REAL-TIME COMMUNICATION IN INTEGRATED SERVICES PACKET NETWORKS

by

Seok-Kyu Kweon

A dissertation submitted in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
(Electrical Engineering: Systems)
in The University of Michigan
1998

Doctoral Committee:

Professor Kang G. Shin, Chair
Associate Professor Farnam Jahanian
Professor Wayne E. Stark
Professor Toby Teorey
Assistant Professor Kimberly M. Wasserman

ABSTRACT

REAL-TIME COMMUNICATION IN INTEGRATED SERVICES PACKET NETWORKS

by
Seok-Kyu Kweon

Chair: Kang G. Shin

One of the main issues in Integrated Services Packet Networks (ISPNs) is providing per-connection Quality-of-Service (QoS) in the heterogeneous traffic environment. To this end, many technical problems such as packet scheduling, resource reservation, traffic policing, and QoS routing must be resolved. In this dissertation, we provide solutions to these problems.

We propose a new cell-multiplexing scheme, named TCRM, which provides per-connection delay guarantees in an ATM network. By guaranteeing a minimum throughput, TCRM can provide deterministic delay guarantees under the leaky bucket constraint of input traffic. A scalable implementation of TCRM is presented. In addition to deterministic delay guarantees, TCRM can be used to provide statistical delay guarantees by statistically multiplexing different real-time connections onto a single virtual channel. Such statistical multiplexing enhances network utilization. Using a histogram-based source model, we investigate the cell loss/delay characteristics of this approach.

Since a large portion of the end users are connected to ISPNs via Ethernets, providing QoS guarantees over Ethernet is important. To this end, we model the MAC protocol of Ethernet with its collision resolution protocol using a Semi-Markov Poisson process. We derive a connection admission control for statistically bounding the medium access time.

Video-on-Demand service requires delivery delay guarantees unlike non-real-time applications, but its delay requirement is not as tight as real-time applications. We propose a new service category, called the *semi-real-time class* for such applications, and present the network control functions necessary for it. For the source traffic characterization necessary to support the semi-real-time class, we define and derive a *statistical traffic envelope*.

For efficient QoS routing, we propose a flooding-based QoS route selection algorithm. It not only dramatically reduces message overhead and operational cost, but also provides good performance compared to existing QoS-routing schemes based on either request flooding or link-state dissemination. The overhead is reduced by limiting the search area within which connection-request messages are flooded using a topological information kept at local nodes.

To My Parents and Eunchong

ACKNOWLEDGEMENTS

I would like to thank all those who have contributed, both directly and indirectly, to this dissertation. My deepest gratitude is to my advisor Professor Kang G. Shin for his guidance throughout the course of this work. He allowed me to pursue my own research interests, and was available to discuss my problems and provide feedback. I would also like to thank the members of my dissertation committee, Wayne Stark, Toby Teorey, Farnam Jahanian, and Kimberly Wasserman for their constructive comments and suggestions.

Thanks also go to the former and present members of the Real-Time Computing Laboratory, especially, Qin Zheng, Chi-Che Chou, Jennifer Rexford, Atri Indiresan, Emmanuel Abram-Profeta, Seungjae Han, Sunghyun Choi, and Sunghwan Moon for their friendship and insightful discussions.

I also appreciate the financial support during the course of my Ph.D. program by the Korean government, the Office of Naval Research, and the National Science Foundation.

Finally, I thank my parents and my wife, Eunchong, for their dedicated support and love.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

Figure

LIST OF APPENDICES

APPENDIX

CHAPTER 1

INTRODUCTION

1.1 Motivation

With the fast development of high-speed network technology and the explosive growth of packet-switching networks, contemporary networks are required to carry various types of traffic including real-time audio/video data, as well as traditional data communication traffic. In particular, the need for supporting such heterogeneous traffic over a single network, which is known as an Integrated Services Packet Network (ISPN), is a broad consensus in the computer networking community. The unified communication infrastructure would offer a multitude of advantages, including vast economies of scale, ubiquity of access, and improved statistical multiplexing.

However, many technical issues must be resolved before such ISPNs are successfully deployed in the real world. In particular, the problem of providing flexible Quality-of-Service (QoS) guarantees in ISPNs has been drawing enormous attention. This is a challenging problem considering the heterogeneous traffic characteristics and a wide range of QoS requirements which are expected to be observed in ISPNs. Real-time traffic requires more precisely-defined QoS in terms of packet delay, throughput and loss rate, compared to best effort traffic generated from traditional data communication applications such as ftp and telnet. Among these performance requirements, delay plays the most important role in real-time applications [79], since the packet error rate can be kept very low with the use of advanced networking technologies. In particular, in the context of real-time communication, delay guarantee indicates *per* packet delay guarantee, not the average delay guarantee. If a packet arrives at the destination after its deadline has expired, its value to the application may be greatly reduced or even worthless. In some circumstances, a packet missing its deadline is considered lost. Thus, the packet delivery delay must be bounded and predictable for real-time applications.

It is impossible to meet different QoS requirements for different real-time applications using the best-effort delivery service of a conventional packet-switching network, because their packet multiplexers do not differentiate between real-time and non-real-time traffic, nor among different real-time traffic. Recently, several packet multiplexing techniques have been proposed to provide per-connection delay guarantee [17, 19, 26, 36, 37, 64, 95, 100]. Although they can provide bounded end-to-end delays in a point-to-point network environment, they differ in input traffic specification, connection admissibility, and implementation complexity. One of the most important design goals of such packet multiplexing schemes is that they must be simple enough to work in a high-speed network environment while keeping network utilization at a reasonable level.

Along with packet multiplexing schemes, ISPNs must be equipped with an appropriate connection admission control (CAC) and resource reservation scheme in order to provide per-connection QoS in an efficient manner [101]. Without these functions, no packet multiplexing scheme can provide the promised QoS to end users. As with packet multiplexing schemes, choosing a good CAC and resource reservation scheme greatly affects the performance of ISPN. Desirable features of a CAC and resource reservation scheme are efficiency, simplicity, low operational overhead, and low cost.

In conjunction with resource reservation and CAC, finding a route which can provide user-requested QoS is another important issue in a point-to-point network environment. This is known as the QoS-routing problem [91]. Without an efficient QoS-routing algorithm, a network may fail to find a route and, as a result, reject the request for a new connection, even if there exists a qualified route with enough resources to honor the request. QoS routing must therefore be considered as a component of *Integrated Service Model* in ISPN [12]. In addition to finding a qualified route for each requested connection, QoS-routing must be able to make good overall network utilization. This also relates to increasing the chance of accepting future requests by making a good route selection for the current request. Finally, QoS-routing algorithms should not be expensive in terms of both operational cost and implementation complexity.

Real-time communication is conventionally classified into two categories according to QoS requirements: deterministic and statistical [1]. In deterministic real-time communication, QoS requirements are specified in absolute terms and no packet losses or deadline misses are allowed. In order to satisfy its absolute QoS requirements, each deterministic real-time connection requires resource reservation based on the worst-case source traffic behavior, thus resulting in severe underutilization of network resources. On the other hand, statistical real-time communication provides QoS guarantees in *statistical* terms so that better network utilization is achieved. That is, statistical real-time communication tolerates a certain percentage of packet losses and deadline misses, and thus, it allows connections to over-book network resources. Statistical real-time communication services are useful to those applications (i) that can tolerate a portion of packet losses and deadline misses and (ii) whose traffic is bursty. The statistical multiplexing gain will be substantial, especially in such Variable-Bit-Rate (VBR) applications as compressed video.

Although applications on ISPNs are usually classified as either real-time or non-real-time, one can think of a “finer” classification as follows: First, applications are classified as *delay-sensitive* or *delay-insensitive*. Delay-insensitive applications do not require per-packet delay guarantees; they are essentially the same as true non-real-time applications. Delay-sensitive applications, on the other hand, require packet-delivery time guarantees from the network. They can be classified further into two sub-categories, depending on the relation between the user-requested delay bound and source traffic characteristics. In general, delay-sensitive applications can be characterized by their traffic parameters such as average and peak packet-generation rates. In order to guarantee the user-requested delay bound for each connection, the network must reserve the bandwidth commensurate with the required QoS. The reserved bandwidth ranges from the average packet-generation rate to the peak packet-generation rate. If the user-requested delay bound is small, the reserved bandwidth tends to be closer to the peak rate; this type of application is said to be *peak-rate-oriented*. Conventional real-time applications fall in this category since they require per-packet delivery delay guarantees [79] and their delay requirements are very tight, which, in turn, entail admission control based on the peak packet-generation rate. Servicing peak-rate-oriented applications is resource-intensive and often leads to very low resource utilization when the source traffic pattern is bursty, or the ratio of peak packet-generation rate to average packet-generation rate is high.

In contrast, when the user-requested delay bound is fairly large (e.g., several seconds to several minutes), the required bandwidth is closer to the average packet-generation rate; we call this type of applications *average-rate-oriented* or *semi-real-time*. A typical example application of this type is VoD (Video-on-Demand) service. Delays in VoD service need not be constrained by the peak packet-generation rate. By reserving bandwidth close to the average packet-generation rate, we can still provide reasonable delay bounds for VoD services at a much lower cost compared to the case of processing them as peak-rate-oriented applications. Although the increased delay bounds require larger buffers in order to avoid excessive packet losses, buffer space can be considered as a more flexible resource to control than bandwidth, thus less expensive. Because of their "loose" delay bounds, semi-real-time applications can also be serviced by using the bandwidth reserved but unused by real-time applications. As a result, introducing semi-real-time applications can significantly improve network utilization while still providing good QoS, unlike the case of using traditional best-effort services for them.

ISPNs, in general, can be considered as point-to-point networks and therefore real-time communication services are assumed to be provided in a point-to-point network environment. In practice, however, an end system is usually connected to a shared-medium Local Area Network (LAN) and LANs are connected together by point-to-point networks. Therefore, it is important to study how real-time communication can be provided through shared-medium LANs in order to consider end-to-end QoS.

Some LANs such as FDDI and FieldBus can be easily modified to provide real-time communication service [81, 82, 105]. This is possible due to the fact that they can provide bounded, or at least predictable, medium access time because of their token-based medium access control protocol. Despite the numerous newly-emerging or already-established high-speed networks, however, the predominant LAN architecture is still, and will be in the future, based on Ethernet, due mainly to its maturity and wide deployment & acceptance. In addition to the most popular 10-Mbps standard Ethernet, newer generations of Ethernets such as 100 Base-T and 100 Base-VG have already been spreading widely in the commercial market [6]. It is a general belief that Ethernet will continue to be the network of choice in a LAN environment, especially at end-host clusters. Moreover, its low price and the availability of Ethernet device drivers on almost all operating systems makes it attractive even for embedded systems like automated factories.

Despite its popularity and usefulness, however, Ethernet has a serious difficulty if it is to be used for real-time communication, because its medium access control (MAC) protocol is contention-based, and thus cannot provide deterministic channel access delays to component stations. Unlike FDDI, Ethernet has no mechanism to control the medium access time.

Deriving a deterministic bound on the channel access time of Ethernet, especially when the packet arrival pattern is not given in a deterministic form, is extremely difficult and of little value since the delay bound derived from the worst-case traffic condition is excessively large even under a very lightly-loaded condition. On the contrary, using a statistical bound on channel access time for CAC will improve network utilization significantly at the expense of a small percentage of packet losses or deadline misses. The difficulty in using a statistical delay bound for real-time communication lies in how to derive the statistical bound itself. This is mainly due to the contention-based MAC protocol of Ethernet.

1.2 Related Work

1.2.1 Deterministic Real-time Communication

In order to provide per-connection delay guarantees over ISPNs, source node and intermediate switches must be equipped with a special packet multiplexing scheme. Recently, several packet multiplexing schemes have been proposed to this end. Virtual Clock (VC) [100], Hierarchical Round Robin (HRR) [36], Stop-and-Go [26], Earliest-Due-Date (EDD), also called Delay-EDD [19], Rate Controlled Static Priority Queueing (RCSP) [95], Weighted Fair Queueing (WFQ) [17], Packet-by-Packet Generalized Processor Sharing (PGPS) [64], and Real-Time Channel (RTC) [37] are a few examples.

VC is a rate-based traffic control algorithm that can be used in packet-switching networks. Although it can provide delay-guaranteed service with an appropriate connection admission control [93], it tends to penalize a connection that has received better than guaranteed service during a certain time interval [64]. Framing strategies like HRR and Stop-and-Go provide bounded end-to-end delays. However, as pointed out in [37, 96, 104], they suffer the problem of coupling between the guaranteeable delay bound and the bandwidth-allocation granularity. Delay-EDD combined with the $(X_{min}, X_{ave}, I, S_{max})$ model can provide bounded end-to-end delays [19] and is optimal in terms of link utilization since it adopts deadline scheduling [54]. However, calculating packet deadlines and sorting the packets based on their deadlines is very expensive in terms of time and hardware. RCSP and RTC are variants of Delay-EDD but do not require the calculation of packet deadlines inside the scheduler. While RTC employs the leaky bucket model as its input traffic specification and allows an arbitrary number of priority levels for arbitrary link-delay guarantees, RCSP uses the $(X_{min}, X_{ave}, I, S_{max})$ model and allows a finite number of static priority levels for the simplicity of implementation. Since they both are rate-controlled service disciplines [95] and, more generally, non-work-conserving service disciplines, they reduce the buffer requirement at the intermediate nodes. In non-work-conserving service disciplines, the traffic-arrival pattern of a connection is reconstructed at every intermediate node so that the traffic-arrival patterns at the intermediate nodes conform to the original input traffic specification at the source. Non-work-conserving service disciplines achieve this goal by using a traffic regulator that holds early packets until their logical arrival times, or eligibility times. Although they can provide the same delay bounds as work-conserving service disciplines, non-work-conserving service disciplines yield larger packet delays than their counterparts. Non-work-conserving service disciplines may seem disadvantageous, but the QoS requirement of importance to real-time applications is bounded packet delays, not small average delays. Let's consider a real-time video conference, for example, in which every video frame must be delivered to the destination before its play-back deadline so that the video stream may be played back smoothly. As long as each video frame arrives at the destination within the play-back deadline, video quality won't be affected regardless whether it arrives earlier or not. In addition, non-work-conserving service disciplines provide a smaller delay jitter than work-conserving service disciplines, thus reducing the buffer requirement at intermediate nodes. Thus, non-work-conserving service disciplines are a better choice for real-time communication.

WFQ is a rate-based packet scheduling scheme and guarantees a minimum throughput to each connection over a period longer than its packet inter-arrival time. However, WFQ itself cannot provide bounded end-to-end delays. Parekh and Gallager proved that PGPS, which is the same as WFQ, can provide bounded end-to-end delays using the leaky bucket model for input traffic specification [65, 66]. WFQ (PGPS) shares the same advantages and disadvantages as Delay-EDD because it also adopts deadline scheduling.[1]In addition, like Delay-EDD, PGPS is a work-conserving service discipline, and thus, requires larger buffer space than RCSP or RTC.

1.2.2 Statistical Real-time Communication

Ferrari and Verma [19] introduced the concept of statistical performance guarantees and proposed a methodology for statistical real-time communication in WANs using the Earliest-Due-Date policy for scheduling packets. They specified channel-establishment conditions for statistical performance guarantees. However, since their channel-establishment process requires the calculation of the probability of deadline overflows that depends on other channels' activities, it is quite complex. Kurose [49] proposed to model a traffic source with a family of random variables that bound the number of packets sent during various time intervals. In his approach, the delay suffered by each packet in the network can be upper bounded, but the bound may be quite loose. Zhang *et al.* applied Kurose's input traf-

[1] Although WFQ (PGPS) is not deadline scheduling but rate-based in essence, it requires sorting packets based on their virtual finish times which can be viewed as their deadlines. Hence, we consider WFQ as deadline scheduling.

fic model to RCSP and provided statistical performance guarantees for individual real-time connections [45, 95, 99]. They assumed that any packet missing its local delay bound at each switch is dropped immediately so that the end-to-end delay overflow probability bound can be decomposed into the local delay overflow probability bounds experienced along the path of the connection. Since a packet does not carry any timing information in many high-speed networks like ATM, it is impossible to identify the individual cells whose deadlines have expired, and thus, this scheme cannot be employed in such high-speed networks. Zhang *et al.* [103] also derived a statistical bound on the end-to-end delay by applying the Exponentially-Bounded Burstiness (E.B.B.) process model [94] to Generalized Processor Sharing (GPS) networks in a similar way Parekh and Gallager [65, 66] derived a deterministic bound using the leaky bucket model [14, 15]. Although their work is theoretically attractive, it is not clear whether it can be applied to real life systems since their scheme assumes an infinite buffer at each node. In addition, the implementation complexity of PGPS must be resolved for use in high-speed networks like ATM [51].

Effective bandwidth has been investigated in order to provide statistically-guaranteed QoS in ATM networks [18, 29, 39]. This approach is based on the large deviation theory and often employs an on-off process as a source traffic model. In particular, Elwalid *et al.* [18] derived the worst-cast traffic parameters for achieving lossless multiplexing and used them to extract multiplexing gains from the statistical independence of traffic processes subject to the constraint of a small buffer overflow probability. They employed the leaky-bucket-regulated periodic on-off process as their input traffic model to this end. In order to calculate the overflow probability in the buffered multiplexing system like an ATM multiplexer, they developed a virtual buffer/trunk system. This model enabled them to transform the two-resource (buffer and link bandwidths) reservation problem into a single-resource reservation problem. By using this model, they were able to use the Chernoff bound as a buffer overflow probability estimate. Although their approach is mathematically elegant, the estimate based on the extremal on-off process is quite pessimistic.

1.2.3 Semi-real-time Communication

Guérin *et al.* [29] proposed the Guaranteed Rate (GR) service which motivated the semi-real-time communication service. They used a *deterministic traffic envelope* in order to provide the GR service. Having recognized the low network utilization and large delay-bound and buffer requirements which resulted from employing the deterministic approach based on the worst-case traffic condition, they suggested the use of a statistical traffic envelope in their future work.

The concept of statistical traffic envelope was used by many researchers to analyze the packet delay/loss behavior of a statistical real-time communication system [7, 43, 49, 94, 97, 103]. In [7, 49, 94, 97, 103], stochastic-bounding approach was used in deriving statistical traffic envelopes. This approach is quite pessimistic in estimating packet loss behavior especially due to the conservative nature of choosing bounding random variables. In [43], Knightly derived a statistical traffic envelope from a deterministic traffic envelope using the variance-maximizing technique. This approach requires the knowledge of deterministic traffic envelopes of multiplexed connections, and obtaining tight deterministic traffic envelopes is indispensable to deriving a tight statistical traffic envelope. In general, it is difficult to derive a tight deterministic traffic envelope without knowledge of much-detailed source traffic characteristics (e.g., entire frame size sequence in VoD applications).

Several researchers have proposed piecewise-constant-rate transmission of packets for VoD-like services [59, 71, 77]. Basically, their approaches require use of real-time communication service in an integrated services network or a dedicated transport system to guarantee a transmission rate. The approach in [59] may cause a large build-up delay and large buffer space requirement at the receiver, depending on the characteristics of the transmitted VBR (Variable Bit Rate) video data. The approach in [77] can alleviate these problems by employing a smoothing technique. However, this results in a large overhead of the network control function due to possibly frequent changes of reserved bandwidth, and requires the calculation of an optimal transmission schedule, which is computation-intensive. Reisslein and Ross [71] proposed a call-admission scheme for VoD-like sources subject to a packet loss constraint. Their approach is based on the Central Limit Theorem and the large deviation approximation. To alleviate the difficulty of calculating the Chernoff bound or the one refined by Bahadur and Rao [2], they suggested several approximation techniques. Since they did not assume any usage of receiver buffer, the real-time communication service must be provided in an integrated services network, or a dedicated transport system must be used to meet the required QoS.

1.2.4 Real-time Communication over Ethernet

Most of the earlier work in the area of supporting real-time communication over Ethernet focused on how to modify the MAC so that the modified MAC can control channel access time. Employing tokens is one such example [89]. Evidently, this approach is quite costly compared to the well-established and widely-used current Ethernet standard.

Most of the work in the area of analyzing delay characteristics of CSMA/CD networks focused on deriving throughput and average packet delay, by assuming that the packet arrival pattern is Poisson. However, the throughput and average delay analysis does not provide detailed information for realizing statistical real-time communication. In statistical real-time communication, the tail distribution of packet delay is more important than such average performance parameters. Beuerman and Coyle [5] derived the tail distribution of packet delay in non-persistent CSMA/CD networks, assuming that the average waiting time for retransmission is sufficiently larger than packet transmission time. However, the characteristic of 1-persistent CSMA/CD networks with BEB is quite different from their result with the non-persistent version.

1.2.5 QoS Routing

There are basically two approaches to QoS routing: source-directed routing and distributed routing. In the source-directed approach [30, 55, 67, 69, 102], the source switch or router selects a path based on connection traffic parameters and available resources in the network. This approach can be refined into static or dynamic routing, depending on whether or not load information at each local node is used for path computation. The former uses static topology information in choosing a route. Although static QoS routing incurs low operational cost, it suffers from a poor connection-establishment rate. In contrast, in dynamic QoS routing, the information on the available resource of each link must be distributed throughout the network, so that any source can have access to the correct information on resources available in the network. The information distributed is often called *link-state*. By applying either Dijkstra's shortest-path algorithm [91] or Bellman-Ford shortest-path algorithm [67] based on link-state information, one can find a qualified route for the requested connection. Examples of source-directed routing protocols are the ATM Forum's PNNI standard [69] and a QoS extension to the OSPF protocol named QOSPF [30, 102]. Because of its high operational overhead in distributing and maintaining link-state information, source-directed routing may not scale well. To improve the scalability in large networks, a hierarchical approach can be adopted to distribute and manage link-state information [69]. In addition to the use of a hierarchical approach, efforts have been made to reduce link-state messages in order to control the overhead. Periodic or triggered distribution of link-state information is a typical example of this effort. Although this approach and the hierarchical approach reduce the overhead by disseminating less frequent and smaller link-state messages, they both cause the inaccuracy of link-state information, thus leading to undue routing and/or signaling failures [28]. A *routing failure* is said to occur if the source cannot find a route based on link-state information kept in its own database even if a qualified route exists. And a *signaling failure* occurs if one of the intermediate nodes on the route determined by the source cannot reserve the resources required for the requested connection's QoS.

In the case of distributed routing, the route is not calculated at the source node but chosen using a distributed routing protocol. In [76, 87, 91], distributed QoS routing algorithms which require each node to maintain a partial global information on current network load condition are presented. In [91], every local node is required to maintain a routing table which has an entry for every destination containing the next hop of the shortest-widest path. In [76], a delay vector and a cost vector are kept at each local node. The delay (cost) vector contains the next hop of the least-delay (least-cost) path for every destination. In this type of distributed routing, the global state must be kept up-to-date at each local node to prevent the undue routing and/or signaling failures as in the source-directed routing. As a result, link-state needs to be distributed as in the source-directed routing.

In [33, 80], distributed QoS routing algorithms in which local nodes are not required to exchange or maintain link-state information for the entire network. They are called flooding-based QoS routing. The source node simply multicasts each connection request to its neighbors, which then relay the request to their neighbors, and so on, until the request reaches the destination. In order to limit the number of request messages, the algorithm does not flood request messages through a link which is found unable to satisfy the QoS requirement that the requested connection presented. Although this approach incurs considerable operational overhead due to the large number of request messages, it still has its own merits. First, there is no need for link-state information distribution and shortest-path calculation, thus reducing operational overhead and implementation complexity. Second, nodes are not required to keep a database of link-state information, saving storage space. Finally, since the latest, thus the most accurate, information kept for each local link is used to determine whether it can accommodate a new connection or not, the algorithm can always find a qualified route, if any, thereby outperforming link-state routing in terms of connection-establishment rate.

1.3 Main Contributions of this Dissertation

The main objective of this dissertation is to provide a real-time communication service over ISPNs. It must not only be simple enough to work in a high-speed network environment, but also achieve good network utilization so that the cost of real-time communication service can be kept low. The main contributions of this dissertation are:

- A new traffic control scheme called the *Traffic-Controlled Rate-Monotonic Priority Scheduling* (TCRM) which provides user-requested delay guarantees in point-to-point networks. It satisfies both the simplicity and efficiency requirements of high-speed networks like ATM. This scheme requires traffic regulation at the user interface and scheduling at each link along the path. We divide the multiplexer of each link into two components: traffic controller and scheduler. Using this mechanism, TCRM (1) has efficiency close to PGPS in terms of channel admissibility; (2) is simple enough to operate in a high-speed switching environment like ATM networks; and (3) requires only a very small buffer space for each real-time channel.

- A framework to provide statistical real-time communication services for VBR video data in point-to-point networks based on TCRM [51]. Since TCRM was originally developed for providing deterministic real-time communication services and does not allow statistical multiplexing among real-time connections, we make slight modifications to TCRM so that it may allow statistical multiplexing among a set of real-time connections. By employing the histogram-based model [85] as the input traffic specification for VBR video stream along with the modified TCRM, we analytically derive a statistical bound for the average cell-loss ratio of each statistical real-time channel. Simulation results support our analysis.

- A methodology for providing the semi-real-time communication service in an ISPN. We first adopt the concept of *traffic envelope* to describe the source traffic characteristics, which is necessary for admission control of new connection requests and assessment of their resource reservation requirements. Instead of using a deterministic traffic envelope [14, 15, 24, 44], we employ a *statistical traffic envelope* derived from exploiting the statistical characteristics of source traffic. We take a direct approach based on the Central Limit Theorem rather than a bounding approach taken in previous work. The statistical traffic envelope enables us to greatly reduce the amount of network resources that need to be reserved and thus can accept more connection requests at the expense of a small percentage of packet losses. Second, we present a traffic regulation scheme which simplifies the derivation of statistical traffic envelopes in a multi-hop environment. In general, the traffic characteristic of a connection changes as the traffic travels through the network, and it is difficult to keep track of such changes and accurately describe the traffic characteristics at each node. Thus, we employ a traffic regulation scheme so as to preserve the source traffic characteristic at every intermediate node along the path of each connection.

- Derivation of a statistical bound on medium access time, and hence packet-delivery delay, by making several assumptions on input traffic and the functions of the MAC protocol of Ethernet. The derived bound is used for providing real-time communication over Ethernet. Our analysis considers the 1-persistent CSMA/CD MAC protocol with the Binary Exponential Backoff strategy (BEB) which is currently used in Ethernet. Using this analysis, we can provide a connection admission control for statistical real-time communication over Ethernet.

- A new flooding-based QoS routing scheme which incurs much lower message overhead yet yields a good connection-establishment rate, compared to the existing flooding-based algorithms. Moreover, unlike the source-directed approach, it does not require on-demand shortest-path calculation, thus lowering the operational cost.

1.4 Organization of the Dissertation

The dissertation is organized as follows. In Chapter 2, we examine the problem of providing deterministic real-time communication over point-to-point packet networks. We propose a new packet multiplexing scheme, TCRM, which provides deterministic delay bounds to individual real-time connections. The efficiency of the proposed multiplexer is shown to be comparable to PGPS while keeping the implementation complexity controllable. Chapter 3 considers the problem of providing statistical real-time communication over point-to-point packet networks. Employing a histogram-based model as the input traffic characteristic of bursty sources, we analyze packet loss statistics when a set of real-time connections are multiplexed over a common channel which is implemented by a modified TCRM. Through a simulation study using VBR video traces, we validate the effectiveness of our approach on statistical real-time communication. In Chapter 4, we propose a new service class, semi-real-time class, in order to provide more diverse choices to end users in ISPNs. To this end, we devise a statistical traffic envelope for an aggregate of semi-real-time connections using the

Central Limit Theorem (CLT). Through a simulation study, we show that semi-real-time communication service can provide reasonable QoS to the semi-real-time class while greatly improving network utilization compared to the case when real-time communication service is employed. Chapter 5 investigates the problem of providing statistical real-time communication over Ethernet. The medium access control protocol of Ethernet, CSMA/CD with binary backoff is modeled as a semi-Markov process. Analyzing the process, we calculate the probability of successful packet transmission over Ethernet, and derive the tail distribution of packet delay. In Chapter 6, a new flooding-based QoS routing algorithm is proposed. Detailed description of the flooding method is given. The effectiveness of the proposed QoS routing scheme is shown in terms of connection establishment rate, operational overhead and implementation complexity. Chapter 7 concludes the dissertation with a discussion of our results and future directions.

CHAPTER 2

TRAFFIC-CONTROLLED RATE-MONOTONIC PRIORITY SCHEDULING FOR DETERMINISTIC REAL-TIME COMMUNICATION

2.1 Introduction

The problem addressed in this chapter is the development of a cell multiplexing scheme that can provide per-channel delay guarantees in an ATM network environment. ATM networks have been drawing significant attention as a main technology for implementing B-ISDN due mainly to its potential for efficiency and flexibility. Therefore, ATM networks must be able to provide delay guarantees to individual virtual channels. In order to provide per-channel delay guarantees, each ATM link must be equipped with a special cell multiplexing scheme other than FIFO, as discussed in Chapter 1. The scheme must be simple enough to operate at high-speed while still achieving reasonable network utilization.

Two types of the ATM Adaptation Layer (AAL) are supposed to support such a delay-guaranteed cell delivery service: AAL1 and AAL2. AAL1 supports a connection-oriented service in which the bit rate is constant. Examples of this service include 64 Kbits/sec voice, fixed-rate uncompressed video and leased lines of private data networks. AAL2 supports a connection-oriented service in which the bit rate is variable. Examples are compressed video and interactive multimedia applications. However, it has not yet clearly specified how to implement these layers [84].

Although Delay-EDD, RCSP, RTC and PGPS can provide bounded end-to-end delays in ATM networks, they differ in input traffic specification, connection admissibility, and implementation complexity. Among them, PGPS is the most efficient in connection admissibility because of its optimal deadline scheduling and the efficiency of its input traffic specification.

However, it may not be a good candidate for realizing real-time communication in ATM networks because of its implementation complexity due to (i) the high overhead associated with virtual finish time calculation and deadline-based sorting and (ii) its large buffer space requirement. Since the number of real-time connections supported by a service discipline can be very large, the scalability of a service discipline is very important, especially when it is implemented in hardware. The large buffer space requirement of PGPS significantly degrades its scalability. In Section 2.4, the implementation issue of PGPS will be examined.

In this chapter, we propose a new cell multiplexing scheme called the *Traffic-Controlled Rate-Monotonic Priority Scheduling* (TCRM) that provides user-requested delay guarantees in ATM networks. As a rate-controlled service discipline, it has the same structure as RCSP, but it tries to simulate the behavior of PGPS. As a result, it achieves connection admissibility close to that of PGPS, and it satisfies both the simplicity and efficiency requirements of an ATM switch. This scheme requires traffic regulation at UNI and scheduling at each link along the path. We divide the multiplexer of each link into two components: traffic controller and scheduler. Using this mechanism, TCRM (1) has efficiency close to PGPS in terms of connection admissibility; (2) is simple enough to operate in a high-speed switching environment like ATM networks; and (3) requires only a very small buffer space for each real-time connection.

The chapter is organized as follows. Section 2.2 describes the mechanism of the proposed scheme and presents an admission-test algorithm for this scheme. Section 2.3 compares the proposed scheme with other schemes using MPEG-coded movie clips. In Section 2.4, we propose a simple implementation of the proposed scheme and discuss its implementation complexity. The chapter concludes with 2.5.

2.2 The Proposed Scheme

A *real-time channel* is defined as a virtual circuit through which a real-time communication service is provided. Before requesting the setup of a new real-time channel, the user must determine the parameters of its input traffic specification and present them to the network service provider along with its QoS requirements. Based on the user's input traffic specification and QoS requirements, the network service provider selects an appropriate path — that is, QoS routing — which traverses multiple nodes and links, and reserves the network resources needed to satisfy the user-specified QoS requirement. At run-time, the service provider guarantees the QoS using the reserved resources.

Before discussing the proposed cell-multiplexing scheme, we must first consider the method to characterize source traffic. One of the most important requirements of a good input traffic specification is that it should be "enforceable" with a simple traffic-policing scheme. If a complex input traffic specification is employed, a traffic-policing scheme that works for each connection at the network entrance will be expensive, and hence, it cannot be used for large-scale high-speed networks. Although many complex input traffic specifications like multiple leaky bucket regulator models or even empirical traffic envelopes were employed for providing end-to-end delay guarantees [44, 75, 92], the usefulness of these models in a real world is difficult to prove. In our approach, we employ a leaky bucket model as an input traffic specification. This model is simple enough to be used for traffic policing at runtime. One can imagine the leaky bucket model, simply called the (σ_i,ρ_i)-model, as placing a smoothing buffer at the network entrance whose size and average drain rate are σ_i and ρ_i, respectively, so that the burstiness of input traffic inside the network is limited, thus lowering the network resource-reservation requirement. In this case, the smoothing buffer works as a traffic policer at the network entry.

The parameter pair of the leaky bucket model for a VBR source, (σ_i,ρ_i), is obtained from the empirical traffic envelope of the source [92]. The empirical traffic envelope $E^*(t)$ is defined as the maximum amount of traffic that has arrived during any time interval of length t, i.e.,

$$E^*(t) = \max_{\tau \geq 0} A[\tau, \tau + t] \qquad \forall t \geq 0,$$

where $A[\tau, \tau + t]$ is the amount of traffic that has arrived during a time interval $[\tau, \tau + t]$. Then, (σ_i,ρ_i) is given as a parameter pair satisfying the following relation,

$$\sigma_i + \rho_i t \geq E^*(t) \qquad \forall t \geq 0.$$

There usually exist multiple pairs satisfying the above relation. For example, in Figure 2.1, two such parameter pairs are shown for a VBR traffic: (σ_1, ρ_1) and (σ_2, ρ_2). They both describe the source traffic while having different burstiness characteristics and throughput requirements. By choosing different parameter pairs, one can provide different delay bounds to a real-time channel under the same cell-service discipline. For instance, for PGPS, (σ_1, ρ_1)-model will provide a smaller delay bound than (σ_2, ρ_2)-model at the expense of higher resource requirements. So, we assume that input traffic specification parameters can be chosen depending on the performance requirement of an application.

Our scheme requires cooperation between the User Network Interface (UNI) and each ATM switch along the path in order to provide real-time communication service. The UNI

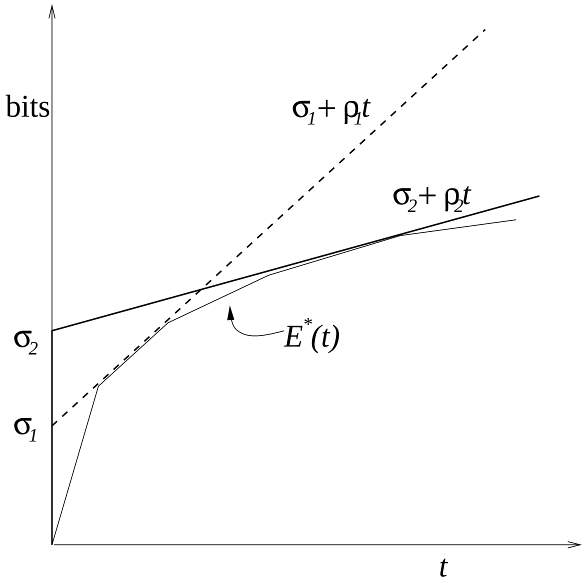


Figure 2.1: Two choices for (σ_i, ρ_i) parameter pair

regulates each channel i's traffic so as to bound the cell-arrival rate at the network entrance by ρ_i. The network service provider ensures that the requested channel i gets its (minimum) service rate ρ_i at every switch along the path through an appropriate admission control and run-time processing.

2.2.1 Traffic Shaping at UNI

Given the traffic model parameter pair (σ_i, ρ_i), the user requests a cell-transmission rate ρ_i from the network. After establishing the channel based on an appropriate admission test, the user begins to transmit its traffic according to the (σ_i, ρ_i) model. At the network entrance, the UNI regulates the incoming traffic in such a way that the maximum cell-transmission rate into the network is smaller than ρ_i, or the minimum cell inter-transmission time is larger than L/ρ_i, where L denotes the length of one cell (53 bytes). That is, when the k^{th} cell of channel i arrived at the UNI at time A_k, its transmission time, X_k, is calculated as:

$$X_k = \begin{cases} A_1 & k = 1 \\ \max(X_{k-1} + \frac{L}{\rho_i},\ A_k) & k \geq 2. \end{cases} \tag{2.1}$$

Until X_k, the UNI holds the cell in its buffer. Since one cell is permitted to be transmitted every time interval of length L/ρ_i, the minimum and maximum guaranteed service rates of the queue are ρ_i over an interval of length L/ρ_i. Since the traffic can arrive in a burst whose maximum size is σ_i, the UNI must have buffer a space of σ_i bits to avoid cell loss.

Unlike most approaches like PGPS, RCSP, and RTC, our scheme does not allow the

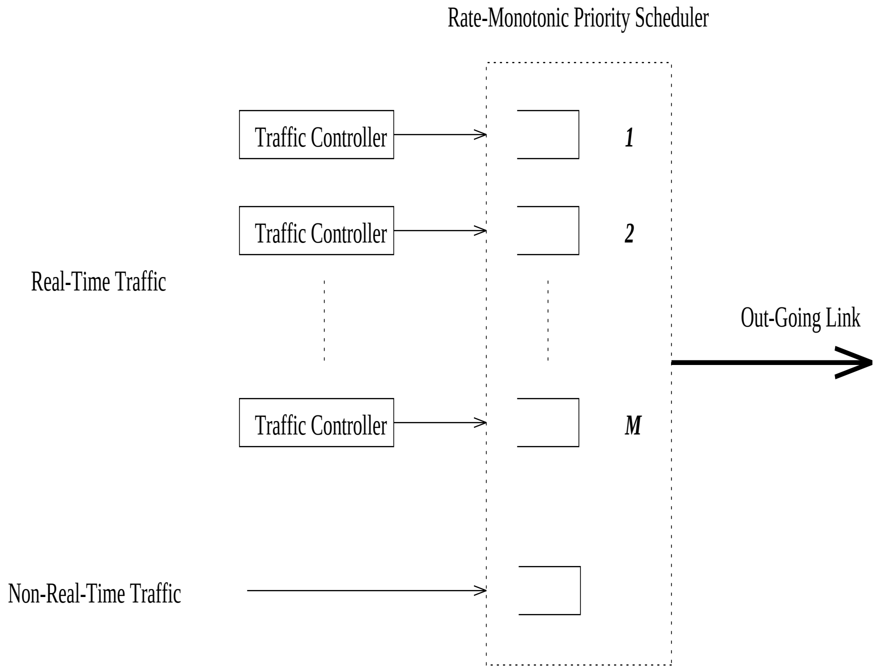


Figure 2.2: Structure of TCRM

burst of cells belonging to channel i to be instantaneously transmitted into the network. Such a strict non-work-conserving service discipline at the network entrance, as we shall see later, does not change the end-to-end delay bound, while significantly reducing the buffer requirement inside the network.

2.2.2 Traffic Regulation and Scheduling inside the Network

We model an ATM switch as an output-buffered multiple-input-multiple-output switch [16]. In this model, no cells are lost due to contention within the switch fabric and competition exists only among those cells sharing the same outgoing link. Assuming that the switching delay is negligible as compared to the queueing delay at an output buffer, we concentrate on controlling the queueing delay at the output buffer in order to achieve a bounded end-to-end delay.

As a rate-controlled service discipline, TCRM consists of *traffic controllers* and a *scheduler* (see Figure 2.2). A traffic controller is assigned to each individual real-time channel and the scheduler is shared by all the real-time channels.

Traffic Controller

The traffic controller executes the same traffic regulation function of the UNI. That is, it keeps the cell-arrival rate at the scheduler below ρ_i. It holds the incoming cells until their supposed arrival times, and then transfers them into the scheduler. This supposed arrival time is called the *logical arrival time* in [37, 95]. The logical arrival time of an incoming cell is calculated based on that of the previous cell of the same channel. Thus, the logical arrival time of the k^{th} cell at the n^{th} node, $X_{k,n}$, is calculated as:

$$X_{k,n} = \begin{cases} A_{1,n} & k = 1 \\ \max(X_{k-1,n} + \frac{L}{\rho_i},\ A_{k,n}) & k \geq 2, \end{cases} \tag{2.2}$$

where $A_{k,n}$ is the actual arrival time of the k^{th} cell at node n. Note that the inter-logical-arrival-time of the incoming cells is at least L/ρ_i. Assuming that the cell-arrival rate at the traffic controller is under ρ_i, we can ensure that at most one cell for channel i can exist in the traffic controller of channel i, since the traffic controller is permitted to transfer a cell every L/ρ_i secs. Hence, the traffic controller requires buffer space for storing only one cell.

Non-Preemptive Rate-Monotonic Priority Scheduling

After the cell stream of real-time channel i passes through the traffic controller, the cell-arrival rate at the scheduler of every switch is bounded by ρ_i. If we can provide the minimum cell drain rate ρ_i at the scheduler, the unbounded accumulation of cells at the scheduler will never happen. In that case, the minimum cell inter-arrival time and the cell delay bound at the scheduler will be given as L/ρ_i. In order to provide the minimum cell-drain rate ρ_i, we employ the well-known rate-monotonic priority scheduling as the scheduling policy of TCRM. Liu and Layland [54] proved that the rate-monotonic priority scheduling is optimal among all fixed-priority scheduling policies when the deadline of each task is the same as the task period. If we treat a cell as a "task", then the rate-monotonic priority cell scheduling is optimal among fixed-priority scheduling policies[1] that achieve the guaranteed throughput, because the cell inter-arrival period is the same as the cell-delivery deadline ($=L/\rho_i$). This scheduling policy assigns higher priority to channels with higher request rates, i.e., higher ρ_i.

Liu and Layland's analysis [54] is based on a preemptive scheduling policy. However, preemptive scheduling is not desirable for cell transmissions since if in-progress transmission

[1] Although fixed-priority scheduling policies are less efficient than deadline-scheduling policies in terms of network utilization [54], we prefer the implementation simplicity of the rate-monotonic priority scheduling.

of a cell is interrupted, the cell will be lost and has to be retransmitted, thus wasting network resources. Thus, we have to use the non-preemptive rate-monotonic priority scheduling policy as the cell scheduling policy.[2] The non-preemptive rate monotonic priority scheduler assigns a priority level to each real-time channel according to its required throughput and in-progress cell transmission will not be preempted. As a result, the scheduler provides the minimum throughput ρ_i to each channel i.

Since the cell inter-arrival time is larger than, or equal to L/ρ_i and one cell is permitted to be transmitted every L/ρ_i, at most one cell of channel i can stay in the scheduler at any time. Hence, the scheduler needs a buffer of one cell for each real-time channel.

Let's look at how the rate-monotonic priority scheduler works. In Figure 2.2, the scheduler transmits cells according to ρ_i values. If there are no cells belonging to real-time channels, cells from non-real-time traffic queue are transmitted. In any case, the cells held at the traffic controllers are not allowed to be transmitted.

2.2.3 Admission Control

In order to provide throughput guarantees at the non-preemptive rate-monotonic priority scheduler, we need an appropriate admission control for real-time channels. The admission-control test involves every node along the path of the real-time channel. If any node along the path fails this test, the channel request must be denied.

