $\lfloor \frac{B_1}{T_1} \rfloor = 3$.

(b) $T_i = 1$ for all i, $B_1 = 3$.

(c) $T_i > B_i$.

(d) $T_i = T$ for all i, $\lfloor \frac{B_1}{T_1} \rfloor = 3$.

Figure 6.13: Examples of worst case for CBR cells.

6.6 Summary

We have presented the performance results of the proposed call admission control framework based on call level and cell level. The workload model presented in chapter 5 was generated to conduct simulation studies of the proposed framework for different mixes. Both homogeneous and heterogeneous traffic streams are used to study the performance of the framework.

The network has a higher call-blocking probability for homogeneous video streams than for audio streams, due to higher peak rate requirements for video traffic. Call-blocking probability decreases as the call arrival interval increases. Also, note that the total number of connections does not affect the call-blocking probability. The call admission control algorithm performs better than the peak rate allocation scheme under different traffic mixes (e.g., call arrival rates, call arrival intervals, ratio of guaranteed traffic, total number of connections).

Bandwidth utilization is higher for audio traffic than for video traffic, because audio streams tend to be less bursty. The efficiency of bandwidth utilization is always less than 50% for audio traffic, regardless of the number of audio connections, which suggests that only 50% of the aggregated peak rate is needed for voice connections. It is also noted that a bandwidth efficiency of 80% for video traffic can be achieved for a single connection, while only a 50% efficiency can be achieved for more than 5 connections. Hence, video traffic needs an amount of bandwidth close to the aggregated peak rates with fewer than 5 connections; otherwise, it needs only 50% of the aggregated peak rate to increase the efficiency.

The call admission control scheme, via dynamically measuring network loads at the connection setup time, results in a low call-blocking probability and a high link bandwidth utilization. We have shown that the peak rate allocation scheme is a special case of our algorithm when all network traffic requires guaranteed services (i.e., the worst case); otherwise it has better performance.

Analysis of the cell delay time shows that our mechanism provides a delay bound based on the adjustable queue size and time-out value in the worst case. An upper bound for queuing delay for a single cell is obtained.

Chapter 7

Summary and Future Work

This chapter summarizes the major contribution of this dissertation research and outlines directions for future work.

7.1 Summary

The high speed of the ATM network makes call admission control an important component of network quality of services. In this thesis we introduce a call admission control framework for different classes of multimedia traffic in ATM networks. This framework consists of three crucial components: traffic measurements, a call admission control algorithm, and a cell-scheduling algorithm. Such a framework exhibits simple call admission and cell scheduling, which makes it practical to implement in an ATM switching node.

Traffic measurements are performed to collect traffic statistics for several important applications from our ATM testbed. These applications are chosen to represent multimedia environments supported by ATM networks (e.g., data, voice, video, and Mosaic). We identify unique traffic characteristics for these applications. These understandings of traffic statistics are valuable for good design and analysis of ATM networks.

A traffic model is built from the statistics collected at the testbed. This model is useful in studying various congestion control functions in ATM networks. However, our work focuses on traffic characterization and source modeling. Such model is used for studying the performance results of the proposed call admission control framework.

Congestion control in our framework is performed at two levels, namely call level and cell level. The call admission control algorithm can be implemented as either a centralized approach or a distributed approach. This algorithm provides guaranteed services for different classes of traffic while at the same time optimizing network link bandwidth utilization by dynamically measuring current network loads. We outline the algorithms for call admission and cell scheduling, and address the methodology of determining network link bandwidth utilization via SNMP.

We investigate the performance of the call admission control algorithm under various traffic mixes. This is carried out via simulations by using input streams generated from our traffic model. Homogeneous traffic streams are used to investigate the behavior of individual applications. Heterogeneous traffic streams are used to study the effectiveness of the call admission control algorithm. It is shown through simulation that our CAC scheme performs better than the peak rate allocation mechanism. Analysis of the cell delay time indicates that our scheduling algorithm provides a delay bound based on the adjustable queue size and time-out value in the worst case.

The framework is most appropriate in providing QoS requirements for different classes of multimedia traffic.

In summary, our work has accomplished its objectives. It is clear that this area will receive more interest and much more study. The following section suggests future directions for our work.

7.2 Future Work

7.2.1 Traffic Measurements Using the Hardware Approach

All of the traffic statistics collected in this dissertation research use a software approach. The data is collected by using the highest time resolution in Solaris routines. A major disadvantage of using software approach to collect traffic statistics is the overhead involved and its low time resolution.