At the scheduler, the throughput guarantee is made not for bit-by-bit, but for cell-by-cell. That is, when each cell arrives at the scheduler with the minimum cell inter-arrival time, L/ρ_i, which is guaranteed by the traffic controller, the scheduler must complete the transmission of the cell before the next cell's earliest arrival time, which is the current cell's arrival time plus L/ρ_i. We need a schedulability test to verify whether or not the worst-case delivery time is smaller than, or equal to, the local delay bound L/ρ_i. Using a method similar to Kandlur's [37], we derive the schedulability test for the proposed scheme. Consider a set of real-time channels $\{i,\ i = 1, \ldots, M\}$ which share a common link ℓ, where M is the number of existing real-time channels on link ℓ. Denote the throughput of channel i by ρ_i, and assume that channels are indexed in the descending order of priority so that $\rho_i \geq \rho_j$ if $i < j$. Then the schedulability test is given as:

$$\sum_{j=1}^{i-1} C \lceil \frac{L/\rho_i}{L/\rho_j} \rceil + 2C \leq \frac{L}{\rho_i} \quad \text{for } i = 1, \ldots, M, \tag{2.3}$$

[2] Employing a non-preemptive policy doesn't affect the optimality of the rate-monotonic priority scheduling since the non-preemptive rate-monotonic priority scheduling policy is optimal among non-preemptive fixed-priority policies, which can be proved using the same arguments in the proof of Theorem 2 in [54].

where C is one cell transmission time, $\frac{L}{\rho_i}$ is the link delay bound of channel i's cell, and all channels j, $1 \leq j < i$, have higher priority than channel i. Note that the first term of Eq. (2.3) denotes the sum of all the transmission times of cells belonging to the channels of higher priority than channel i in the worst case.[3] C of the second term denotes the time to complete in-progress cell transmission. The other C denotes the transmission time of a cell belonging to channel i. Conceptually, the schedulability condition implies that the transmission of a cell of channel i must be finished within its link delay bound even in the worst case. If the schedulability condition fails, the cells of channel i cannot be transmitted in the worst case. Therefore, the schedulability condition is also the necessary condition.

Using this argument, we can show that TCRM emulates circuit-switching in the cell level. Let's define $T_i(t,s)$ as channel i's traffic transmitted over a link during a time interval $[t,s)$ for any t and s such that $t \leq s$. Then,

$$L\lfloor\frac{s-t}{L/\rho_i}\rfloor \leq T_i(t,s) \leq L\lceil\frac{s-t}{L/\rho_i}\rceil \tag{2.4}$$

The lower bound is derived from the fact that a local delay bound is guaranteed by the scheduler and the upper bound comes from traffic regulation by the traffic controller. Therefore, the average traffic service rate of channel i, $R_i(t,s)$, during the interval $[t,s)$ is given as:

$$R_i(t,s) = \rho_i, \tag{2.5}$$

where $s-t = k(L/\rho_i)$ and $k = 1, 2, \ldots$. In other words, the throughput of channel i is guaranteed to be ρ_i during any time interval of length L/ρ_i, implying that TCRM emulates circuit-switching (TDMA) in the cell level. Since, however, our scheme allows best-effort traffic to be transmitted when time slots reserved by bursty real-time connections are not used, it does not lose statistical multiplexing gain of ATM networks unlike TDMA.

Next, we derive a simple admission-control test from Eq. (2.3). The schedulability test can be rewritten as:

$$\sum_{j=1}^{i-1}\lceil\frac{\rho_j}{\rho_i}\rceil + 2 \leq \frac{L}{\rho_i C} \quad \text{for } i = 1, \ldots, M.$$

When a new channel of priority k and cell service rate ρ'_k is requested, we need to conduct the following schedulability test for all $i \geq k$:

$$\lceil\frac{\rho'_k}{\rho_i}\rceil + \sum_{j=1}^{i-1}\lceil\frac{\rho_j}{\rho_i}\rceil + 2 \leq \frac{L}{\rho_i C}.$$

[3] Here, the worst case means that all the channels of higher priority than channel i generate their cells concurrently with channel i.

By moving the second and third terms from the left-hand side to the right-hand side, we get

$$\lceil \frac{\rho'_k}{\rho_i} \rceil \leq \frac{L}{\rho_i C} - \sum_{j=1}^{i-1} \lceil \frac{\rho_j}{\rho_i} \rceil - 2. \tag{2.6}$$

Now, let's define the residual link capacity sequence R_i:

$$R_i = \frac{L}{\rho_i C} - \sum_{j=1}^{i-1} \lceil \frac{\rho_j}{\rho_i} \rceil - 2, \quad i = 1, \ldots, M. \tag{2.7}$$

Notice that R_i indicates how many more channels with the throughput requirement ρ_i can be accepted. Using R_i, we can construct a new schedulability test algorithm as follows.

<u>**Schedulability Test**</u>

1. When a new channel of cell service rate ρ'_k and priority k is requested,

   Step 1. Calculate R_k using ρ'_k. If $R_k \geq 0$, go to Step 2. Otherwise, reject the channel request.

   Step 2. Check if $\lceil \frac{\rho'_k}{\rho_i} \rceil \leq R_i$ for $i \geq k$. If that is true, go to Step 3. Otherwise, reject the channel request.

   Step 3. Update R_i's and ρ_i's for $i \geq k$ as:

   $$R_{i+1} \leftarrow R_i - \lceil \frac{\rho'_k}{\rho_i} \rceil$$

   $$\rho_{i+1} \leftarrow \rho_i$$

2. When an existing channel k is disconnected,

   Update R_i's and ρ_i's for $i > k$ as:

   $$R_{i-1} \leftarrow R_i + \lceil \frac{\rho'_k}{\rho_i} \rceil$$

   $$\rho_{i-1} \leftarrow \rho_i$$

By storing the residual link capacity sequence R_i, we can reduce the amount of calculation for the channel schedulability test. The computational complexity of the above algorithm is $O(M)$.

2.2.4 Bounding End-to-End Delays in Multi-Hop Connections

Using the fact that TCRM guarantees the minimum throughput ρ_i for channel i, we can derive the end-to-end delay bound of channel i. Given the input traffic specification (σ_i, ρ_i) of channel i, during any time interval of length t, the amount of traffic generated by the user may not exceed $\sigma_i + t \cdot \rho_i$. For convenience, we assume that σ_i is an integer multiple of the length of one cell, L. Due to the UNI's traffic regulation, the cell-arrival rate at the network entrance is limited by ρ_i and the burst is held at the buffer of the UNI. We assign a buffer space of σ_i for channel i.

We now show the boundedness of end-to-end delivery delays. First, we show that the queue size at the input buffer of the UNI cannot be larger than σ_i. During a time interval $[s, t)$ for any s, t such that $s \leq t$, the maximum amount of traffic that has arrived at the UNI, $x_i(s, t)$, is given by:

$$x_i(s, t) = \sigma_i + \rho_i(t - s)$$

under the leaky bucket model. However, since the traffic is transmitted cell-by-cell, the maximum number of cells that have arrived at the UNI is given by

$$x_i(s, t) = \sigma_i + L\lfloor \frac{s - t}{L/\rho_i} \rfloor.$$

During the same interval, the minimum number of cells transmitted at the UNI, $y_i(s, t)$, is given by

$$y_i(s, t) = L\lfloor \frac{s - t}{L/\rho_i} \rfloor,$$

which comes from the fact that channel i is guaranteed to have the minimum throughput ρ_i in the cell level. Therefore, the maximum backlog at the input buffer during the interval $[s, t)$ is given by $B_{max} = x_i(s, t) - y_i(s, t) = \sigma_i$, and the maximum number of cells that can exist at the buffer is σ_i/L.

Using this fact, we can derive the following theorem on the end-to-end delivery delay bound in ATM networks.

Theorem 2.1 *If a real-time channel i is specified by (σ_i, ρ_i) and its guaranteed throughput is ρ_i, then the end-to-end delivery delay of any cell belonging to channel i is bounded by*

$$D_i = \frac{\sigma_i}{\rho_i} + N\frac{L}{\rho_i} + \sum_{k=1}^{N} e_k, \tag{2.8}$$

where N is the number of hops that channel i must take and e_k is the propagation delay at the k^{th} link.

Proof: The proof of this theorem is straightforward, since the end-to-end delivery delay D_i is given by

$$\begin{aligned} D_i &= \text{the maximum queueing delay at the UNI} \\ &+ \sum_{k=1}^{N} \text{the maximum queueing delay at the } k^{th} \text{ node} \\ &+ \sum_{k=1}^{N} \text{the propagation delay at the } k^{th} \text{ link} \\ &= \frac{\sigma_i}{L} \cdot \frac{L}{\rho_i} + N\frac{L}{\rho_i} + \sum_{k=1}^{N} e_k. \end{aligned} \tag{2.9}$$

The first term in Eq. (2.9) comes from the fact that the maximum number of cells in the buffer of the UNI is σ_i/L, and each cell's queueing delay is L/ρ_i because of the UNI's traffic regulation. The reason the delay at the traffic controller of each link is omitted in the second term, is that the traffic controller does not hold any cell if it has arrived at the latest arrival time from the previous node. In such a case, the logical arrival time is the same as the actual arrival time. Even if the cell has arrived earlier than its latest arrival time, it is held at the traffic controller only until its latest arrival time. This is self-evident from the calculation of the logical arrival time in Eqs. (2.1) and (2.2). The third term is needed because of the propagation delay at each switch. By arranging the equation, we obtain

$$D_i = \frac{\sigma_i}{\rho_i} + N\frac{L}{\rho_i} + \sum_{k=1}^{N} e_k. \qquad \square$$

In general, because the burst size will be much larger than the length of one cell, the end-to-end delay bound D_i will be dominated by σ_i/ρ_i.

2.3 Comparative Evaluation of Channel Admissibility

To demonstrate the efficiency of TCRM, it is necessary to investigate the maximum link utilization by real-time traffic. Liu and Layland proved that the least upper bound of link utilization of the preemptive version of the rate-monotonic priority scheduling is approximately 70% when the number of real-time channels is large, while the entire link capacity can be used by the deadline-driven scheduling [54]. The least upper bound of link utilization does not mean that link utilization greater than the bound cannot be achieved. A higher utilization can be achieved if task periods are suitably related to each other. But still its link utilization is lower than that of the deadline-driven scheduling. This is similar to the case of non-preemptive rate-monotonic priority scheduling, although its link utilization is lower than that of the preemptive version. For example, the maximum link utilization of non-preemptive rate-monotonic priority scheduling for one channel is 50%, which can be calculated easily from Eq. (2.3), while the preemptive version can achieve 100%. From this, we may conclude that the link utilization of our scheme can be very low compared to PGPS in some cases. As in a preemptive version, however, higher utilization can also be achieved for a non-preemptive version. Rather than deriving a theoretical link utilization bound, in this section we investigate empirical link utilizations of PGPS, RTC, RCSP, and TCRM using traces of MPEG-compressed videos.

As stated earlier, both RTC and PGPS adopt the (σ_i,ρ_i) model while RCSP employs the (X_{min}, X_{ave}, I, S_{max}) model. Here S_{max} is the length of an ATM cell, i.e., 53 bytes. Although RTC adopts the leaky bucket model as its input traffic specification, it interprets ρ_i not as the average cell-arrival rate but as the maximum traffic-arrival rate [38]. For this reason, RTC can accept real-time channels only when the sum of the peak traffic arrival rates is smaller than the link capacity.

We consider a homogeneous ATM network which has 11 serially-connected nodes as shown in Figure 2.3. We use a multi-hop network in order to account for inter-dependency of delays at different links under PGPS and TCRM (see Eq. (2.8)). Also, we consider such a simple network configuration rather than a more general configuration in which real-time channels are routed and switched at intermediate nodes because the end-to-end delay bound of a real-time channel is not affected by switching and routing because of its *Firewall* property [104]. As long as each intermediate link provides a local delay bound, the end-to-end delay bound of a real-time channel does not change regardless whether or not it joins and leaves other channels at the intermediate links. In Figure 2.3, the first node is the sender and the 11^{th} is the receiver. Thus, the path from the sender to the receiver has 10 intermediate links. Each link has the transmission bandwidth of 100 Mbits/sec. The traffic data used in the calculations are obtained from two MPEG-coded movie clips: *Starwars* (Sequence 1) and *Honey, I Blew Up the Kids* (Sequence 2). An MPEG-coded video yields an exemplary type of VBR traffic. These two sequences consist of frames generated once every 1/30 sec. The frame sizes of two sequences are plotted in Figure 2.4, where the x-axis is the frame number and the y-axis is the frame size measured in cells. In this example, we consider two cases: one is multiplexing homogeneous traffic, and the other is multiplexing heterogeneous traffic. In multiplexing homogeneous traffic, we consider either sequence alone and calculate the end-to-end delay bound against the number of real-time channels established. In multiplexing heterogeneous traffic, we attempt to establish real-

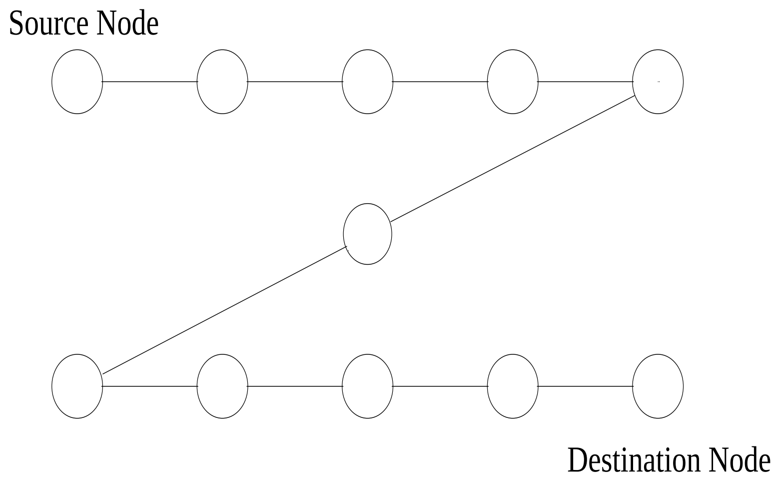
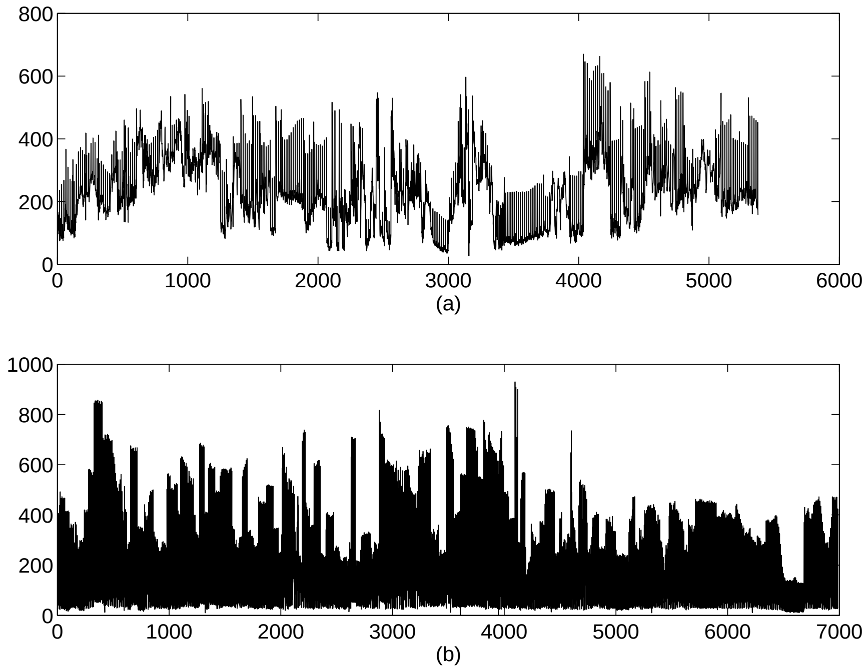


Figure 2.3: An ATM network model



(a) *Starwars* (b) *Honey, I Blew Up the Kids*

Figure 2.4: Frame sizes of compressed video sequences

time channels for both sequences together, and calculate the number of real-time channels of Sequence 1 that can be established while varying the number of real-time channels of Sequence 2.

2.3.1 Multiplexing Homogeneous Traffic

First, let's consider Sequence 1. The maximum number of cells per frame is 670, and the average number of cells per frame is 227.69. Since the size of a cell is 424 bits (53 bytes $\times$ 8 bits/byte = 424 bits), the peak traffic-generation rate is 8.52 Mbits/sec, and the average traffic generation rate is 2.89 Mbits/sec. We set the user's end-to-end delay requirement to $10 \times 1/30$ sec. Then, the local delay bound at each link is 1/30 sec. We now want to compute the number of real-time channels that can be established. By reserving a bandwidth equal to the peak traffic generation rate for each channel, the number of channels that can be established in a circuit-switched network is

$$\lfloor \frac{100 \text{ Mbits/sec}}{8.52 \text{ Mbits/sec}} \rfloor \;=\; 11(\text{channels}).$$

To calculate the end-to-end delay bounds under TCRM, we first determine the burst size σ_i and the required throughput ρ_i from the frame size sequence $\{f_i\}_{i=1,\ldots,N}$, where N is the number of frames in the sequence. We assume that the link capacity is evenly distributed to the real-time channels on the link. Then, the throughput ρ_i is obtained by dividing the link capacity by the number of channels plus one, which is needed to pass the schedulability test. Note that this throughput must be larger than the average traffic-generation rate of a real-time channel (here 2.89 Mbits/sec) to bound end-to-end delays. The bucket size σ_i is given as the maximum number of cells accumulated at the UNI when the average cell drain rate is ρ_i. Then,

$$\sigma_i = \max_m \max_n \lceil \sum_{l=n}^{n+m} (f_l - \frac{\rho_i}{30L}) \rceil \qquad m, n \in \{1, \ldots, N\}. \tag{2.10}$$

By substituting σ_i and ρ_i into Eq. (2.8), we calculate the end-to-end delay bounds for Sequence 1. In this example, we assume that the propagation delay is negligible. The results are plotted in Figure 2.5: one can establish a maximum of 21 channels, given that the user-requested end-to-end delay bound is 1/3 sec.

For PGPS, we derive (σ_i, ρ_i) in the same way as we did for TCRM, except that we obtain ρ_i by dividing the link capacity by the number of channels, since PGPS uses the link utilization test for its admission control. Using the formula in [65], we calculate the end-to-end delay bounds. Given that the user-requested bound is 1/3 sec, a maximum of 22 channels can be established.

With RTC, we can establish 11 channels based on the schedulability test in [37]. This number is the same as that of a circuit-switching network. This is due to the fact that RTC interprets ρ_i as the maximum traffic-arrival rate, as stated earlier. The end-to-end delay bounds are derived based on the worst-case response time in [37].

For RCSP, we use here only one priority level with a local delay bound of 1/30 sec because the characteristics of all the channels are assumed to be homogeneous. In [99], the local delay bound is calculated differently, depending on whether the peak utilization exceeds 1 or not. We calculated the parameters X_{min}, X_{ave} and I from the frame size sequence as follows: First, the minimum cell inter-arrival time X_{min} is simply given by a frame interval divided by the maximum frame size. For the average cell inter-arrival time X_{ave}, we must consider the average cell-arrival rate. As with TCRM and PGPS, the average traffic arrival rate is given by the link bandwidth divided by the number of real-time channels established. Then, the average cell-arrival rate is the average traffic arrival rate divided by the cell size, and X_{ave} is given by the reciprocal of the average cell-arrival rate. I is determined so that the average cell inter-arrival time during any time interval over I is larger than X_{ave}. According to our calculation, I becomes very large as the peak link utilization exceeds 1. This is the reason that the delay bound jumps suddenly as the peak link utilization exceeds 1. Given that the user-requested end-to-end delay bound is 1/3 sec, a maximum of 15 channels can be established.

In Figure 2.5, until the 11^{th} channel is established, all four schemes show reasonable end-to-end delay bounds. However, when the peak utilization exceeds 1 (i.e., the number of channels is greater than 11), RCSP is shown to exhibit rapidly-increasing delay bounds. RTC does not guarantee bounded delays when the peak utilization exceeds 1. TCRM and PGPS show reasonable end-to-end delays even when the number of real-time channels is fairly large. The reason why TCRM is slightly less efficient than PGPS is that TCRM employs a fixed-priority scheduling, while PGPS adopts an optimal deadline-based scheduling.

Next, let's consider Sequence 2. The maximum number of cells per frame is 930, and the average number of cells per frame is 118.29. Thus, the peak traffic-generation rate is 11.83 Mbits/sec, and the average traffic generation rate is 1.50 Mbits/sec. Then, the number of channels that can be established in a circuit-switched network is 9, and the maximum number of channels that can provide bounded delays based on the average traffic generation

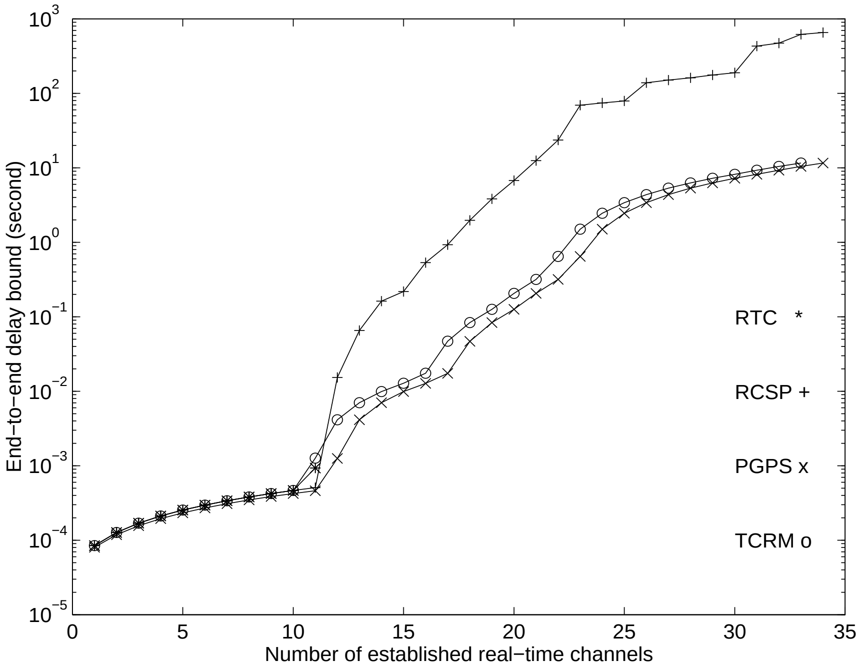


Figure 2.5: Achievable end-to-end delay bounds: ***Starwars***

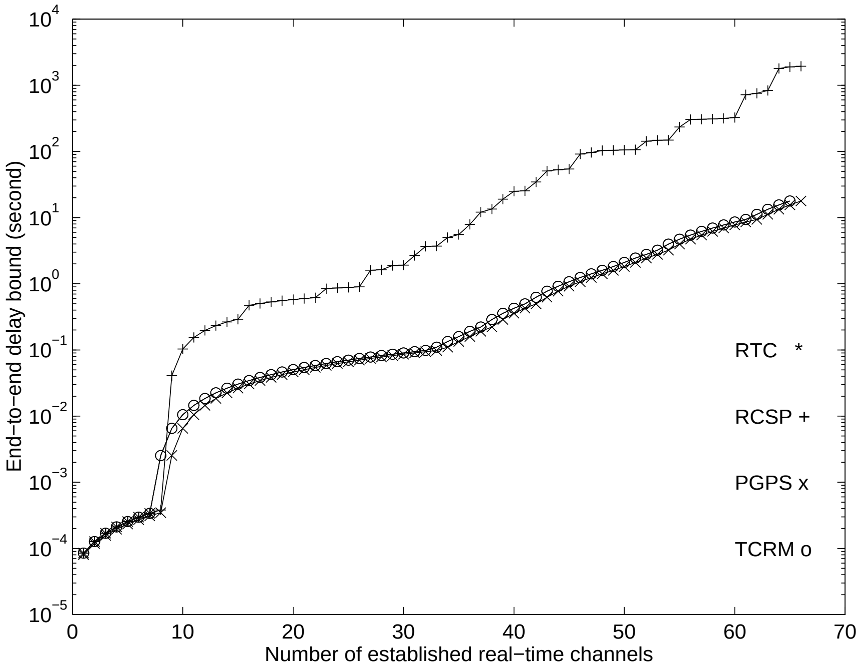


Figure 2.6: Achievable end-to-end delay bounds: ***Honey, I Blew Up the Kids***

		Sequence 1	Sequence 2
TCRM	σ_i	9.761×10^5 (bits)	7.334×10^5 (bits)
	ρ_i	4.762×10^6 (bits/sec)	2.564×10^6 (bits/sec)
PGPS	σ_i	9.761×10^5 (bits)	7.334×10^5 (bits)
	ρ_i	4.762×10^6 (bits/sec)	2.564×10^6 (bits/sec)
RCSP	X_{min}	4.975×10^{-5} (sec)	3.584×10^{-5} (sec)
	X_{ave}	6.36×10^{-5} (sec)	6.784×10^{-5} (sec)
	I	0.1 (sec)	0.1 (sec)

Table 2.1: Traffic description parameters for TCRM, PGPS and RCSP

is 66. The end-to-end delay bounds are plotted in Figure 2.6. This figure shows a result similar to that of Sequence 1.

2.3.2 Multiplexing Heterogeneous Traffic

In order to compare the channel admissibility in a heterogeneous environment, we calculate the number of establishable real-time channels of Sequence 2 with a fixed number of real-time channels of Sequence 1 already established. To this end, we need to fix the traffic description parameters. We use the same network model and user requirements considered in the homogeneous case, and we omit the analysis of RTC since its channel admissibility is quite low, as is shown in the case of multiplexing homogeneous traffic.

Based on the user requirements and traffic characteristics, we derived the traffic description parameters for TCRM, PGPS, and RCSP as shown in Table 2.1. We obtain these values from the case when the maximum number of real-time channels are established, given that the user end-to-end delay bound is 1/3 sec. For example, the parameters of RCSP for Sequence 1 are derived when the number of real-time channels is 15 (see Figure 2.5). Interestingly, the parameters of TCRM and PGPS have the same values. This is because the end-to-end delay bounds of both methods are very close to each other (see Eq. (2.8) and the end-to-end delay bound equation of PGPS [64]). Note that Sequence 1 requires a higher bandwidth than Sequence 2 although the peak traffic generation of Sequence 2 is higher than that of Sequence 1. This indicates that Sequence 2 is more bursty than Sequence 1.

Using the derived traffic parameters, we first establish a fixed number of real-time channels consisting of Sequence 1, then determine the maximum number of establishable real-time channels consisting of Sequence 2. As the number of channels of Sequence 1 increases,

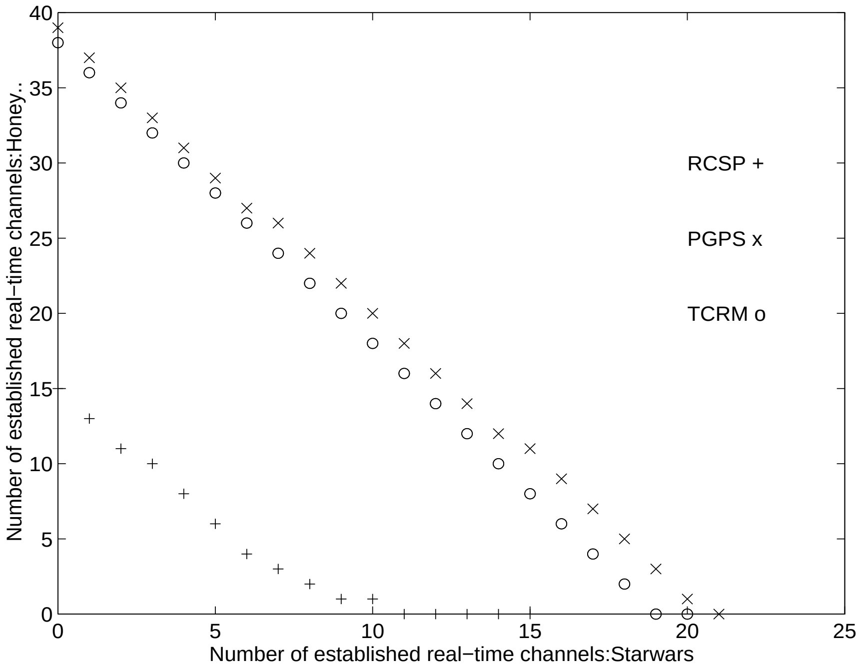


Figure 2.7: Channel accommodability for the heterogeneous case

that of Sequence 2 decreases. Figure 2.7 shows the results for RCSP, PGPS and TCRM. Note that the channel admissibility of TCRM is very close to that of PGPS while that of RCSP is not very good.

In this example,[4] we have verified that the performance of PGPS is superior to that of RCSP and RTC. This is, as pointed out in [25], because the end-to-end delay bounds are simply given as the sums of the worst case delays at intermediate links along the path of a channel in RCSP and RTC. They ignore the inter-dependency of link delays at different intermediate links unlike PGPS. In PGPS [64], if we ignore minor terms, the end-to-end delay bound is given by

$$D_{end-to-end} = \frac{\sigma_i + KL}{\rho_i}, \tag{2.11}$$

where K is the number of hops of the path that channel i takes. Thus, the worst-case cell delay under PGPS consists of the time needed to clear the maximum burst of size σ_i at a drain rate ρ_i and the time needed to transmit an ATM cell at ρ_i at all the intermediate links. Although the end-to-end delay bound of TCRM in Eq. (2.8) consists of the same terms used in Eq. (2.11) except propagation delay, its physical meaning is different from that of PGPS. While backlogs can be built up at any intermediate link in PGPS because

[4] Delay-EDD [19] is omitted in this comparison, as it shares many similarities with RTC and RCSP. In particular, it shares the same input traffic specification with RCSP. But its admission control test for deterministic performance guarantees is based solely on the peak traffic generation rate, and thus, its channel admissibility is very poor.

of its work-conserving policy, bursts are always held at the UNI in TCRM. Also, at the intermediate links, both the maximum and the minimum throughput guarantees are given as ρ_i in TCRM, whereas only the minimum throughput is guaranteed in PGPS.

Holding the burst at the network entry was also considered in [25]. Although their work has been done independently of our work, their approach and ours have many similarities. Besides the function of the UNI, service disciplines employed inside the network share the same structure. Their service discipline, Rate-Controlled Service with Earliest-Deadline-First (RCS-EDF), consists of rate controllers and a link scheduler like TCRM. The only difference is that their link scheduler is a deadline-based scheduler while our link scheduler is a rate-monotonic priority scheduler. Whereas they gave a theoretical result on how to simulate the performance of PGPS using a rate-controlled service discipline, our goal is to provide a service discipline which is simple to implement while still achieving reasonably good performance.

2.4 Implementation

In order to provide deterministic real-time communication services in large-scale high-speed ATM networks, it is very important to implement a fast and scalable cell scheduler in hardware. Let's consider the following example to get a feel for operation speed. In a 2.5 Gbps ATM network, a cell can be transmitted once every 0.17 μsecs. In the worst-case, the link scheduler must determine which cell to transmit next within 0.17 μsecs, while accepting new cells from all incoming links within the same 0.17 μsec period. If the link scheduler cannot keep up with link speed, it should not be used for real-time communication in high-speed networks (as it will keep the link idle thus under-utilizing the link bandwidth). Scalability is another important factor since switches in backbone networks can easily have hundreds of thousands of concurrent real-time channels with different delay requirements. So, the hardware implementation of a service discipline must be scalable with respect to the number and the type of real-time channels.

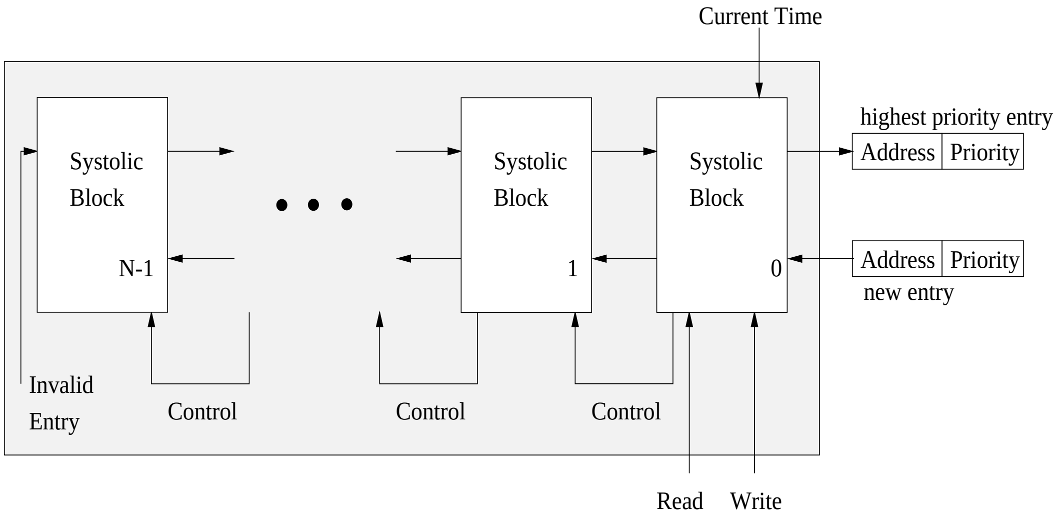
The two components of TCRM, the traffic controller and the rate-monotonic priority scheduler, are discussed in this section.

2.4.1 Traffic Controller

As stated in Section 2.2, a traffic controller is assigned to each real-time channel. In an actual hardware implementation, however, one traffic controller is made responsible for all real-time channels for cost and flexibility reasons. Had one traffic controller been assigned to each real-time channel, a traffic controller must be assigned (de-assigned) every time a real-time channel is established (terminated). Although the traffic controller can be implemented with a modified calendar queue as discussed in [51], its scalability is limited for two reasons. First, the number of the time bins of the calendar queue depends on the range of logical arrival times. For instance, if the link capacity is 1 Gbps and if the minimum throughput that can be allocated to a real-time channel is 1 Kbps, a cell of a particular channel may arrive once every 10^6 cell-transmission times. Then, the number of the time bins needed is at least 10^6 if a single time bin corresponds to one cell-transmission time. As the link capacity grows while the minimum allocable throughput is fixed, the number of the time bins also increases. Although a hierarchical structure can mitigate this problem, the result is that the hardware implementation becomes more complex. Second, the storage requirement of each time bin is the maximum number of real-time channels that can be established, assuming that cells are directly stored in the time bins. Consider the worst-case scenario that all the channels have cells with the same logical arrival times. In such a case, all cells are stored in the same time bin, leaving the other time bins empty. In order to avoid cell loss, each time bin must be able to store a cell per channel. In this example, the number of real-time channels that can be established simultaneously is 10^6, and thus, the total storage requirement of the calendar queue is 10^{12} cells. Since the number of cells that can co-exist in the queue is at most 10^6, the memory utilization will be extremely poor ($= 10^{-6}$). In addition, such an excessive storage requirement limits the scalability of a traffic controller implemented with the calendar queue.

Employing a dynamic priority queue is one way to avoid this inefficient memory usage. In the dynamic priority queue approach, even if all cells have the same logical arrival time, they are not stored in the same time bin but stored in separate entries while keeping the same priority. Thus, the number of entries needed is the maximum number of cells that can simultaneously reside in the traffic controller. In this approach, each cell is indexed with its logical arrival time, and the priority queue sorts cells using their indices. Thus, the cell with the earliest logical arrival time is assigned the highest priority. Only the cell in the highest priority level will be checked for its eligibility. If the cell is *on-time*, i.e., its logical arrival time equals the current time, it will be transferred to the scheduler. Otherwise, the cell is held in the priority queue until its logical arrival time is reached.

There are several hardware implementations of a dynamic priority queue [9, 10, 52, 53, 68, 73, 88]. Among them, the systolic array priority queue [52, 53], despite its high im-



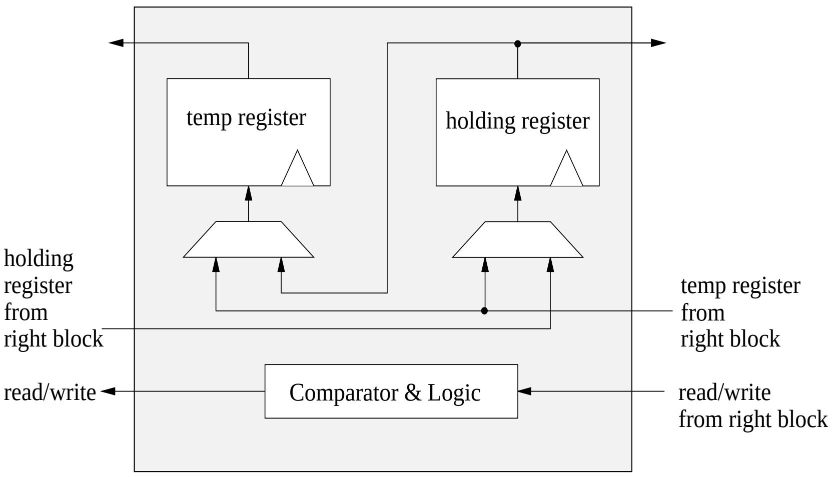


Figure 2.8: The systolic array priority queue (above) and systolic array block (below)

plementation cost, is the only candidate known to date that satisfies both the speed and scalability requirements. Hence, we use a systolic array priority queue to design the traffic controller. The example traffic controller based on the systolic array priority queue in Figure 2.8 consists of an array of identical blocks, with each block holding a single entry. When a new cell of channel i arrives, it is stored in a shared memory, and its logical arrival time is calculated using Eq. (2.2). The calculation can be done using a simple adder which needs two state variables: channel i's minimum cell inter-arrival time, (L/ρ_i), and the time when the last cell of channel i was transferred to the scheduler. The former is constant for channel i and the latter is recorded every time a cell is transferred to the scheduler. Then, the address and the logical arrival time of the cell are recorded in the address field and the priority field of the new entry in the systolic array priority queue, respectively. That is, the logical arrival time of the cell is assigned to the priority index of the entry whose address field contains the location of the cell in the shared memory. When a newly-arrived cell is inserted into the new entry of the queue, only the 0^{th} block compares the priority of its entry with that of the new entry. During the next cycle the cell with the larger logical arrival time is inserted into the left neighbor's block. The left neighbor's block repeats the same process of comparing and sending the lower-priority entry to the next block each cycle. Thus, the systolic array does not become fully sorted until several cycles after inserting the newly-arrived cell. However, both insertion and removal of an entry are constant-time operations from the viewpoint of an outgoing link. Because each block passes lower-priority entries to the next block, the 0^{th} block always holds the highest-priority entry (i.e., the one with the smallest logical arrival time) in the queue. The logical arrival time of the entry in the 0^{th} block is compared with the current time. If they match, the entry is removed from the priority queue, and the corresponding cell is *logically transferred* to the scheduler, meaning that the cell's address and the throughput reserved for channel i are transferred into the scheduler. Once an entry is removed from the block, the 0^{th} block gets the entry from its left neighbor block, performing a right shift on the entire queue.

Each systolic array block consists of a holding register, which stores the entries in a sorted order, as well as a temporary register, which holds entries en route to the next block to the left. The lower-priority entry in a block is passed during an enqueue operation. Multiplexors, a comparator, and decision logic also make up the rest of the block. The diagram of a systolic array block is given in Figure 2.8. Queue capacity is increased by adding more blocks to the end of the queue (without using a central controller), thus making the systolic array priority queue scalable. Also, there is no bus loading problem which usually limits the scalability of the other priority queue structures [62]. As a result, the traffic controller built with a systolic array priority queue has good scalability.

The number of priority levels (i.e., the number of possible logical arrival times) affects the cost and delay of the comparator. It increases linearly with each extra bit in the priority field. The delay of the comparator can be controlled by employing a parallel comparator [8] at the expense of higher implementation cost. Another drawback is that the systolic array priority queue requires twice as many registers as the other structures [9, 10, 52, 53, 68, 73, 88] because of the temporary register in each block. Moon *et al.* proposed a modified systolic array priority queue in order to solve this problem [62] which can be used to implement a traffic controller.

2.4.2 Rate-Monotonic Priority Scheduler

In the rate-monotonic priority scheduler a channel's priority is fixed according to its throughput requirement, and thus, it can be implemented with a fixed priority queue, e.g., a FIFO priority queue [10]. In this approach, each channel is assigned its own private FIFO queue. Thus, when a cell of channel i arrives, it is inserted into the channel's FIFO queue. During a cell-transmission stage, a priority encoder scans the heads of these FIFO queues (in decreasing order) and removes a cell from the first non-empty FIFO queue. Although this architecture does not require any complex function and thus, is simple to implement, it has poor scalability because increasing priority levels requires adding more FIFO queues, which results in (i) added hardware cost and increased complexity of the priority encoder and (ii) increased delay in the priority encoder.

The scalability of a systolic array priority queue makes it also suitable for implementing a rate-monotonic priority scheduler. The scheduler built with a systolic array priority queue has basically the same structure as the traffic controller built with it except for the following differences. First, while the priority index indicates a cell's logical arrival time in the traffic controller, in the scheduler it indicates the reserved throughput of a real-time channel. Since the reserved throughput of a channel is constant, the priority index need not be calculated every time a cell arrives at the scheduler. Second, the cell in the 0^{th} block, i.e., the one with the highest priority, is immediately transmitted via an outgoing link when the link is idle, whereas in the traffic controller the cell with the highest-priority is held until its logical arrival time is reached.

2.4.3 Implementation Cost

As mentioned earlier, in spite of its high implementation cost, the use of a systolic array priority queue in implementing the traffic controller can be justified because of its good scalability. Let N denote the maximum number of real-time channels that can co-exist. Then, the traffic controller needs N systolic array blocks to guarantee no cell losses since each channel can have at most one cell in the scheduler. The size of an entry's priority index in a systolic array block depends on the range of logical arrival times in the traffic controller. For example, if the link speed is 1 Gbps and the minimum allocable bandwidth is 1 Kbps, cells of a particular channel may be generated once every 10^6 cell-transmission times. If the minimum "grain" of logical arrival time is one cell-transmission time, the traffic controller can have 10^6 different logical arrival times. Then, the size of priority index is given by $\log_2 10^6 \approx 20$. [5] A 20-bit comparator is expensive, but this cost is unavoidable in any scheme that uses timing information in scheduling cells, *e.g.*, PGPS, RCSP.

Let's consider the implementation cost of a rate-monotonic priority scheduler. The number of systolic array blocks in the scheduler also equals the maximum number of real-time channels that can be established. The difference is in the size of priority indices of the entries in each block. In the scheduler, the priority indices represent the bandwidths allocated to real-time channels. Theoretically, TCRM may assign arbitrary bandwidths to individual real-time channels, and thus, in the worst-case scenario, the number of priority levels equals the number of real-time channels. In reality, however, this is not the case. In most applications, certain typical bandwidths are assigned to individual channels, *e.g.*, 64 Kbps voice channels, 1.5 Mbps video channels, etc. Thus, it is reasonable to assume a set of "standard" bandwidths that can be assigned to individual real-time channels. In this case, if two channels have the same bandwidth requirement, the cells of both channels are assigned the same priority index. If there are more than two cells with the same priority index in the scheduler, ties are broken arbitrarily. If a real-time channel with a non-standard bandwidth requirement is requested, one may assign the nearest higher standard bandwidth. Then if the number of allowed bandwidth levels is 1000, a 10-bit comparator is needed for each systolic array block.

The other implementation costs of TCRM come from a table for storing throughputs of real-time channels and a table for storing the logical arrival times of the cells seen last in the scheduler for each channel. In order to avoid the calculation of the minimum cell inter-arrival time, L/ρ_i, one may store the minimum cell inter-arrival time instead of the throughput requirement. In this case, the minimum cell inter-arrival time is used to find the priority level of a cell in the scheduler. Storing per-connection state information is unavoidable in any cell scheduling scheme when per-connection QoS guarantees need to be provided.

In terms of implementation cost and scalability, the implementation of TCRM with a systolic array priority queue is much better than the other schemes that are known to provide optimal performance: PGPS [65, 66] and RCS-EDF [25]. Let's first consider PGPS. PGPS can also be implemented using the same systolic array priority queue as TCRM. Considering scalability, this implementation is the only choice for PGPS. Since PGPS is a work-conserving service discipline, it does not require two priority queues but a single priority queue. It may seem advantageous to have a single priority queue, but

[5] In order to avoid cell losses when multiple cells have an identical logical arrival time, the time grain must be smaller than one cell-transmission time.