As described in chapter 4, we used a Solaris routine, **nanosleep()**, to collect traffic statistics. The **read/write** operations that are used to record traffic statistics have a minimum overhead of 40 *nanoseconds*, which makes time-stamping each individual cell impossible. As ATM technology becomes more mature, hardware equipment is available to collect traffic statistics based on the cell interarrival time information. Traffic measurements with higher time resolution will provide more insight into different levels (e.g., call level, burst level, and cell level).

7.2.2 Self-Similarity Studies of Individual Applications

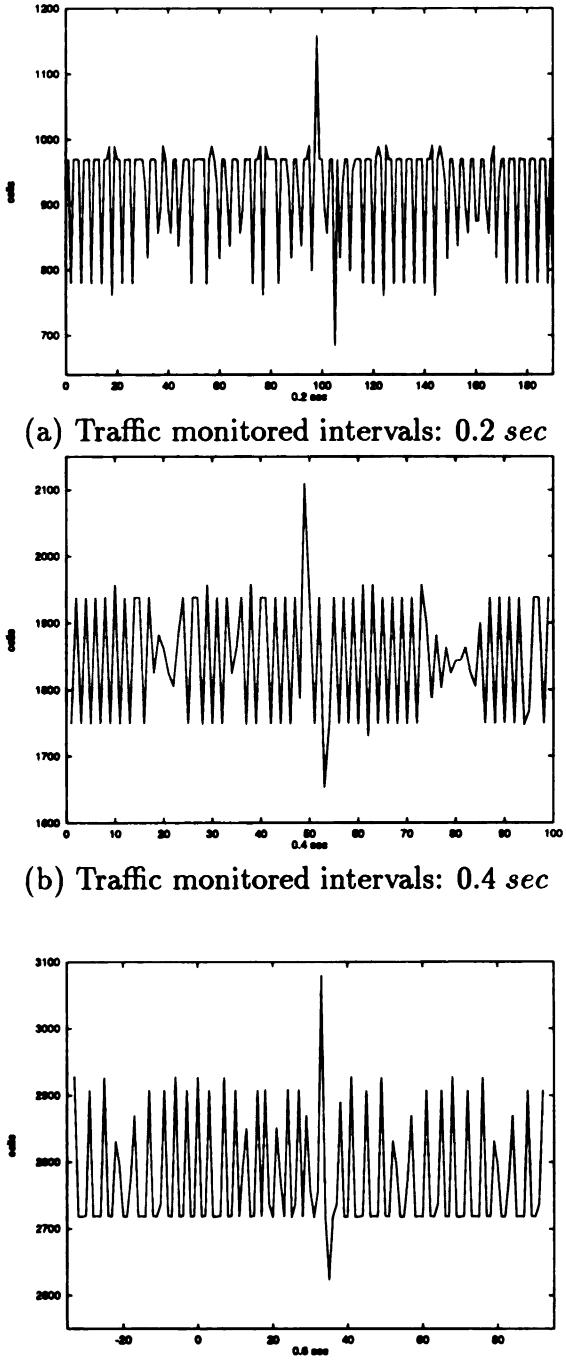
Coupled with the idea of the hardware approach to traffic measurements are self-similar traffic studies. The traffic model developed in chapter 5 is based on empirical traffic measurements. Traffic descriptors (peak rate, mean rate, burstiness) are obtained from the collected data, where traffic statistics are collected every Δt seconds. We have observed that the traffic descriptors obtained depend heavily on the value of Δt used, that is, different Δt results in different values of traffic descriptors.

Our future objective is to study individual application traffic behavior by using a fractal model, which has the same autocorrelation function for capturing traffic behavior, regardless of the different time scales that are used to study traffic behavior. Figures 7.1 and 7.2 show our early work on a pictorial proof of self-similarity for `audio` and `video` traffic, respectively.

The following definitions are used to construct self-similar graphs for audio and video traffic as shown in Figures 7.1 and 7.2. Let $X = (X_t : t = 0, 1, 2, \cdots)$ be a series of cell statistics from our traffic measurements, where X is obtained from a very small value of Δt. For various values of m (e.g., $m = 1, 2, 3, \cdots$), let $X^{(m)} = (X_k^{(m)} : k = 1, 2, 3, \cdots)$ denote a new series obtained by summing the original cell statistics X over non-overlapping blocks of size m. That is, for each $m = 1, 2, 3, \cdots$, $X^{(m)}$ is given by $X_k^{(m)} = (X_{km-m+1} + \cdots + X_{km}), k = 1, 2, 3$. Figures 7.1(a), (b), and (c) plot the X_k and $X_k^{(m)}$ series for `audio` traffic, whereas Figures 7.2(a), (b), (c), and (d) are the series for `video` traffic.