PGPS requires excessive storage space because of its work-conserving service policy; that is, the buffer requirement of PGPS increases as we move along the downstream nodes [65]. Even without considering the increase of buffer requirement at the downstream nodes, the minimum buffer requirement for channel i at every link is larger than σ_i. The burst size σ_i actually depends on applications, but it is generally large. For instance, the burst size of Sequence 1 in the previous section is 2302 cells ($9.761 \times 10^5/424$). Thus, PGPS requires a buffer of 2302 cells for a channel of Sequence 1 whereas the buffer requirement of TCRM is only 2 cells. Moreover, the number of systolic array blocks in the priority queue must be equal to the sum of burst sizes of all the real-time channels. In contrast, TCRM requires only 2 blocks for each channel.

Besides its buffer requirement, PGPS has a large range of deadlines because of its work-conserving service policy. The range of deadlines in PGPS is close to the worst-case end-to-end delay bound, i.e., approximately, σ_i/ρ_i for channel i. The range of deadlines determines the size of the priority index of comparators and a longer range makes the implementation of PGPS costlier. By contrast, TCRM has a smaller range of logical arrival times and thus, requires cheaper comparators.

Finally, PGPS requires the calculation of the virtual finish time of each cell, which is quite complex and computationally expensive. Although some approximation approaches like Self-Clocked Fair Queueing (SCFQ) [27, 72] may simplify the calculation, they cannot avoid the large buffer requirement and the large range of deadlines because they still employ work-conserving service policies.

Now, let's consider RCS-EDF which achieves the same performance as PGPS. Since RCS-EDF is a non-work conserving service discipline, however, its implementation cost is much lower than that of PGPS. In RCS-EDF, the rate controller and the EDF scheduler basically have the same structure since they both sort cells using timestamps which are logical arrival times in the rate controller and deadlines in the EDF scheduler. Because of the non-work-conserving service property, the ranges of logical arrival times and deadlines are identical. The only difference is that the EDF scheduler does not need a comparator which monitors the logical arrival time of the highest priority cell. Note that the highest-priority cell is not held but transmitted immediately in the EDF scheduler. Overall, RCS-EDF has two systolic array priority queues which are almost identical in terms of implementation cost. Compared to TCRM, RCS-EDF has a rate controller which is the same to a traffic controller of TCRM but has a different scheduler. Thus, we need only to compare schedulers of TCRM and RCS-EDF in terms of implementation cost. First, both schemes require the same number of systolic array blocks since the maximum number of cells that can coexist is the same because of their non-work conserving nature. However, considering the component blocks, the EDF scheduler of RCS-EDF is more expensive than the rate-monotonic priority scheduler if we assume a standard set of bandwidths employed in TCRM. In the above example, if the number of allowed bandwidth levels is 1000 and the minimum allocable bandwidth is 1 Kbps while the link speed is 1 Gbps, TCRM requires a 10-bit comparator for each systolic array block. On the other hand, RCS-EDF requires a 20-bit comparator since its priority indices are not bandwidth levels but cell deadlines, and since cell deadlines are independent of bandwidth levels.

2.5 Conclusion

In this chapter, we have proposed a cell multiplexer, TCRM, to realize real-time communication in ATM networks. TCRM (i) provides bounded end-to-end delays which are essential for real-time communication, and (ii) is simple enough to operate in large-scale high-speed ATM networks. We have achieved this goal by employing traffic shaping at the UNI and separating the traffic controller from the rate-monotonic priority scheduler. TCRM requires only a small buffer space inside the network, i.e., buffer space for only two cells for each real-time channel: one for the traffic controller and the other for the scheduler. TCRM is shown to not only emulate circuit-switching in the cell level but also provide VBR services without losing statistical multiplexing gain of ATM networks.

We have developed a simple admission-test algorithm and also presented an implementation of TCRM using a systolic array priority queue. This implementation scales well and requires far less memory than the implementation of PGPS. TCRM has been compared to RTC, RCSP, and PGPS in terms of simplicity and efficiency. TCRM is similar to RCSP and RTC in terms of implementation complexity, but can achieve high channel admissibility similar to PGPS which is much more complex to implement than TCRM.

By disallowing interaction among real-time channels, TCRM provides deterministic real-time communication services. Using deterministic real-time communication services does not necessarily result in inefficient usage of network resources since the bandwidth which is reserved but not used by real-time channels can be used for transmitting non-real-time traffic. However, it limits the number of real-time channels that can be accommodated. In order to accommodate more real-time channels, different real-time channels must be multiplexed statistically. In the next chapter, we investigate the problem of achieving statistical real-time communication using TCRM.

CHAPTER 3

STATISTICAL REAL-TIME COMMUNICATION OVER ATM NETWORKS

3.1 Introduction

In Chapter 2, we proposed a cell multiplexing scheme for realizing deterministic real-time communication over ATM networks. As discussed in Chapter 1, deterministic real-time communication underutilizes network resources severely, since one must reserve network resources based on the worst-case source traffic-generation behavior in order to provide deterministic QoS guarantees. In this chapter, we consider statistical real-time communication in which network utilization is greatly enhanced by exploiting statistical multiplexing gain. Statistical real-time communication specifies QoS requirements in statistical terms instead of deterministic terms, thus tolerating a certain percentage of cell losses and deadline misses.

In this chapter, we propose a framework to provide statistical real-time communication services for VBR video streams in ATM networks based on TCRM. TCRM was originally developed for providing deterministic real-time communication services and does not allow statistical multiplexing among real-time connections. We modify TCRM so that it may allow statistical multiplexing among a set of real-time connections. TCRM guarantees the timely delivery of every cell as long as it is not lost due to buffer overrun. Thus, cell losses due to buffer overrun must be probabilistically bounded for realizing statistical real-time guarantees. By employing the histogram-based model [85] as the input traffic specification for VBR video data along with the modified TCRM, we analytically derive a statistical bound for the average cell-loss ratio of each statistical real-time channel. Simulation results are shown to support our analysis.

Our approach differs in several aspects from the effective bandwidth approach [18, 29, 39] in providing statistical real-time communication services. First, our approach can provide a framework that can control the capacity of a trunk over which statistical real-time channels are multiplexed using the TCRM. Therefore, it can be used not only for a large ATM network but also for a small system that multiplexes only dozens of real-time channels. Second, compared to the effective bandwidth approach, ours provides much tighter cell-loss estimates that can be used for channel-admission control. Lastly, we can reduce the complexity of channel admission control by adjusting the number of bins in the histogram while the Chernoff bound approach requires solving non-linear equations in calculating cell-loss estimates.

The remainder of this chapter is organized as follows. Section 3.2 defines a real-time connection with statistical performance guarantees (i.e., a statistical real-time channel) and reviews the histogram-based model for VBR video sources as well as TCRM. In Section 3.3, we describe the mechanism of the proposed scheme and analytically derive the cell-loss ratio bound for both single-hop and multi-hop cases. Section 3.4 presents a simulation study with real VBR video data and compares our approach with the effective bandwidth approach. The chapter concludes with Section 3.5.

3.2 Background

Providing statistical real-time communication service requires source-traffic modeling, resource reservation, and an appropriate scheme for cell-multiplexing and buffer-management. This section discusses the source-traffic model and the cell-scheduling scheme employed in our approach.

A *statistical real-time channel* is a unidirectional virtual circuit that provides QoS guarantees in statistical terms, i.e., the probability of losing a cell of this channel is less than a given number Z:

$$Pr(\text{end-to-end cell loss}) \leq Z. \tag{3.1}$$

Although a statistical real-time channel can also be defined in terms of delay as was done in [11], we consider only the cell losses due to buffer overrun, because the modified TCRM used in our scheme can guarantee the delivery deadline of every cell as long as it is not lost due to buffer overrun. (This will be discussed further in Section 3.3.3.)

3.2.1 The Histogram-Based Model for VBR Video Traffic Sources

In order to reduce the large amount of multimedia traffic such as video, audio and graphical data, a number of data compression techniques have been proposed and used.

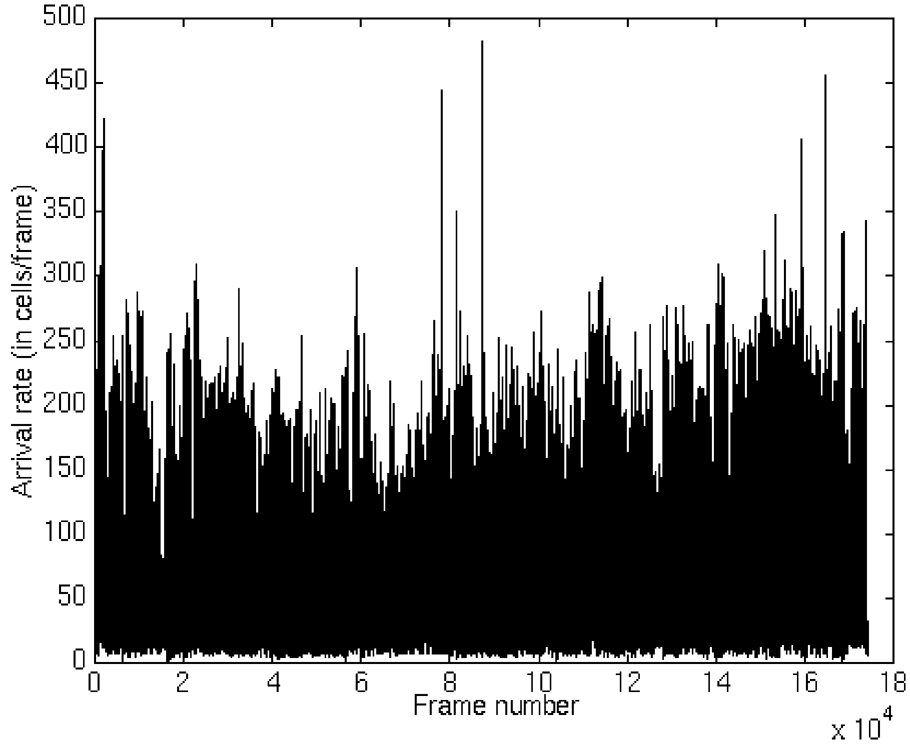


Figure 3.1: Arrival rate of MPEG-coded video sequence: *Starwars*

Compression attempts to keep the quality of played-back data at the receiving end constant at the expense of changing the bit rate, i.e., VBR characteristics of compressed data. Consider an MPEG-coded movie sequence, *Starwars*,[1] in Figure 3.1. The sequence shows extremely high burstiness as I (Intra-coded), P (Predictive) and B (Bidirectional) frames alternate. Accurate characterization of these compressed data streams is essential for real-time transport of such data over ATM networks. By appropriately modeling the traffic, it is possible to dimension network resources while satisfying the QoS requirements.

Many models were previously proposed for VBR video under various compression schemes [3, 20, 22, 31, 32, 34, 35, 46–48, 58, 60, 61, 63, 70, 85]. Since the VBR behavior of a video stream strongly depends on the compression technique used, many of these models do not characterize MPEG-coded video which is currently widely-accepted as a standard for transmission and storage of video data in many multimedia systems. The method to characterize MPEG-coded video streams has been investigated by many researchers [34, 35, 46–48, 63]. They addressed the modeling of MPEG streams using a model fitting or an analytic approach. However, none of them presented an analytic solution to the bandwidth-allocation problem for multiplexed video streams. For instance, in [46–48] Krunz *et al.* characterized

[1] The original sequence was generated by Garrett and Vetterli [21].

a video stream using its frame-size histogram and generated synthetic streams possessing the same characteristics as the original stream. These synthetic streams are then used for a *simulation* study of bandwidth-allocation and buffer-dimensioning problems. Compared to the analytic approach considered in this chapter, their approach doesn't scale well (i.e., has a limitation in dealing with a large number of multiplexed video streams).

Skelly *et al.* [85] also proposed the histogram-based model in order to describe a VBR video traffic, and Shroff and Schwartz [83] presented an analytic solution to the bandwidth-allocation problem for multiplexed video streams under the assumption that each stream was deterministically smoothed at the source. In the histogram-based model, the traffic-generation rate from a VBR video source is assumed to be constant during a fairly long period based on the fact that the bit rate changes very slowly because compressed video is highly correlated. Thus, the traffic-generation rate from a VBR video source is assumed to be constant during a certain period, T, of time. The traffic-generation rate is sampled once every T seconds and the sampled rate is considered to be the generation rate during that period. Although the bit rate changes rapidly as a new frame starts in MPEG video unlike the motion JPEG video considered in [83, 85], the change of bit rate can still be considered slow compared to a cell's worst-case link delay in ATM networks if the ratio of buffer size to the output rate is sufficiently small. That is, if a video frame is decomposed into ATM cells for transmission and cells are not injected into the network in a burst mode (i.e., cell arrivals at the network entry is smoothed either deterministically or randomly), the cell-arrival rate during a frame period can be considered constant. For example, if a video buffer can hold 100 cells and the output rate of the buffer is 2000 cells/frame, the worst-case cell delay at the buffer is one twentieth frame which is negligible compared to a single frame period. To the link scheduler, the cell-arrival rate will appear constant during a fairly long period of time. As a result, we extend the use of the histogram-based video model to more general video streams including MPEG-coded video. In this chapter, we set the period as one frame interval (= 1/24 second) and measure the traffic-generation rate in cells per frame. In this way, we obtain the sequence of cell-generation rates from a VBR video source, and then represent the VBR video source with the histogram of its cell-generation rates.

Let's assume that transmissions of ATM cells are randomly scattered within a frame interval, i.e., random smoothing is executed at the source. Then, we can think of the arrival process formed by a real-time video as a modulated Poisson process whose modulating process has a constant frame period. The value that the modulation process takes is the cell-arrival rate during a frame interval, and thus, the frequency of cell-arrival rates can be obtained from the rate-histogram of frame sizes. In the histogram-based model, a queueing system (an ATM multiplexer) whose input is a modulated Poisson process reaches steady-state fast enough to allow for steady-state analysis of the system. In other words, the modulating process (change of arrival rate) is much slower than the modulated process (time to transmit a cell). Thus, in steady-state, the arrival process at the queueing system can be considered a Poisson process whose rate is determined by its rate-histogram.

When video streams are multiplexed by an ATM multiplexer, the input process of the aggregate traffic is also modeled as a modulated Poisson process. The rate-histogram of the aggregate traffic determines the frequency of cell-arrival rates in the modulated Poisson process. In this case, determining the modulation period becomes complicated since, in general, start times of frames of component streams are not synchronized. If frames of all the component streams are generated synchronously, the modulation period is simply one frame period. Otherwise, modulation periods are smaller than one frame period and depend on the realization of multiplexed streams. However, we can still use synchronous multiplexing in modeling the aggregate video stream for the purpose of providing statistical cell-loss guarantees. Because synchronous multiplexing provides the worst-case scenario in terms of cell loss, the rate-histogram obtained by convolving rate-histograms of component streams can be used to find a cell-loss ratio bound for the aggregate video stream. To understand why synchronous multiplexing provides the worst-case scenario in terms of cell losses, let's consider the buffer overrun condition. Buffer overrun and thus cell losses occur when the cell-arrival rate of the aggregate video stream exceeds the multiplexor's capacity for a certain time period. When all the streams are synchronized, the time during which large frames are multiplexed together and thus cell losses take place is longer than that in the unsynchronized case. That is, the synchronized multiplexing does not utilize smoothing on the time axis. Hence the synchronized multiplexing is used to model the aggregate video stream with the histogram-based model, and the aggregate video stream can be modeled as a modulated Poisson process whose modulation period is one frame interval. Moreover, the rate-histogram of the aggregate MPEG-coded video stream is obtained by simply convolving rate-histograms of component video streams. When a large number of video streams are multiplexed, the assumption of using random smoothing at the source can be relaxed since a large number of similar and independent sources can be considered as a Poisson process [4]. Thus, as long as the cells belonging to the same frame do not arrive in burst (i.e., some form of smoothing is executed at the source), we can model the cell arrivals of the aggregate video as a modulated Poisson process.

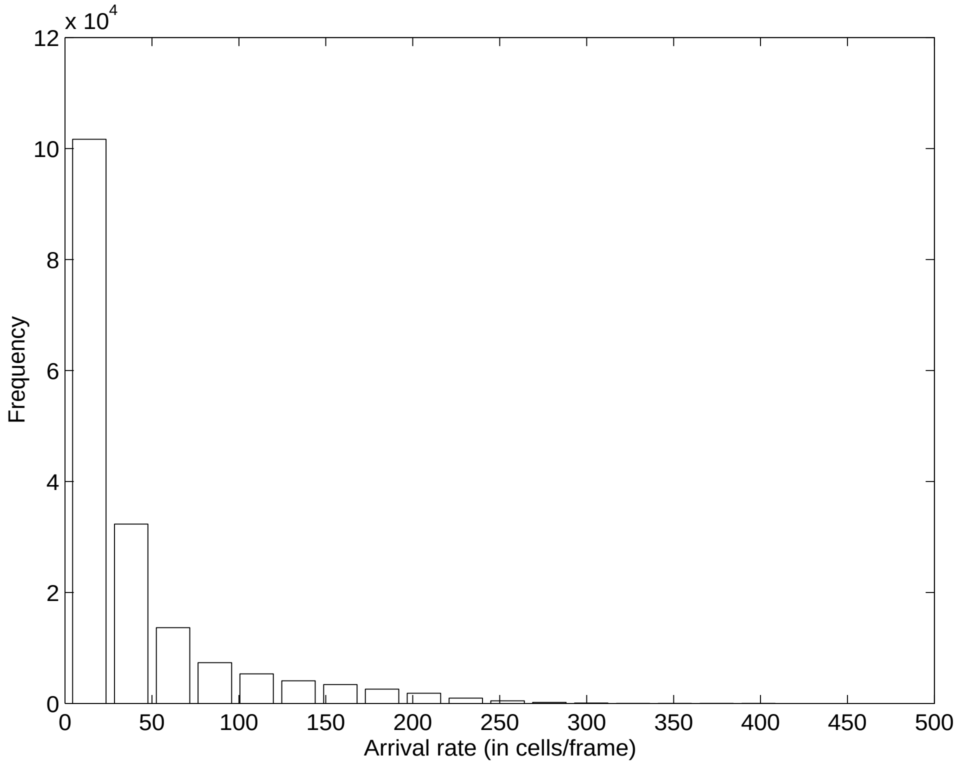


Figure 3.2: Arrival rate histogram of *Starwars*

Figure 3.2 shows the histogram of the traffic-generation rate of the sequence in Figure 3.1, where a frame duration is the period for measuring the rate. The MPEG sequence is IBBPBBPBBPBB IBB.... Since I frames appear once every 12 frames, the frequency of large frames in the histogram is very low compared to that of small frames.

3.3 A Framework for Statistical Real-Time Communication

Using the guaranteed throughput service provided by TCRM, we now build a framework for statistical real-time communication on ATM networks.

3.3.1 Macro-Channel

The QoS guarantee is given to a *set* of statistical real-time channels, rather than to a single real-time channel as in the deterministic approach. The set of statistical real-time channels are multiplexed onto a common "macro" real-time channel which is guaranteed to receive the minimum throughput provided by TCRM. A macro-channel is defined as a single-hop real-time channel with parameters (ρ, N) over a link where ρ is the bandwidth (in cells/sec) guaranteed to this channel by TCRM and N is the size of buffer needed at the traffic controller of this channel. Recall that the buffer space for only one cell is reserved for each real-time channel at its traffic controller in the original TCRM. In this paper, we change the TCRM's buffer size from 1 to N in order to reduce the cell-loss probability when multiple cells arrive at the macro-channel in a very short time. The cell-drain rate from the buffer is ensured to be ρ using the cell logical arrival times. Since the admission control in Section 2.2 requires only ρ as a parameter, the change of the buffer size does not require any other modifications in the structure of TCRM.

Within a macro-channel, we do not differentiate statistical real-time channels from one another. All the cells arriving at this macro-channel are transmitted on a FIFO (First-In-First-Out) basis. This policy simplifies significantly channel management within a macro-channel compared to the case of treating individual statistical real-time channels separately. Note, however, that the cells of a macro-channel are serviced separately from those of the other macro-channels, deterministic real-time channels, and best-effort traffic. Since all statistical real-time channels sharing a common macro-channel are teated *equally* on a FIFO basis, all cells in the macro-channel are given the *same* loss probability irrespective of the cell's channel membership. This implies that individual statistical real-time channels sharing a common macro-channel have the same cell-loss ratio of the macro-channel.

Given input traffic specifications of all of its component statistical real-time channels, we can derive the parameters (ρ, N) of a macro-channel based on its QoS requirement, or its cell-loss ratio bound Z.

Figure 3.3 shows a scenario in which various statistical real-time communication services are provided. Macro-channels with different parameters (ρ, N) are established in order to provide different QoS guarantees in a single ATM network. Although Figure 3.3 shows one macro-channel per link, one can establish an arbitrary number of macro-channels with different cell-loss ratios over a link as long as there are sufficient resources.

A statistical real-time channel C may run through a (fixed) multi-hop path between its source and destination. In such a case, since a macro-channel is established over each hop, we need to concatenate a series of "appropriate" macro-channels each of which is selected from the macro-channels established over each link along the path, and multiplex the statistical real-time channel C into them. By an "appropriate" macro-channel, we mean that it must guarantee the cell-loss ratio required for this statistical real-time channel C. We will discuss how to choose macro-channels when we consider admission control later in this section.

Given the above setting, the problem is how to determine the bandwidth ρ and the

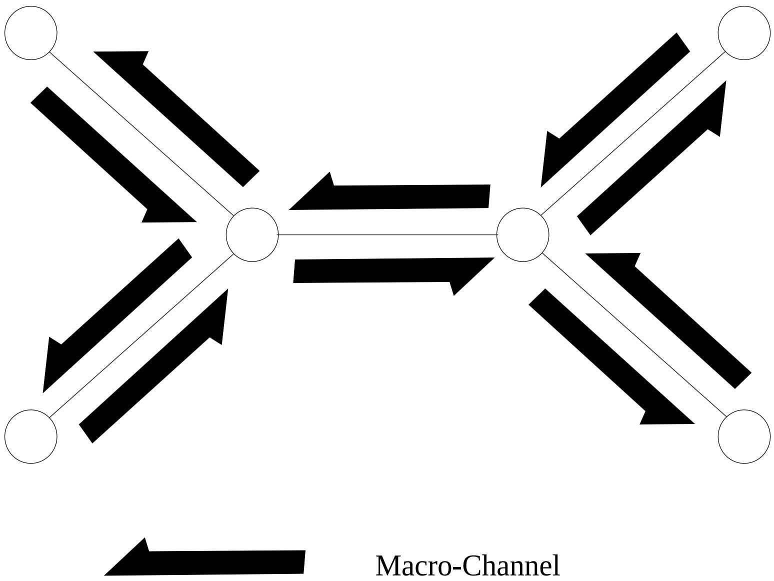


Figure 3.3: Macro-channel

buffer space N needed to meet the given delay and cell-loss requirements. Since the TCRM bounds the delay over each link when the buffer size N and the minimum throughput ρ are fixed, we will first concentrate on the cell-loss ratio.

3.3.2 Cell-Loss Ratio within a Macro-Channel

We want to derive the cell-loss ratio of a macro-channel using the histogram-based model for aggregate video sources. For now, we consider only the case in which all statistical real-time channels are established over only a single hop, resulting in all the cell streams feeding into a macro-channel directly from external sources. We will in Section 3.3.3 relax this assumption.

In order to determine the cell-loss ratio of a macro-channel, we need the input traffic specification of the aggregate of statistical real-time channels multiplexed over the macro-channel. Since we have chosen the histogram-based model for source traffic, we need the rate histogram of the aggregate of statistical real-time channels, which can be obtained by simply taking the convolution of the rate histograms of component sources, as discussed in the previous section.

According to the histogram-based model, a queueing system reaches steady-state fast enough to allow for steady-state analysis of the system. We can then think of the cell-arrival process as a Poisson in steady-state. Under this assumption, the queueing system can be analyzed in two steps as follows. First, we fix cell-arrival rate to a constant, say, λ, and

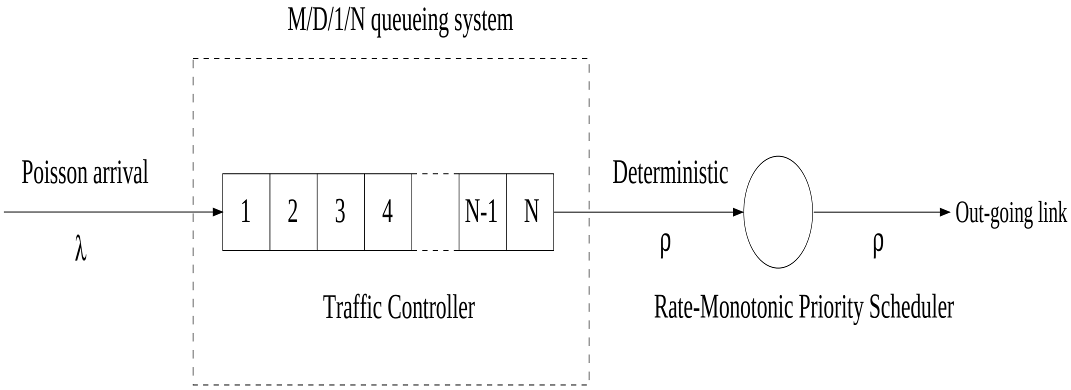


Figure 3.4: An $M/D/1/N$ queueing system.

analyze the cell-loss behavior of the system with a Poisson arrival input whose rate is λ. Then, using the rate-histogram, we calculate the weighted sum of cell-loss ratios in order to obtain the cell-loss statistics of the overall system.

Assume that the throughput guaranteed to macro-channel j is ρ and that the buffer space reserved at the traffic controller for macro-channel j is N. Then, the buffer of macro-channel j can be seen as an $M/D/1/N$ queueing system since the cell inter-transfer time from the traffic controller's buffer to the rate-monotonic priority scheduler is constant, which is L/ρ. This is illustrated in Figure 3.4. Although there can be multiple queueing systems when multiple macro-channels run through a link, we specify only a single macro-channel since different macro-channels are virtually isolated from one another by TCRM.

When the cell-arrival rate is λ in an $M/D/1/N$ system, the cell-blocking probability $P_b^{MD}(\lambda)$ can be calculated using the following $O(N^2)$ algorithm [13]:

$$\phi'_0 = 1, \tag{3.2}$$

$$\phi'_{k+1} = \phi'_k - \sum_{j=1}^{k} \phi'_j a_{k-j+1} - a_k, \quad 0 \le k \le N-1, \tag{3.3}$$

$$\phi_0 = \left(\sum_{k=0}^{N} \phi'_k\right)^{-1}, \tag{3.4}$$

$$P_b^{MD}(\lambda) = 1 - \frac{1}{\phi_0 + \lambda/\rho}, \tag{3.5}$$

where

$$a_k = \frac{(\lambda/\rho)^k}{k!} e^{-\frac{\lambda}{\rho}}. \tag{3.6}$$

The cell-loss ratio of the macro-channel, P_{macro}, is given as the weighted sum of cell-loss ratios where weights are given by the rate histogram of the aggregated video sources. Thus,

$$P_{macro} = \frac{\sum_{i=1}^{M} P_b^{MD}(\lambda_i) f_i \lambda_i}{\sum_{i=1}^{M} f_i \lambda_i}, \tag{3.7}$$

where M is the number of intervals (bins) in the histogram and f_i is the probability mass function (p.m.f.) of arrival rate λ_i which is given by the histogram of the cell-arrival rate of the aggregate traffic.

Although the $M/D/1/N$ analysis is exact, it may not be practical to use for establishing a statistical real-time channel because of its computational complexity even when the buffer size N is moderate. So, we employ the $M/M/1/N$ analysis instead in our scheme, as the cell-blocking probability of an $M/D/1/N$ system is upper-bounded by that of an $M/M/1/N$ system. In Section 3.4 we will compare the difference between the cell-blocking probabilities of an $M/D/1/N$ system and an $M/M/1/N$ system, showing that the difference between the two is quite small. In an $M/M/1/N$ system, the cell-blocking probability $P_b^{MM}(\lambda)$ is given in [42] as:

$$P_b^{MM}(\lambda_i) = \frac{(\lambda_i/\rho)^N}{1-(\lambda_i/\rho)^{N+1}}\{1-(\lambda_i/\rho)\}. \tag{3.8}$$

As with Eq. (3.7),

$$P_{macro}^{MM} = \frac{\sum_{i=1}^{M} P_b^{MM}(\lambda_i) f_i \lambda_i}{\sum_{i=1}^{M} f_i \lambda_i}, \tag{3.9}$$

and since

$$P_b^{MD}(\lambda_i) < P_b^{MM}(\lambda_i), \tag{3.10}$$

$$P_{macro} < P_{macro}^{MM}. \tag{3.11}$$

As discussed earlier, although P_{macro}^{MM} is a bound on cell-loss ratio of the set of statistical real-time channels multiplexed over the macro-channel, it also works as a cell-loss ratio bound for each individual component channel, because all the cells in the macro-channel are treated equally by the FIFO service policy and thus have an identical cell-loss ratio.

3.3.3 Cell Losses in an End-to-End Connection

In Section 3.3.2, we considered only the single-hop case in which the Poisson-arrival approximation holds. However, in general point-to-point networks, cell streams take multiple hops before arriving at their destination nodes. In our framework, a statistical real-time channel is multiplexed over a *series* of macro-channels on the links along the channel path. We first describe our assumptions on the traffic switched and routed via multiple hops and then derive the cell-loss ratio bound for the end-to-end connection.

Effects of Switching

As cells are switched and routed from one macro-channel to another, the traffic pattern may change depending on the traffic condition at each macro-channel. This raises two questions on our assumptions about the traffic from aggregate sources in a single hop. One is the accuracy of the Poisson-arrival assumption on the traffic from aggregate sources. The other is the derivation of the new cell-arrival rate histograms at intermediate nodes, because the histogram defined at the source may change depending on the conditions of the intermediate nodes. That is, if other statistical real-time channels sharing the same macro-channel at the upstream nodes have large amounts of traffic, a statistical real-time channel may lose a large portion of its cells at those nodes and the probability mass function (the histogram) at the downstream links may change.

For the time being, let's assume that the histogram defined at the source node remains unchanged at all intermediate nodes. In general, the output process of an $M/D/1/N$ queue is not a Poisson process and cell inter-departure times are correlated [23]. This poses difficulty in analyzing a multi-hop statistical real-time channel. This is also the case in an $M/M/1$ system analysis. In the $M/M/1$ system, the packet inter-arrival and service times are correlated. To handle this difficulty, Kleinrock proposed to use "Independence Approximation" in analyzing a communication network using a general queueing network like the Jackson network [4, 40]. It asserts that, in an $M/M/1$ system, merging several cell streams on a transmission link has an effect akin to restoring the independence of inter-arrival times and service times. In particular, he emphasized the independence of service times of a packet at different nodes, which is not true in real communication networks. Since the length of a cell is fixed in ATM networks, the correlation of cell-service time is not important in our problem. What matters in the $M/D/1/N$ analysis is the independence of cell inter-arrival times. As with Kleinrock's independence approximation, we assume that the cell inter-arrival time at a macro-channel at an intermediate link is exponentially-distributed if multiple cell streams routed from different macro-channels on different links and externally-fed cell streams are merged into this macro-channel. Then, the new aggregate traffic arriving at this new macro-channel can be approximated as a Poisson process, thus enabling the application of the $M/D/1/N$ analysis result at any macro-channel as long as the number of multiplexed statistical real-time channels are large enough and the routing processes are uncorrelated. Our simulation study in Section 3.4 confirms the validity of this assumption.

Next, let's consider the rate-histogram of a video stream switched and routed inside the network. As the traffic passes through downstream nodes, the original traffic pattern at the source node will change and, in general, may become burstier during a short period. However, in the rate-histogram model, we assume that the cell-arrival process is a Poisson and that the arrival rate is constant for a fairly long time (the steady-state condition). Under the steady-state condition, we assume that the cell-arrival rate from a statistical real-time channel is constant over an infinite period of time, and is equal to, say, λ. This cell stream is multiplexed with streams of other statistical real-time channels onto a macro-channel, M_a. After departing macro-channel M_a, the cell stream is separated from the other statistical real-time channels and then multiplexed onto a new macro-channel, M_b. While being multiplexed at M_a, some cells of this stream may be lost due to buffer overrun. Therefore, the number of cells of the stream at M_b cannot be larger than that at M_a. Over an infinite period of time, the arrival rate of this stream at M_b, denoted by $I(\lambda)$, cannot be larger than that at M_a, λ. That is,

$$I(\lambda) \leq \lambda. \tag{3.12}$$

Now, let's consider the stream as a modulated Poisson process, then the steady-state condition will still hold within each frame period. Let the rate-histogram of the process be given by $\{\lambda_i, f_i\}_{i=1,\ldots,M}$, where M is the number of bins and f_i is the p.m.f. of arrival rate λ_i. Let Λ_a and Λ_b denote the arrival rates of the stream at M_a and M_b, respectively. Then,

$$\begin{aligned} Pr\{\Lambda_b \geq I(\lambda_k)\} &= \sum_{i=k}^{M} I(\lambda_i) \cdot f_i \\ &\leq \sum_{i=k}^{M} \lambda_i \cdot f_i \\ &= Pr\{\Lambda_a \geq \lambda_k\} \end{aligned} \tag{3.13}$$

Since $I(\lambda_k) \leq \lambda_k$,

$$Pr\{\Lambda_b \geq \lambda_k\} \leq Pr\{\Lambda_b \geq I(\lambda_k)\}. \tag{3.14}$$

Thus,

$$Pr\{\Lambda_b \geq \lambda_k\} \leq Pr\{\Lambda_a \geq \lambda_k\}. \tag{3.15}$$

This relation shows that the histogram of a statistical real-time channel at the intermediate nodes is *probabilistically bounded* from above by the histogram at the source node. That is,

$$\begin{aligned} &Pr(\text{cell-arrival rate at the source node} \geq x) \geq \\ &\quad Pr(\text{cell-arrival rate at the intermediate nodes} \geq x). \end{aligned} \tag{3.16}$$

In terms of QoS guarantees, it is still effective to use the source's traffic characteristics for intermediate nodes since the cell-loss probability can still be bounded by using the same traffic characteristics. It allows for a simple run-time channel-establishment process at the expense of slightly conservative resource reservation. The amount of over-reservation of resources at the intermediate nodes is negligible when the cell-loss probability is quite small, which is the case of most statistical real-time applications in ATM networks, as will be discussed in Section 3.4.

Cell-Loss Ratio Bound in an End-to-End Connection

Based on the above arguments, the end-to-end cell-loss probability of a statistical real-time channel is given by

$$Pr(\text{end-to-end cell loss}) \leq 1 - \prod_{j=1}^{K}(1 - P_{macro,j}), \tag{3.17}$$

where K is the number of hops the statistical real-time channel takes and $P_{macro,j}$ is the cell-loss probability of the macro-channel at the j^{th} hop. Notice that, although $P_{macro,j}$ is the cell-loss ratio of the macro-channel at the j^{th} hop, it is the cell-loss ratio of individual statistical real-time channels multiplexed onto the macro-channel. For the sake of simplicity of the run-time channel-establishment process, we can employ the $M/M/1/N$ analysis. Then, we get from Eq. (3.11)

$$Pr(\text{end-to-end cell loss}) \leq 1 - \prod_{j=1}^{K}(1 - P_{macro,j}^{MM}), \tag{3.18}$$

where $P_{macro,j}^{MM}$ is the cell-loss probability at the j^{th} hop obtained from the $M/M/1/N$ analysis.

Although we mainly focused on deriving a cell-loss ratio bound, it must be stressed that our approach also guarantees statistically a real-time cell's delivery delay. That is, the probability that a cell is delivered to its destination before its deadline is larger than Z. This is because a cell which is not lost is guaranteed to be delivered to its destination within a bounded time because buffer size is fixed and the minimum buffer drain rate is guaranteed at each link by TCRM. The end-to-end delay bound of the statistical real-time channel is given as:

$$D_{end-to-end} = \sum_{j=1}^{K}(N_j + 1)L/\rho_j \tag{3.19}$$

where N_j and ρ_j are the buffer size in cells and the bandwidth of the macro-channel at the j^{th} hop, respectively, and $(N_j + 1)L$ is the maximum backlog at the macro-channel when a cell has arrived at the macro-channel. The reason for adding 1 to the buffer size is to account for the delay at the rate-monotonic priority scheduler, as seen in Figure 3.4.

3.3.4 Extension of Macro-Channel to Virtual Path

Thus far, we have described the macro-channel as a single-hop real-time channel. The macro-channel can be easily extended to the concept of Virtual Path (VP) by establishing a macro-channel not as a single-hop real-time channel, but as a VP which takes multiple hops. As long as each VP is guaranteed to have a constant throughput at intermediate links by TCRM, it can be considered as a transparent ATM trunk with constant transmission capability. That is, a VP can be considered as a single-hop real-time channel even if it passes through multiple hops. The only change in the VP extension is the end-to-end delay bound given by Eq. (3.19), which must be modified as:

$$D_{end-to-end} = \sum_{i=1}^{K}(N_j + l_j)L/\rho_j \tag{3.20}$$

where l_j is the number of hops the j^{th} VP takes. The l_j is added to represent the delay at the rate-monotonic priority scheduler at each hop. Except for this additional delay, a statistical real-time channel using VP's is exactly the same as the single-hop version.

3.4 Simulation and Discussion

In order to demonstrate the usefulness of the histogram-based model for statistical real-time communication, we have conducted a simulation study using MPEG-coded video traces. Since the end-to-end delay deadline of every cell which is not lost due to the buffer overrun is met by TCRM, we will consider only the cell-loss ratio as the QoS parameter.

3.4.1 The Simulation Model

Figure 3.5 shows the topology of an ATM network used for the simulation study which consists of 5 nodes and 4 links. All the links are simplex, and thus, cells are transmitted only in the direction of arrows shown in the figure. Also, for the sake of simplicity, we assume that there exists only one macro-channel over each link. That is, there is no deterministic real-time traffic, and other statistical real-time traffic or non-real-time traffic except for the statistical real-time traffic is multiplexed over each macro-channel on each link. Since TCRM provides a virtual circuit with a guaranteed throughput over an ATM link, a macro-channel can be considered as an ATM link with the guaranteed throughput ρ and the input buffer of size N.

In this network, we multiplexed 20 statistical real-time channels on each link. The parameters of each statistical real-time channel are randomly selected from the clips of the

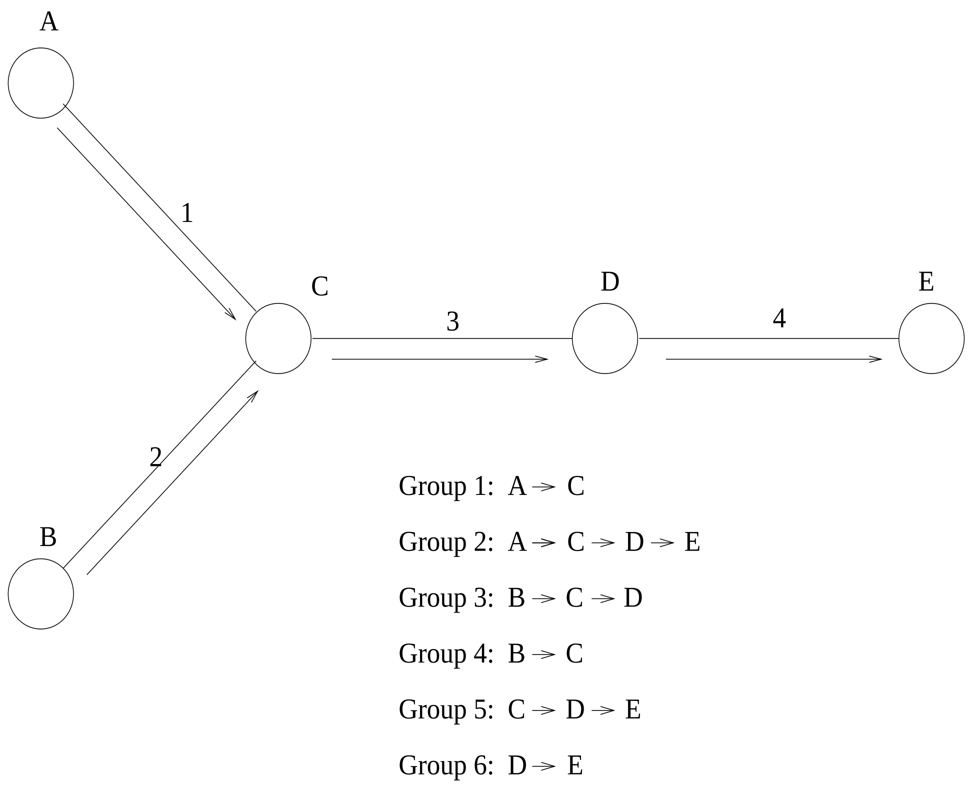


Figure 3.5: The network topology for simulation

movie *Starwars* in Figure 3.1, and 17 different MPEG-coded video clips.[2] The length of each stream is 1000 frames, and each run lasts about 50 seconds since one frame interval is 1/24 second. First, we conducted an experiment using streams only from the *Starwars* sequence in order to investigate cell-loss ratios in a homogeneous-traffic environment. Using convolution, we derived the p.m.f. of the arrival rate of the aggregate of 20 streams seen in Figure 3.6. We derived a 20-bin histogram from the sequence, which requires simple operations for the convolution. The average cell-generation rate of the aggregate traffic is about 822 cells/frame and the maximum cell-generation rate is about 9,000 cells/frame.

Next, in order to investigate the heterogeneous-traffic case, we have conducted a similar experiment using the 17 different sequences. We selected as many streams as needed for the simulation from these sequences. In particular, we chose 14 streams once and the other streams twice in order to feed 20 channels which are multiplexed over link 1. In this case, the average cell-arrival rate of the aggregate of 20 streams was 988 cells/frame and the maximum was about 12,000 cells/frame.

To investigate the various cases, the multiplexed streams are grouped according to their paths: six groups are shown in Figure 3.5. For example, group 2 consists of channels whose

[2]These sequences were generated by Rose [74].

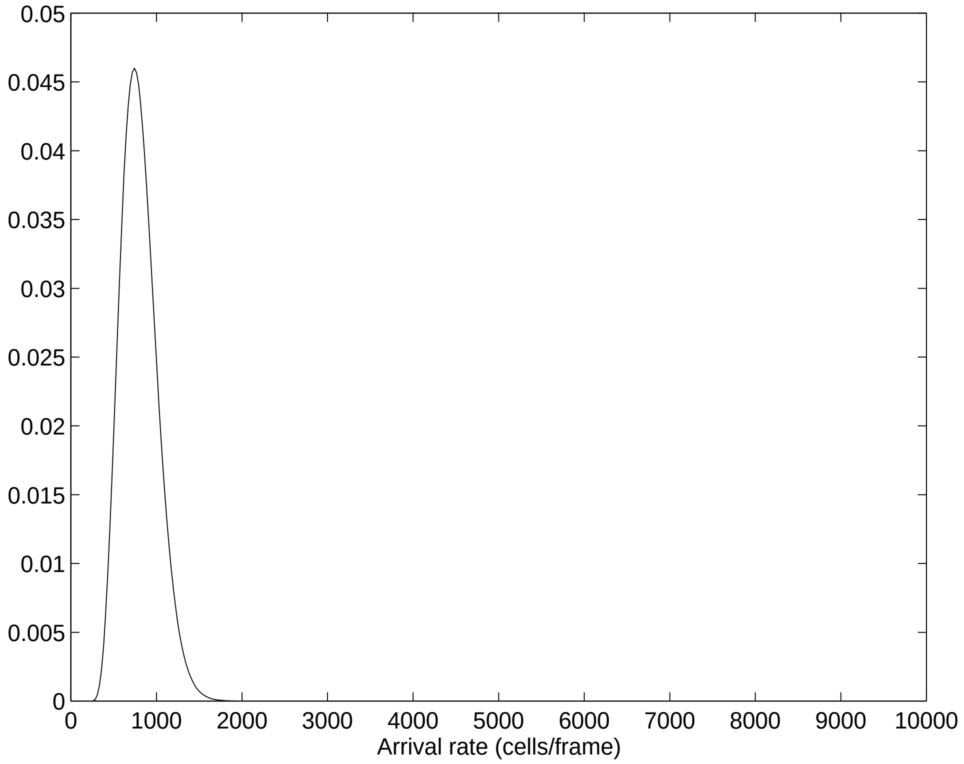


Figure 3.6: Probability mass function of arrival rate of an aggregate of 20 statistical real-time channels of *Starwars*

sources (destinations) are node A (node E), and that pass through node C and D. Only the channels of group 1 and 2 traverse link 1. As a result, through link 1, no routed cells are transmitted, but only external input traffic from node A are transmitted. On link 3, group 2 and 3 are routed from link 1 and 2, respectively, and group 5 is directly fed from node C.