For stationary processes, such aggregations would result in uncorrelated white noise [47]. These graphs in Figures 7.1 and 7.2 not only retain significant correlations, but are also quite similar in appearance. The goal is to analyze, model, and generate self-similar audio and video traffic. To achieve this goal, we first need to consider a fairly large amount of traffic data, especially with higher time resolution, which will enable self-similar traffic studies over extended time scales.

The future work of analyzing self-similar traffic includes: (1) obtaining the log-log plots of complementary cumulative distributions (right-tail behavior) and the cumulative distributions (left-tail behavior); (2) studying probability densities, time correlations, and long-range dependencies; (3) estimating the *Hurst parameter*, which defines a certain relationship of autocorrelations over all time scales; (4) constructing a self-similar source traffic model for **`audio`** and **`video`** traffic. This empirically trace-driven model is valuable for studying various aspects of QoS issues of ATM networks.

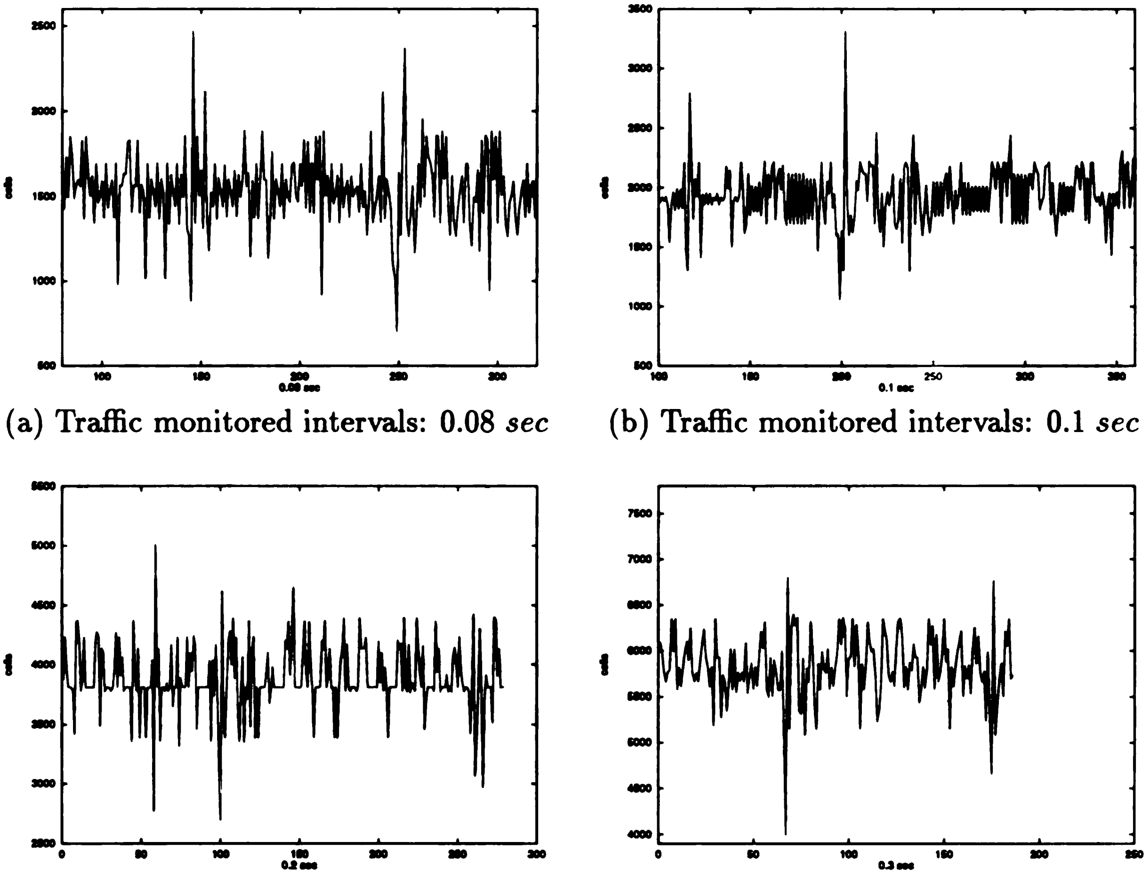


(a) Traffic monitored intervals: 0.2 *sec*

(b) Traffic monitored intervals: 0.4 *sec*

(c) Traffic monitored intervals: 0.6 *sec*

Figure 7.1: Pictorial proof of self-similarity: `audio` traffic traces on three different time scales (0.2 *sec*, 0.4 *sec*, and 0.6 *sec*)



(a) Traffic monitored intervals: 0.08 *sec*

(b) Traffic monitored intervals: 0.1 *sec*

(c) Traffic monitored intervals: 0.2 *sec*

(d) Traffic monitored intervals: 0.3 *sec*

Figure 7.2: Pictorial proof of self-similarity: `video` traffic traces on four different time scales (0.08 *sec*, 0.1 *sec*, 0.2 *sec*, and 0.6 *sec*)

Bibliography

[1] B. W. Abeysundara and A. E. Kamal, "High speed local area networks and their performance: A survey," *ACM Computing Surveys*, vol. 23, June 1991.

[2] J. Gechter and P. O'Reilly, "Conceptual issues for ATM," *IEEE Network*, pp. 14–16, January 1989.

[3] F. Vakil and H. Saito, "On congestion control in ATM networks," *IEEE LTS*, August 1991.

[4] H. Suzuki, T. Murase, S. Sato, and T. Takeuchi, "A burst traffic control strategy for ATM networks," in *Proceedings of IEEE Global Telecommunications Conference'90*, 1990.

[5] J. J. Bae and T. Suda, "Survey of traffic control protocols in ATM networks," in *Proceedings of IEEE Global Telecommunications Conference'90*, 1990.

[6] T. F. L. Porta and M. Schwartz, "Architectures, features, and implementation of high-speed transport protocols," *IEEE Network Magazine*, 1991.

[7] R. Guérin, H. Ahmadi, and M. Naghshineh, "Equivalent capacity and its application to bandwidth allocation in high-speed networks," *IEEE Journal on Selected Areas in Communications*, vol. 9, pp. 968–981, September 1991.

[8] D. Ferrari and D. Verma, "Quality of service and admission control in ATM networks," Tech. Rep. TR-90-064, International Computer Science Institute, Berkeley, California, December 1990.

[9] B. J. Vickers, J. B. Kim, T. Suda, and D. P. Hong, "Congestion Control and Resource Management in Diverse ATM Environment," *IEICE Trans. Comm.*, 1994.

[10] A. Forum, *ATM User-Network Interface Specification Version 3.0.* Englewood Cliffs, NJ: Prentice Hall, 1994.

[11] J. S. Turner, "New directions in communications," *IEEE Communications Magazine*, vol. 24, pp. 8–15, October 1986.

[12] L. Cheng and H. Hughes, "On the bursty behavior of the ATM traffic," in *Summer Computer Simulation Conference*, (Ottawa, Canada), 1995.

[13] J. M. Hyman, A. A. Lazar, and G. Pacific, "Real-time scheduling with quality of services constraints," *IEEE Journal on Selected Areas in Communication*, vol. 9, pp. 1052–1063, September 1992.

[14] J. M. Ferrandiz and A. A. Lazar, "Admission control of real-time sessions of an integrated node," in *Proceedings of IEEE INFOCOM'91*, pp. 553–559, April 1991.

[15] K. W. Ross and D. H. K. Tsang, "Optimal circuit access policies in an ISDN environment: A Markov decision approach," *IEEE Trans. on Comm.*, vol. 37, pp. 934–939, September 1989.

[16] G. Ramamurthy, R. S. Dighe, and D. Raychaudhuri, "An UPC-based traffic control framework for ATM networks," in *Proceedings of IEEE Global Telecommunications Conference'94*, pp. 600–605, November 1994.

[17] S. Jamin, S. Shenker, L. Zhang, and D. Clark, "An admission control algorithm for predictive real-time service," in *Proceedings of 3rd International Workshop on Network and Operating System Support for Digital Audio and Video*, November 1992.

[18] D. D. Clark, S. Shenker, and L. Zhang, "Supporting real-time applications in an integrated services packet network: Architecture and mechanism," *Proc. ACM SIGCOMM*, August 1992.

[19] A. Kolarov and G. Ramamurthy, "Comparison of congestion control schemes for ABR service in ATM local area networks," in *Proceedings of IEEE Global Telecommunications Conference'94*, pp. 913–918, November 1994.

[20] G. Ramamurthy and R. S. Dighe, "Distributed source control: A new access control for integrated broadband networks," *IEEE Journal on Selected Areas in Communication*, vol. 9, September 1991.

[21] G. Ramamurthy and R. S. Dighe, "A multidimensional framework for congestion control in B-ISDN," *IEEE Journal on Selected Areas in Communication*, vol. 9, December 1991.

[22] D. Ferrari and D. Verma, "A scheme for real-time channel establishment in wide-area networks," *IEEE Journal on Selected Areas in Communication*, vol. 8, pp. 368–379, April 1990.