During the simulation, the cell transmission from each source is randomly distributed over one frame duration, and all the cells belonging to a frame must be transmitted from the source within one frame duration. At intermediate nodes, cells are transmitted on a FIFO basis regardless of their channel identities.

3.4.2 Simulation Results

In order to investigate the validity of our assumption on the Poisson arrival process at intermediate nodes, we have considered a case in which some links, in addition to the routed traffic from upstream links, are fed with external inputs. We have assigned 13 channels to group 1, 6 to group 2, 7 to group 3, 13 to group 4, 6 to group 5, and 7 to group 6 so that 20 cell streams traverse each link. Note that link 1 and link 2 are not fed with any routed traffic.

First, we have considered a homogeneous traffic environment in which we multiplex only streams from the *Starwars* sequence. We have varied the bandwidth assigned to a macro-channel established over each link from 700 cells/frame to 2,300 cells/frame. The buffer size N is 50 (cells), and thus, the worst-case cell delay in a single hop is 1/20 frames, i.e., 2.1 msec if the throughput guaranteed to a macro-channel is 1000 cells/frame. This is small enough to satisfy the steady-state condition presented in Section 3.2. The cell-loss ratios are compared to the analysis of the $M/D/1/N$ and $M/M/1/N$ systems in Figure 3.7. We only show the average cell-loss ratios because loss guarantees to a macro-channel and individual statistical real-time channels are given in a statistical form. When the capacities of macro-channels are large (i.e., utilizations of macro-channels are low), the cell-loss ratios of all links for the two systems do not show any notable difference. All the cases show that the cell-loss ratios are bounded by the $M/D/1/N$ and $M/M/1/N$ results.[3]On the other hand, when the capacities of the macro-channels are small, the cell-loss ratios of link 3 and link 4 are smaller than the bounds while those of link 1 and link 2 match the bound almost exactly. As we mentioned in Section 3.3.3, the tail distributions of the aggregate traffic at link 3 and link 4 decrease because of the cell losses at the upstream links, despite the fact that the decrease is negligible when the cell-loss probability is small. This explains the smaller cell-loss ratios at link 3 and link 4 which have the routed cell streams from upstream links 1 and 2 when the cell-loss probability is large. From this observation, we conclude that if we use a macro-channel with a high cell-loss ratio bound, our scheme will result in over-reservation of network resources. However, for a macro-channel with a very small cell-loss ratio bound (e.g., 10^{-4}), our scheme provides good cell-loss estimates, and thus, enables efficient use of network resources. Lastly, Figure 3.7 shows that the analysis result of the $M/M/1/N$ system is very close to that of the $M/D/1/N$ system. Therefore, using the $M/M/1/N$ analysis for deriving a cell-loss ratio bound doesn't result in link bandwidth over-reservation. We can reduce the degree of over-reservation; in fact, the analysis result of an $M/M/1/N$ system can be made arbitrarily close to that of an $M/D/1/N$ system by increasing N [83].

In Figure 3.7, we also show the Chernoff bound estimate of the cell-loss ratio which is calculated using the derived periodic on-off random process suggested by Elwalid *et al.* [18] and the approximation by a Gaussian distribution which is based on Central Limit Theorem (CLT) [29]. Both approaches employed the buffer-overflow probability as a QoS parameter

[3]In all the cases in this simulation, 99 % confidence-level intervals are 10^{-4}, so any value below 10^{-4} is considered to be noisy.

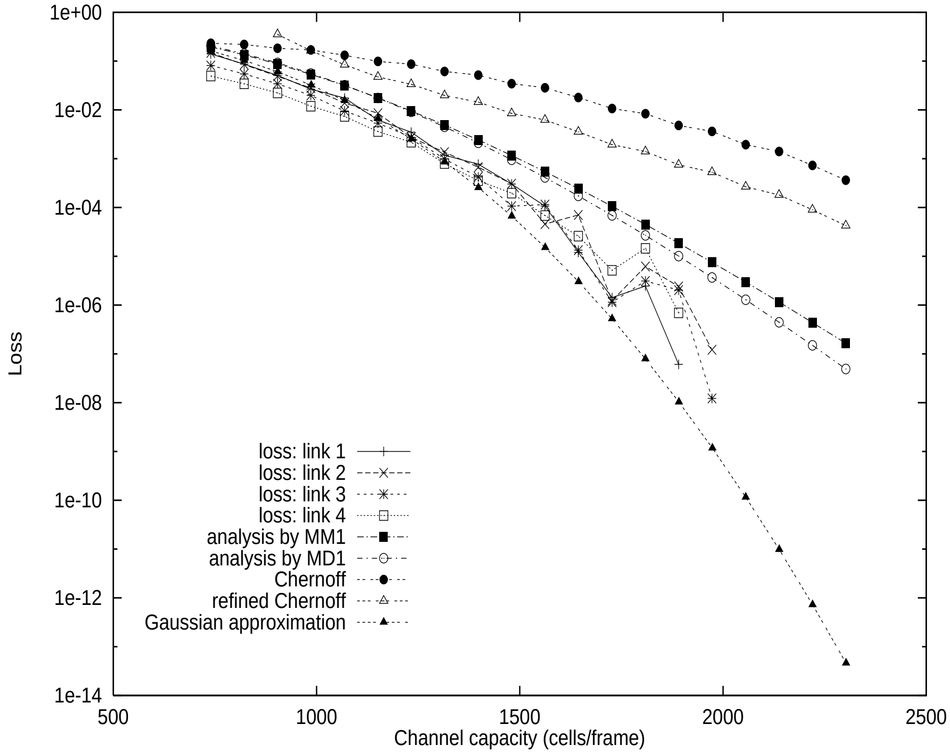


Figure 3.7: Cell-loss ratio in all external input case – homogeneous traffic

while ours uses the cell-loss ratio. For the purpose of comparison, we derived the cell-loss estimates from the buffer-overflow probability given by both the methods.

The parameters of the on-off process derived from the *Starwars* sequence are as follows. The peak rate is 230 cells/frame,[4]the mean rate is 41 cells/frame, and the bucket size of the leaky-bucket regulator is 46,286 cells. The on and off periods derived from the parameters are 2,444 frame intervals and 11,395 frame intervals, respectively. By substituting these parameters into the Chernoff bound estimate according to the step suggested in [18], we obtained the result plotted in Figure 3.7. In addition to the Chernoff bound, we also show the more accurate refined Chernoff bound by Bahadur and Rao [2]. Compared to our analysis results based on the $M/M/1/N$ and the $M/D/1/N$ systems, both the Chernoff bound and the refined Chernoff bound estimates are too pessimistic. Considering the fact that Elwalid's approach is based on the extremal traffic description, one can anticipate the pessimistic result in Figure 3.7. In contrast, the $M/M/1/N$ analysis based on the rate histogram provides a very accurate cell-loss estimate with only a 20-bin histogram which is not so complex in executing convolutions. Specifically, when the cell-loss ratio bound is

[4] Originally, the peak rate for achieving lossless multiplexing was 483 cells/frame, but it resulted in a too pessimistic cell-loss ratio estimate. In this Chapter, we instead choose the 99.9 percentile from the cell-arrival rate histogram as a peak rate.

set to 10^{-4}, our scheme requires reservation of 1,712 cells/frame while Bahadur and Rao's approach requires reservation of 2,170 cells/frame.

In the CLT approximation, buffer size is ignored and only bandwidth is considered as a reservable resource. Ignoring buffer size may result in pessimistic cell-loss estimates. However, as argued in [29], the CLT approximation is shown to be too optimistic in estimating cell losses for very bursty traffic like MPEG. By contrast, the $M/M/1/N$ analysis provides a reasonable cell-loss ratio bound lying between the Chernoff bound estimate and the Gaussian approximation.

We have conducted the same experiment using 17 different video clips in order to investigate the validity of our model in a heterogeneous-traffic environment. We followed the same procedure as before and obtained the result in Figure 3.8. The only difference is choosing the peak rate of the on-off process. Instead of 99.9 percentile, we used the average cell-generation rate of I frames as a peak rate in order to favor Elwalid *et al.*'s approach, but it is not justifiable in a strict sense since the original peak cell-generation rate is required in order to obtain the parameters for a lossless multiplexing system [18]. In Figure 3.8, we show the simulation result on link 1 only since each link has a different trunk capacity depending on the characteristics of the aggregate traffic. However, we obtained similar results on the other links. In the figure, the $M/M/1/N$ analysis provides a good estimate of cell-loss ratios as in the homogeneous-traffic case, compared to the Chernoff bound estimate and the CLT approximation. The Chernoff bound estimate is a little closer to the simulation result than the homogeneous case due to the choice of a smaller peak rate.

Next, we considered the case in which there exists only routed traffic without any external input traffic at intermediate nodes. To this end, we disable group 5 in Figure 3.5 and change the number of channels in each group accordingly. We assign 10 channels to each of groups 1, 2, 3, 4 and 6. Note that the number of channels multiplexed over each macro-channel on each link is kept 20. In this case, there is no external input traffic at the macro-channel on link 3. In Figure 3.9, we only show the homogeneous-traffic case using the *Star wars* sequence. The loss at the macro-channel on link 3 does not make any difference from that on links 1, 2 and 4 when the cell-loss probabilities are small. When the cell-loss probability is large (i.e., the reserved bandwidth of the macro-channel approaches the average cell-generation rate of the aggregate channel), the loss of the macro-channel on link 3 is smaller than others. However, the trend is clear that the cell-loss probability is bounded by the analysis result and that the difference between the simulation and analysis

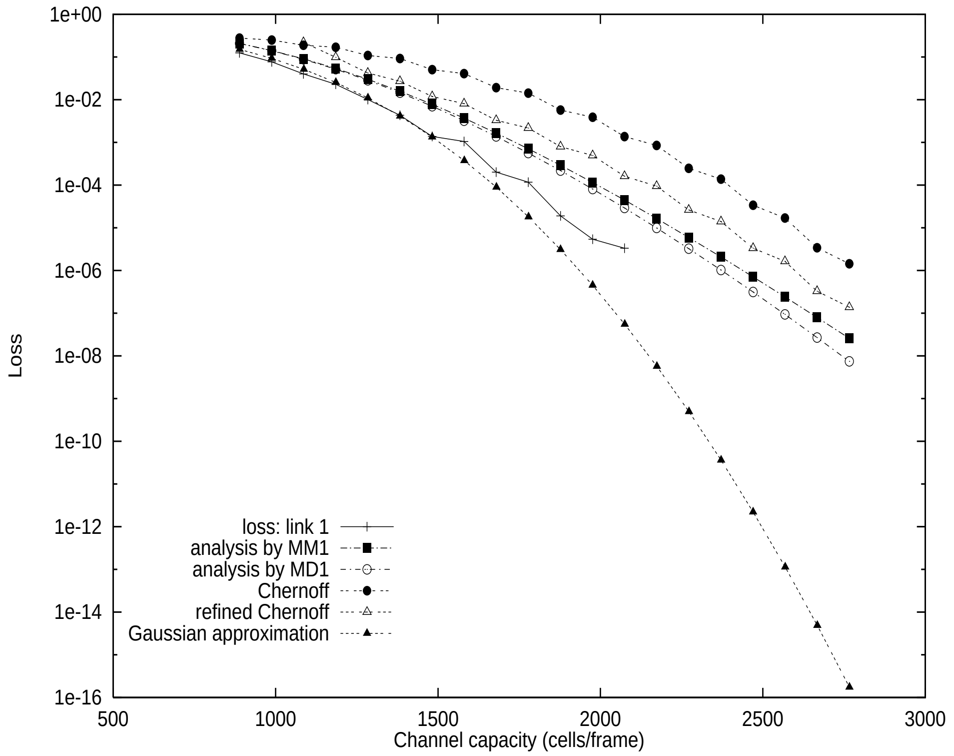


Figure 3.8: Cell-loss ratio in all external input case — heterogeneous traffic

results is small when the cell-loss ratio is small. Therefore, the Poisson-arrival assumption can be applied even when there is no external input traffic at the intermediate nodes.

Next, we investigated the validity of our framework when the number of statistical channels multiplexed over a macro-channel is quite large. We have set the number of channels to 100 and the buffer size N to 100 cells. In Figure 3.10, we show the simulation result along with the $M/M/1/N$ analysis, the Chernoff bound approximation and the CLT approximation. The Chernoff bound estimates remain too pessimistic. The $M/M/1/N$ analysis provides a good cell-loss ratio bound. Interestingly, the CLT approximation shows a pretty accurate cell-loss estimate in this figure unlike the case when 20 channels are multiplexed. This result is due to the large number of channels multiplexed and the small buffer size, which is the basic assumption of the CLT approximation. Although the CLT approximation provides a good cell-loss estimate for the case of a large number of channels, our approach provides good cell-loss bounds regardless whether the number of channels multiplexed is large or small.

The statistical multiplexing gain achieved by increasing the number of channels multiplexed is depicted in Figure 3.11 in which the cell-loss ratios are plotted against link utilization when 5, 10 and 20 channels are multiplexed. The link utilization is normalized

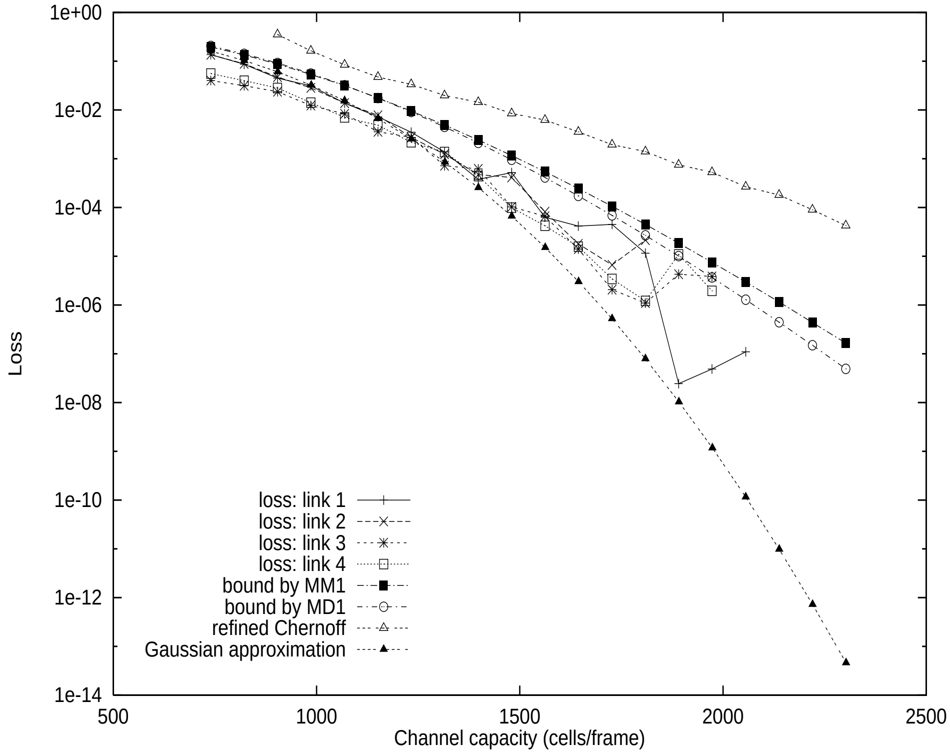


Figure 3.9: Cell-loss ratio in no external input case – homogeneous traffic

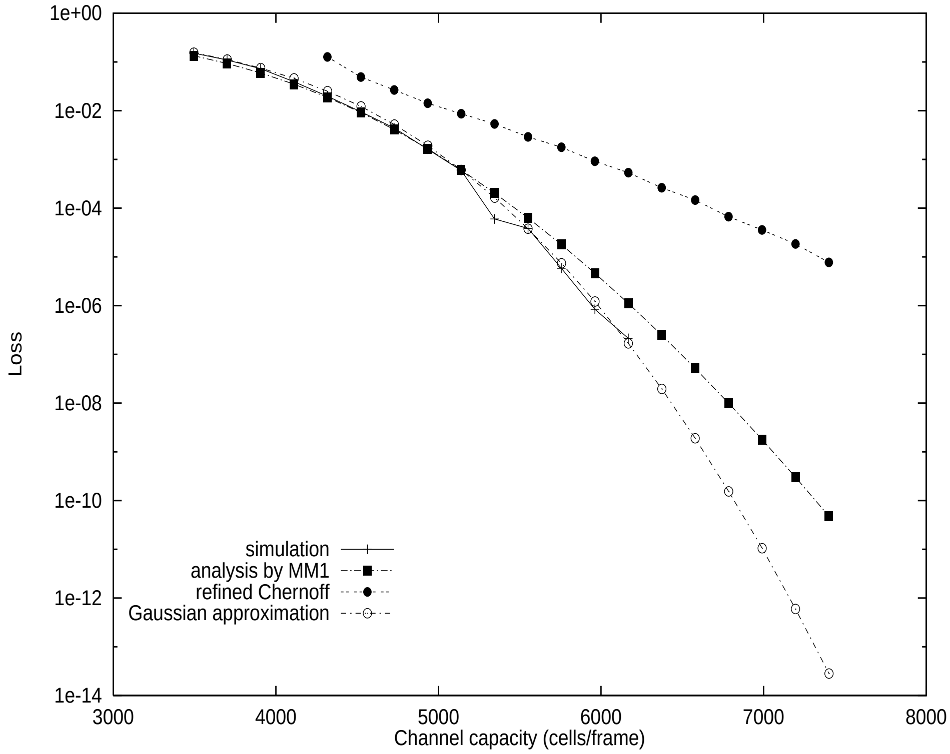


Figure 3.10: Cell-loss ratio of 100 channels

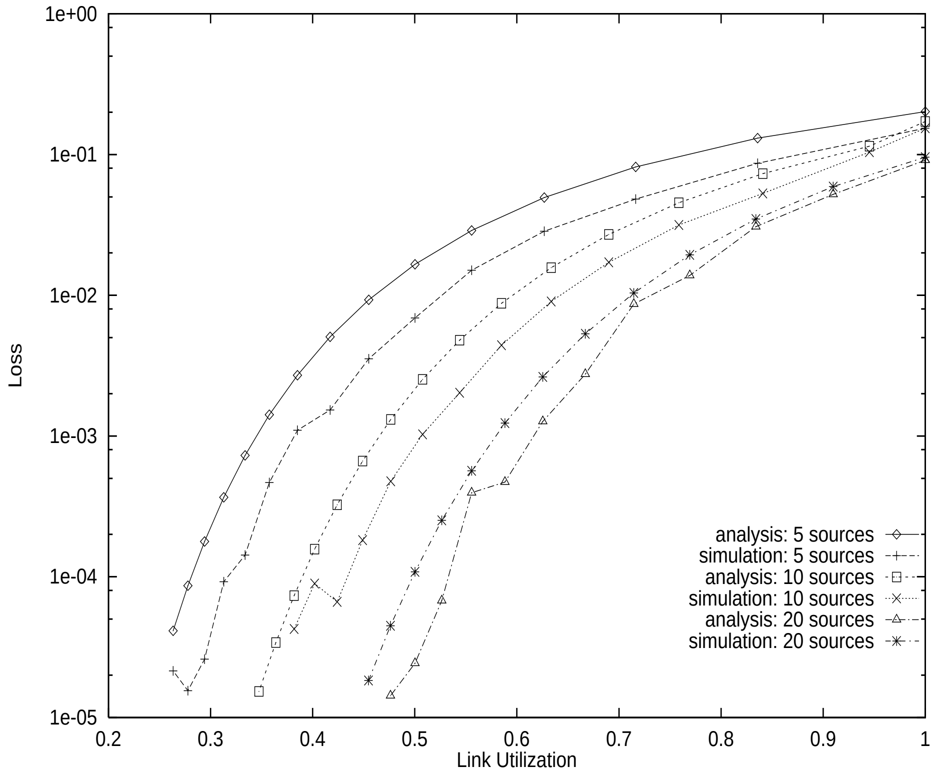


Figure 3.11: Statistical multiplexing gain

against the average cell-arrival rate of the aggregate sources. We show only the $M/M/1/N$ bound. One can see that the loss ratios are bounded for all three cases and thus, the histogram-based model satisfies our requirements regardless of the number of statistical real-time channels multiplexed. Moreover, the statistical multiplexing gain is shown to increase with the number of channels multiplexed. However, in order to establish a macro-channel with the cell-loss probability of 10^{-4}, we need to reserve the bandwidth which is twice the average cell-generation rate. This results from the high burstiness of MPEG data and is inevitable in order to satisfy the QoS requirement. Although the macro-channel's utilization is about 0.5, it does not necessarily mean the waste of bandwidth since the unused bandwidth by the macro-channel can be used for transmission of best-effort traffic by TCRM, as in the case of real-time channels [51].

Finally, we investigated the effects of buffer size on the cell-loss ratio. We set the buffer size to 500 cells. The cell-loss ratios obtained from the simulation are much smaller than the bound as seen in Figure 3.12. This indicates that the assumption that a macro-channel with the aggregate input reaches the steady-state quickly is violated in a large-buffer system. The queue trajectory in Figure 3.13 confirms this reasoning. The buffer size N is 500 cells, the cell-arrival rate is 2,000 cells/frame, and the service rate is 1,300 cells/frame. Therefore, the

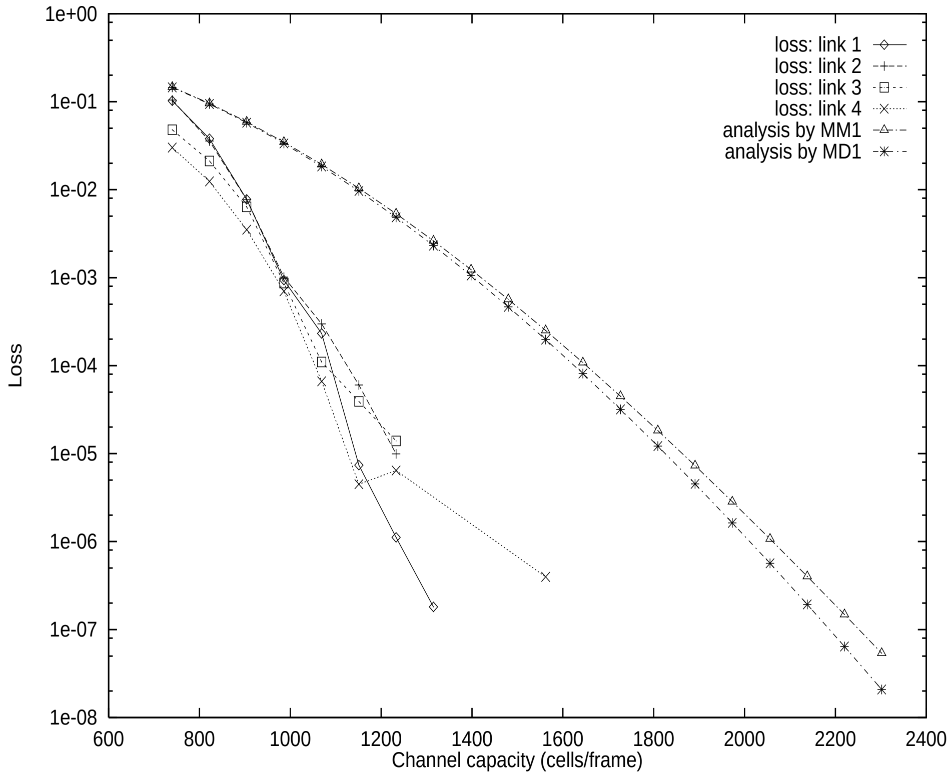


Figure 3.12: Effect of large buffer size

system is overloaded and cells are lost in a steady-state. Starting from the 'empty' state, it takes several frame intervals for the queue to reach its steady-state. Considering the fact that the cell-generation rate remains constant for at most one frame interval, the queueing system cannot reach the steady state under most overloaded conditions obtained from the histogram-based approach, and thus, the cell-loss ratio is quite low as compared to the estimate. This, in turn, indicates that our framework results in over-reservation of network resources when the buffer size is very large, but it still satisfies the basic requirements of statistical real-time communication since the cell-loss probabilities are bounded.

3.5 Conclusion

In this chapter, we have proposed a framework for statistical real-time communication in ATM networks. The framework is based on TCRM. While only one application is assigned to a single real-time channel in deterministic real-time communication, multiple statistical real-time channels are aggregated as a macro-channel over each ATM link and cells from different component channels of the macro-channel are serviced on a FIFO basis. To quantify the cell-loss ratio of a macro-channel, we have proposed to use a histogram-based model for

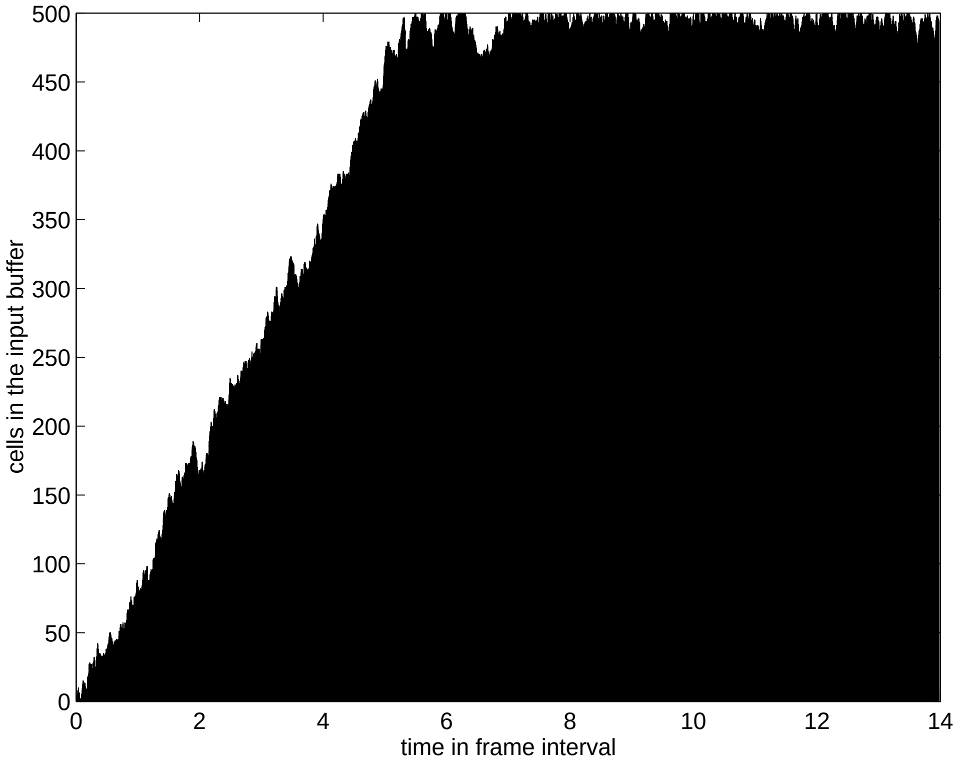


Figure 3.13: Queue trajectory in a large buffer system

the input traffic specification of VBR video sources. The histogram-based model specifies a VBR video source with the histogram of time-sampled traffic-generation rate. Also, in the histogram-based model, input traffic specification for aggregate sources is derived by taking the convolution of the histograms of component sources, making it simple to apply this model to a set of statistical real-time channels. Assuming a constant cell-arrival rate, we have modeled the arriving traffic at the macro-channel as a Poisson process and derived the cell-loss ratio from the $M/D/1/N$ analysis. Then, the cell-loss ratio of the macro-channel is given as the weighted sum of cell-loss ratios, and the weight was given by the rate-histogram. For the channel-establishment process, we have employed the $M/M/1/N$ analysis result for simplicity. Based on this analysis of a macro-channel, we have extended a statistical real-time channel to the multiple-hop case. In order to consider the traffic change in macro-channels at intermediate links, we have made an assumption of independence of cell inter-arrival times at the intermediate links. Lastly, our scheme was extended in the context of VPs.

We have conducted a simulation study using MPEG-coded video data in order to evaluate the effectiveness of our framework for statistical real-time communication and validated the assumptions used. The simulation results have reasonably well matched the analysis which is based on the assumptions such as the histogram-based modeling and Poisson arrival at each link, although, in some cases, over-reservation of network resources has been observed.

CHAPTER 4

SEMI-REAL-TIME COMMUNICATION

4.1 Introduction

In this chapter, we propose a new service category, the semi-real-time class for VoD-like applications, and present the network control functions necessary. As discussed in Chapter 1, if the user-requested delay bound is fairly large (e.g., several seconds to several minutes), the required bandwidth is closer to the average packet-generation rate than the peak packet-generation rate. We defined such characteristics average-rate-oriented, or semi-real-time in Chapter 1.

Instead of using a deterministic traffic envelope [14, 15, 24, 44], we employ a *statistical traffic envelope* derived by exploiting the statistical characteristics of source traffic. The statistical traffic envelope enables us to greatly reduce the amount of network resources to be reserved and thus accept more connection requests at the expense of a small percentage of packet losses. Then, we present a traffic regulation scheme which simplifies the derivation of statistical traffic envelopes in a multi-hop environment. In general, the traffic characteristic of a connection changes as the traffic travels through the network, and it is difficult to keep track of such changes and accurately describe the traffic characteristics at each node. Thus, we employ a traffic regulation scheme in order to preserve the source traffic characteristic at every intermediate node along the path of each connection.

The chapter is organized as follows. Section 4.2 introduces some basic definitions and concepts needed for the development of our approach. Sections 4.3 and 4.4 describe the proposed approach for providing VoD services. Section 4.5 evaluates the proposed approach and demonstrates its effectiveness in providing VoD services on an integrated services network. The chapter concludes with Section 4.6.

4.2 Background and Problem Statement

In this section, we define our service model for an integrated services packet network, review the deterministic traffic envelope from which a statistical traffic envelope is derived, and introduces a statistical traffic envelope.

4.2.1 Service Model for an Integrated Services Packet Network

For efficient handling of diverse traffic in an integrated services network, we categorize traffic into three classes according to their QoS requirements: real-time, semi-real-time, and best-effort. Real-time, semi-real-time, and best-effort classes of traffic are serviced with the highest, the second-highest, and the lowest priority, respectively.

The real-time service guarantees a bounded delivery delay for *each* individual packet at every switch along a fixed path so that its end-to-end delay may be bounded. For typical real-time applications, this delay bound must usually be within a very small range.[1]In order to guarantee such a small delay bound, different packet scheduling algorithms have been proposed and used for runtime packet transmissions after reserving resources along the path and performing an appropriate admission test based on the characteristics of source traffic [17,19,26,36,37,51,64,95,100]. In particular, the worst-case traffic-generation rate (i.e., peak packet-generation rate or minimum packet inter-arrival time), rather than the long-term average traffic-generation rate, plays a key role in the admission control. As a result, using the real-time service often leads to very low network utilization.

By contrast, many traditional data communication applications such as ftp and telnet do not require per-packet delay guarantee, but an average throughput is the only parameter of interest to them. These applications belong to the best-effort class.

There are other applications which lie between the real-time class and the best-effort class in terms of QoS requirement. They do not require very tight delay bounds like the real-time class does, yet providing packet delay bounds to them improves user-perceived QoS significantly. The requested delay bounds are much looser than those of the real-time class, *e.g.*, ranging from several seconds to several minutes. For example, guaranteeing loose packet-delivery delay bounds enables VoD service provider to dimension the required network resource and service quality. The semi-real-time class introduced here targets this type of applications. In our approach, each semi-real-time class packet is guaranteed to have a bounded delivery delay at each switch in a “statistical” sense, but the delay bounds

[1] For example, end-to-end delay bounds are within the range of several tenths to hundredths of a second for voice communication.

are much larger than those provided in the real-time service. This helps the semi-real-time service employ the network resources left unused by higher-priority (real-time) class traffic.

While each real-time connection receives an individual QoS guarantee, all semi-real-time connections receive the same QoS *collectively*, and thus, share the common delay characteristic and buffer space. The best-effort class is only provided with a *long-term* average throughput.

4.2.2 Deterministic Traffic Envelope

Originally, the deterministic traffic envelope was employed to analyze the delay and backlog characteristics of a real-time connection based on its worst-case traffic pattern and available network resources [14, 15]. Its function is to describe a deterministic upper bound on the amount of a real-time connection's traffic. Specifically, we define a traffic envelope function, $\hat{A}(\tau)$, such that

$$\hat{A}(\tau) = \sup_{t \geq 0} A[t, t+\tau], \qquad \tau \geq 0, \tag{4.1}$$

where $A[t, t+\tau]$ is the amount of traffic generated by the source of a real-time connection in the interval $[t, t+\tau]$. Note that $\hat{A}(\tau)$ is a time-invariant deterministic bound since it constrains the amount of source traffic over any time interval of length τ. $\hat{A}(\tau)$ is also called the *empirical envelope* in [44], or the *minimum envelope* process in [7].

Now, we can characterize the aggregate traffic behavior using each real-time connection's traffic envelope when multiple real-time connections are established over a link. Suppose real-time connections can be characterized by the envelopes $\hat{A}_m(\tau), 1 \leq m \leq M$. Note that these envelopes could be different from the traffic characteristics at the network entrance because of traffic fluctuations inside the network. But, for the time being, we focus on the single-hop case and do not consider the effect of traffic fluctuations inside the network. (The multi-hop case will be considered later.) The traffic envelope of aggregated real-time connections is then simply given as the sum of all the connections' traffic envelopes, $\sum_{m=1}^{M} \hat{A}_m(\tau)$, and, since semi-real-time connections use the resources left unused by real-time connections, they collectively receive a service of at least

$$S_{SR}(\tau) = \{\ell\tau - \sum_{m=1}^{M} \hat{A}_m(\tau)\}^+, \tag{4.2}$$

during any time interval of length τ where ℓ is the link capacity and $[x]^+ \stackrel{def}{=} \max\{0, x\}$. $S_{SR}(\tau)$ is called the *minimum service curve*. Assuming that the aggregate traffic of semi-real-time connections is characterized by the traffic envelope $\hat{A}_{SR}(\tau)$, the delay experienced

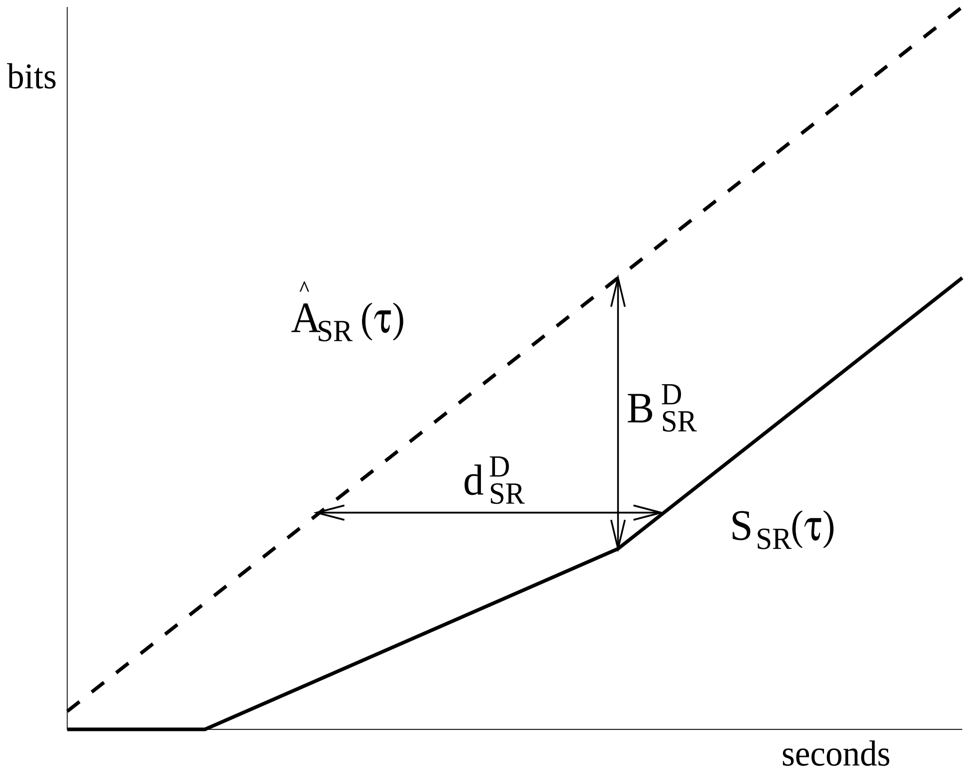


Figure 4.1: Maximum delay and maximum backlog

by any semi-real-time packet is upper bounded by

$$d^D_{SR} = \max_{t \geq 0} \ \min\{\tau : \tau \geq 0 \text{ and } \hat{A}_{SR}(t) \leq S_{SR}(t+\tau)\}, \tag{4.3}$$

and the backlog at the switch will be upper bounded by

$$B^D_{SR} = \max_{\tau \geq 0}\{\hat{A}_{SR}(\tau) - S_{SR}(\tau)\}. \tag{4.4}$$

The maximum backlog B^D_{SR} is the buffer requirement for loss-free semi-real-time service. Eqs. (4.3) and (4.4) are simply the horizontal and the vertical distance between the traffic envelope $\hat{A}_{SR}(\tau)$ and the service curve $S_{SR}(\tau)$, respectively. Figure 4.1 shows a graphical representation of delay computation.

4.2.3 Statistical Traffic Envelope

Since semi-real-time connections are admitted or blocked depending on the amount of resources left unused by real-time connections, it might be advantageous to have smaller traffic envelopes of real-time connections. However, since a traffic envelope is, by definition, a deterministic function, there is no room for deriving a tighter bounding function as long as the traffic envelope is deterministic. On the other hand, we can obtain a smaller traffic envelope by taking a statistical approach instead of a deterministic one. Unlike a deterministic traffic envelope which describes the worst-case traffic behavior, a statistical traffic envelope tries to provide a *statistically meaningful* bound on the amount of arrived traffic. This statistical traffic envelope is used to calculate the resources reserved but left unused by the real-time class traffic, and thus gives the semi-real-time class a larger pool of available resources than the deterministic traffic envelope can give to the semi-real-time class. As a result, by deriving and using a statistical traffic envelope for the real-time class traffic — for which resources have already been reserved based on the corresponding deterministic envelope — one can accommodate more semi-real-time class traffic than one would otherwise. Note that this statistical traffic envelope for the real-time class traffic is *not* used to derive the QoS parameters, like delay or packet-loss ratio, for real-time class traffic, but only for the semi-real-time class traffic. More specifically, when providing the required QoS for real-time, semi-real-time, and best-effort classes in an integrated services packet network environment, we represent the real-time class traffic with two traffic descriptors: deterministic and statistical traffic envelopes. First, the deterministic traffic envelope of real-time class traffic is used to calculate its QoS parameters. The network service provider then calculates the (statistical) amount of resources reserved but left unused by the real-time class traffic using the statistical traffic envelope of real-time class traffic. This calculated amount is used to service the semi-real-time class traffic. Since the statistical traffic envelope of real-time class traffic is given only in statistical terms, the actual amount of resources left unused by real-time class traffic can be smaller than the value calculated using the statistical traffic envelope. As a result, a certain percentage of semi-real-time packets may be lost. However, the QoS degradation of semi-real-time class connections is *bounded* by a pre-specified loss tolerance. Note that the QoS of real-time traffic is not affected by employing a statistical traffic envelope of real-time class traffic for servicing the semi-real-time class, as the QoS of real-time traffic is determined solely by its corresponding deterministic traffic envelope and real-time traffic will be serviced with higher priority than semi-real-time traffic.

While its deterministic counterpart is obtained by simply adding up the traffic envelopes of component connections, a statistical traffic envelope must be derived based on the statistical nature of the aggregated connections. A statistical traffic envelope is defined for both real-time connections and semi-real-time connections as follows.

Definition 4.1 *A statistical traffic envelope, $\tilde{A}(\tau)$, for a connection or a set of connections when the amount of traffic arrived during the time interval $[t, t+\tau]$ is given by $A[t, t+\tau]$,* *is defined as:*

$$\tilde{A}(\tau) \stackrel{def}{=} \min\{x: \ Pr\{A[t, t+\tau] > x\} < Z, \quad t \geq 0, \tau \geq 0\}. \tag{4.5}$$

The physical meaning of Definition 4.1 is that the amount of traffic arrived from the set during any time interval of length τ is smaller than or equal to $\tilde{A}(\tau)$ with the probability of $1 - Z$ or larger.

Using both the statistical traffic envelopes of the real-time class and the semi-real-time class, we can derive the statistical bounds of packet delay and packet loss of the semi-real-time class. First, we need to redefine the minimum service curve of the semi-real-time class in *statistical* terms as follows: If the statistical minimum service curve of the semi-real-time class is given by $S_{SR}(\tau)$, then, for any time interval of length τ, the semi-real-time class is guaranteed to receive a service of at least $S_{SR}(\tau)$ with the probability of $1 - Z$ or larger. Using Definition 4.1, we can obtain $S_{SR}(\tau)$.

Theorem 4.1 *Let us assume that we are given a statistical traffic envelope, $\tilde{A}_R(\tau)$, for the aggregate of all real-time connections. Then,*

$$S_{SR}(\tau) = \{\ell\tau - \tilde{A}_R(\tau)\}^+. \tag{4.6}$$

Proof: Let $S^0_{SR}[t, t+\tau]$ denote the amount of service provided to the semi-real-time class during $[t, t+\tau]$ for $t \geq 0$ and $\tau \geq 0$. Then,

$$S^0_{SR}[t, t+\tau] = \{\ell\tau - A_R[t, t+\tau]\}^+, \tag{4.7}$$

where $A_R[t, t+\tau]$ is the amount of traffic arrived from the real-time class during $[t, t+\tau]$. Then, from Definition 4.1,

$$A_R[t, t+\tau] > \tilde{A}_R(\tau) \tag{4.8}$$

with the probability less than Z. Let us assume that Eq. (4.8) holds. Then, by arranging terms, we obtain

$$\ell\tau - A_R[t, t+\tau] < \ell\tau - \tilde{A}_R(\tau). \tag{4.9}$$

If we take the $[x]^+$ function on both sides of Eq. (4.9), the l.h.s becomes $S^0_{SR}[t, t+\tau]$, but the relationship between both sides changes. That is, if

$$\tilde{A}_R(\tau) \geq \ell\tau, \tag{4.10}$$

$$\{\ell\tau - \tilde{A}_R(\tau)\}^+ = 0, \tag{4.11}$$

and from Eq. (4.9)

$$\{\ell\tau - A_R[t, t+\tau]\}^+ = 0. \tag{4.12}$$

Thus,

$$\{\ell\tau - A_R[t, t+\tau]\}^+ = \{\ell\tau - \tilde{A}_R(\tau)\}^+. \tag{4.13}$$

Otherwise, the inequality remains intact. Therefore, by taking the $[x]^+$ function on both sides of Eq. (4.9), we get

$$\{\ell\tau - A_R[t, t+\tau]\}^+ \leq \{\ell\tau - \tilde{A}_R(\tau)\}^+, \tag{4.14}$$

and thus,

$$S^0_{SR}[t, t+\tau] \leq \{\ell\tau - \tilde{A}_R(\tau)\}^+. \tag{4.15}$$

This holds with the probability less than Z as stated earlier. Therefore,

$$Pr[\ \ S^0_{SR}[t, t+\tau] > \{\ell\tau - \tilde{A}_R(\tau)\}^+\ \ ] \quad \geq \quad 1 - Z. \tag{4.16}$$

Adding the equality inside the probability function does not affect Eq. (4.16). Therefore,

$$Pr[\ \ S^0_{SR}[t, t+\tau] \geq \{\ell\tau - \tilde{A}_R(\tau)\}^+\ \ ] \quad \geq \quad 1 - Z. \tag{4.17}$$

Thus,

$$S_{SR}(\tau) = \{\ell\tau - \tilde{A}_R(\tau)\}^+. \tag{4.18}$$

□

In order to further reduce the delay bound and the buffer requirement of the semi-real-time class, we employ the statistical traffic envelope for the semi-real-time class as well as the real-time class, which is defined in the same way as the real-time class. Let $\tilde{A}_{SR}(\tau)$ denote the statistical traffic envelope of the semi-real-time class for any time interval of length τ with loss tolerance Z_2, and let $\tilde{A}_R(\tau)$ and $S_{SR}(\tau)$ denote the statistical traffic envelope and the statistical minimum service curve of the real-time class for any time interval of length τ with loss tolerance Z_1. We still assume that the link capacity is ℓ.