[23] M. Ilyas and H. T. Mouftah, "Performance evaluation of congestion avoidance in broadband ISDNs," in *Proceedings of IEEE International Conference on Communications'90*, pp. 439–442, 1990.

[24] S. Y. R. Li, "Algorithms for flow control and call set-up in multihop broadband ISDN," in *Proceedings of IEEE INFOCOM'91*, pp. 889–895, 1990.

[25] P. W. Tse and M. Zukerman, "Connection admission control in ATM network," in *Proceedings of IEEE Global Telecommunications Conference'94*, pp. 1790–1794, November 1994.

[26] E. W. Knightly and H. Zhang, "Traffic characterization and switch utilization using a deterministic bounding interval dependent traffic model," in *Proceedings of IEEE INFOCOM'95*, 1995.

[27] E. W. Knightly and R. F. Mines, "Test applications for a heterogeneous real-time network testbed," in *Proceedings of the Third International Conference on Computer Communications and Networks*, pp. 183–189, September 11–14 1994.

[28] H. Zhang and E. Knightly, "Providing end-to-end statistical performance guarantees with interval dependent stochastic models," in *Proceedings of ACM SIGMETRICS*, 1990.

[29] G. Ramamurthy and R. S. Dighe, "Performance analysis of multilevel congestion controls in BISDN," in *Proceedings of IEEE INFOCOM'94*, vol. 1, pp. 1c.1.1–1c.1.9, June 1994.

[30] D. C. Verma, "Quality of services in ATM networks," tech. rep., University of California and International Computer Science Institute, 1990.

[31] J. M. Hyman, A. A. Lazar, and G. Pacific, "Joint scheduling and admission control for ATS-based switching nodes," in *Proceedings of ACM SIGCOMM*, August 1992.

[32] J. A. Garay and I. S. Gopal, "Call preemption in communication networks," in *Proceedings of IEEE INFOCOM'92*, pp. 1043–1050, 1992.

[33] M. Peyravian, "Providing different levels of network availability in high-speed networks," in *Proceedings of IEEE Global Telecommunications Conference'94*, pp. 941–945, November 1994.

[34] L. Cheng, S.-M. Chang, and H. Hughes, "A connection admission control algorithm based on empirical traffic measurements," in *Proceedings of IEEE International Conference on Communications'95*, (Seattle, WA), June 1995.

[35] L. Cheng and H. Hughes, "Performance study of congestion control schemes for ATM networks," in *Summer Computer Simulation Conference*, 1992.

[36] L. Cheng, H. Hughes, and P. Yegani, "Priority Scheduling in an ATM Network," in *Summer Computer Simulation Conference*, 1994.

[37] I. Nikolaidis and R. O. Onvural, "A bibliography on performance issues in ATM networks," in *Proceedings of ACM SIGCOMM'93*, 1993.

[38] D. S. Holtsinger, H. G. Perros, and A. A. Nilsson, "Analysis of traffic measurements in the VISTAnet Gigabit networking testbed," tech. rep., Center for Communications and Signal Processing, July 1993.

[39] L. J. Bottomley and A. A. Nilsson, "Traffic characterization in a wide area network," in *Proceedings of IEEE TriCom*, 1992.

[40] R. Cáceres, *Multiplexing Traffic at the Entrance to Wide-Area Networks.* PhD thesis, University of California Berkeley, December 1992.

[41] K. C. Claffy, *Internet Traffic Characterization.* PhD thesis, University of California, San Diego, 1994.

[42] W. E. Leland, M. S. Taqqu, W. Willinger, and D. V. Wilson, "On the self-similar nature of ethernet traffic," in *ACM SIGCOMM'93*, (San Francisco, CA), September 1993.

[43] J. R. S. Fernandez, *A Burst-Oriented Traffic Control Framework and Associated Call Admission Control Schemes for ATM Networks.* PhD thesis, Michigan State University, 1995.

[44] H. Heffes and D. M. Lucantoni, "A Markov modulated characterization of packetized voice and data traffic and related statistical multiplexer performance," *IEEE Journal on Selected Areas in Communications*, vol. 4, pp. 856–868, Sep 1986.

[45] R. Jain and S. Routhier, "Packet trains — measurements and a new model for computer network traffic," *IEEE Journal on Selected Areas in Communications*, September 1986.

[46] H. J. Fowler and W. E. Leland, "Local area network traffic characteristics, with implications for broadband network congestion management," *IEEE Journal on Selected Areas of Communications*, vol. 9, pp. 1139–1149, 1991.

[47] M. W. Garrett and W. Willinger, "Analysis, modeling and generation of self-similar vbr video traffic," in *ACM SIGCOMM'94*, (London England UK), September 1994.