Theorem 4.2 *Using $\tilde{A}_R(\tau)$, $\tilde{A}_{SR}(\tau)$, and $S_{SR}(\tau)$, we define the following quantities:*

$$d_{SR} \stackrel{def}{=} \max_{t \geq 0} \ \min\{\tau : \tau \geq 0 \text{ and } \tilde{A}_{SR}(t) \leq S_{SR}(t+\tau)\}, \tag{4.19}$$

and

$$B_{SR} \stackrel{def}{=} \max_{\tau \geq 0}\{\tilde{A}_{SR}(\tau) - S_{SR}(\tau)\}. \tag{4.20}$$

Then, both (i) the probability that any semi-real-time packet is transmitted no later than d_{SR} and (ii) the probability that the backlog of the semi-real-time class is smaller than or equal to B_{SR} are larger than or equal to $(1 - Z_1) \cdot (1 - Z_2)$, respectively.

Proof: Without loss of generality, let us assume that a busy period of the semi-real-time class starts at time 0 and ends at time T. A busy period of the semi-real-time class is defined as the time interval during which packets belonging to the semi-real-time class are waiting for transmission, i.e., backlogged. Consider a semi-real-time packet p which has arrived at time t, where $0 < t < T$. Then, the amount of traffic arrived from the semi-real-time class including p during $[0, t]$ is smaller than or equal to $\tilde{A}_{SR}(t)$ with probability $1 - Z_2$ or larger by Definition 4.1. Let us assume that the amount of traffic arrived in the semi-real-time class including p during $[0, t]$ is smaller than or equal to $\tilde{A}_{SR}(t)$. Then, in order to ensure the completion of transmission of p, the semi-real-time class must receive a service of transmitting at least $\tilde{A}_{SR}(t)$. Since the semi-real-time class is guaranteed to receive at least $S_{SR}(s)$ during $[0, s]$ with probability $1 - Z_1$ or larger, $\tilde{A}_{SR}(t)$ will be provided to the semi-real-time class no later than $t + d^t_{SR}$ with probability $1 - Z_1$ or larger, where d^t_{SR} is given by

$$d^t_{SR} = \min\{\tau : \tau \geq 0 \text{ and } \tilde{A}_{SR}(t) \leq S_{SR}(t + \tau)\}. \tag{4.21}$$

Therefore, with probability $1 - Z_1$ or larger, p's delay is smaller than or equal to d^t_{SR}. If we relax the assumption that the amount of traffic arrived to the semi-real-time class including p during $[0, t]$ is smaller than or equal to $\tilde{A}_{SR}(t)$, p's delay is smaller than or equal to d^t_{SR} with probability $(1 - Z_1) \cdot (1 - Z_2)$ or larger. Lastly, since

$$d_{SR} \geq d^t_{SR}, \tag{4.22}$$

the probability that p's delay is smaller than or equal to d_{SR} is larger than or equal to $(1 - Z_1) \cdot (1 - Z_2)$.

Next, let us consider the backlog. As in the previous case, we assume that a busy period of the semi-real-time starts at time 0 and ends at T. For any τ, $0 \leq \tau \leq T$, the amount of traffic arrived to the semi-real-time class during $[0, \tau]$ is at most $\tilde{A}_{SR}(t)$ with probability $1 - Z_2$ or larger. Since at least $S_{SR}(\tau)$ is provided to the semi-real-time class during $[0, \tau]$ with probability $1 - Z_1$ or larger, the backlog of the semi-real-time class is at most $\tilde{A}_{SR}(t) - S_{SR}(\tau)$ with probability $(1 - Z_1) \cdot (1 - Z_2)$ or larger. Since by definition

$$B_{SR} \geq \tilde{A}_{SR}(t) - S_{SR}(\tau), \tag{4.23}$$

the backlog of the semi-real-time class is upper bounded by B_{SR} with probability $(1 - Z_1) \cdot (1 - Z_2)$ or larger. $\square$

Note that B_{SR} also indicates the buffer requirement for the semi-real-time class when the maximum overflow probability is given by $1 - (1 - Z_1) \cdot (1 - Z_2)$. Since all the packets are not lost during buffer overflow, the probability of packet loss is smaller than the buffer overflow probability. Still, we can use $1-(1-Z_1)\cdot(1-Z_2)$ as the bound on packet loss ratio.

4.3 Deriving Statistical Traffic Envelope

Unlike its deterministic counterpart that requires knowledge of the entire traffic pattern of a connection, the statistical characterization of traffic sources requires the knowledge of some statistical traffic parameters, such as average packet-arrival rate, peak packet-arrival rate, peak-to-average ratio, and burst duration. Based on these statistical parameters, we can derive a statistical traffic envelope.

One way to derive a statistical traffic envelope for an aggregate of multiple traffic sources is to use a stochastic-bounding approach [49, 98]. The idea of the bounding approach is to find a random process which stochastically bounds the amount of traffic generated by the source, and use it — instead of using the original pattern — for analysis. Selection of such a random process is critical in estimating the amount of resources necessary to achieve the given QoS, but no general solution to it has been reported thus far. In [18], on/off periodic sources with uniformly-distributed independent phases were used to find the network behavior in the worst case. Each on/off periodic source is supposed to bound one of original traffic sources. One can adopt this on/off periodic process as a bounding random process and find a statistical traffic envelope by using appropriate stochastic bounds, such as the Chebychev or Chernoff bound. However, our evaluation has shown this approach to be too pessimistic, due mainly to the conservative nature of the bounding approach in estimating the amount of traffic, as pointed out in [50]. Specifically, when the length of interval was even slightly large, the derived statistical traffic envelope was much bigger than the original deterministic traffic envelope. This is because an on/off bounding random process shows extremity in traffic burstiness irrespective of the length of an interval during which the amount of traffic is observed, unlike the deterministic traffic envelope that shows clear smoothing effect with a longer interval. As a result, the derived statistical traffic envelope was of little use in saving resources. The stochastic-bounding approach using an on/off periodic random process may serve as a guideline for estimating required network resources when the source traffic is purely stochastic or unknown, but does not give an efficient solution when the source traffic is somewhat deterministic as in VoD-like applications.

Instead of employing the stochastic-bounding approach, we take a simple and direct approach based on the Central Limit Theorem (CLT) [86]. This approach assumes existence of a trace of each source traffic, as in most VoD-like applications. For simplicity, we consider a discrete-time (instead of continuous-time) version of a statistical traffic envelope. By adjusting the sampling rate, we can make the discrete-time version as close to the continuous-time version as desired.

Now, let us divide the time axis into intervals of constant length called *slots*, and determine the traffic envelope at integer multiples of a slot. For example, a frame interval for MPEG video sequences can be viewed as a slot of length s. Then, our intermediate goal is to find the probability distribution of $A(ns)$, $Pr(A(ns) \leq x)$ for $n = 1, 2, 3, \cdots$, where $A(ns)$ is the amount of traffic generated by the aggregate of K connections during any n consecutive slots. Let $x_k(i)$ denote the amount of traffic arrived at connection k during the i_{th} slot. We assume that $x_k(i)$'s are wide-sense stationary. Then, we can define the following parameters for connection k:

$$\mu_{k1} \stackrel{def}{=} E\{x_k(i)\}, \tag{4.24}$$

$$\sigma_{k1}^2 \stackrel{def}{=} E\{x_k(i) - \mu_{k1}\}^2, \tag{4.25}$$

$$R_k(m) \stackrel{def}{=} E[\{x_k(i) - \mu_{k1}\}\{x_k(i+m) - \mu_{k1}\}] \tag{4.26}$$

for any i. The distribution of $x_k(i)$ is the same as $A_k(s)$ because of the assumed wide-sense stationarity. Now, assuming that K is sufficiently large, i.e., large enough to apply CLT and that connections are independent of each other, the amount of the aggregated traffic is distributed as:

$$\sum_{k=1}^{K} A_k(s) \sim N(\sum_{k=1}^{K} \mu_{k1}, \sum_{k=1}^{K} \sigma_{k1}^2). \tag{4.27}$$

Thus, we obtain the distribution function for the amount of the aggregated traffic during a single slot.

Now, let us consider a period of n consecutive slots. The mean of connection k's traffic is obtained easily as:

$$\begin{aligned} \mu_{kn} &= E\{A_k(ns)\} \\ &= E\{\sum_{i=1}^{n} x_k(i)\} \\ &= \sum_{i=1}^{n} E\{x_k(i)\} \\ &= \sum_{i=1}^{n} \mu_{k1} \\ &= n\mu_{k1}. \end{aligned} \tag{4.28}$$

In order to derive the variance of connection k's traffic for an interval of n slots, we need its autocovariance function $R_k(i)$. Thus,

$$\begin{aligned}
\sigma_{kn}^2 &= E\{\sum_{i=1}^{n} x_k(i) - \mu_{kn}\}^2 \\
&= E\{\sum_{i=1}^{n} x_k(i) - n\mu_{k1}\}^2 \\
&= E[\sum_{i=1}^{n} \{x_k(i) - \mu_{k1}\}]^2 \\
&= E[\sum_{i=1}^{n} \{x_k(i) - \mu_{k1}\}^2 + 2 \sum\sum_{i \neq j} \{x_k(i) - \mu_{k1}\}\{x_k(j) - \mu_{k1}\}] \\
&= \sum_{i=1}^{n} E\{x_k(i) - \mu_{k1}\}^2 + 2 \sum\sum_{i \neq j} E[\{x_k(i) - \mu_{k1}\}\{x_k(j) - \mu_{k1}\}] \qquad (4.29) \\
&= n\sigma_{k1}^2 + 2 \sum_{i=1}^{n-1} (n-i) R_k(i). \qquad (4.30)
\end{aligned}$$

Eq. (4.30) is obtained simply by rearranging the terms of Eq. (4.29). Assuming that streams from different connections are independent from each other, as in the case when $n = 1$, we get

$$\sum_{k=1}^{K} A_k(ns) \sim N(\sum_{k=1}^{K} \mu_{kn}, \sum_{k=1}^{K} \sigma_{kn}^2), \tag{4.31}$$

when the number, K, of connections is sufficiently large.

Now, from the normal distribution function, we can derive the statistical traffic envelope. Given the loss tolerance Z, the statistical envelope for a time interval of n slots is given by

$$\tilde{A}(ns) = erfc_n^{-1}(Z) \tag{4.32}$$

where

$$erfc_n(x) \stackrel{def}{=} \frac{1}{\sqrt{2\pi \sum_{k=1}^{K} \sigma_{kn}^2}} \int_x^{\infty} e^{-\frac{(y - \sum_{k=1}^{K} \mu_{kn})^2}{2 \sum_{k=1}^{K} \sigma_{kn}^2}} dy. \tag{4.33}$$

The use of CLT in estimating bandwidth for VoD systems was mentioned by Reisslein and Ross [71]. While bandwidth is the only parameter considered in their work, bandwidth and buffer requirements are considered together in our approach, so that one can save bandwidth at the expense of an additional buffer space.

4.4 Traffic Regulation for Intermediate Switches

A VoD service can be provided through a dedicated network or a general integrated services packet network. In the former case, the network is a single-hop network in many

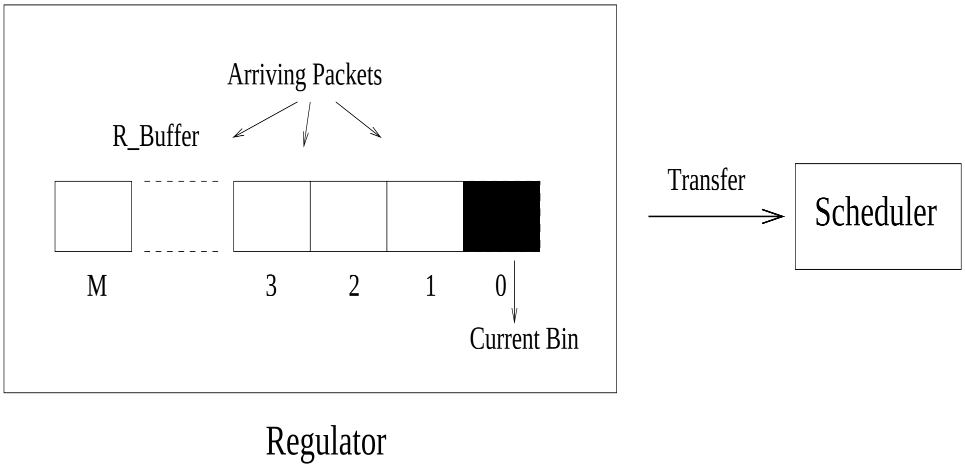


Figure 4.2: Traffic regulator

cases, e.g., satellite network or FDDI network. In this case, no traffic regulation is needed. In the latter case, however, connections usually run through multiple switches. Due to interferences by other concurrent streams, the traffic arrival pattern at the source node of a connection changes as its packets travel through the network, thus making it difficult to describe the traffic arrival pattern inside the network and obtain an appropriate statistical traffic envelope at switches. To avoid this difficulty, we enforce an appropriate traffic regulation at every switch. For the same reasons, many packet service disciplines have employed traffic regulation mechanisms along with a packet scheduling scheme [38, 51, 95]; see [95] for more elaboration on the benefit of traffic regulation.

In order to achieve traffic regulation that works well with our statistical traffic envelope, the parameters used in deriving the statistical traffic envelope must remain constant. That is, the mean, the variance, and the autocovariance of traffic measured over any slot must be kept constant. One way of achieving this goal is to confine traffic fluctuation within each slot, i.e., to prevent traffic fluctuation from spreading outside a slot.

Figure 4.2 shows the structure of a local switch equipped with such a traffic regulation function. Our scheme requires every local switch, including the source node, to cooperate with each other. At the source node, packets are stamped with their slot IDs representing the slot number during which they arrived. Packets which have arrived in a slot are given the same ID, and the slot IDs are consecutive numbers so that they may be represented with the smallest modulo number.

A traffic regulator consists of a clock and a receive buffer to store arriving packets. The receive buffer consists of M bins. The clock ticks once every slot, and points to one of the bins. The bin pointed to by the clock is called the *current bin*. When packets with the first slot ID from a particular connection arrive at the switch, the regulator inserts them into the current bin and record the slot ID as the connection's current slot. Whenever the clock ticks, the current bin of the regulator and the current slot ID increment to the next bin and the next slot ID, respectively. When a packet with a slot ID other than the current one arrives at the switch, it is inserted into one of the bins whose index is the difference between its slot ID and the current slot ID plus the index of the current bin. It is assumed that connections keep track of the series of their slot IDs.

When a bin becomes "current," all the packets in the bin become eligible for transmission and hence are transferred to the scheduler. The regulator is somewhat similar to the one used in [51]. The main difference is that packets generated in a slot share the same eligibility time. The number, M, of bins must be determined in such a way that M times the slot size must be equal to the delay bound at the previous node, so that those packets which have arrived early may not be lost as a result of insufficient buffer space.

A regulator can be implemented in either dedicated hardware or software. In a software implementation, a simple linked-list can work as a bin. All the packets with the same eligibility are stored in a single linked-list.

Although a regulator is assumed to serve a single connection, the switch can be designed so that a single regulator may be shared among all the connections passing through the switch. In such a case, the number of bins must be set to the largest among the delay bounds of all the connections at each connection's previous switch.

Since a regulator prevents traffic fluctuation from spreading outside one slot, the traffic characteristic within a slot is kept constant at all switches along the path, and thus, the statistical traffic envelope approach described earlier works in exactly the same way at any switch inside the network.

4.5 Empirical Results

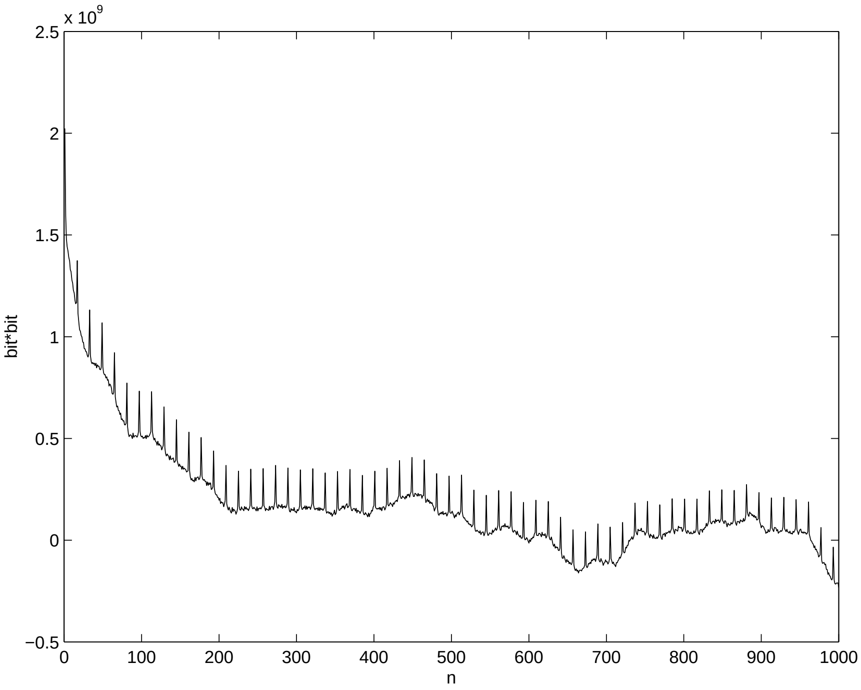
In this section, we present trace-driven simulation results using a set of MPEG-coded motion video traces, and demonstrate the effectiveness of the proposed statistical traffic envelope. We also comparatively evaluate the statistical traffic envelope and the deterministic envelope in terms of network utilization.

Although we defined the statistical traffic envelope in terms of the distribution of the amount of traffic which has actually arrived during a given time duration, the envelope can also act as an estimate of the worst-case traffic envelope, depending on the choice of the loss tolerance Z. To verify this aspect, we derived the worst-case traffic pattern for any number of slots from an aggregated set of video sequences, and compared it against the statistical traffic envelope derived with the procedure described in Section 4.3.

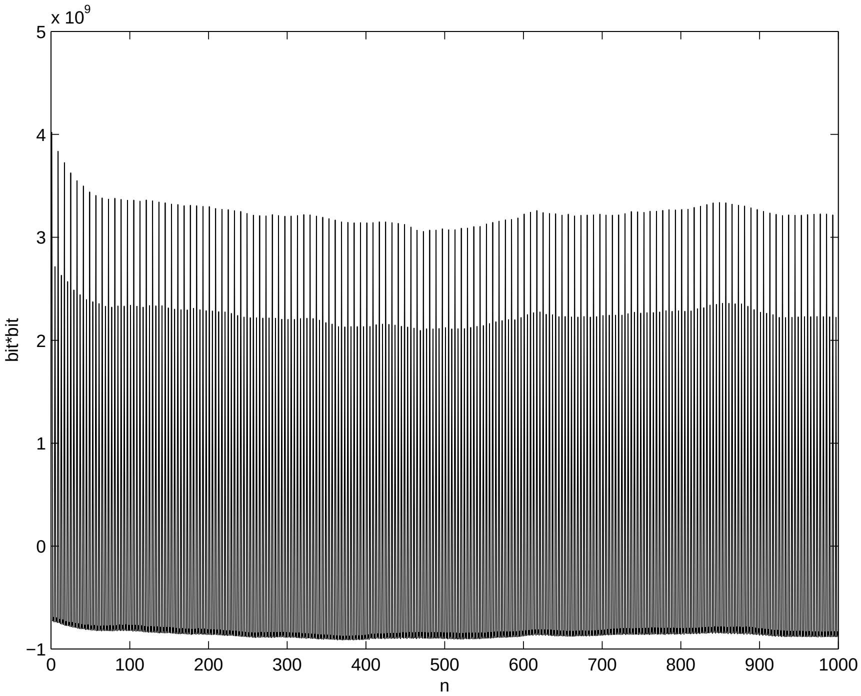
We employ the compressed video sequences shown in Figure 2.4 for the empirical evaluation of the derived statistical traffic envelope: *Starwars* (Sequence 1) and *Honey, I Blew Up the Kids* (Sequence 2). Sequence 1 has more scene changes than Sequence 2, and thus, the correlation is higher with quite large lags, as seen in Figure 4.3 which plots the autocovariance of each sequence. In Sequence 1, the differences in I, P, B frames are not as significant as in Sequence 2 because of its high active scene changes, and this trend is captured in the autocovariances.

From these two traffic sources, we composed N different video sequences in such a way that we first wrapped around them and selected the starting points randomly. We then multiplexed them together and obtained the aggregate traffic of N connections. For this aggregate traffic, we derived the worst-case traffic pattern for a certain length of time, which is the worst-case traffic envelope for the aggregate traffic. We ran this simulation 1000 times each for $N = 50, 100$ and 200, respectively. Figures 4.4 and 4.5 plots the results when N =100 for Sequence 1 and Sequence 2, respectively. The solid-line curves indicate the worst-case traffic pattern for each run. For the purpose of comparison, we plotted both deterministic and the statistical traffic envelopes. The deterministic traffic envelope for 100 sequences was derived by first calculating the deterministic traffic envelope for a single sequence and then multiplying it with N =100. The statistical traffic envelope for 100 sequences was derived using the procedure in Section 4.3. Here, we set the duration of a frame (1/30 sec) as a slot and set $Z := 10^{-9}$. This extremely small loss tolerance was chosen to reflect the fact that the statistical traffic envelope must be able to estimate a near worst-case traffic envelope. In Figures 4.4 and 4.5, the average amount of traffic arrived during the corresponding duration is also shown for the purpose of comparison. The worst-case traffic patterns for real traces are shown to be estimated accurately by the statistical traffic envelope and much smaller than the deterministic traffic envelope. Although the statistical traffic envelopes seem to be a linear function of time duration, they are not really linear because of the nonlinear variance functions derived by using autocovariances in Figure 4.3. Sequence 1 has a statistical traffic envelope closer to the worst-case traffic envelope, compared to Sequence 2, due to its highly active scene changes.

We conducted the same experiments with longer sequences (40,000 frames long or approximately 20 mins) and verified a trend similar to those in Figures 4.4 and 4.5. Figure



(a) Sequence 1



(b) Sequence 2

Figure 4.3: Autocovariances of frame size sequences

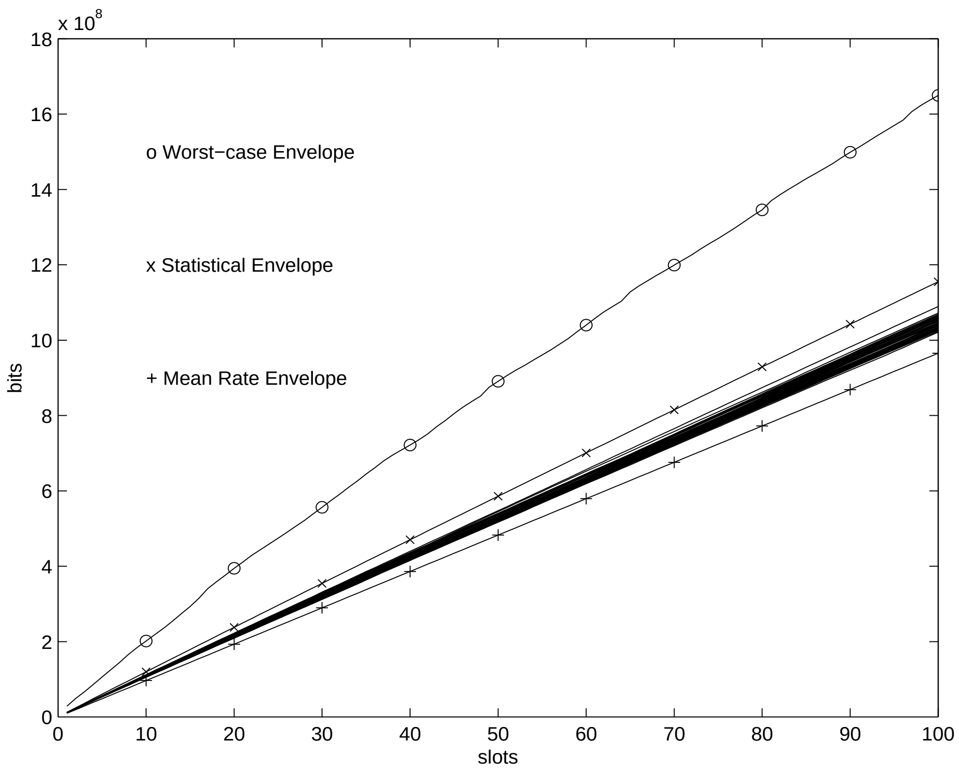


Figure 4.4: Empirical traffic envelope for 100 sequences of sequence 1

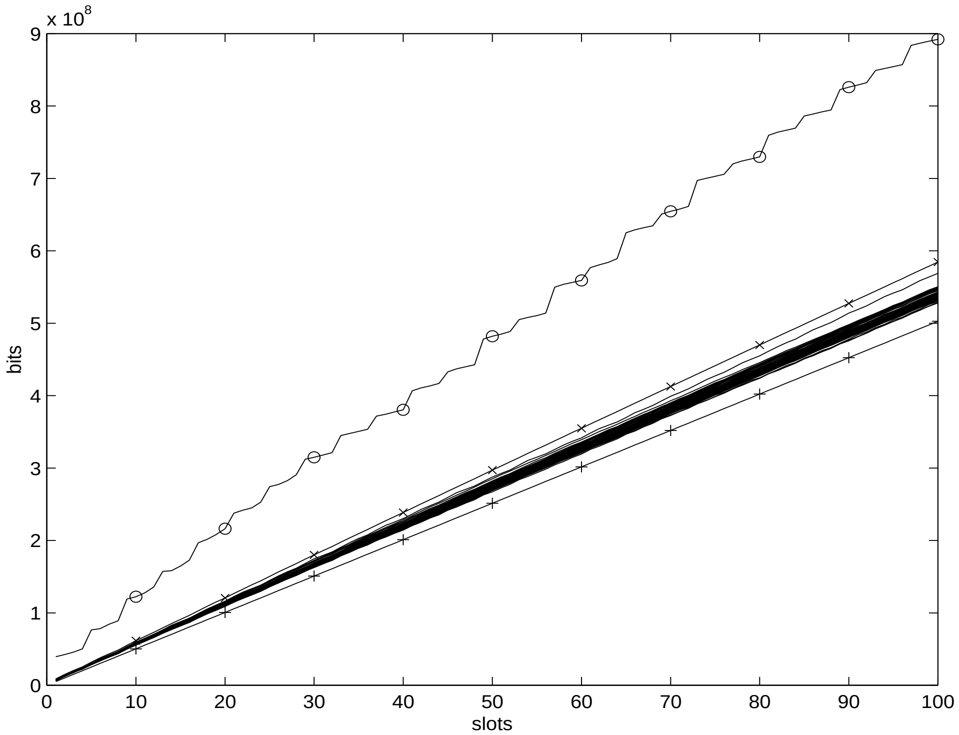


Figure 4.5: Empirical traffic envelope for 100 sequences of sequence 2

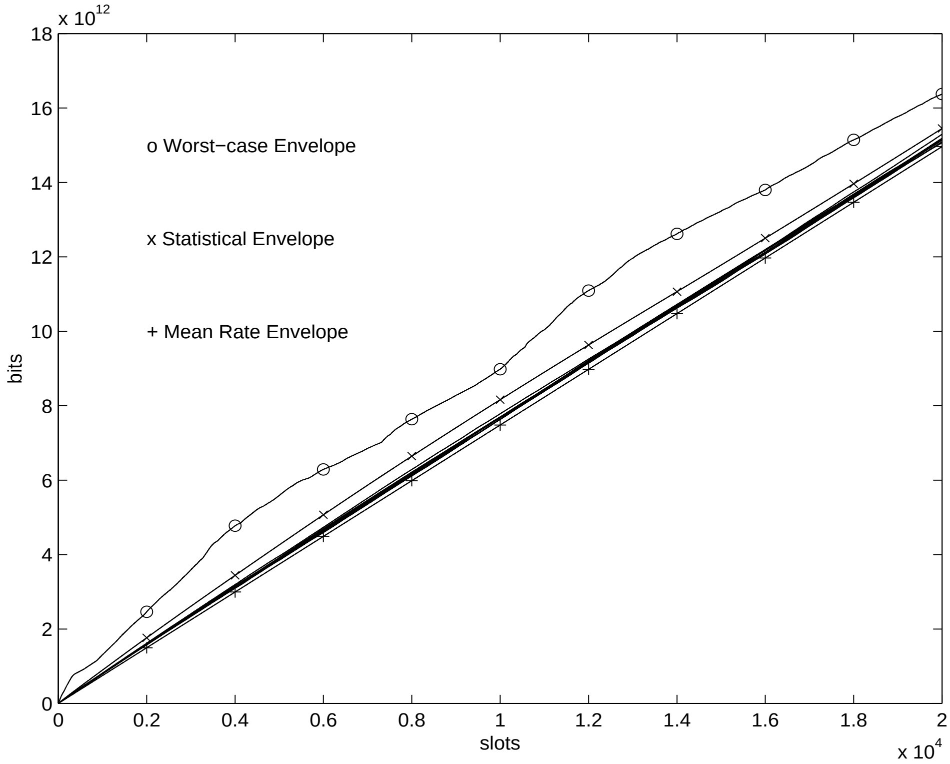


Figure 4.6: Empirical traffic envelope for 100 sequences of a 40,000 frame long clip

4.6 plots the result of using one of the longer sequences.

After verifying the effectiveness of the statistical traffic envelope as an estimate for the worst-case traffic, we investigated its usefulness by applying it to a VoD application. Since we have asserted that the identical traffic characteristics can be obtained at any intermediate node in the multi-hop environment by using traffic regulation, it suffices to consider single-hop semi-real-time communication. We consider a 155 Mbps link as a physical medium, over which three priority classes of traffic are transmitted: real-time, semi-real-time, and best-effort. Since the best-effort class doesn't affect the other two classes of traffic, it will not be considered further in our discussion.

Sequence 2 is used as the source traffic for the real-time class, and its average arrival rate is 1.50 Mbps. By setting the local link delay bound to 2.29 slots (76 msec), we were able to establish up to 30 connections using the admission test of the WFQ scheme [65]. The choice of this particular scheme for real-time communication is arbitrary and doesn't alter the conclusion drawn in this section. Since the aggregated average traffic arrival rate for the 30 sources is 45.0 Mbps, the network utilization is about 0.29. As argued in the introduction, the network utilization is very low when only the real-time service is provided for delay-sensitive applications. By providing the semi-real-time class, the remaining bandwidth can be used for other delay-sensitive applications. Under the condition that 30 connections of Sequence 2 are established over the link, we attempt to provide a VoD service by using the semi-real-time service. This time Sequence 1 is used as source traffic for the VoD service, and its average traffic arrival rate is about 2.89 Mbps. While varying the number of semi-real-time connections but keeping the 30 real-time connections made of Sequence 2, we derived the worst-case delay and buffer requirements of the VoD connections using the deterministic traffic envelope as well as the statistical one following the steps explained in Sections 2 and 3. The results are plotted in Figures 4.7 and 4.8. For a given delay bound, we were able to accommodate a much larger number of VoD connections by using statistical traffic envelopes than deterministic ones. Especially, when the delay bound is set to 5 slots (165 msec), 6 connections can be established with the deterministic traffic envelope and 24 connections with the statistical traffic envelope, thus a 400 % increase in network utilization. In terms of the buffer requirement, the statistical traffic envelope is far more advantageous than the deterministic traffic envelope as seen in Figure 4.8. When the number of connections is fixed, the difference in the buffer requirements shows an order of magnitude improvement even with an extremely small loss tolerance for the statistical traffic envelope, e.g., $Z_1 = Z_2 = 10^{-9}$.

Figures 4.9 and 4.10 show the results of comparing the delay bound and buffer requirement when the real-time class of Sequence 2 was removed, i.e., no real-time class traffic. In this case, the entire capacity of the link can be used for VoD applications. Considering the average traffic arrival rate of Sequence 1, up to 53 connections (the total traffic arrival rate is 153.2 Mbps) can be established over the 155 Mbps link. The delay bound and buffer requirement show similar trends as in Figures 4.7 and 4.8.

From Figure 4.9, we can draw an interesting conclusion on the usefulness of the semi-real-time class for VoD services: if we employ the real-time communication service for the VoD application, where the delay bound is set to, say, 1 slot (33 msec), then only 18 connections can be established. On the other hand, if we employ the semi-real-time communication service where the delay bound is 104 slots (3.43 sec), then the number of connections acceptable is increased to more than 51 (i.e., reserved bandwidth per connection is only 3.04 Mbps which is very close to the average traffic arrival rate). Thus, an almost 300 % increase in network utilization is achieved at the cost of extremely small packet loss probability and reasonable buffer requirement (66 Mbytes). This result indicates that, for VoD applica-

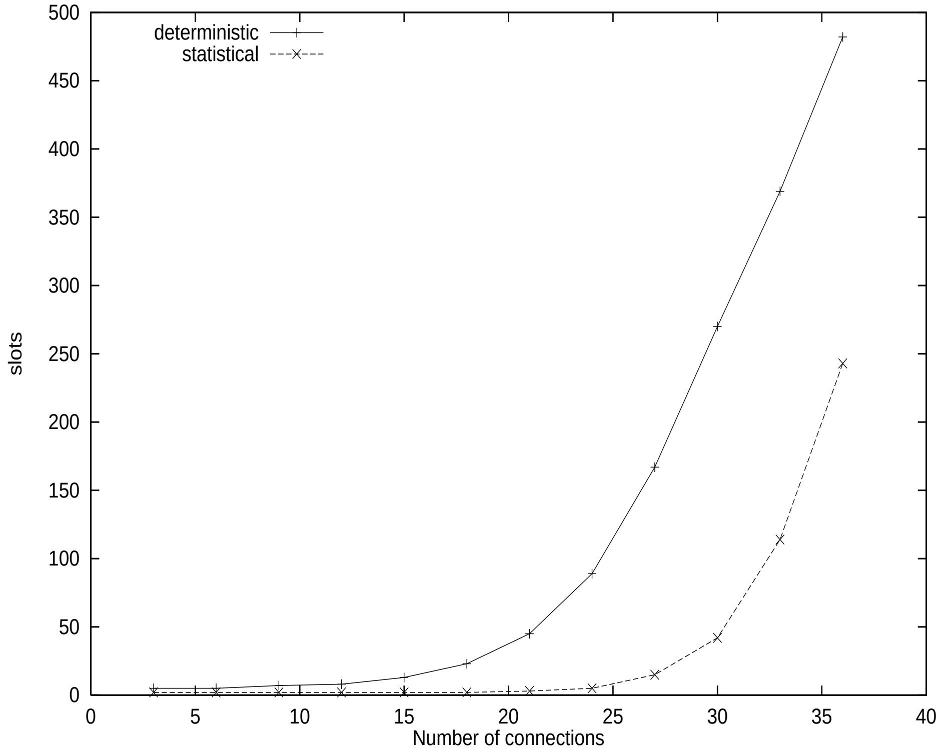


Figure 4.7: Worst-case delay comparison in the presence of real-time class

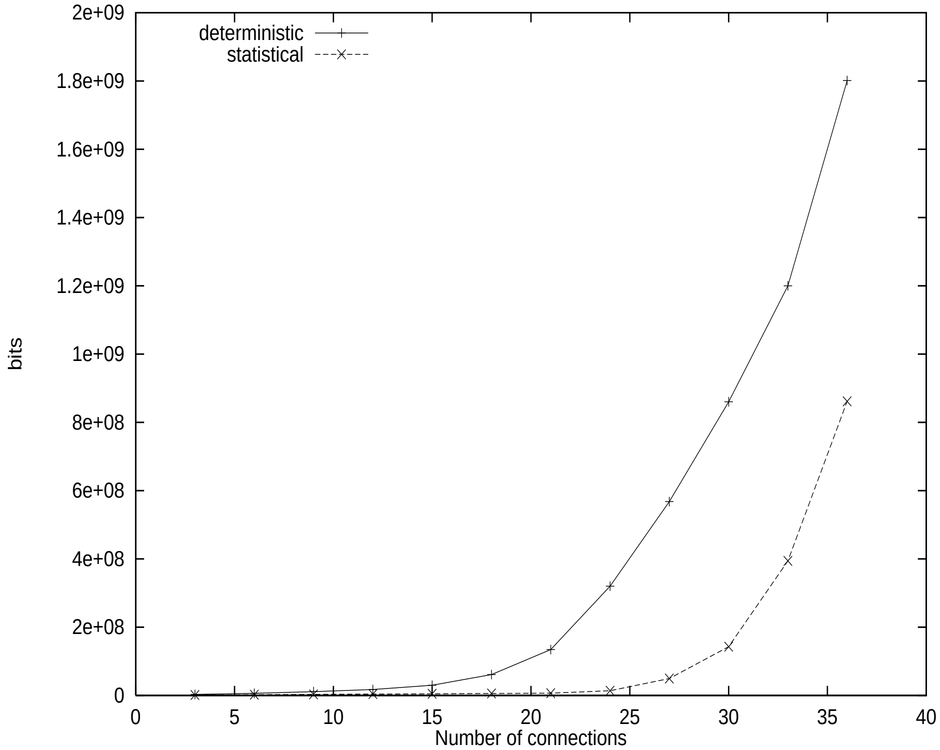


Figure 4.8: Buffer requirement comparison in the presence of real-time class

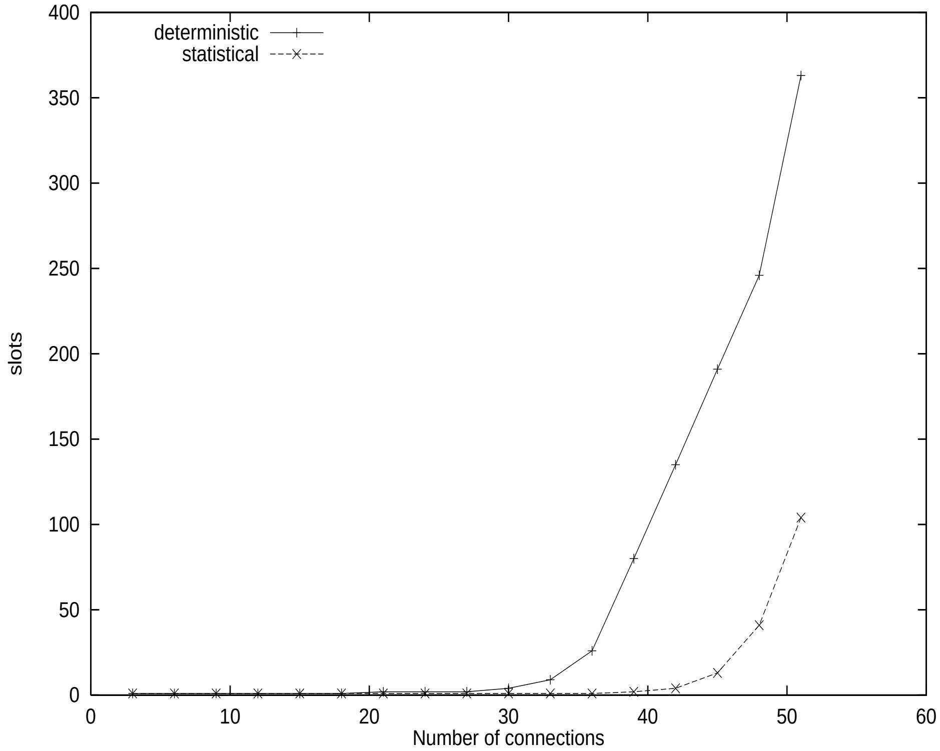


Figure 4.9: Worst-case delay comparison in the absence of real-time class

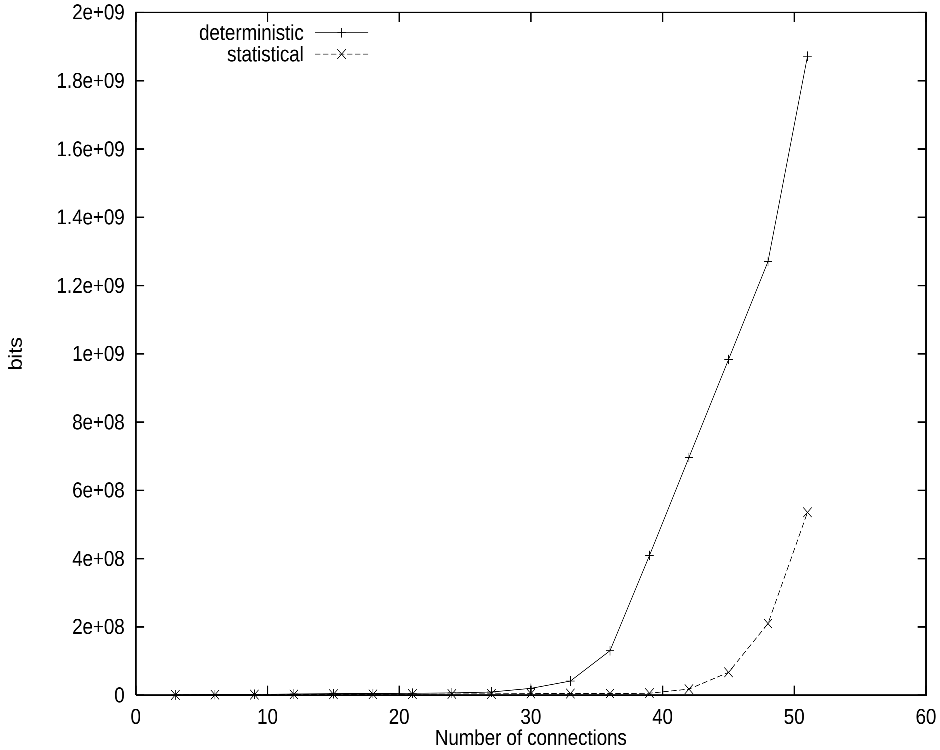


Figure 4.10: Buffer requirement comparison in the absence of real-time class

tions, the semi-real-time communication with statistical delay/loss guarantee is far more advantageous than the real-time communication service in terms of network resource utilization. In particular, in Figure 4.10, the semi-real-time communication service is shown to be able to accept more VoD connections by allowing larger input buffer space. This is unique compared with Reisslein and Ross' work [71] where they did not assume the usage of input buffer. Since we assume that packets belonging to a video frame arrive instantly, our approach always requires input buffer irrespective of the number of multiplexed VoD connections. On the other hand, Reisslein and Ross assumed that packets' arrival times are scattered over a frame period. If we follow such a smoothed traffic arrival assumption, the buffer requirement of the semi-real-time communication service will be lower. Assuming smoothed traffic arrival pattern, we calculated the number of connections that can be established subject to packet loss ratio, 10^{-9}, in a no buffer system. The number of connections of Sequence 1 was 39. As you can see in Figure 4.10, the semi-real-time communication service can support a maximum of 51 connections even without traffic smoothing, using a reasonable input buffer size ($= 5.36 \times 10^8$ bits).