[48] R. Cáceres, P. B. Danzig, S. Jamin, and D. J. Mitzel, "Characteristics of wide-area TCP/IP conversations," in *Proceedings of ACM SIGCOMM'91*, 1991.

[49] P. B. Danzig, S. Jamin, R. Cáceres, D. J. Mitzel, and D. Estrin, "An empirical workload model for driving wide-area TCP/IP network simulations," *Journal of Internetworking: Practice and Experience*, vol. 3, pp. 1–26, March 1992.

[50] V. Paxson, "Measurements and models of wide area TCP conversations," Tech. Rep. LBL-30840, Computer Systems Engineering Department, Lawrence Berkeley Laboratory, University of California, Berkeley, May 1991.

[51] R. Cáceres, "Measurements of wide-area internet traffic," Tech. Rep. UCB/CSD 89/550, University of California, Berkeley, December 1989.

[52] P. B. Danzig and S. Jamin, "tcplib: A library of TCP internetwork traffic characteristics," Tech. Rep. USC-CS-91-495, University of California, Los Angeles, September 1991.

[53] D. Sanghi, A. K. Agrawala, O. Gudmundsson, and B. N. Jain, "Experimental assessment of end-to-end behavior on Internet," in *Proceedings of IEEE INFOCOM'93*, vol. 3, 1993.

[54] E. P. Rathgeb, "Modeling and performance comparison of policing mechanisms for ATM networks," *IEEE Journal on Selected Areas in Communications*, vol. 9, pp. 325–334, April 1991.

[55] W. E. Leland, "Window-based congestion management in broadband atm networks: The performance of three access-control policies," in *Proceedings of IEEE INFOCOM'89*, pp. 49.7.1–49.7.7, 1989.

[56] P. Yegani, "Performance models for ATM Switching of Mixed Continous-Bit-Rate and Bursty Traffic with Threshold-Based Discarding," *IEEE ICC*, pp. 354.3.1–354.3.5, 1992.

[57] P. Yegani, "Priority Schemes in High Performance Fast-Packet-Switched Networks," *Proc. IBM Performance ITL Symposium*, April 1992.

[58] I. W. Habib and T. N. Saadawi, "Controlling flow and avoiding congestion in broadband networks," *IEEE Communication Magazine*, October 1991.

[59] T.-Y. Huang and J.-L. Wu, "Performance analysis of a dynamic priority scheduling method in ATM networks," *IEEE Proceedings*, vol. 140, August 1993.

[60] J. J. Bae and T. Suda, "Survey of traffic control schemes and protocols in ATM networks," *Proceedings of the IEEE*, vol. 79, pp. 170–189, February 1991.

[61] H. J. Chao, "Design of leaky bucket access control schemes in atm networks," in *Proceedings of IEEE International Conference on Communications'91*, pp. 6.1.1–6.1.8, 1991.

[62] G. Gallassi, G. Rigolio, and L. Fratta, "ATM: Bandwidth assignment and bandwidth enforcement policies," in *Proceedings of IEEE Global Telecommunications Conference'88*, pp. 49.6.1–49.6.6, 1989.

[63] M. Hirano and N. Watanabe, "Characteristics of a cell multiplexer for bursty atm traffic," in *Proceedings of IEEE International Conference on Communications'89*, pp. 0399–0403, 1989.

[64] A. E. E. Jr., D. T. Luan, and D. M. Lucantoni, "Meeting the challenge: Congestion and flow control strategies for broadband information transport," in *Proceedings of IEEE Global Telecommunications Conference'89*, pp. 49.3.1–49.3.5, 1989.

[65] M. G. Hluchyj and N. Yin, "A second-order leacky bucket algorithm to guarantee qos in atm networks," in *Proceedings of IEEE ATM Workshop '95*, November 1995.

[66] Y. Lim and J. Kobza, "Analysis of a delay-dependent priority discipline in an integrated multiclass traffic fast packet switch," *IEEE Transactions on Communication*, vol. 38, pp. 659–665, May 1990.

[67] Kraimeche, G. Pacifici, D. E. Pendarakis, G. Ramamurthy, B. Sengupta, P. A. Skelly, F. Vakil, and Z. Zhang, "Call admission control for TeraNet," Tech. Rep. CTR # 319-92-29, Center for Telecommunications Research, Columbia University, March 2 1993.

[68] R. Gidron and A. Temple, "TeraNet: A multihop multichannel lightwave network," in *Proceedings of the IEEE International Conference on Communications*, (Denver, Colorado), pp. 602–608, June 1991.

[69] G. Hebuterne and A. Gravey, "A space priority queueing mechanism for multiplexing ATM channels," *ITC Specialist Seminar*, 1989.

[70] D. W. Petr, L. A. D. Jr., and V. S. Frost, "Priority discarding of speech in integrated packet networks," *IEEE Journal on Selected Areas in Communication*, vol. 7, pp. 644–659, June 1989.

[71] Kroner, G. Hebueterne, P. Boyer, and A. Gravey, "Priority management in ATM switching nodes," *IEEE Journal on Selected Areas in Communications*, vol. 9, pp. 418–427, April 1991.

[72] D. Ferrari, "A new call admission control method for real-time communication in an internetwork," tech. rep., University of California at Berkeley and International Computer Science Institute, 1992.

[73] H. Zhang and S. Keshav, "Comparison of rate-based service disciplines," *Proc. ACM SIGCOMM*, pp. 113–122, September 1991.

[74] L. Cheng, H. Hughes, and P. Yegani, "Priority scheduling schemes in an ATM switching node," in *Proceedings of the Third International Conference on Computer Communications and Networks*, (San Francisco, CA), pp. 72–76, 1994.

[75] A. Forum, *ATM User-Network Interface Specification Version 3.0.* Englewood Cliffs, NJ: Prentice Hall, 1993.

[76] H. Armbrüster, "The flexibility of ATM: Supporting future multimedia and mobile communications," *IEEE Communications Magazine*, pp. 76–84, February 1995.

[77] A. S. Acampora and M. Naghshineh, "Control and quality-of-service provisioning in high-speed microcellular networks," *IEEE Personal Communications*, pp. 36–43, Second Quarter 1994.

[78] D. Raychaudhuri and N. D. Wilson, "ATM-based transport architecture for multiservices wireless personal communication networks," *IEEE Journal on Selected Areas of Communications*, vol. 12, pp. 1401–1414, October 1994.

[79] H. T. Kung and R. Morris, "Credit-Based Flow Control for ATM Networks," *IEEE Network*, vol. 9, pp. 40–48, March/April 1995.

[80] H. T. Kung, T. Blackwell, and A. Chapman, "Credit-Based Flow Control for ATM Networks: Credit update protocol, adaptive credit allocation, and statistical multiplexing," *Proceedings of ACM SIGCOMM '94 Symposium on Communications Architectures, Protocols and Applications*, pp. 101–114, August 31 – September 2 1994.

[81] H. T. Kung and K. Chang, "Receiver-oriented adaptive buffer allocation in credit-based flow control for ATM networks," in *Proceedings of IEEE INFOCOM'95*, April 1995.

[82] N. Yin and M. G. Hluchyj, "On closed-loop rate control for atm cell relay networks," in *Proceedings of IEEE INFOCOM'94*, pp. 99–108, 1994.

[83] W. Stallings, *SNMP, SNMPv2, and CMIP: The Practical Guide to Network-Management Standards.* One Jacob Way: Addison Wesley, 1993.

[84] J. Case, M. Fedor, M. Schoffstall, and J. Davin, "A simple network management protocol (SNMP)." Request for Comments 1157, May 1990.

[85] K. McCloghrie and M. Rose, "Management information base for network management of TCP/IP-based internets." Internet Working Group, Request for Comments 1213, Network Information Center, SRI International, Menlo Park, California, March 1991.

[86] J. Case, K. McCloghrie, M. Rose, and S. Waldbusser, "Introduction to version 2 of the internet-standard network management framework." Request for Comments 1441, April 1993.

[87] J. Case, K. McCloghrie, M. Rose, and S. Waldbusser, "Protocol operations for version 2 of the simple network management protocol (SNMPv2)." Request for Comments 1448, April 1993.

[88] K. McCloghrie and F. Kastenholz, "Evolution of the interfaces group of MIB-II." Request for Comments 1573, January 1994.

[89] L. Cheng and H. D. Hughes, "Quality of services based on both call admission and cell scheduling," *Special Issue of Computer Networks and ISDN Systems Journal on Signaling and Management of ATM Networks*, 1996.

[90] Fore Systems, Inc., 1000 Gamma Drive, Suite 504 Pittsburgh, PA 14238-2940, *ForeRunner ASX-100 ATM Switch Documentation Overview*, 1993.

[91] Fore Systems, Inc., 1000 Gamma Drive, Suite 504 Pittsburgh, PA 14238-2940, *ForeRunner SBA-200 ATM Sbus Adapter User's Manual*, 1993.

[92] J. Postel and J. Reynolds, "File transfer protocol (FTP)." Request for Comments 959, October 1985.