In Figures 4.4, 4.5, and 4.6, we have shown that statistical envelopes with packet loss ratio bound 10^{-9} are good estimates of empirical worst-case traffic envelopes of aggregate sources. However, comparing traffic envelopes does not provide us with the exact packet loss behavior of VoD connections. To investigate the packet loss prediction capability of statistical traffic envelopes, we conducted the following trace-driven simulation. We multiplexed 53 connections of Sequence 1 over a 155 Mbps link, i.e., the maximum number of connections that can be supported by the link, and measured packet loss ratio by using different buffer sizes. The used buffer sizes were $10^7, 5 \times 10^7$, $10^8, 5 \times 10^8$, and 10^9 bits. The result is shown in Figure 4.11. When the buffer size was set to 10^9 bits, the measured packet loss ratio was 0, and thus is not shown in the figure. The 99 % confidence interval was set to 10^{-9}. To compare the simulation results with the analysis, we obtained the required buffer size subject to packet loss ratios, $10^{-1}, 10^{-3}, 10^{-5}, 10^{-7}$, and 10^{-9} using Theorem 4.2 in Section 4.2. As shown in Figure 4.11, the analysis results upper bound the simulation results.

We also investigated through simulation the case that real-time class traffic affects the packet loss behavior of semi-real-time class traffic. As in the previous case, we multiplexed 30 connections of Sequence 2 as real-time class (their delay bound was set to 2.29 slot (76 msec)) and different numbers of connections of Sequence 1 as semi-real-time class. The numbers of semi-real-time connections that can be established were 15, 18, 21, 24, 27, 30,

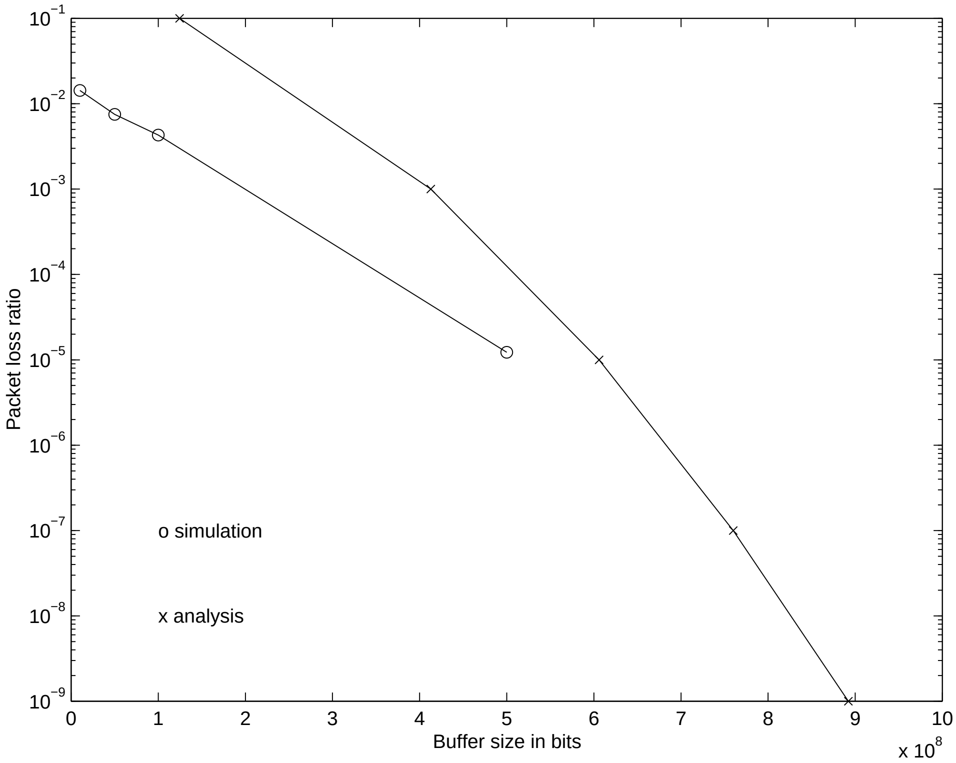


Figure 4.11: Loss ratio vs. buffer size in the absence of real-time class

33, and 36. The packet loss ratio was set to 10^{-9}, and delay bounds and buffer sizes were set to the values shown in Figures 4.7 and 4.8. For the 99 % confidence interval of 10^{-9}, the measured packet loss ratios were 1.0361×10^{-6} and 6.4947×10^{-9} when the numbers of semi-real-time connections were 15 and 18, respectively. In these cases, the measured packet loss ratios were much larger than the target value, 10^{-9}. In the other cases, the measured packet loss ratios were 0. This result verifies that our approach based on CLT is valid only when the number of multiplexed sources is sufficiently large.

We conducted more simulations using a larger set of video sequences, and observed similar trends without any significant difference.

The empirical study shows that a VoD service can be provided by reserving bandwidth similar to the sum of average bandwidths of each movie and provisioning a large input buffer, and most current VoD services follow this philosophy without any mathematical proof of performance guarantee. The semi-real-time communication service enables us to provide the same service with the similar amount of network resources but provides statistical QoS guarantees specifically.

4.6 Conclusion

In this chapter, we proposed a new service class called the semi-real-time class in an ISPN environment. Specifically, it is intended to provide the VoD service more efficiently than the case of using the real-time or best-effort class. We defined a statistical traffic envelope to characterize source traffic necessary for the semi-real-time service, so that high network utilization may be achieved at the expense of a small percentage of packet losses. Using a statistical traffic envelope allows us to estimate bounds of the buffer overflow probability and deadline-miss probability without conducting queueing analysis. Using the Central Limit Theorem, we obtained a statistical traffic envelope. Trace-driven simulations have shown the statistical traffic envelope to perform as intended as a worst-case traffic envelope in a statistical meaning. Through a numerical evaluation, we showed that VoD-like applications can be better serviced using the semi-real-time service in terms of consumed network resource. Compared to the case of using the real-time service, using the semi-real-time service resulted in nearly a 300 % increase in network utilization for an example case.

CHAPTER 5

STATISTICAL REAL-TIME COMMUNICATION OVER ETHERNET

5.1 Introduction

In Chapters 2 and 3, we considered the problem of providing real-time communication over ATM networks. Although ATM networks are deployed and being deployed as backbone networks in many areas and ATM LANs are being deployed, many different networks currently exist in the real world. In order to provide end-to-end QoS guarantees over such a heterogeneous network environment, one must be able to provide QoS guarantees in each individual network. In particular, most end systems are connected to WANs through LANs, and the most popular LANs are Ethernet. Therefore, providing predictable delay over Ethernet is a first step toward achieving end-to-end delay guarantees in the general internetwork environment. In addition, the low price and the availability of Ethernet device drivers on almost all operating systems make it attractive even for embedded systems like automated factories.

In this chapter, we derive a statistical bound on the channel access time of Ethernet by making several assumptions on input traffic and the functions of its MAC protocol. Specifically, we derive a relationship between the statistical bound and the allowed input level analytically. This analysis provides us with a connection admission control for statistical real-time communication over Ethernet. Our analysis considers the 1-persistent CSMA/CD MAC protocol with the Binary Exponential Backoff strategy (BEB) which is currently used in Ethernet.

The chapter is organized as follows. Section 5.2 defines the concept of a statistical real-time channel in the context of Ethernet. Section 5.3 derives the tail distribution of packet delay over Ethernet. We show the effectiveness of the derived CAC through an in-depth simulation study in Section 5.4. The paper concludes with Section 5.5.

5.2 Problem Statement

We first review some preliminary concepts necessary to state our main problem and discuss a software architecture for realizing real-time communication over Ethernet.

We define a *statistical* real-time channel for packet-switched networks in a similar way as we did for ATM networks in Chapter 3. A statistical real-time channel is defined as a unidirectional virtual circuit that guarantees the timely delivery of packets in statistical terms, i.e., the probability that a packet is lost during its transmission or misses its delivery deadline is less than a certain loss tolerance, Z:

$$Pr(\text{end-to-end packet loss}) \leq Z, \tag{5.1}$$

or

$$Pr(\text{packet delay} > \text{delay bound}) \leq Z. \tag{5.2}$$

Unlike WAN or other LANs (e.g., FDDI and FieldBus), Ethernet cannot control the medium access time of individual stations or connections. As a result, (1) only one type of QoS is provided to the entire component stations and all the connections established over the network, and (2) only one statistical real-time channel can be supported over a single Ethernet. For this reason, instead of using the above definition of a statistical real-time channel which was defined for general packet-switching networks, we introduce a new definition of a statistical real-time channel running over an Ethernet. In an Ethernet, if a newly-arrived packet results in a collision during its transmission, the transmission is stopped and the packet is rescheduled for transmission after some random delay. Hence, the delay that a packet experiences depends on the number of trials until its successful transmission. We therefore use the number of trials taken for a packet until its successful transmission as a performance measure in place of packet delay. A packet of a statistical real-time channel over the Ethernet must satisfy the following condition:

$$Pr(n \leq K) > 1 - Z, \tag{5.3}$$

where n is the number of trials taken to transmit the packet successfully. The counting starts when the packet arrives at the network from the application layer. By relating the number of trials to the delay that a packet experiences before its successful transmission, we can easily transform Eq. (5.3) to a delay value. Let D_K^* be the worst-case delay of a packet when its transmission has succeeded at its K_{th} trial. Then, the following condition

$$Pr(D \leq D^*_K) > 1 - Z \tag{5.4}$$

is guaranteed to hold according to Eq. (5.3).

By employing an appropriate traffic admission control, one must ensure Eq. (5.4) or Eq. (5.3) to hold in order to realize statistical real-time communication over Ethernet. The admission control must be performed in both the connection-level and the packet-level. In the connection-level, the network service provider determines whether to accept a new connection request from a component station into the statistical real-time channel based on the new aggregate input rate of the statistical real-time channel. After a connection is accepted through the connection-level admission control, it transmits its packets over the network. At this stage, each station must monitor whether each connection's packet-generation rate conforms to the negotiated value determined by the connection-level admission control. If a connection's packet-generation rate exceeds this value, the station must delay or discard those violating packets. This is a packet-level traffic admission control and called *rate control* or *traffic policing* in the literature. Since a statistical performance guarantee is provided to the entire set of connections of the network, traffic policing is indispensable for fairly distributing network resources to individual connections.

Figure 5.1 shows the software architecture employed in our approach for realizing statistical real-time communication over Ethernet. In order to minimize the required changes in the current Ethernet standard, we insert the middleware layer between the application and transport layers. Traffic policing/rate control is performed at this layer so that the aggregate packet-arrival rate at the Ethernet may be kept under a pre-defined target rate. Our analysis will provide the target input rate for a given loss tolerance and delay bound. As a transport layer protocol, we employ UDP instead of TCP. This eliminates the possibility of additional traffic generation other than from the application layer, which simplifies the delay analysis in Ethernet. Automatic Repeat Request (ARQ) of TCP generates additional traffic other than from the application layer, which makes it complicated to analyze packet delay in Ethernet.

5.3 Analysis of 1-Persistent CSMA-CD Protocol

We examine the relation between the throughput and the channel offered traffic rate, and derive the probability of successfully transmitting a packet during each trial in the context of BEB.

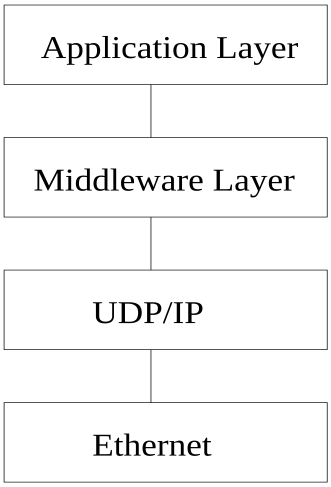


Figure 5.1: Software architecture

5.3.1 Modeling 1-persistent CSMA/CD networks

Our analysis is based on the following assumptions.

A1. The traffic source of Ethernet consists of an infinite number of users who collectively form an independent Poisson source with an aggregate mean packet-generation rate of λ packets/sec. This assumption is introduced to make our analysis tractable, and has been widely used for the performance analysis of contention-based MAC protocols [41]. It proved to be valid when the number of stations is sufficiently large, *e.g.*, over 20. Moreover, the arrival process formed by the times at which packets are scheduled for transmission or retransmission, called the *channel offered traffic*, is assumed to be generated from another infinite population by a Poisson process with a time-varying parameter μ, which we call the channel offered traffic rate. Obviously, $\mu \geq \lambda$. This infinite population model assumes that each station has at most one packet requiring transmission at any time. Therefore, we ignore the queueing delay at each station in our analysis. We employed a time-varying parameter for the channel offered traffic rate in order to handle the bursty traffic characteristics of Ethernet.

A2. The channel reaches the steady state when the throughput stays at a constant (steady-state) value. Let each packet be of constant length requiring T seconds to transmit, and let $S = \lambda T$. Then, S is the average number of packets generated per packet transmission time. Under the steady-state condition in which the traffic arrival rate is equal to the traffic output rate, S can also be referred to as throughput.

A3. The new packet-arrival rate is sufficiently low compared to the network's transmission capacity (10 Mbps for Ethernet and 100 Mbps for fast Ethernet). This assumption is unavoidable to realize real-time communication over Ethernet. We will later determine how low the arrival rate should be. If the arrival rate is not sufficiently low, a portion of packets will experience several collisions and hence large delays. This low arrival rate assumption will make the probability of a packet encountering multiple collisions negligible small.

A4. When a collision occurs, there are only two packets involved in that collision. In case of a low arrival rate, this is reasonable to assume. Also, we have to know the number of packets involved in a collision in order to derive the conditional collision probability when packets are retransmitted.

A5. The propagation delay between any two stations is equal to a constant, a. This is realistic for a 100 Base-T network which employs a star-topology wiring. For other topologies like a bus, one must set a to the largest propagation delay between any pair of stations. Since the collision probability increases as the propagation delay increases, choosing the largest propagation delay as a network parameter enables us to derive the worst-case (or an upper bound of) performance parameters.

Before describing our model, let's examine the operation of BEB briefly. When a packet collides with another packet, BEB sets the backoff time for the packet indicating when to try its retransmission. The backoff time is randomly chosen from $[0, 1, \cdots, 2^{n-1}]\times$ $slot\_time$, where n is the number of collisions the collided packet has experienced and $slot\_time$ is 512 bit times (in Ethernet, 51.2 μsecs, and in Fast Ethernet, 5.12 μsecs). Since the range of backoff time increases with the number of times a packet collided with other packets, the packet-arrival rate due to retransmissions is high when there are a small number of times packets experience collision.

Based on the above assumptions and observations, we model a 1-persistent CSMA/CD network as a semi-Markov process (SMP):[1]the SMP has two types of operating modes, depending on the packet-arrival rate: QUIET and BURST. These two operating modes are introduced to reflect the bursty nature of traffic arrival on Ethernet. In QUIET mode, the arrival rate of retransmitted packets is low, as compared to the arrival rate of new packets. This happens when retransmitted packets have already experienced a number of collisions, so that their backoff times are quite large. According to Assumption A1, in this mode, the

[1]This modeling draws on Vo-Dai's work [90] which we modified to accommodate the bursty nature of Ethernet traffic.

arrival process of both new packets and retransmitted packets composes a Poisson process with rate g. Upon occurrence of a collision, the system goes into BURST mode. In this mode, the channel offered traffic rate, β, is dominated by the arrival rate of retransmitted packets, and the collision probability is much higher than in QUIET mode. In an Ethernet where the arrival rate of new packets is quite low, if no collision occurs for a sufficiently long time, the channel offered traffic rate is very small and close to the arrival rate of new packets. However, once a collision occurs, the channel offered traffic rate increases abruptly due to the small backoff time. In this state, the probability of another collision when they are scheduled for retransmissions is very high. BURST mode is introduced to represent this situation. We can analyze the behavior of Ethernet in BURST mode by considering the worst-case scenario in terms of collisions. The worst-case scenario occurs when packets involved in a collision are newly-arrived. From Assumption A4, only two packets are involved in a collision at a time. In addition, we assume that the arrival process of both new and retransmitted packets become a Poisson process with rate β. Since the arrival rate of new packets is negligible compared to that of retransmitted packets, β is approximated by the arrival rate of retransmitted packets. Using these assumptions, we can determine β as follows. First, packets' backoff times are either 0 or 5.12 μsecs (51.2 μsecs) in Fast Ethernet (Ethernet)[2]since collided packets are all newly-arrived. Then, the probability that no packets will be scheduled during the first of two $slot\_time$ periods which are respectively $[0, 5.12)$ μsecs and $[5.12, 2 \times 5.12)$ μsecs, is 1/4. Thus, the average arrival rate in BURST mode, β, is obtained from the relation:

$$Pr(0 \text{ packets arrive during one } slot\_time) = 1/4.$$

Since we assumed that the arrival process is Poisson,

$$Pr(0 \text{ packets arrive during one } slot\_time) = e^{-\beta \cdot slot\_time},$$

and thus,

$$e^{-\beta \cdot slot\_time} = \frac{1}{4}.$$

Rearranging the terms,

$$\beta = \log 4/slot\_time \approx 1.3863/slot\_time.$$

Based on these two modes, we obtain the following SMP model for Ethernet with 9 states each of which belongs to either QUIET or BURST mode. (Figure 5.2 depicts each state.)

[2] From this point, we use fast Ethernet in all examples.

0: The QUIET-mode transmission state. A single station's transmission has lasted for the period of propagation delay, a, without interference from any other station and still continues, and the packet started transmission from the station in QUIET mode. Since all users are now aware of this transmission, they will not interfere with it.

1: The idle state. No station is transmitting packets on the channel. This state belongs to QUIET mode.

2: The QUIET-mode single contention state. One station started transmitting a packet in QUIET mode and continues transmission on the channel, but it has not been heard by all of the other stations. We assume that this state continues until the first transmission is heard by all the stations whether or not other stations transmit packets. Thus, the sojourn time of state 2 is the propagation delay, a.

3: The BURST-mode collision state. Two or more stations have been transmitting simultaneously and the first transmission was heard by all the stations. After sensing the collision, the stations stop transmitting their packets and transmit a jamming signal.

4: The QUIET-mode multi-contention state. Two or more stations start transmitting packets at exactly the same time in QUIET mode but their transmissions have not yet been heard by each other, or by others. This happens only when two or more stations schedule their packets for transmission in the same QUIET-mode transmission state (state 0), and thus, start transmitting packets as soon as the channel is free.

5: The BURST-mode singular collision state. This state is subordinate to states 4 and 7, and is entered from states 4 and 7 only when no other packets, except those which originally initiated states 4 and 7, are being transmitted.

6: The BURST-mode single contention state. One station started transmitting a packet in BURST mode and continues transmitting on the channel, but it has not been heard by all of the other stations. We assume that, as in state 2, this state continues until the first transmission is heard by all the stations whether or not other stations transmit. Therefore, the sojourn time of state 6 is the propagation delay, a, as in state 2.

7: The BURST-mode multi-contention state. Two or more stations start transmitting packets at exactly the same time in BURST mode but their transmissions have not yet been heard by each other or by others. This happens only when two or more stations schedule their packets for transmission in the same BURST-mode transmission state

(state 8), the same BURST-mode collision state (state 3), or the same BURST-mode singular collision state (state 5) (and thus, start transmitting packets as soon as the channel becomes free).

8: The BURST-mode transmission state. A single station's transmission time has lasted for the propagation delay, a, without interference from any other station and the packet has arrived during BURST mode. Since all users are now aware of this transmission, they will not interfere with it.

At any time, the channel can be in one of these 9 states. In this model, the channel starts in state 1 in QUIET mode. Upon occurrence of a collision, the channel enters, and stays in, BURST mode until the channel becomes idle.

For the embedded Markov chain of the SMP, the following steady-state equation must hold:

$$\pi = \pi M, \tag{5.5}$$

where $\pi = (\pi_0 \ \pi_2 \ \cdots \ \pi_8)$, π_i's are steady-state probabilities, and M is the state-transition matrix given by

$$M = \begin{pmatrix} 0 & A_1 & A_2 & 0 & 1 - A_1 - A_2 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ e^{-ga} & 0 & 0 & 1 - e^{-ga} & 0 & 0 & 0 & 0 & 0 \\ 0 & A_3 & 0 & 0 & 0 & 0 & A_4 & 1 - A_3 - A_4 & 0 \\ 0 & 0 & 0 & 1 - e^{-ga} & 0 & e^{-ga} & 0 & 0 & 0 \\ 0 & A_5 & 0 & 0 & 0 & 0 & A_6 & 1 - A_5 - A_6 & 0 \\ 0 & 0 & 0 & 1 - e^{-\beta a} & 0 & 0 & 0 & 0 & e^{-\beta a} \\ 0 & 0 & 0 & 1 - e^{-\beta a} & 0 & e^{-\beta a} & 0 & 0 & 0 \\ 0 & A_7 & 0 & 0 & 0 & 0 & A_8 & 1 - A_7 - A_8 & 0 \end{pmatrix}$$

where

$$A_1 = \int_0^\infty e^{-gx} dF(x),$$

$$A_2 = \int_0^\infty gxe^{-gx} dF(x),$$

$$A_3 = \int_0^\infty e^{-\beta(a+s+x)} dG(x),$$

$$A_4 = \int_0^\infty \beta(a + s + x)e^{-\beta(a+s+x)} dG(x),$$

$$A_5 = e^{-\beta(s+a)},$$

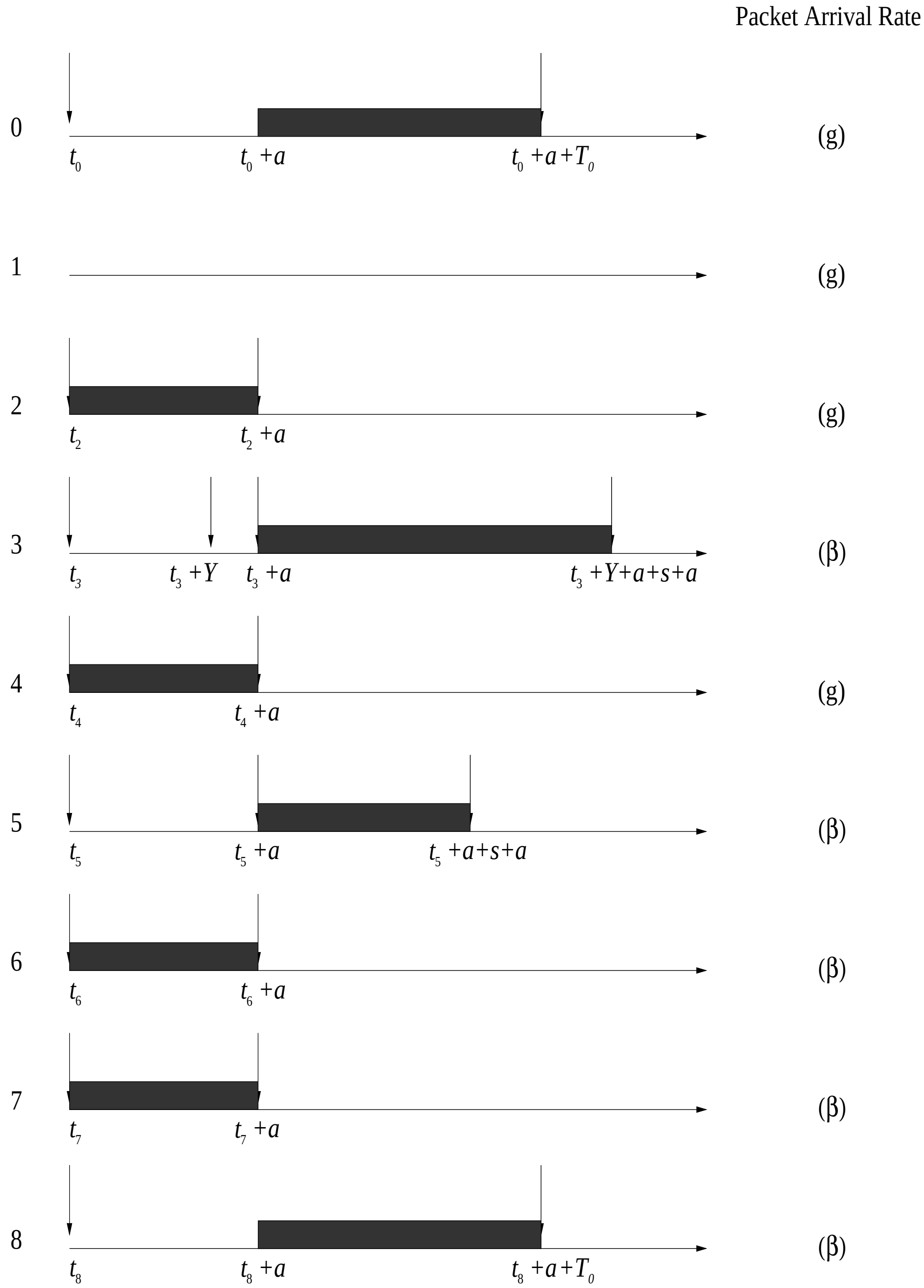


Figure 5.2: Channel states

$$A_6 = \beta(s+a)e^{-\beta(s+a)},$$

$$A_7 = \int_0^\infty e^{-\beta x} dF(x),$$

and

$$A_8 = \int_0^\infty \beta x e^{-\beta x} dF(x).$$

s is the jamming time[3]; $F(x)$ is the probability distribution function of packet-transmission times; and $G(x)$ is the probability distribution function of the latest transmission initiated within a contention period. In state 3 of Figure 5.2, $t_3 + Y$ indicates the latest transmission initiated within the contention period. See [90] for a detailed account of derivation of $G(x)$. Note that $G(x)$ depends on the packet-arrival rate in both contention states 2 and 6. Considering that the arrival rates for states 2 and 6 are g and β, respectively, let $G(x; g)$ and $G(x; \beta)$ denote the probability distribution functions of the latest transmission initiated in state 2 and state 6, respectively. Then, it is easy to show

$$G(x; g) \geq G(x; \beta), \quad \forall\ x.$$

That is, $Y(\beta)$ is *probabilistically larger than* $Y(g)$ [98], where $Y(g)$ and $Y(\beta)$ are the latest transmissions initiated in states 2 and 6 in reference to their starting times, respectively. Then, $G(x)$ is given as the weighted sum of $G(x; g)$ and $G(x; \beta)$. However, since one needs to consider the worst case in terms of packet loss, we choose $G(x; \beta)$ to be $G(x)$. Thus, we obtain

$$G(x) = \begin{cases} 0 & x < 0 \\ (e^{\beta x} - 1)e^{-a\beta}(1 - e^{-a\beta}) & 0 \leq x < a. \\ 1 & x \geq a \end{cases}$$

The transition probability is derived using the Poisson arrival assumption. For example, M_{02} given by A_2 is obtained by considering the fact that there must be only one packet arrival during one packet-transmission time, in order for the system to go from state 0 to state 1. Other elements of M are obtained in a similar way.

The steady-state probabilities, $\pi_i, i = 0, 1, \cdots, 8$ are a solution to both Eq. (5.5) and the following equation:

$$\sum_{j=1}^{8} \pi_j = 1.$$

Here we omit the algebraic solution for this system of equations.

[3]We merged the interframe gap into the jamming time and the packet-transmission time, respectively.

Let T_i denote the time spent in each state of the SMP. Then, the expected time spent in each state of the SMP, $E[T_i]$, is given by

$$\begin{aligned} E[T_0] &= \int_0^\infty x dF(x), \\ E[T_1] &= \frac{1}{g}, \\ E[T_2] &= a, \\ E[T_3] &= a + s + \int_0^a x dG(x), \\ E[T_4] &= a, \\ E[T_5] &= a + s, \\ E[T_6] &= E[T_2], \\ E[T_7] &= E[T_4], \end{aligned} \tag{5.6}$$

and

$$E[T_8] = E[T_0].$$

Eq. (5.6) is derived from the assumption that the arrival process is Poisson with rate g.

Based on the steady-state probabilities and the expected sojourn time in each state, we can derive the throughput S which is defined as the fraction of time spent on actual transmissions:

$$S = \frac{\pi_0 E[T_0] + \pi_8 E[T_8]}{\sum_{j=0}^{8} \pi_j E[T_j]}.$$

From the steady-state assumption, S can be considered as the input rate due to newly-arriving traffic at all stations.

Now, in order to derive the probability of a packet being transmitted successfully at its first trial, we insert a *probe* packet into this system. First, we calculate the probability of the probe packet finding the system in state i upon its arrival at the system as:

$$P_i = \frac{\pi_i E[T_i]}{\sum_{j=0}^{8} \pi_j E[T_j]}, \qquad i = 0, \cdots, 8.$$

This is based on the assumption that the new arrivals follow a Poisson process, and thus, that the probe packet's arrival time is uniformly-distributed in time.

Next, for each state, we can calculate the probability of the probe packet being transmitted successfully in that state as:

$$u_0 = \int_0^\infty e^{-gx} dF(x) \cdot e^{-ga}, \tag{5.7}$$

$$u_1 = e^{-ga},$$

$$u_2 = 0,$$

$$u_3 = \int_0^\infty e^{-\beta(a+s+x)} dG(x) \cdot e^{-\beta a},$$

$$u_4 = 0,$$

$$u_5 = e^{-\beta(s+a)} \cdot e^{-\beta a},$$

$$u_6 = 0,$$

$$u_7 = 0,$$

and

$$u_8 = \int_0^\infty e^{-\beta x} dF(x) \cdot e^{-\beta a}.$$

u_0 is obtained as follows. When the probe packet has arrived at the system which is in state 0, i.e., the QUIET-mode transmission state, it can be successfully transmitted only if there are no packet arrivals during the on-going packet-transmission time and no packet arrival during the probe packet's contention period, $[0, a]$. The probabilities of these events are given by $\int_0^\infty e^{-gx} dF(x)$ and e^{-ga}, respectively. Thus, we obtain u_0; similarly, other u_i's are obtained.

Finally, the probability of the probe packet being transmitted successfully at its first trial is given as a weighted sum:

$$P_s = \sum_{j=0}^{8} u_j P_j.$$

5.3.2 Calculating Success Probability upon Retransmission

Although we employed the Poisson arrival assumption on the packet-arrival process of the channel, in which a packet's waiting time until its transmission is independent of the number of trials, the collision probability of a packet depends heavily on the number of collisions that the packet has experienced, as well as the channel offered traffic rates, g and β, in Ethernet, because of its BEB strategy. It makes the probability of a packet's successful transmission dependent on the number of (re)transmission trials. So, a packet's success probability at its second trial is different from that at its first trial, and each trial for retransmission has a different success probability.

We approach this problem by considering the conditional probability that the probe packet is transmitted successfully at its second trial, given that its first trial has failed. In this case, the BEB strategy makes the station choose one of two backoff times, $0 \times slot\_time$

and $1 \times slot\_time$. In order to calculate the probability of collision at the second trial, we need information about the number of packets which have been involved in the first collision. As we did when determining β, we assume that the number of packets involved in the first collision is 2, including the probe packet itself. As we argued earlier, the network load condition in which real-time communication can be provided will be lightly-loaded. Thus, we expect the above assumption to be valid. Under this assumption, the two packets involved in the first collision must choose different backoff times in order for the second trial to be successful. Let the probe packet choose the first backoff time 0 and the other packet choose the second backoff time, $slot\_time$. The probability of this event is 1/4, and the probability of the probe packet being successfully transmitted in this event, P_{21}, is given by

$$\begin{aligned} P_{21} &= Pr(\text{no arrivals during the collision period}) \times \\ &\quad Pr(\text{no arrivals during the contention period of the probe packet}). \end{aligned} \tag{5.8}$$

Since we are dealing with collided packets separately from the other packet arrivals, the channel offered traffic rate in BURST mode is not β but g in this calculation, because the number of packets which were involved in the collision is assumed to be 2. Then, the first term in Eq. (5.8) is given by

$$Pr(\text{no arrivals during the collision period}) = e^{-g(s+2a)},$$

The length of the collision period is the sojourn time of state 3, and we take its maximum value, $s+2a$, in order to obtain the worst-case success probability. The second term is given by

$$Pr(\text{no arrivals during the contention period of the probe packet}) = e^{-ga}.$$

Thus,

$$P_{21} = e^{-g(s+3a)}.$$

Next, we consider P_{22}, the success probability when the probe packet chooses the second backoff time. Since the other packet has chosen the first backoff time, the probe packet must wait for the other packet to finish its transmission if it does not collide with a third packet. Otherwise, the situation gets more complicated. That is, when the other packet has collided with a third packet, we have too many possibilities of successfully transmitting the probe packet. In order to avoid such complexity, we simply set the success probability of the probe packet to zero in that case, which is the worst-case lower bound of the success probability. This affects our analysis very little since the probability that the other packet collides with a third packet is very small. Then,

$$\begin{aligned} P_{22} &= Pr(\text{no arrivals during the collision period}) \times \\ &\quad Pr(\text{no arrivals during the contention period of the other packet}) \times \\ &\quad Pr(\text{no arrivals during the transmission period of the other packet}) \times \\ &\quad Pr(\text{no arrivals during the contention period of the probe packet}). \end{aligned}$$

Substituting all the terms,

$$P_{22} = e^{-g(s+2a)} \cdot e^{-ga} \cdot \int_0^\infty e^{-gx} dF(x) \cdot e^{-ga}.$$

Then, the conditional probability that the probe packet is transmitted successfully at its second trial given that its first trial has failed, $P_{2|1}$, is given by

$$\begin{aligned} P_{2|1} &= \frac{1}{4}P_{21} + \frac{1}{4}P_{22} \\ &= \frac{1}{4}\{e^{-g(s+3a)} + e^{-g(s+4a)} \cdot \int_0^\infty e^{-gx} dF(x)\}. \end{aligned}$$

Similarly, one can approximate the conditional probability of the probe packet being transmitted successfully at the third or later trials given that it has failed at its previous trials. However, since we have already employed many assumptions in obtaining the conditional probability for the second trial, such approximations will add larger and larger errors as the number of trials increases. Thus, we use the conditional probability for the second trial as the estimate for the conditional probability for later trials instead of approximating them. Because of its smallest backoff time, the second trial provides the worst-case situation in terms of packet-arrival pattern for later trials. Thus, $P_{2|1}$ can be used as the worst-case estimate for later trials' conditional success probabilities. Thus, for $n > 2$, we set $P_{n|n-1} := P_{2|1}$.

Finally, we can obtain the probability of a packet being transmitted successfully within K trials using the conditional success probabilities. It is given in the following recursive form:

$$P_K = Pr(n \leq K) = P_{K-1} + (1 - P_{K-1})P_{K|K-1},$$

where $P_0 = 0$ and $P_{1|0} = P_s$.

In our approach, a packet which did not get transmitted within K trials is considered missing its delivery deadline, and thus, lost. In this case, the delivery deadline of a packet is given as the delay that a packet which is transmitted successfully in its K_{th} trial experiences in the worst scenario possible. Such a packet experiences the maximum delay possible when it experienced the longest waiting time and the longest backoff time until the $(K-1)_{th}$ trial in all its previous trials and finally succeeded in the K_{th} trial. Again, we pick such a packet as a probe packet. Let's derive this worst-case delay that the probe packet experiences when it succeeded in its K_{th} trial. The worst case, the probe packet being delayed before being scheduled for its first transmission trial, happens when the probe packet arrives for transmission at the system which is at the beginning of state 0. In this case, the probe packet must wait for the in-progress packet to finish its transmission before being transmitted. Since the length of the transmission time of the in-progress packet is given as T_0, its maximum is given as $\max(T_0)$. Upon completion of the transmission of the in-progress packet, the probe packet starts transmission. If the probe packet collides with other packets during this transmission period, the transmission of the probe packet will be stopped, and this interrupted transmission time is counted toward the delay of the probe packet. We call it the *collision period*. The length of the collision period is given by the sum of the sojourn times of states 2 and 3, i.e., T_2 and T_3. Since we are considering the worst case, we take the maximum value of T_3, $s + 2a$, so the maximum collision period is $3a+s$. After experiencing its worst-case collision period, the probe packet will be scheduled for retransmission (for the second trial). Again, in the worst case, the probe packet will take the largest backoff time, $1 \times slot\_time$, instead of $0 \times slot\_time$. If the system has entered state 0 by another packet just before the probe packet is scheduled for transmission in its second trial, the probe packet must wait for the in-progress transmission to finish again. The probe packet can then finish its transmission unless another collision happens. Thus, $2 \cdot \max(T_0) + T_2 + \max(T_3) + slot\_time$ is the worst-case delay of the probe packet before being transmitted in its second trial. The worst-case delay of the probe packet when it succeeds in its K_{th} trial is obtained using a similar argument. Thus, the worst-case delay when a packet is successfully transmitted within K trials is given by

$$D^*(K) = K \cdot \max(T_0) + (K-1) \cdot \{T_2 + \max(T_3)\} + \sum_{j=1}^{K-1} (2^j - 1) \cdot slot\_time + \max(T_0) \quad (5.9)$$

The last term, $\max(T_0)$, is to consider the probe packet's worst-case transmission time. Arranging terms, we obtain

$$D^*(K) = (K+1) \cdot \max(T_0) + (K-1) \cdot (3a+s) + (2^K - K - 1) \cdot slot\_time. \quad (5.10)$$

5.4 Simulation Results

To validate our analytic model, we simulated a 100BASE-T network with the following parameters. The transmission capacity is 100 Mbits/sec and the propagation delay between any two stations is 1.7 μsec assuming that the collision domain diameter (maximum distance between stations) is 400 m. The slot time ("unit" time for backoff) is 512 bit times or 5.12 μsecs. The interframe gap ("gap" time) is 0.96 μsecs and the jamming time is 0.48 μsecs. The number of stations is set to 20. Both fixed- and varying-size packets were simulated. For the case of fixed-size packets, the packet size was set to 64, 500, 1000 and 1500 bytes. For the case of varying-size packets, a mixture of packets of 64 and 1500 bytes were simulated. In the varying-size case, the analytic results were more optimistic in estimating the packet-loss rate than the simulation results, due mainly to the violation of the infinite population assumption. That is, even under lightly-loaded conditions, packet queues were built up by shorter packets during the transmission of longer packets. (Note that the number of packets is much larger if the packet size gets smaller while keeping the same load level.) This violates our assumption that each station can hold at most one packet. So our Ethernet model using the SMP process is ineffective in deriving a CAC for realizing real-time communication in the varying-size case. In this dissertation, we only present the comparison results for the fixed-size case.

We simulated the model for several seconds to dozens of seconds for various traffic loads. The number of packets arrived from the stations is approximately 50,000 for each load considered. We count the number of trials until a newly-arrived packet succeeds in its transmission and calculate the conditional success probability for each trial and derive the packet-loss ratio within n trials where $n = 1, \cdots, 5$.

Figure 5.3 shows the analytic result of the throughput for varying the channel offered traffic rate, g. Note that g is the rate of both new packet arrivals and retransmission trials in QUIET mode. Since we have two channel offered traffic rates, g and β in our model, Figure 5.3 must read differently from the results in [41, 90]. Figure 5.3 indicates the relation between g and the input load from outside the system. We determine g from Figure 5.3, given the input load, using the steady-state condition assumption as discussed in Section 5.3. Unlike g, β is a fixed parameter that is independent of the throughput.

Using the channel offered traffic rate g obtained from Figure 5.3, we calculated the conditional success probability for the first and second trials following the steps explained in Section 5.3, and plotted them in Figures 5.4, 5.5, 5.6 and 5.7 along with the simulation results. For all packet sizes, the analytic results precisely match the simulation results for

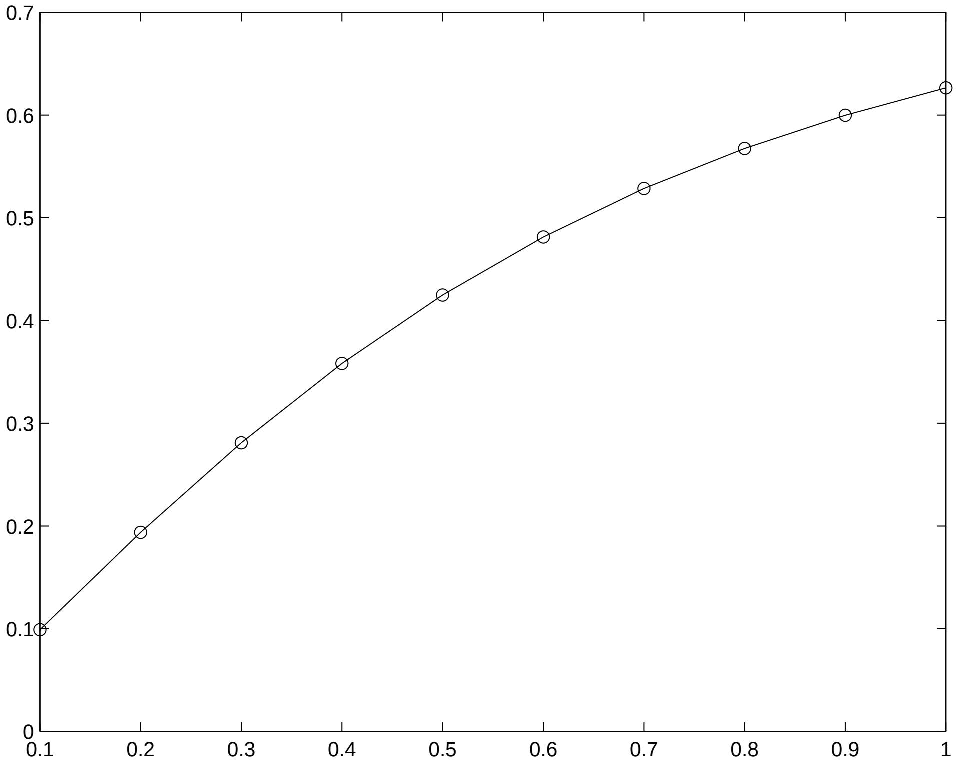


Figure 5.3: Throughput – 1500 bytes

the first trial under lightly-loaded conditions which are of prime interest to the design of real-time communication.

For the second trial, the analytic results provide a reasonable lower bound for the simulation results except for very lightly-loaded conditions, where the analytic results are more optimistic than the simulation results. However, before checking the correctness of the analysis, we need to investigate the credibility of the simulation. Among 50,000 packets generated for each simulation run, over 90% of them succeeded at their first trial, and thus, only several thousand packets were available for the second trials. This resulted in much larger confidence intervals for the second trials than in the first trials. Specifically, for the first trials, the 99% confidence intervals were less than 1%. For the second trials, they were 3.7% (for 64 bytes case), 7.7% (500 bytes), 9.3% (1000 bytes) and 9.9% (1500 bytes). For later trials, the confidence intervals were much larger, because the number of packets observed for the second trials decreased as the packet size increases. From this observation, we conclude that the analytic results can act as an estimate or a lower bound of the conditional success probabilities with acceptable errors for the input load range of interest.

For later trials, as we argued in Section 5.3.2, the estimate for the second trials is found to work as a good lower bound despite the large confidence intervals of the simulation

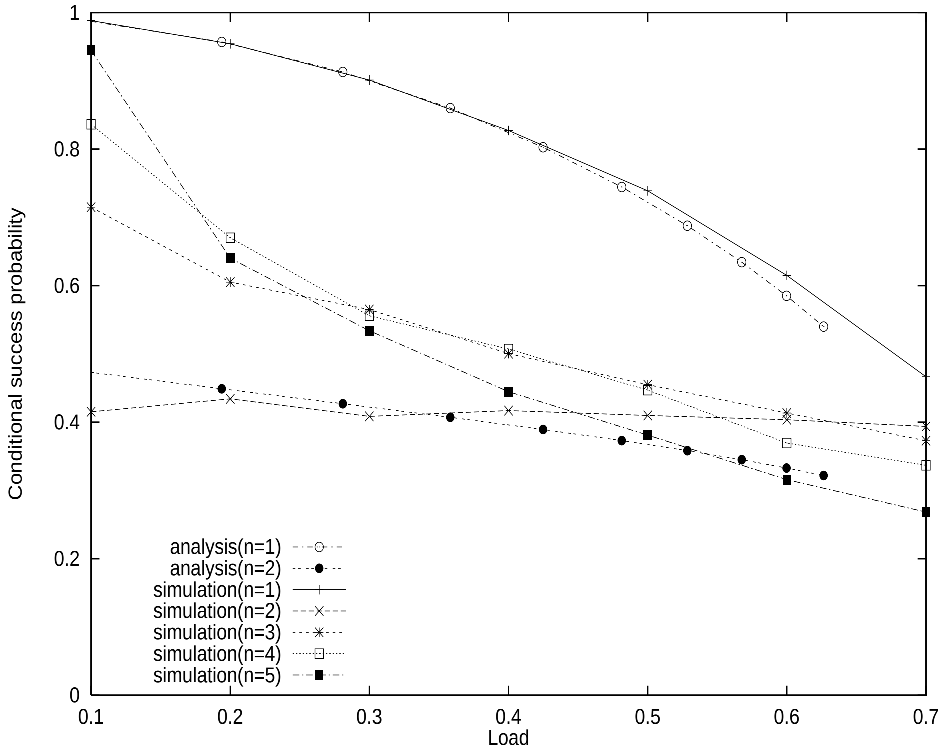


Figure 5.4: Conditional success probability – 1500 bytes

results.

Figures 5.8, 5.9, 5.10 and 5.11 show packet-loss ratios when a packet which did not succeed in its transmission within n trials is considered lost. We calculated the worst-case delay bound for each trial using Eq. (5.10). This is shown in Table 5.1. From Figure 5.8, one can see that the input load must be kept under 30% in order to achieve the packet-loss ratio under 1% if the packet-delivery deadline is set to 878 μsecs. For more stringent packet-loss ratios, we must either increase the delay bound or decrease the input-load level.

As the packet size decreases, the performance gets worse. That is, network utilization must be lowered significantly in order to keep the same packet-loss tolerance for a given n. This is because packet-arrival rates are higher in smaller packet sizes under the same input load. Of course, the delay bound gets smaller in case of smaller packet sizes for the same trial number as shown in Table 5.1. From this result, we can conclude that, for realizing real-time communication over Ethernet with reasonable packet-loss ratios, choosing a larger packet size is more advantageous as long as this does not result in an unreasonably large delay bound.

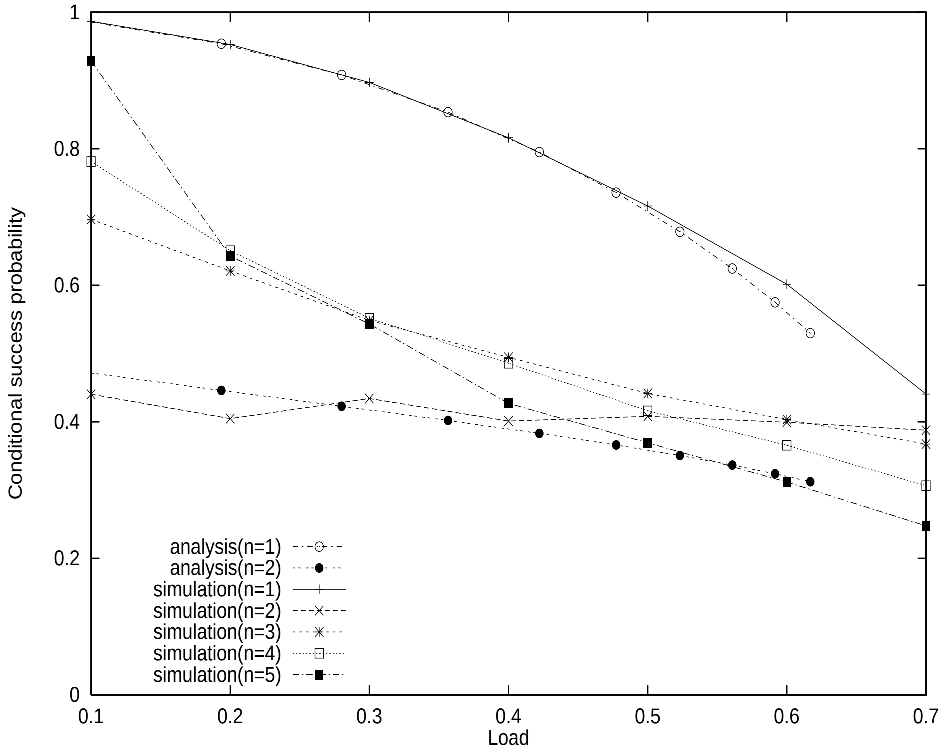


Figure 5.5: Conditional success probability – 1000 bytes

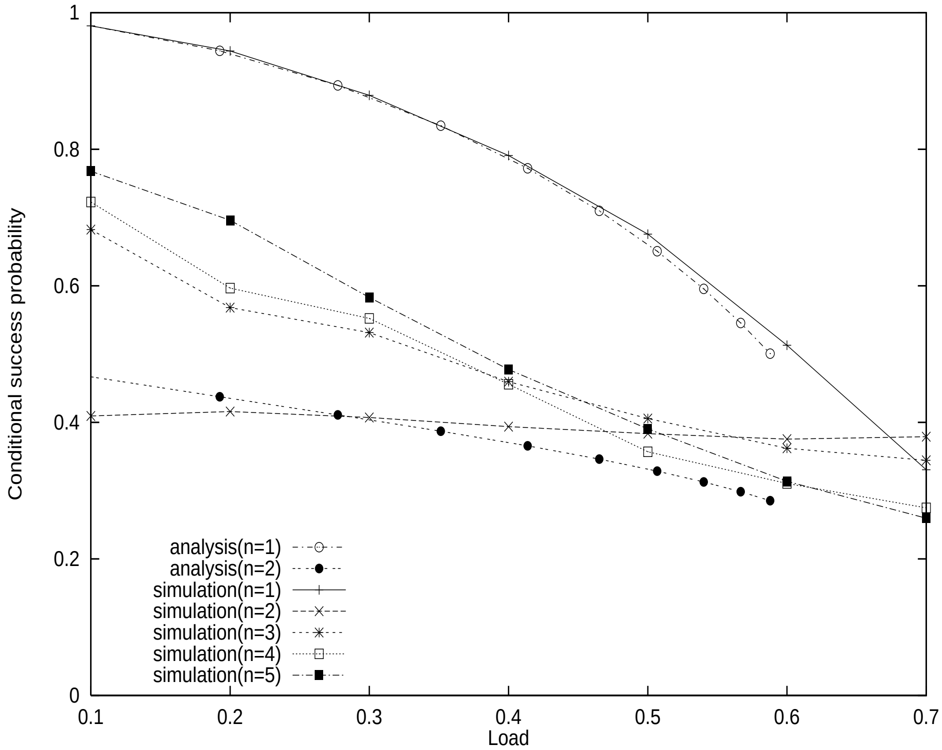


Figure 5.6: Conditional success probability – 500 bytes

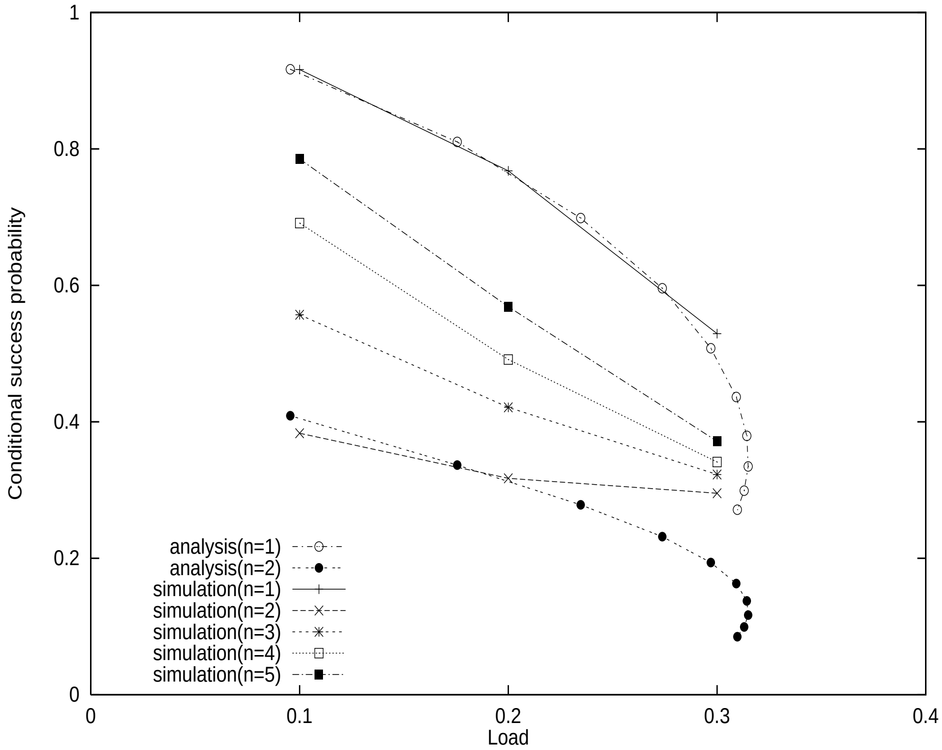


Figure 5.7: Conditional success probability – 64 bytes

packet size (bytes)	$n = 1$	$n = 2$	$n = 3$	$n = 4$	$n = 5$
64	10.24	25.18	48.74	89.55	164.8
500	80.00	131.4	192.8	274.1	395.5
1000	160.00	251.5	353.1	475.0	637.4
1500	240.0	371.5	513.2	675.3	878.0

Table 5.1: Delay bounds in μsecs

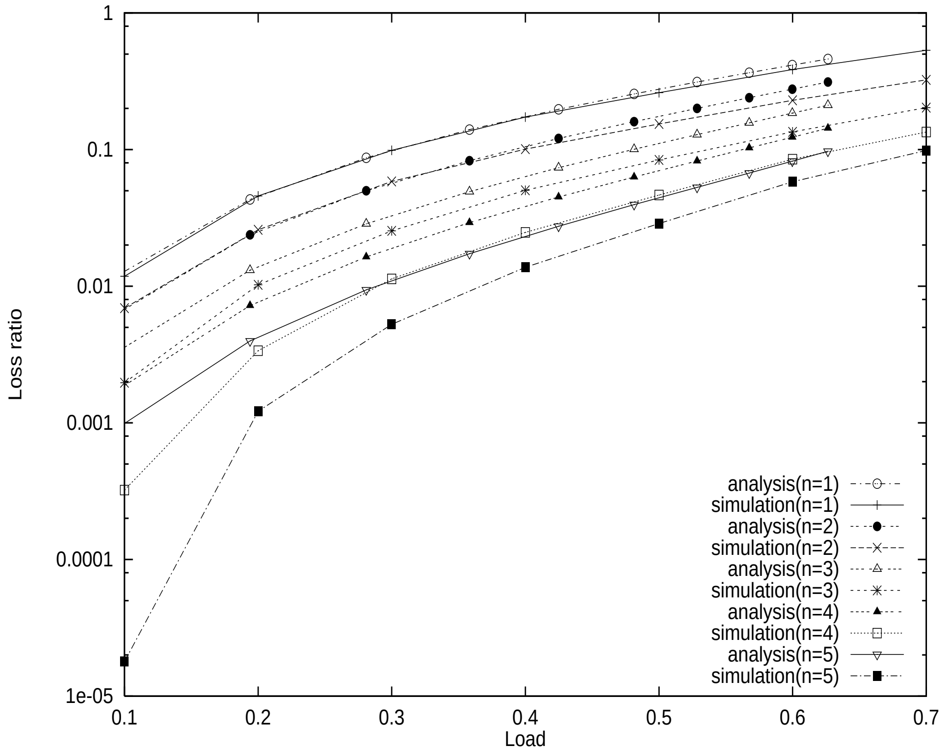


Figure 5.8: Loss ratio – 1500 bytes

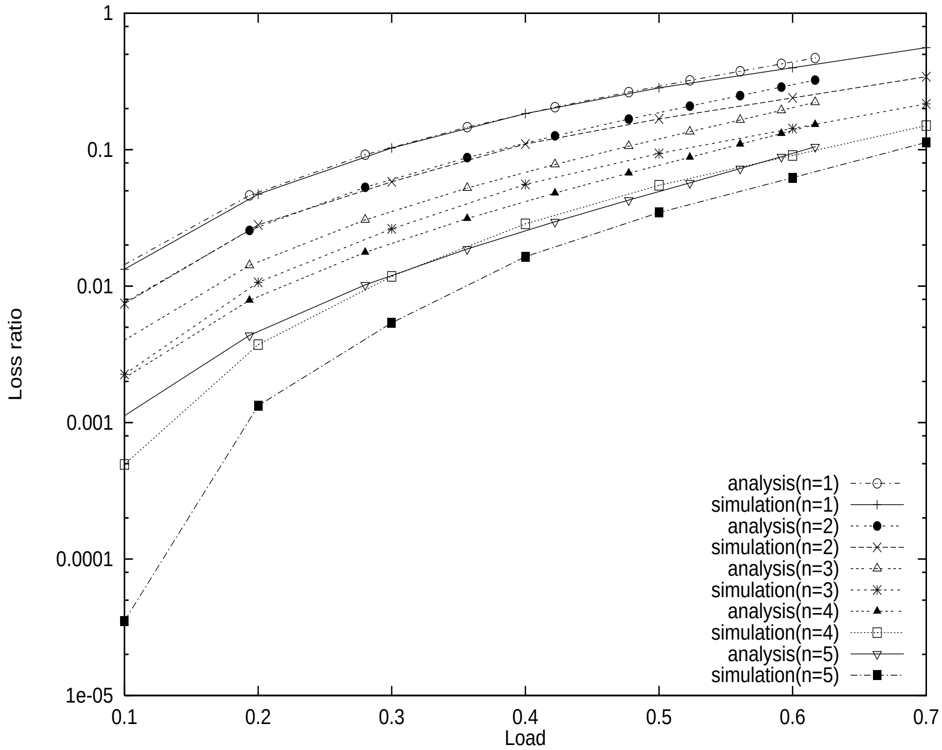


Figure 5.9: Loss ratio – 1000 bytes

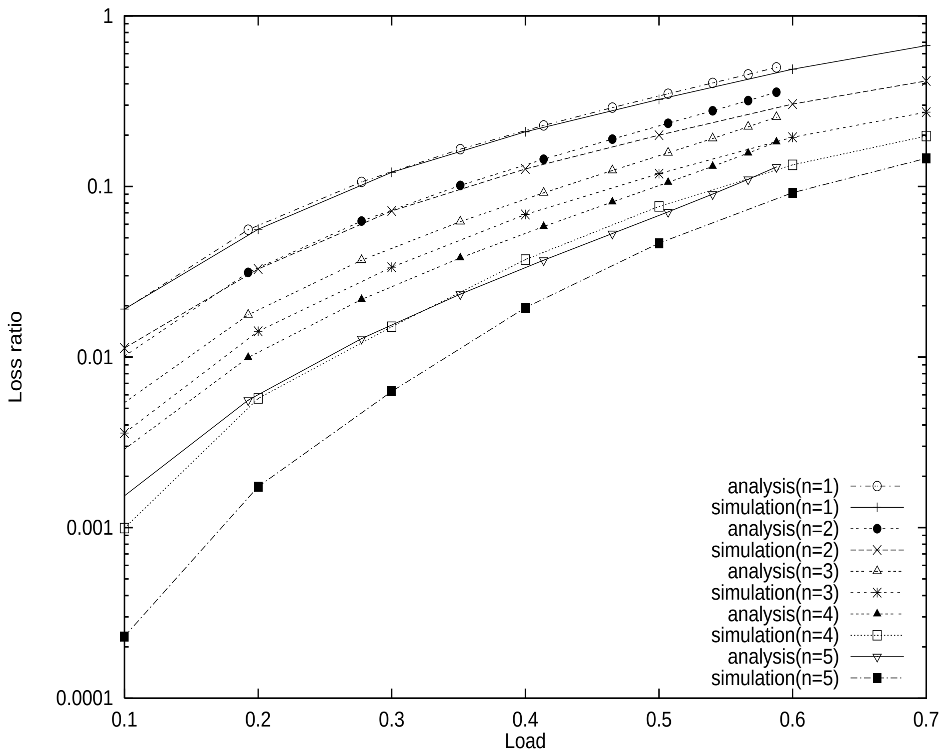


Figure 5.10: Loss ratio – 500 bytes

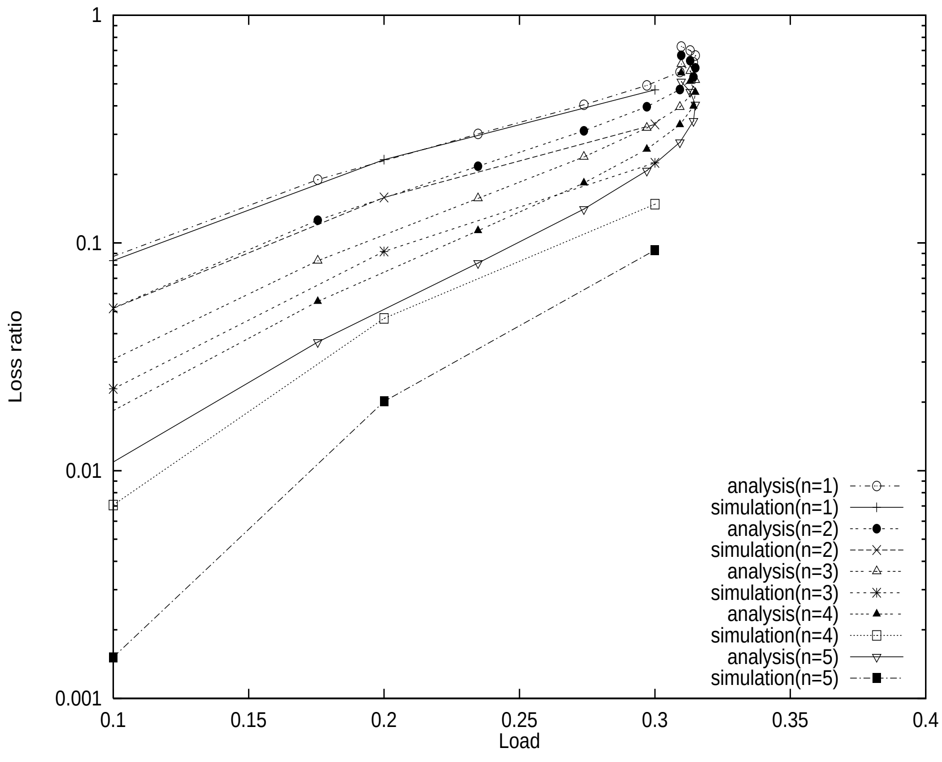


Figure 5.11: Loss ratio – 64 bytes

5.5 Conclusion

In this chapter, we presented a methodology for connection admission control (CAC) for Ethernet and Fast Ethernet. Under the assumption that the aggregate packet arrivals consisting of new arrivals and retransmissions form a Poisson process, we modeled the 1-persistent CSMA/CD with BEB as an SMP process and derived conditional success probability of each packet in its transmission, depending on the number of trials for transmission. We employed the SMP process to accommodate the bursty nature of Ethernet traffic due to BEB. Using the derived figure, we obtained the tail distribution of packet delay over Ethernet. In-depth simulation results have shown our analytic model to provide reasonably accurate estimates for the packets' deadline miss ratio when all the packets are of the same size.

CHAPTER 6

DISTRIBUTED QOS ROUTING USING BOUNDED FLOODING

6.1 Introduction

In the previous chapters, we presented network control functions needed to provide QoS guarantees in an ISPN – connection admission control and packet scheduling scheme. In this chapter, we address another important issue in providing QoS guarantees in ISPNs – QoS-routing. Without an efficient QoS-routing algorithm, a network may fail to find a route and reject the request for a new connection with QoS requirements, even if there exists a qualified route with enough resources to provide the requested QoS.

As discussed in Chapter 1, there are basically two approaches to QoS routing: source-directed and flooding-based. Although source-directed routing, specifically link-state routing, incurs smaller signaling overhead, it has the problem of signaling and routing failure due to the inaccurate information kept at each node [28, 78] and generates huge overhead when link-state information is distributed. On the other hand, flooding-based QoS routing can avoid such failures due to out-dated information, but it incurs huge signaling overhead since signaling is executed through message flooding.

In this chapter, we propose a variation of flooding-based QoS routing which incurs much lower message overhead yet yields a good connection-establishment rate, compared to the existing flooding-based algorithms. In order to reduce the flooding overhead, request messages are allowed to take only those routes of hop count smaller than a pre-specified limit. The hop count limit is chosen such that as many alternate routes as possible are searched while keeping overhead below a given level. Nodes are therefore required to keep the network-topology information at each node which is not link-state but node connectivity. Since node connectivity changes (due to loss and addition of nodes/links) much less frequently than that of dynamic link-state, distribution of topology changes incurs little overhead compared to link-state distribution. Using this 'almost static' topology information, network nodes decide whether to relay connection-request messages to their neighbors. That is, if the route via one of its neighbors has a larger hop-count than the pre-specified limit, the node does not relay the request message to the neighbor. By narrowing the flooding area using this type of "pruning," the proposed algorithm is shown to be able to reduce the message overhead dramatically. Moreover, unlike the source-directed approach, it does not require on-demand shortest-path calculation, thus lowering the operational cost. Note, however, that this reduced message overhead and operational cost may be achieved at the expense of loss in connection-success rate; fortunately, this loss is shown to be acceptably low. In general, there is a tradeoff between reduction of overhead and operational cost, and loss in connection-establishment rate, in both flooding-based and link-state schemes. That is, by either increasing the frequency of link-state distribution or increasing request messages, one can achieve a higher connection-success rate. Although it is impossible to derive an analytical relation between overhead reduction and performance loss, our scheme can control overhead by adjusting the hop count limit of candidate routes subject to the required connection-success rate.

The rest of the chapter is organized as follows. After providing some background in Section 6.2, we describe the proposed QoS routing in Section 6.3.4. Using extensive simulations, the proposed and other existing algorithms are comparatively evaluated in Section 6.4.3. The chapter concludes with Section 6.5.

6.2 Work Related to QoS Routing

Most service disciplines for timeliness-QoS guarantees assume connection-oriented services due to their conceptual ease in resource reservation and management. The goal of QoS routing is then to find a route or a virtual circuit through which real-time data of the corresponding connection will be transported. This is the fundamental difference between datagram routing and QoS routing. In datagram networks, including the current Internet, each datagram is routed in a connectionless fashion.

Since real-time communication services are provided by employing a special form of service discipline at each switch or router, the metric which QoS routing must consider for its route-selection strongly depends on the service discipline employed. The metric could be delay, loss rate, bandwidth, jitter, or a combination of thereof. In [91], an optimal routing scheme for multiple QoS parameters was considered. Although QoS requirements and network resources may be characterized by multiple parameters, under most of the Weighted-Fair-Queueing service disciplines [64], one can satisfy user-requested QoS requirements by providing guaranteed throughput or bandwidth to each connection. In some service disciplines such as delay-EDD (Earliest-Due-Date) and RCSP (Rate-Controlled Static-Priority), guaranteeable delay and reserved bandwidth are *indirectly* related. In this case, reserving bandwidth is not enough for guaranteeing a delay bound. Since our approach can be easily modified to handle these sophisticated service disciplines, we will consider bandwidth as each connection's metric for its QoS or resource requirement. Depending on the underlying admission-control policy, a connection's QoS requirement can be given as a peak, or average, or effective bandwidth. The network service provider must reserve resources commensurate with the bandwidth requirement along the path chosen by QoS routing in order to provide the QoS promised to the end user. For link-state routing, a node's link-state database stores information about the utilization of each link which is defined as the sum of reserved bandwidths for the connections running over the link.

There are several strategies on how to choose an optimal route for real-time communication. For example, minimum-hop routing, shortest-widest path [91], widest-shortest path [30], shortest-distance path [57], and minimum-load routing [78] are among them. Since non-minimal routing algorithms, e.g., shortest-widest path, often select circuitous routes that "occupy" more network resources, they possibly cause rejection of future connection requests. The proposed routing scheme favors the selection of a minimum-load route to balance network utilization, but it can be easily modified to accommodate other routing strategies.

6.3 QoS Routing with Bounded Flooding

Our scheme is designed for an arbitrary point-to-point network; it can be a distributed system or a wide-area network. All links in the network are assumed to be bidirectional.[1]

6.3.1 Overview of the Proposed Approach

We employ a bounded, not brute-force, flooding; in order to reduce the overhead of flooding request messages, we limit the number of hops each request message can take before reaching its destination. That is, when an intermediate node receives a request

[1] This assumption can be relaxed trivially.

message, it will decide whether to forward the message to one of its neighbors by checking if the minimum-hop path via that neighbor can lead the request message to the destination within the source-defined hop count limit. This approach can be interpreted as flooding with a limited search area. Hop count is selected as a basis because restricting the hop count of a path limits the degree to which the scheme can find a circuitous route around congested links, thus not blocking future connection requests, as argued in [56]. In order to determine whether a request message can reach the destination within the hop count limit via a particular outgoing link, each intermediate node uses its almost static network-topology information. Instead of calculating the minimum-hop path via an outgoing link to the destination every time a request message arrives, we use a (distance) table storing the hop count of the minimum-hop path through each outgoing link to every other node.

6.3.2 Composing Distance Tables

Before starting the QoS-routing service and/or when the network topology changes due to failures and additions of nodes and links, every node must compose or update its distance tables, using new topology information. Note that this topology information is almost static, because it does not contain any link-state information but describes only the physical connectivity between nodes. By calculating inter-node distances off-line, our scheme dramatically reduces run-time operational cost.

Distance from node i to node j via node i's outgoing link ℓ is defined as the hop count of the minimum-hop route among all the routes from node i to j via link ℓ. In order to prevent the minimum-hop route from traversing ℓ in the reverse direction, we exclude this "reverse-directional link"[2] from minimum-hop route calculation. Its purpose is to exclude the case when request messages bounce back to nodes from which they came. This prevents request messages from oscillating between nodes. The minimum-hop route can be easily calculated using Dijkstra's shortest-path algorithm or Bellman-Ford shortest-path algorithm. All links are given the same weight except the reverse-directional link of ℓ whose weight is set to ∞.

Although we employed hop count in composing distance tables, other metrics can be used as distance, depending on the environment. For example, if the network is not homogeneous, i.e., each link has different capacity, the "cost" of a path can be employed as distance.

[2] Actually, there is only one bidirectional link, but it is viewed as consisting of two unidirectional links.

6.3.3 The Proposed QoS-Routing Algorithm

Upon generation of a connection request, the source node must check the "search-scope" of the connection. The search-scope is the maximum number of hops the request message can take to reach its destination. In order to increase the chance of granting the requested connection, multiple alternate, not necessarily disjoint, paths must be given an opportunity to run the connection over them. The search-scope must therefore be determined by making a tradeoff between the message overhead and the request-acceptance probability. In this dissertation, we allow at least two alternate paths to be tried for each connection request. Thus, the search-scope is given by the second smallest distance between each source-destination pair. If more than two minimum-hop paths exist for a request, we consider the hop count of the minimum-hop paths as the second smallest distance. For example, in Figure 6.1, the distance $d(A, B; a)$ between source A and destination B via a is 2, and the distance via b, $d(A, B; b)$, is also 2. So, the second smallest distance is 2 and thus, the search-scope of a connection between A and B is set to 2. If A is the source and C the destination, then $d(A, C; a) = 2, d(A, C; b) = 4$, and $d(A, C; d) = 4$. The second smallest distance is 4, and thus, the search-scope is 4. In general, the search-scope can be given as the hop count of the n^{th} minimum-hop route for a given pair of source and destination.

Since we are considering only the connectivity between nodes, the search scope for every pair of source and destination can be calculated *a priori* by source nodes before servicing any connection request. The pre-calculated search scopes are stored in a table at each node. Whenever the topology changes, albeit very infrequently, these search scopes must be recalculated. On the other hand, the search scope can be determined on-line for a given connection request. In this case, the source node may use the current load condition which is estimated by using variables like local link bandwidth usage and the measured request-acceptance probability. Here we use the first approach to minimize the operational cost at the expense of a slight performance loss.

Upon receiving a connection request from an application program, the source node generates a request message, m. A connection request message contains the following fields.

- Connection identifier $Req.ID$ which uniquely identifies the corresponding real-time connection. For the uniqueness of each connection ID, an identifier is composed of two parts: the node ID (or address) and connection number (unique within a source). This composition of connection IDs ensures their uniqueness throughout the network.

- Source identifier $Req.src$ of the requested connection.

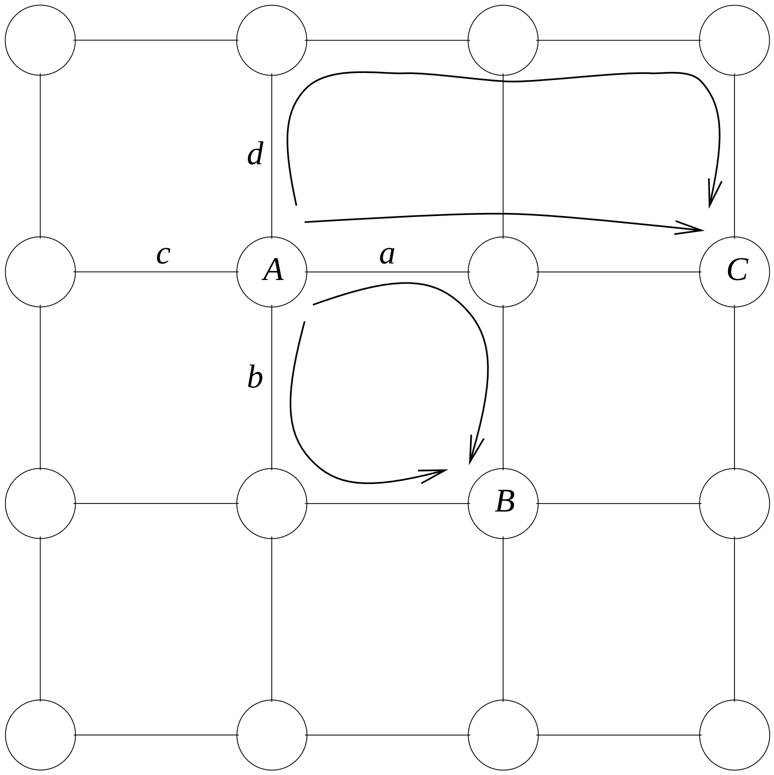


Figure 6.1: An example of determining the search scope

- Destination identifier $Req.dest$ of the connection.

- Timeout. This field is used to specify the request's time to live (TTL). After its TTL, a request message is no longer valid and thus discarded. How to determine this value will be discussed when the connection-confirmation process is described later in this section.

- Search-scope Sc of the requested connection.

- Hop count H of the path taken by the request message to the current node.

- The connection's bandwidth requirement $bw(\cdot)$. If no outgoing link has available bandwidth larger than this, the request will be discarded.

- List of intermediate node IDs that the message has traversed thus far. Every time the request message is relayed to the next node, the new node ID is appended to this field. This information is needed for the destination node to confirm the establishment of the requested connection.

- Accumulated utilization u of the path. This is the sum of all the intermediate links' loads, where load is defined as the bandwidth reserved for real-time connections. This field is needed for intermediate nodes and the destination to be able to choose the least-loaded path. As discussed earlier, our scheme favors selection of the minimum-load route based on this field. For other selection strategies, this field must be filled in with an appropriate parameter. For instance, the field may store the minimum of loads of the intermediate links the request message has traversed if the strategy is the shortest-widest path.

Since the information of existing connections is necessary for a new connection's admission test as well as for the run-time scheduling of messages belonging to those connections already established, each node has to maintain two sets of tables for existing connections and pending connections. The first set is the tables of established connections (TECs), one for each of its outgoing links. Each entry of a TEC represents a real-time connection which goes through the corresponding link and consists of the following two data fields.

- Connection identifier: this is the same as the one in the connection request message.
- The connection's bandwidth requirement.

Using a connection's bandwidth requirement, the service policy determines the priority of data messages belonging to this connection.

The second set of tables each node has to maintain are tables for temporarily-pending connection requests, also one for each of its outgoing links. These tables will be referred to as "tables of pending requests" (TPRs). Each entry of a TPR represents a connection request (or a pending connection) and contains the following fields.

- Connection identifier: same as the one in the connection request message. When a connection request is conditionally-accepted (that is, the out-going link is able to accommodate the requested connection), it is copied from the connection ID field of the request message.
- Timeout: must be larger than that of a request message, in order to prevent deletion of a connection request from TPR before the confirmation message arrives. More on this will be discussed in the confirmation process. Upon expiration of the timer, the connection request is deleted from TPR.
- The connection's bandwidth requirement.
- Accumulated utilization u^* of the path traversed by the most-recently accepted request message for establishing the same connection.

When a connection request is conditionally-accepted to run the connection over a link, fields of the corresponding entry of the link's TPR are copied from the corresponding fields of the message. In addition to entries for connection requests, TPR must have a field to keep track of the remaining bandwidth of the link. The remaining bandwidth is calculated by subtracting the sum of bandwidth requirements of both the established real-time connections and conditionally-accepted real-time connections from the link capacity, $cap(\cdot)$. This is used for a new connection's admission test.

Apart from TECs and TPRs, a node has to maintain the table of conditionally-accepted connections (TAC) if it has received a request message whose *Req.dest* is the node itself. The function of a TAC is to allow the destination to choose the best among the routes which request messages traversed. Each entry of this table consists of the following fields.

- Connection identifier.
- Accumulated utilization u^* of the most-recently saved path.
- List of IDs of intermediate nodes the message has traversed.
- Timeout: same as that of the request message. This field tells when to initiate the connection-confirmation process.

Upon receiving a connection request from an application program, the source node sends a request message, m, through each of its outgoing links only if it satisfies the following two conditions:

distance test (source):

$$distance(Req.src, Req.dest, \ell) \leq search\text{-}scope(m), \qquad \text{and} \tag{6.1}$$

bandwidth test:

$$util(\ell) + bw(m) \leq cap(\ell), \tag{6.2}$$

where $util(\ell)$ is the utilization of link ℓ by real-time communication traffic (i.e., the bandwidth reserved for real-time connections), and $bw(m)$ is the bandwidth requirement of connection m.

The flowchart in Figure 6.2 shows the actions to be taken by an intermediate node or by the destination upon receipt of a connection request message. First, the node checks whether *Req.dest* of the message matches its own ID, $Node.ID$. If they match, then it checks whether its TAC already has the connection ID identical to that of the request message. If yes, the node has already received at least one copy of the connection request.

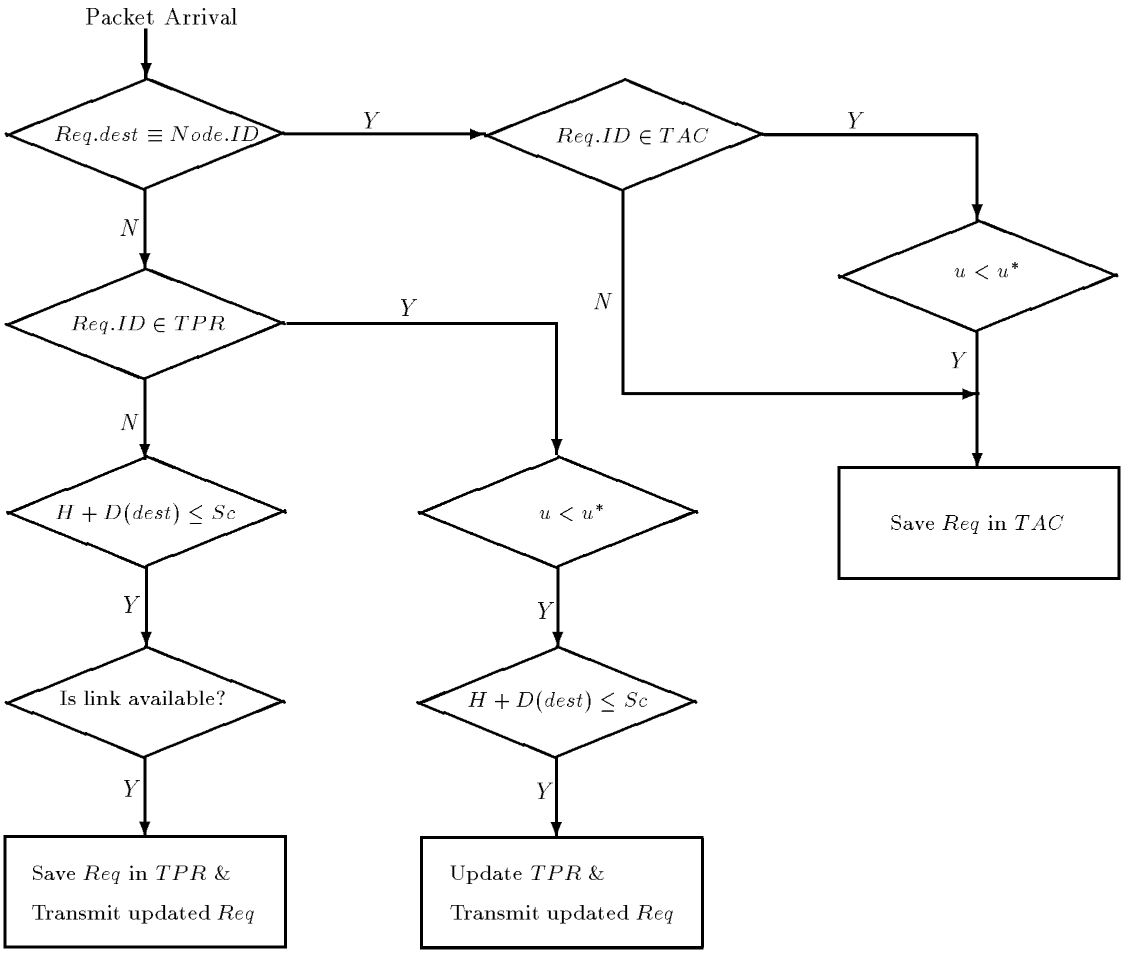


Figure 6.2: Node actions upon arrival of a request message.

In this case, the node checks if the newly-arrived request message contains a better route than the one in TAC (**better route test**). By "a better route," we mean that the request message contains a smaller accumulated utilization than that of the old one in TAC. As mentioned earlier, this is to favor a less-loaded path. If the request message carries a better route, the node updates the corresponding entry of its TAC. Otherwise, the request message is discarded. If the request is new, the node stores the request in its TAC.

If the destination contained in the request message does not match, the node checks, for each outgoing link, if the ID is in the link's TPR, that is, if other request messages carrying the same connection request have already passed through the link. If yes, as done with TAC, the node checks if the new request message contains a smaller accumulated utilization than that of TPR. If yes, the node executes the following distance test.

distance test (intermediate nodes):

$$H(m) + distance(node, Req.dest, l) \leq search\text{-}scope(m), \tag{6.3}$$

where $H(m)$ is the hop count of the path message m has traversed so far before reaching this node. If it passes this test, then the node updates and forwards the request message to a neighbor node via the outgoing link, and updates TPR by replacing u^* of the connection by the accumulated utilization of the newly-received request message. The request message is updated by adding the utilization of the outgoing link to the accumulated utilization of the message, incrementing H by 1, and appending the node ID to the list of intermediate nodes of the path the message has traversed thus far. If this test fails, the request message is discarded.

In order to reduce the overhead of request messages, we do not relay them to the connection's 'upstream' nodes. A connection's upstream nodes are the ones from which at least one copy of the same request has come. The necessary information for this action can be maintained and easily checked by recording the request message' ID in the TPR of the reverse-directional link of the link through which the request message has come. By assigning 0 to u^*, we can use a better-route test for this.

If the ID of the request message matches none of the IDs of connection requests stored in its TAC and TPRs, the request is new, and the distance test in Eq. (6.3) and the bandwidth test in Eq. (6.2) are conducted. If it passes both tests, the node updates and relays the request message to a neighbor through the outgoing link. In addition, it saves the connection request in the TPR of the outgoing link.

We have already discussed the destination's action upon arrival of a request message. The destination does not respond immediately to the arrival of a request message. It does not initiate the connection-confirmation process until it reaches the timeout of the request which is kept in the TAC of the destination. Before describing the confirmation process, we need to discuss timeout fields of request messages and the tables. First, in order to make our scheme work as intended, we must ensure that as many request messages as possible arrive at the destination before the timer expires and that the timer is set to as small a value as possible to reduce the chance of rejecting new requests due to unnecessary temporary reservation as seen in the flooding-based approach. For this purpose, connection request messages are transmitted through a dedicated signaling channel which can be realized by opening a permanent virtual connection. By using a dedicated signaling channel, one can reduce a request message's link delay. From the arrival pattern of real-time connection requests, we can estimate the bound of request message delays over a link. We assume that the processing delay at each node is included in the link delay. Then, the source can set the timeout field of a request message using the search scope of the connection. Since the search scope indicates the maximum hop count of candidate routes, the timeout must be set larger than, or equal to, the link delay bound multiplied by the search scope. Choosing a large timeout value may not cause inefficient use of network resources since relaying request messages to neighbors depends on search scopes and distances as well as the timeout value of a request message. The chosen value also works as the connection-confirmation initiation time since we set the timeout of request messages equal to that of the connection in TAC. Thus, the timeout of request messages and TAC is set to the link delay bound multiplied by the search scope plus the processing time at the destination in order to minimize the time to set up a connection.

In addition to connection request messages, connection-confirmation messages are also transmitted through the signaling channel. Thus, we can use the same link delay bound for confirmation messages. The link delay bound enables us to calculate the maximum traversal delay of a confirmation message, which is used to determine the timeout field of TPRs. Since conditional reservation of resources at the intermediate nodes must be kept until the confirmation message reaches the source, the timeout in TPRs must be larger than the maximum traversal time of a confirmation message plus the timeout of a request message. In order to minimize unnecessary temporary reservation[3] of network resources, we select the maximum traversal time of a confirmation message plus the time-to-live of a request message as the timeout in TPRs.

Now, let's consider the connection-confirmation process. Since, as discussed earlier, the most recently-saved request message in its TAC contains the minimum-load route among those which safely reached the destination, the destination simply chooses the one in its TAC as the best route found. Although our approach is not guaranteed to find the global minimum-load route because of its bounded search, it is likely to choose the best one due to its selection process based on the better-route test. Upon expiration of the timer of the connection in TAC, the destination sends a confirmation message in the reverse direction of the path recorded in the connection's entry of its TAC and deletes the entry from its TAC. Upon the arrival of the confirmation message, the intermediate nodes along the route update their TECs by recording the requested connection's profile, and also delete the connection's entry from TPRs.

When a real-time communication is terminated by an application program, a disconnect

[3]Temporary reservation that lives longer than necessary.

process is initiated by either the source or the destination. In this process, a disconnect message is passed along the path of the real-time connection. Upon arrival of the disconnect message, the intermediate nodes delete the corresponding entry of their TECs and update the remaining link bandwidth in their TPRs.

6.3.4 Memory Requirement

Although our scheme is very efficient in reducing operational cost and message overhead, the efficiency is achieved at the expense of increased memory requirement. Compared to the existing flooding approach, our scheme requires additional memory for distance tables and search-scope tables. Other tables such as TPRs, TECs and TACs are also required by the existing flooding approach. When the number of nodes is N and the maximum number of outgoing links of a node is L, the number of entries needed for distance tables in the entire network is $O(N^2L)$, and, for search-scope tables, $O(N^2)$.

Link-state routing doesn't need TPRs and TACs, but requires a table similar to TECs for maintenance of link-state for each link and the run-time scheduling of real-time messages. Moreover, each node must have a table to store link-states of the entire network which must be updated more frequently than TPRs and TACs. Memory requirement for this table is the same as the one needed for distance tables in our scheme, i.e., $O(N^2L)$. If the maximum number of connection requests which are pending simultaneously in the network is N, the memory requirements for TPRs and TACs in the proposed scheme are $O(N^2L)$ and (N^2), respectively, as the number of TPRs and TACs are $O(NL)$ and $O(N)$, respectively. Since, in terms of memory requirement, TPRs are a dominant factor among TPRs, TACs and search-scope tables, additional memory of our scheme compared to that of link-state routing is $O(N^2L)$. If the maximum number of connection requests which are pending simultaneously in the network is much smaller than N, a dominant factor comes from search-scope tables whose memory requirement is $O(N^2)$.

6.4 Simulation and Discussion

We have conducted an in-depth simulation study to comparatively evaluate the proposed and other QoS routing schemes in terms of their performance and overhead. In this study, we measured the probability of establishing connections successfully under various load conditions and network configurations. We also evaluated the overhead incurred for establishing a connection with all of the routing schemes considered.