[93] Fore Systems Inc., 1000 Gamma Drive, Suite 504 Pittsburgh, PA 14238-2940, *SPANS: Simple Protocol for ATM Network Signaling*, 1993.

[94] Sun Microsystems Computer Corporation, *Desktop SPARC Hardware Owner's Guide*, December 1992.

[95] Sun Microsystems Computer Corporation, *SunVideo 1.0 User's Guide*, October 1993.

[96] Sun Microsystems Computer Corporation, *Installing SBus Cards in Desktop SPARCstations*, October 1992.

[97] Harrenstien, Stahl, and Feinler, "NICName/Whois.." Request for Comments 954, SRI International, October 1985.

[98] E. CCITT/ISO, Gloucester, "CCITT/ISO. the directory, part 1: Overview of concepts, models and services." CCITT Draft Recommandation X.500/ISO DIS 9594-1, December 1988.

[99] A. Emtage and P. Deutsch, "Archie - an electronic directory service for internet," in *Proceedings of USENIX Winter Conference*, pp. 93–110, Janurary 1992.

[100] T. Berners-Lee, R. Cailliau, J. Groff, and B. Pollermann, "World-Wide Web: The information universe," in *Electronic Networking: Research, Applications and Policy*, vol. 2, (Westport CT), pp. 52–58, Meckler Publications, Spring 1992.

[101] T. Berners-Lee, R. Cailliau, J. Groff, and B. Pollermann, "World-Wide Web: An information infrastructure for high-Energy physics," in *Artificial Interlligence and Software Engineering for High Energy Physics* (D. Perret-Gallix, Ed.), (La Londe, France), World Scientific, Singapore, January 1992.

[102] B. Kahle and A. Medlar, "An information system for corporate users: Wide area informaiton servers," in *ConneXions – The Interoperability Report*, Interop, Inc., Novermber 1991.

[103] R. E. Droms, "Access to heterogeneous direcory services," in *Proceedings of 9th Joint Conference of IEEE Computer and Communications Societies (InfoCom)*, June 1990.

[104] M. F. Schwartz and P. G. Tsirigotis, "Experience with a semantically cognizant internet white page directory tool," *Journal of Internetworking: Research and Experience*, vol. 2, pp. 23–50, March 1991.

[105] M. McCahill, "The internet gopher: A distributed server information systems," in *ConneXions – The Interoperbility Report*, vol. 6(7), pp. 10–14, Interop, Inc., July 1992.

[106] J. E. Pitkow and M. M. Recker, "Results from the first World-Wide Web survey," *Special Issue of Computer Networks and ISDN Systems*, vol. 27, no. 2, 1994.

[107] J. E. Pitkow and M. M. Recker, "Using the web as a survey tool: results from the second WWW user survey," *Special Issue of Computer Networks and ISDN Systems*, vol. 27, pp. 809–822, April 1995.

[108] L. Dittmann and S.B.Jacobson, "Statistical multiplexing of identical bursty sources in an ATM network," in *Proceedings of the IEEE Global Telecommunication Conference*, pp. 39.6.1–39.6.5, 1988.

[109] M. Decina, T. Toniatti, P. Vaccari, and L. Verri, "Bandwidth assignment and virtual call blocking in ATM networks," in *Proceedings of IEEE INFOCOM'90*, 1990.

[110] M. Decina and T. Toniatti, "On bandwidth allocation to bursty virtual connections in atm networks," in *IEEE International Conference on Communications*, pp. 318.6.1–318.6.8, 1990.

[111] V. Paxson and S. Floyd, "Wide-area traffic: The failure of Poisson modeling," in *The Proceedings of ACM SIGCOMM'94*, 1994.

[112] J. Y. Hui and E. Arthurs, "A broadband packet switch for integrated transport," *IEEE Journal on Selected Areas of Communications*, vol. SAC-5, pp. 1264–1273, October 1987.

[113] K. Sriram and W. Whitt, "Characterizing superposition arrival processes in packet multiplexers for voice and data," *IEEE Journal on Selected Areas of Communications*, vol. SAC-4, pp. 833–846, September 1986.

[114] J. J. Bae, T. Suda, and R. Simha, "Analysis of a finite buffer queue with heterogeneous Markov modulated arrival processes: A study of the effects of traffic burstiness on individual packet loss," in *Proceedings of IEEE Global Telecommunications Conference'92*, pp. 219–230, 1992.

[115] R. Jain, Ed., *The Art of Computer Systems Performance Analysis.* John Wiley & Sons, 1991.

[116] L. Cheng and H. D. Hughes, "An ATM traffic model based on empirical traffic measurements," in *Proceedings of the IEEE Global Telecommunication Conference*, (Singapore), November 1995.