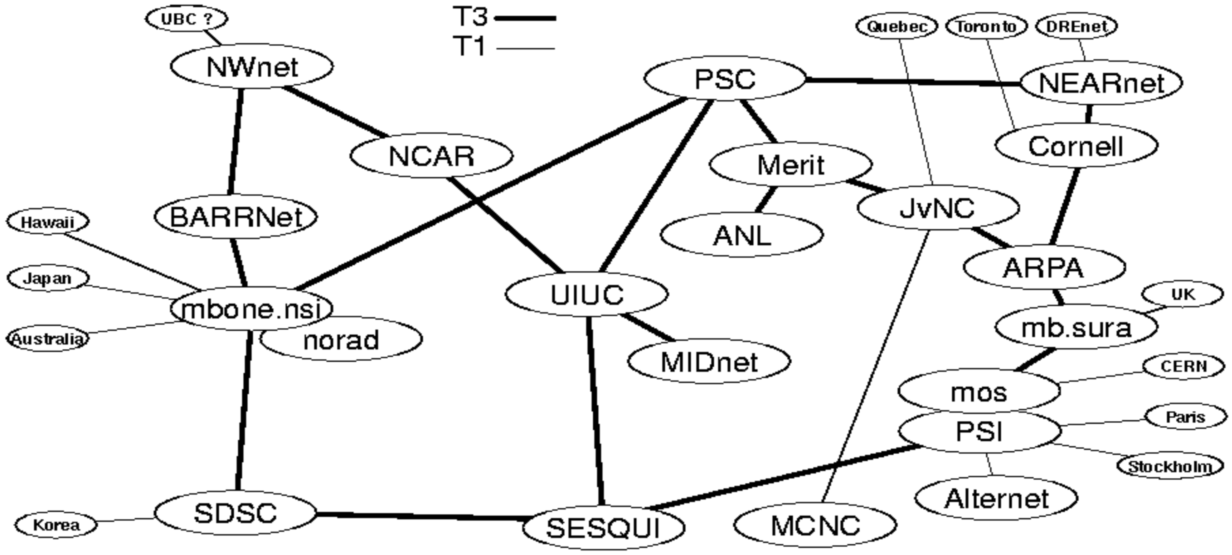
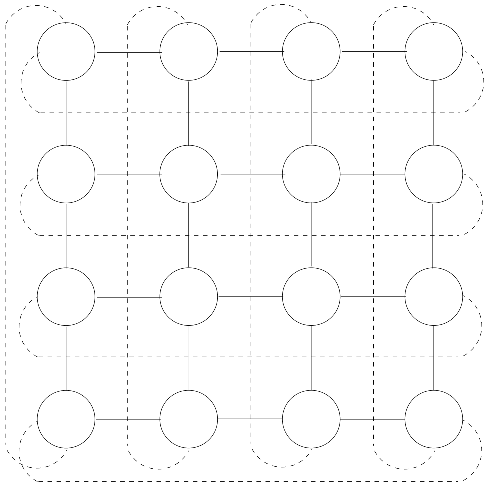


Figure 6.3: MBONE topology in North America.

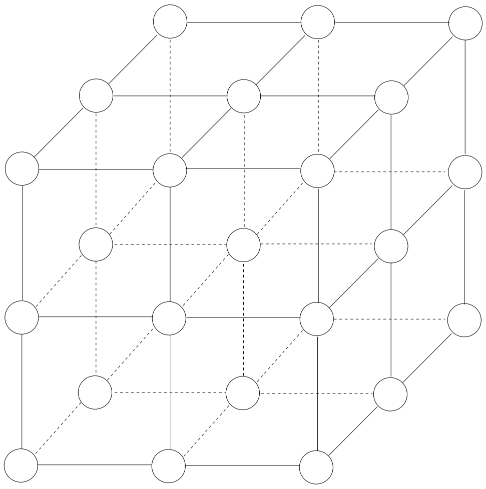
6.4.1 Simulation Model

In order to investigate the performance of the proposed scheme under different network configurations with a wide range of node connectivity, we selected the following networks: 5-ary 2-cube, 5-ary 3-cube, 10-ary 2-cube, 5-ary 3-cube, and the MBONE topology in the North America region in Figure 6.3. In the MBONE topology, all T3 links and nodes which are connected via T3 links are included in the simulation. Each node acts as a router or switch, and links are assumed to be bidirectional, with 'unit' capacity in each direction. A k-ary n-cube is an n-dimensional cube with k nodes in each dimension. Each node is connected to two other nodes in both the directions of each dimension in a 'wrapped-around' fashion. The left side of Figure 6.4 shows a 4-ary 2-cube with the wrap links indicated by dotted lines and the right side shows a 3-ary 3-cube with the wrap links omitted for simplicity. Each edge represents two unidirectional links. The MBONE topology in Figure 6.3 was selected to investigate the performance of each routing scheme under a network configuration with (currently) realistic connectivity. Its connectivity is very poor compared to the other topologies considered. (Note that future backbone networks are expected to have higher connectivity, so k-ary n-cube topologies are reasonable to consider.) Table 6.1 shows the characteristics of the networks chosen for evaluation.

The load is defined as the bandwidth reserved for real-time connections. For each network configuration, we tried to establish real-time connections under a certain load using



(a) 4-ary 2-cube



(b) 3-ary 3-cube

Figure 6.4: Example k-ary n-cube topologies.

Topology	Nodes	Links
10-ary 2-cube	100	400
5-ary 3-cube	125	750
5-ary 2-cube	25	100
MBONE	18	42

Table 6.1: Characteristics of topologies

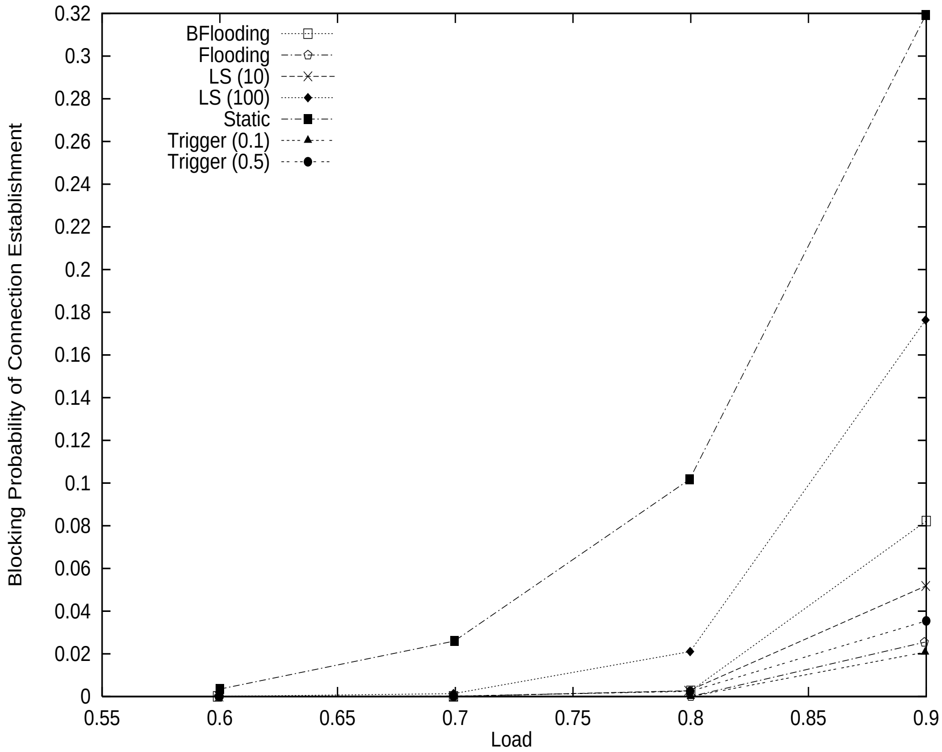
the proposed scheme (labeled as BFlooding in the figures), static routing (labeled as Static), QoS routing by flooding (labeled as Flooding) [33, 80], link-state routing with periodic link-state distribution whose periods are 10 and 100 times the average connection inter-arrival time (labeled as LS(10) and LS(100), respectively), and lastly, link-state routing with triggered link-state distribution with trigger levels of 0.1 and 0.5 (labeled as Trigger(0.1) and Trigger(0.5), respectively). In Trigger(0.1) (Trigger(0.5)), link-state information is distributed when the available link bandwidth changes by more than 10% (50%) from the recently-distributed value. Static routing determines a QoS route solely based on the static topology of the network. Although it incurs a very small overhead and operational cost, its performance will later be shown unacceptable in most cases.

The simulation study uses homogeneous traffic patterns, with uniform random selection of source and destination nodes. For simplicity, it also assumes exponentially-distributed connection-request inter-arrival times. Instead of using more realistic traffic models, we opted for simple traffic patterns, because our goal is to comparatively evaluate the proposed scheme and others, as opposed to providing accurate absolute performance figures. Connection bandwidths are uniformly-distributed within a pre-specified interval. For instance, if the bandwidth range is set to 20%, connection bandwidth, b, is randomly chosen from an interval $(0.0, 0.2]$, resulting in $b \sim U(0.0, 0.2]$. In order to keep the load constant, an appropriate number of randomly-selected old connections are disconnected when a new connection is established. At the beginning, no connections are deleted until the load reaches a steady-state level. The simulations were run until connection blocking rates' 99% confidence intervals were within 1%.

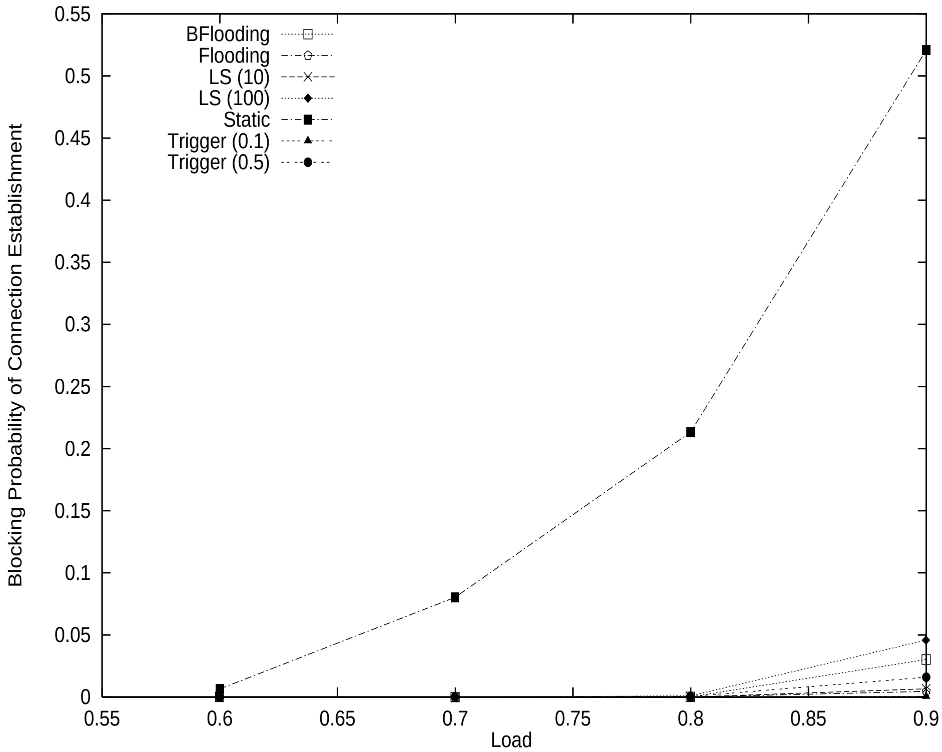
6.4.2 Simulation Results

Depending on the network configuration and bandwidth usage by the requested connections, the seven schemes considered exhibit a wide range of connection-success/blocking probability.

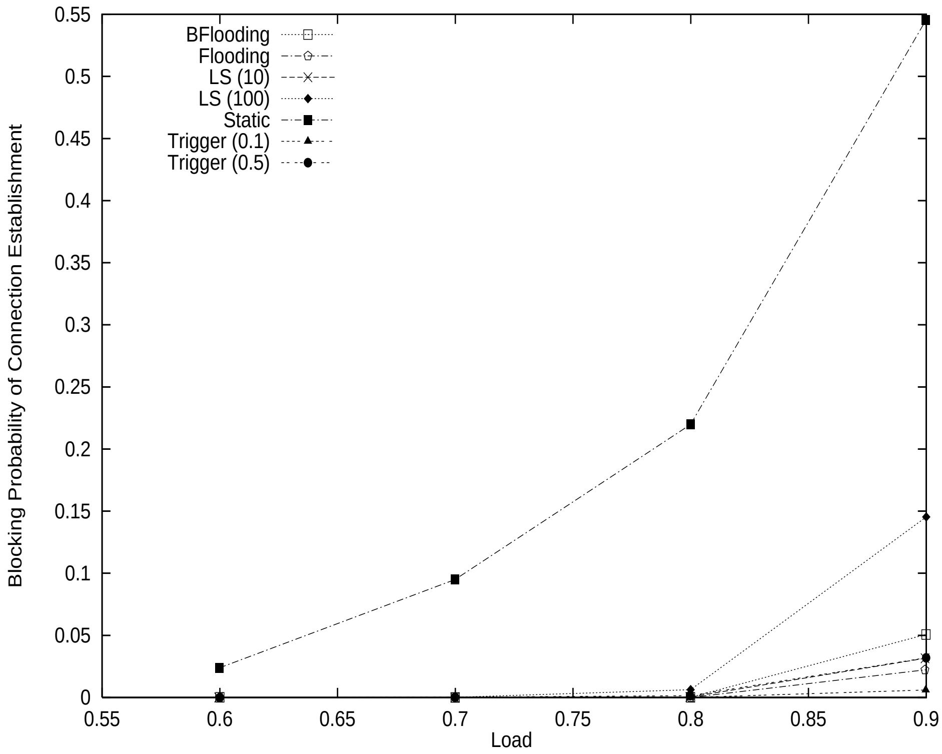
Figure 6.5 shows the connection-blocking probability of each scheme when the bandwidth range is 4% of the link capacity. Such a small bandwidth range allows us to look at a realistic scenario in realizing real-time communication service in general ISPNs. The bandwidth requirement of a real-time connection is usually quite small compared to the link capacity. In the case of k-ary n-cubes, the blocking probability increases in the order of Trigger(0.1), Flooding, Trigger(0.5), LS(10), BFlooding, LS(100), and Static. The blocking probabilities of Flooding and Trigger(0.1) are almost the same except when load



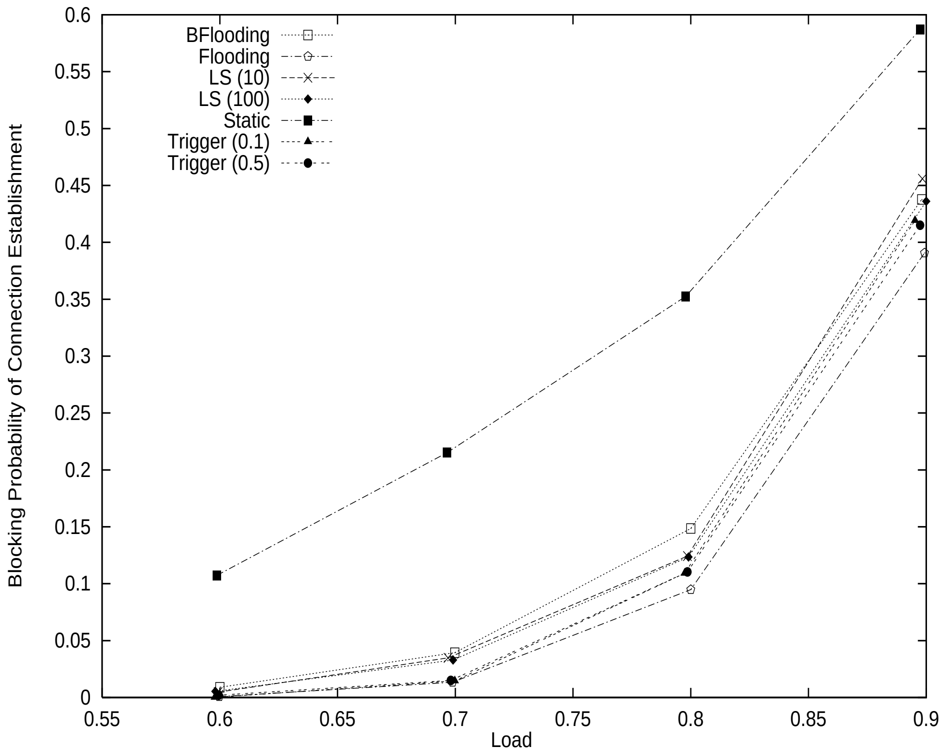
(a) 5-ary 2-cube



(b) 5-ary 3-cube



(c) 10-ary 2-cube



(d) MBONE

Figure 6.5: Connection-blocking probability, $b \sim (0, 4\%]$

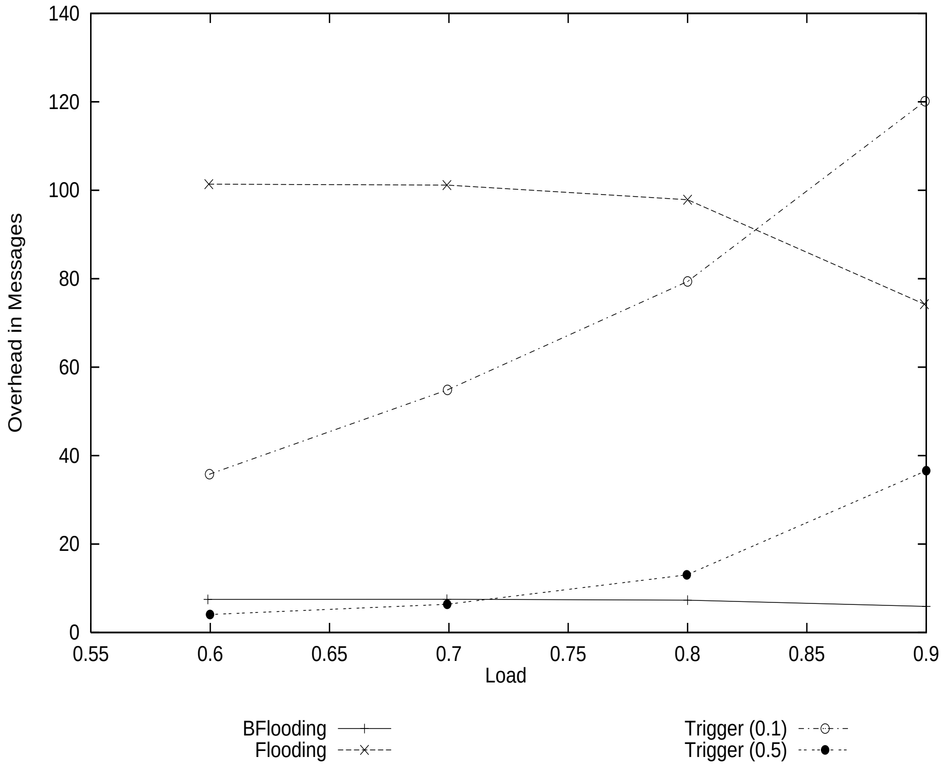
Topology	BFLOODING	FLOODING	LS (10)	LS (100)
10-ary 2-cube	25–33	376– 479	4000	400
5-ary 3-cube	19 – 23	866–1062	9375	936
5-ary 2-cube	6–8	74 – 101	250	25
MBONE	5 – 12	10 – 24	76	8

Table 6.2: Overheads for different routing schemes.

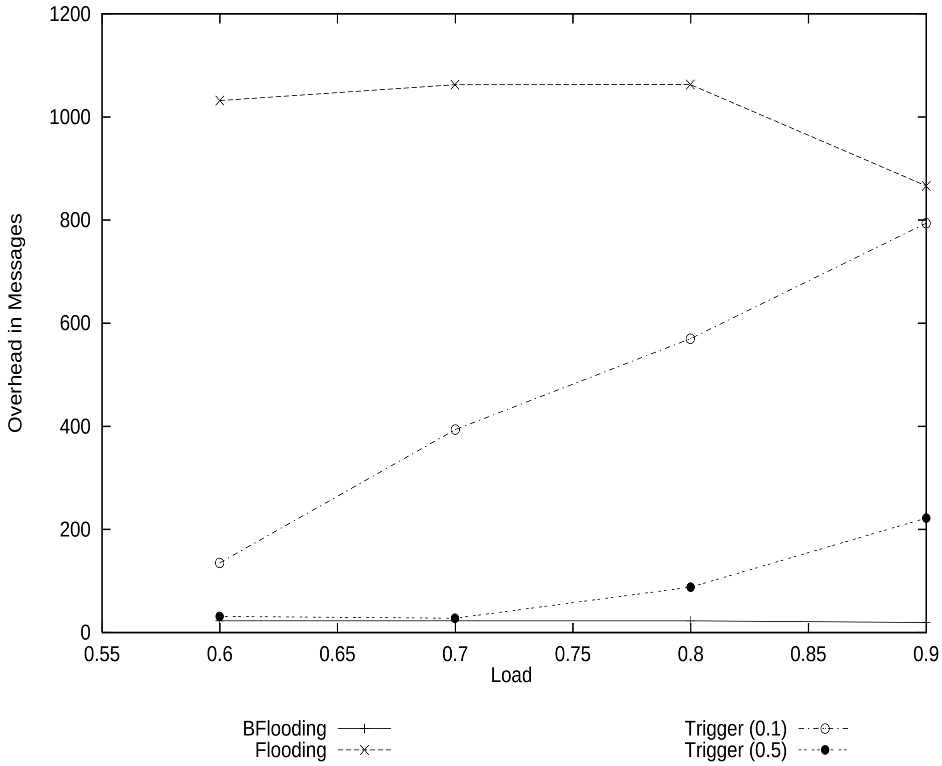
= 0.9. Flooding theoretically provides the best performance in terms of connection-success rate since it searches all the possible routes, but, because of the temporarily-reserved link bandwidth due to flooding, its blocking is higher than that of Trigger(0.1) under such a heavily-loaded condition. The performance of the proposed scheme lies between LS(10) and LS(100), but its blocking probability is very small, showing almost no difference from those of other schemes except when load = 0.9. Even when load = 0.9, its blocking probability is under 5%.

For the MBONE topology that has poorer connectivity, all but Static show similar performance, because all the schemes have a very few alternate paths in this sparsely-connected network, and thus, almost all candidate routes are always considered in determining a qualified route. While our scheme shows a slightly higher blocking probability, its performance is much closer to those of Flooding or link-state routing than Static.

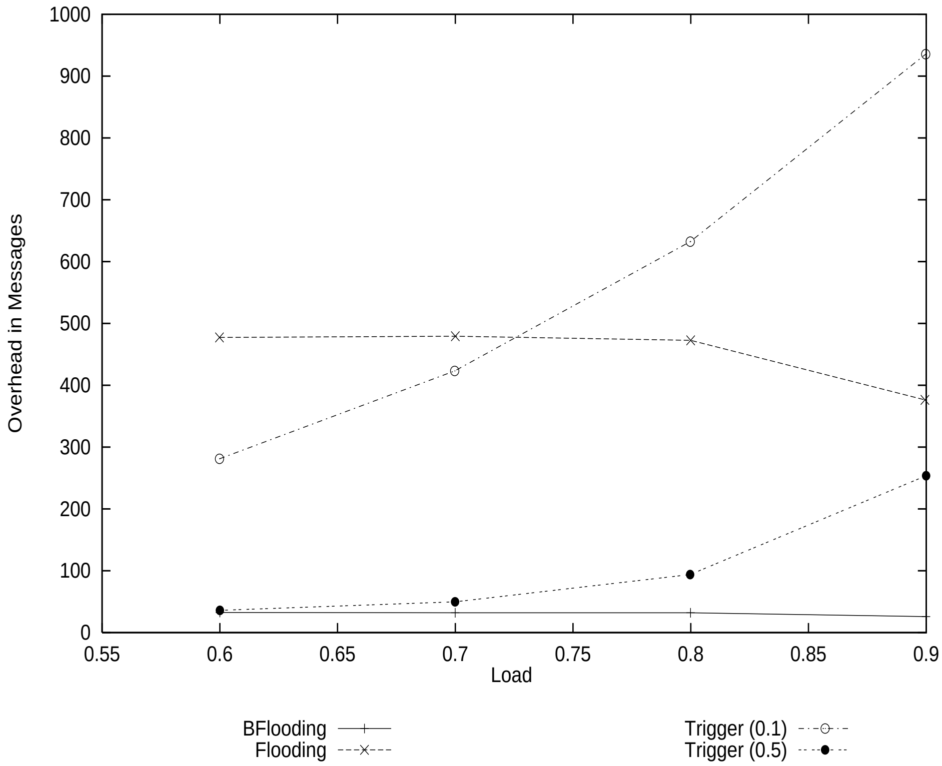
The slightly-deteriorated performance of our scheme can be offset by its small overhead and low operational cost because it doesn't require path calculation. For the bandwidth range of 4%, the overheads are given in Table 6.2 and Figure 6.6. We consider only the number of messages as overhead, regardless whether messages are generated for establishing a connection as in BFlooding and Flooding, or distributing link-state information as in LSs and Triggers. This comparison may not be fair since link-state messages and connection-request messages contain uncorrelated information. However, they both are transmitted over links and consume bandwidth and processing resources. So, we consider the number of messages generated during an average connection inter-arrival time as the overhead, regardless whether messages are for either distributing link-state or requesting a connection setup. (In this comparison, we assume that the network topology is static. Thus, the distribution of topology change information is not considered for overhead calculation. Dynamic changes of network topology will be discussed in the next subsection). Let's consider Flooding which shows the best performance in most load ranges. In Figure 6.6, the number of messages



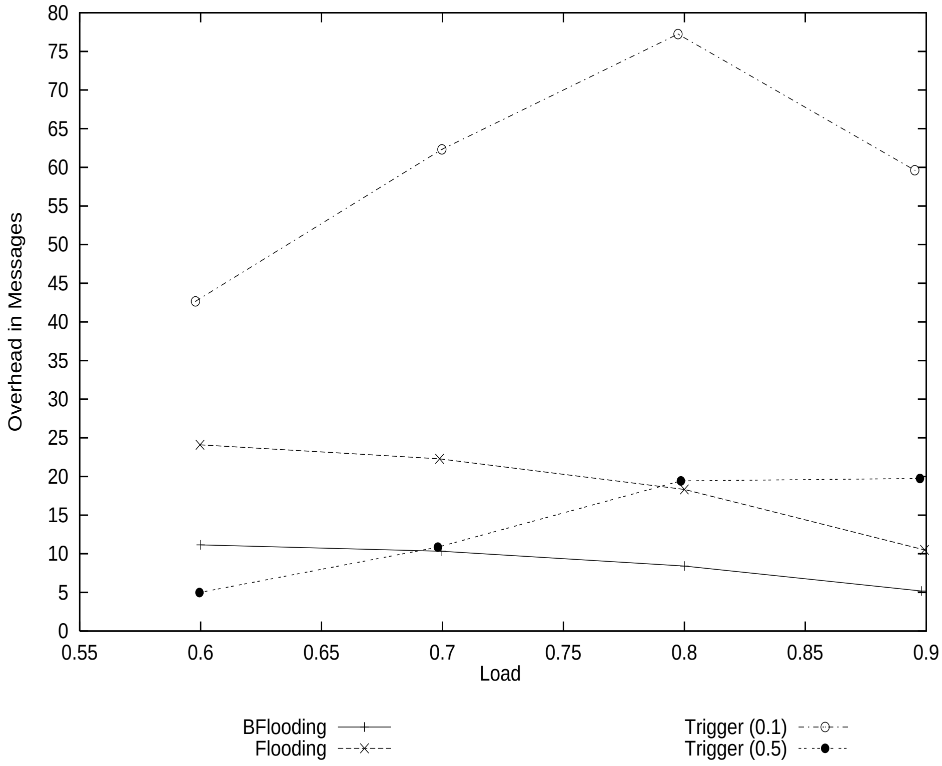
(a) 5-ary 2-cube



(b) 5-ary 3-cube



(c) 10-ary 2-cube



(d) MBONE

Figure 6.6: Overheads of different routing schemes, $b \sim (0, 4\%]$

generated per connection request is around 100, 1000, 500, and 20 in 5-ary 2-cube, 5-ary 3-cube, 10-ary 2-cube, and MBONE, respectively. As load increases, the number of request messages decreases due to the early pruning at heavily-loaded links. By contrast, BFlooding incurs a very small number of request messages, as shown in Table 6.2 and Figure 6.6. This is the result of limiting the search area. Now, let's consider Trigger(0.1) and Trigger(0.5) in Figure 6.6. The small trigger level of Trigger(0.1) generates a large number of link-state broadcasts. In densely-connected networks like the 10-ary 2-cube, the number of link-state messages ranges between 250 and 900. This large overhead makes Trigger(0.1) impractical for QoS routing in spite of its good performance. The overhead increases with load in Trigger(0.1), indicating frequent triggers under heavily-loaded conditions as expected. However, this does not apply to the MBONE topology in which the load condition changes less frequently even under heavily-loaded conditions, as the connection-success probability is much smaller due to scarce alternate paths, and thus, less frequent changes in link-state.

Compared to Trigger(0.1), Trigger(0.5) shows much more reasonable overhead. Especially, in the low-to-medium range of load, its overhead is comparable to that of our scheme. Considering its performance and overhead, Trigger(0.5) appears quite a promising choice for QoS routing. However, as with Trigger(0.1), the overhead of Trigger(0.5) increases with load dramatically. In contrast, BFlooding's overhead remains small under all load conditions. For example, in the MBONE topology, BFlooding generates only 5–12 request messages for setting up a real-time connection.

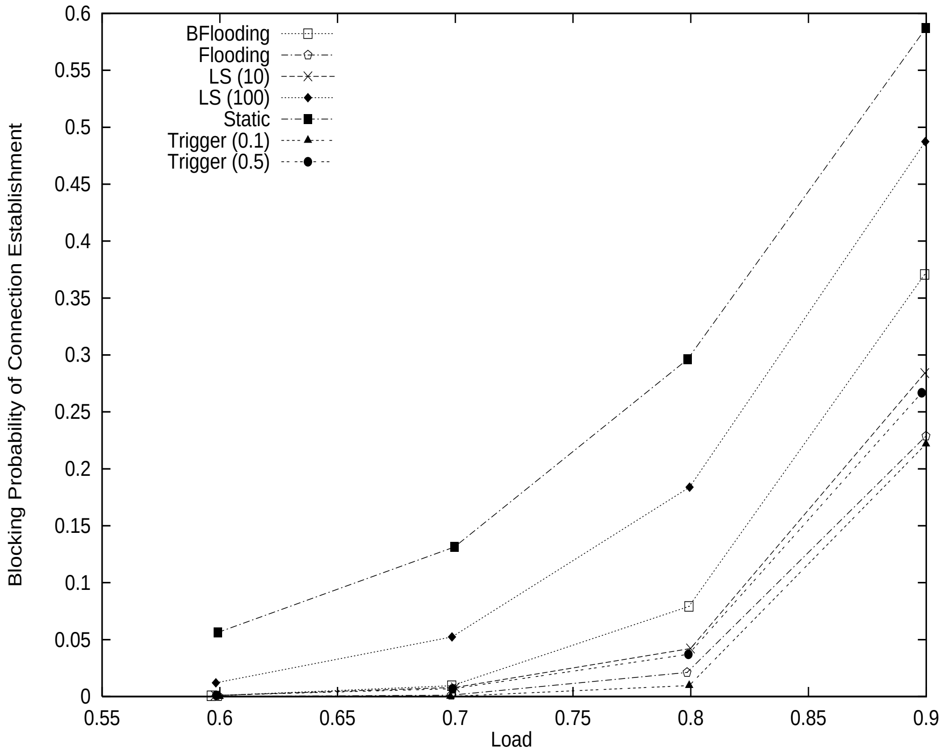
In LS(10) and LS(100), the number of link-state messages is constant irrespective of load ranges or conditions; this is why their overheads in Table 6.2 are shown separately from those of Triggers. During every updating period, each link generates its new link-state messages.[4]The new state of a link must be distributed to all the nodes in the network. The distribution can be done using either flooding or broadcasting through a minimum spanning tree. The latter approach generates a smaller number of messages. Assuming broadcasting through a minimum spanning tree, 40,000 messages are transmitted in a 10-ary 2-cube in an update period. The number of messages was derived from the number of nodes and links in Table 6.2. That is, on average, LS(10) generates 4,000 messages per request, and LS(100) does 400. As seen in Table 6.2, the overheads of LS(10) and even LS(100) are extremely large except the case when LS(10) is applied in the MBONE topology. Although LS(100) and LS(10) are comparable to BFlooding in terms of performance, their overhead is much larger than BFlooding, and, in some cases, larger than that of Flooding.

[4]Updates must not be synchronized to prevent the network from being overloaded with a burst of link-state messages.

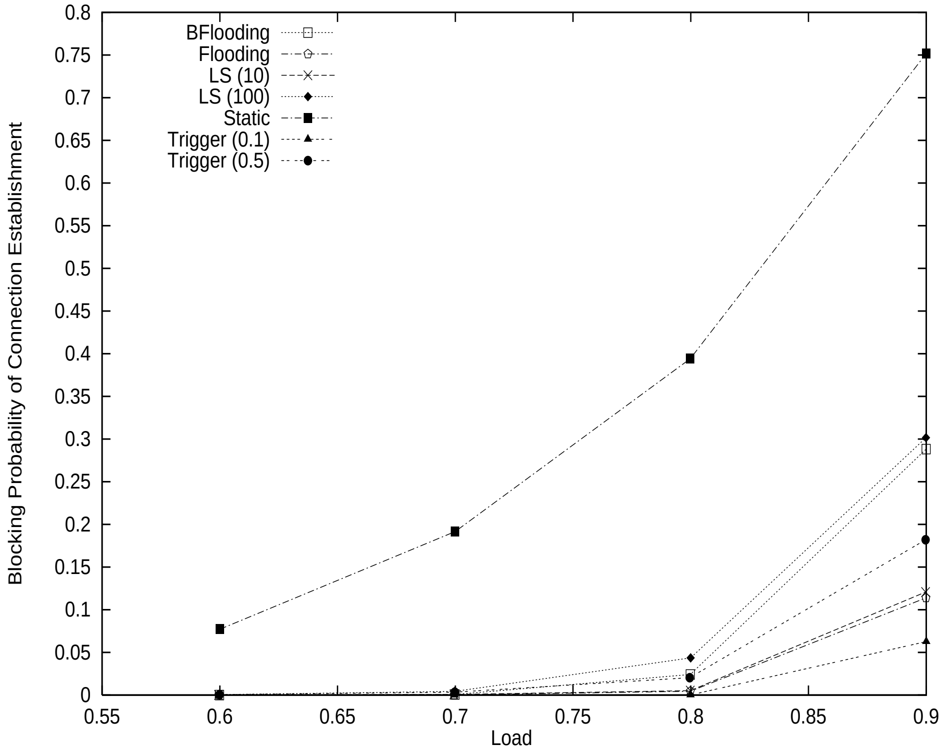
Figures 6.7 and 6.8 show the connection-blocking probabilities when the bandwidth ranges are 10% and 15%, respectively. Overall, the blocking probability increases as the bandwidth range increases, verifying the fact that a connection with a larger bandwidth requirement is less likely to be established. Moreover, as the bandwidth range gets larger, the gap between blocking probabilities of our scheme and Trigger(0.5) increases, which is the main comparison target in terms of performance and overhead. However, the gap between the overheads of our scheme and Trigger(0.5) also increases. The overhead of Triggers increases with bandwidth range because link-state changes more rapidly, triggering more frequent link-state distribution when bandwidth range gets large. In contrast, the overhead of our scheme does not change with bandwidth range.

In order to investigate the tradeoff between message overhead and performance loss, we varied the search scope of a connection. By increasing the search scope, we expected performance improvement in terms of connection-establishment success probability. In this experiment, additional hops were added to the original search-scope which was set so as to search at least two alternate paths. We only considered the case when the bandwidth range is 10%. In Figures 6.11 and 6.12, the connection-blocking probability and message overhead for the 10-ary 2-cube and the MBONE topology are plotted for our approach with different search-scopes, Flooding and Trigger(0.5). In BFlooding(n), we added n hops to the original search-scope used by BFlooding. For the 10-ary 2-cube, increasing the search-scope by one or two hops does not add any route to be searched, thus causing no change in performance or message overhead. Thus, the search-scope was increased by 3, 6, 9, and 12 hops[5].When the increment is 3, the connection-blocking probability is lower than that of Trigger(0.5) or even Flooding. However, the number of messages generated per request is much larger than that in BFlooding. Compared to Trigger(0.5), BFlooding(3) incurs larger overhead under a mid-range load, but smaller under a heavily-loaded condition. This is because, under a heavily-loaded condition, a smaller number of request messages are relayed due to the bandwidth test failure in BFlooding(3) while more frequent link-state distributions are required in Trigger(0.5). Little gain is made by adding more hops to the search-scope as seen in Figure 6.11. In fact, the best performance was achieved when the increment was 6. As the increment gets larger, the performance deteriorates, since temporarily-reserved link bandwidth prevents new connections from passing the bandwidth test as with Flooding. Despite the little gain in performance, the increase in message overhead is very large in BFlooding(6), implying that increasing the search scope beyond some limit does not help

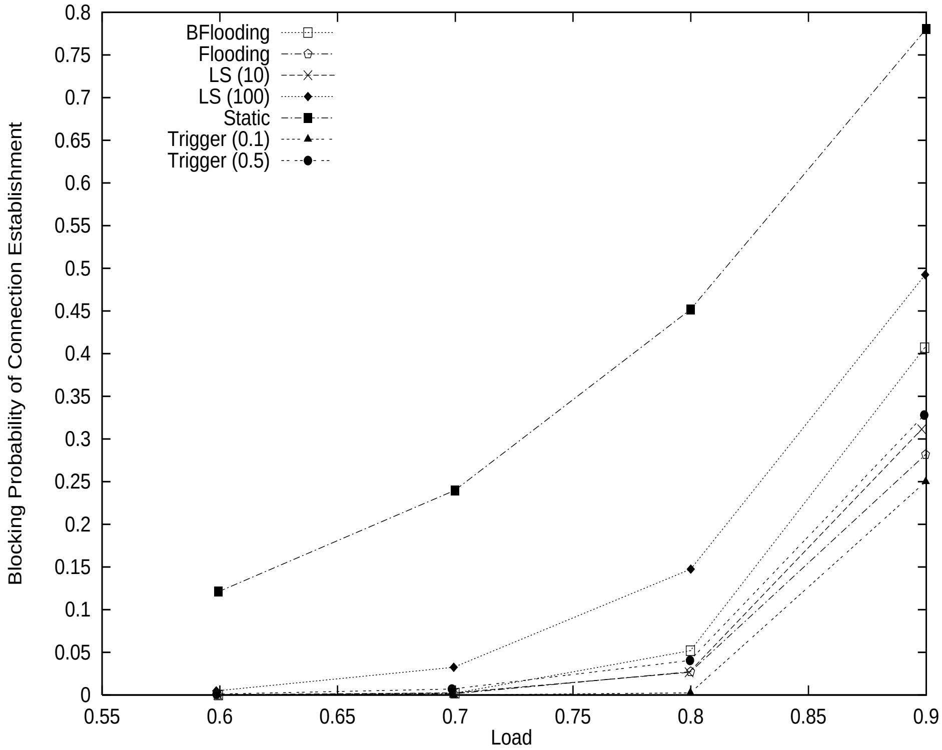
[5] Some cases are omitted in the figures for clarity.



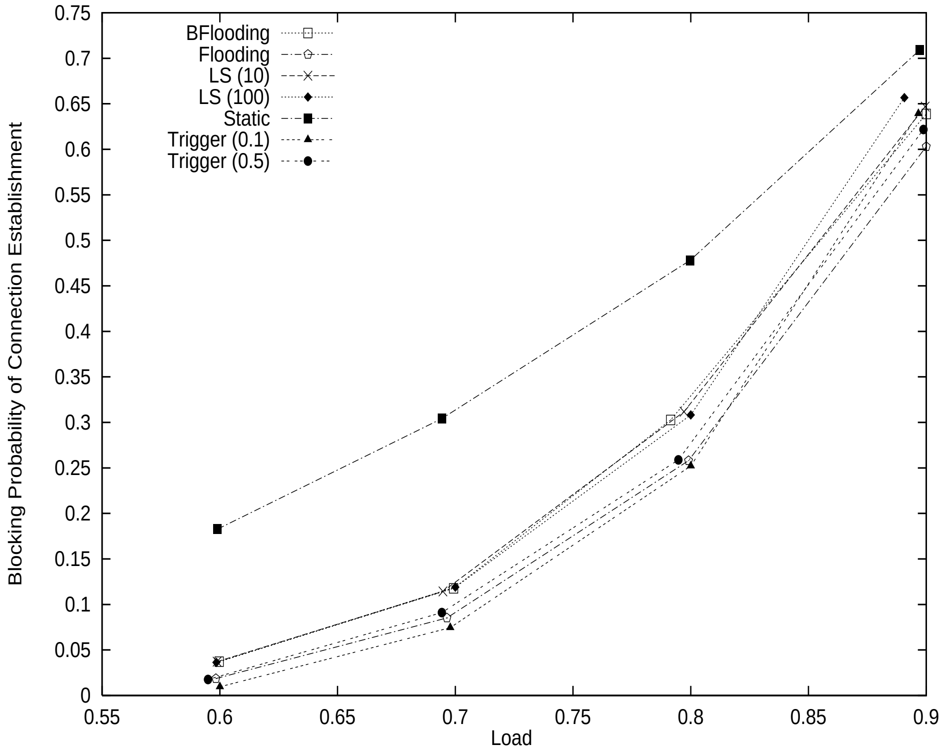
(a) 5-ary 2-cube



(b) 5-ary 3-cube

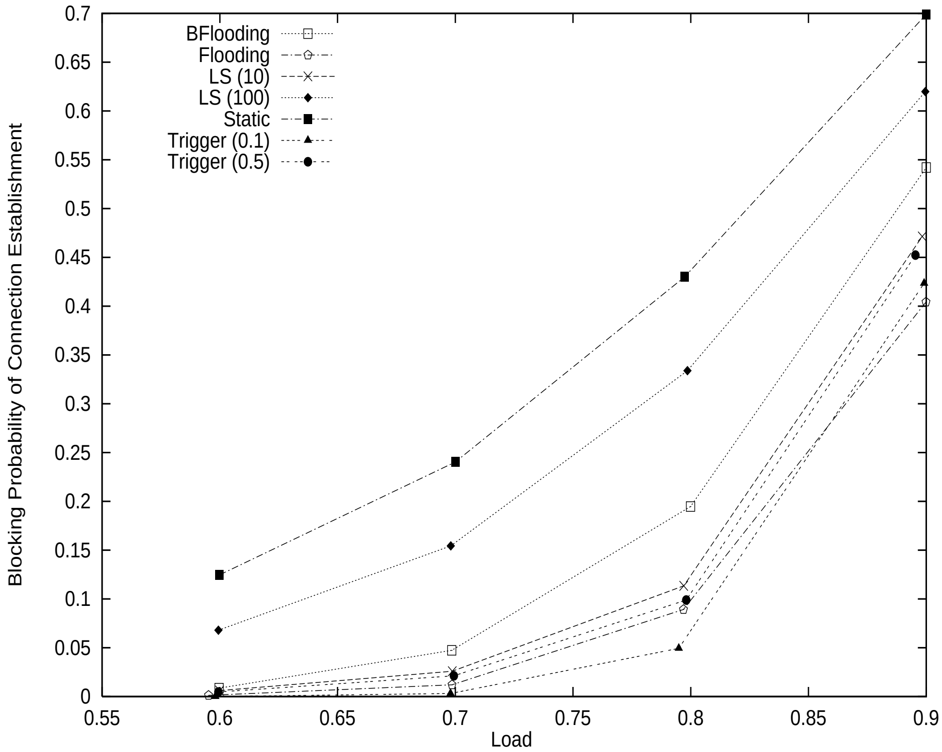


(c) 10-ary 2-cube

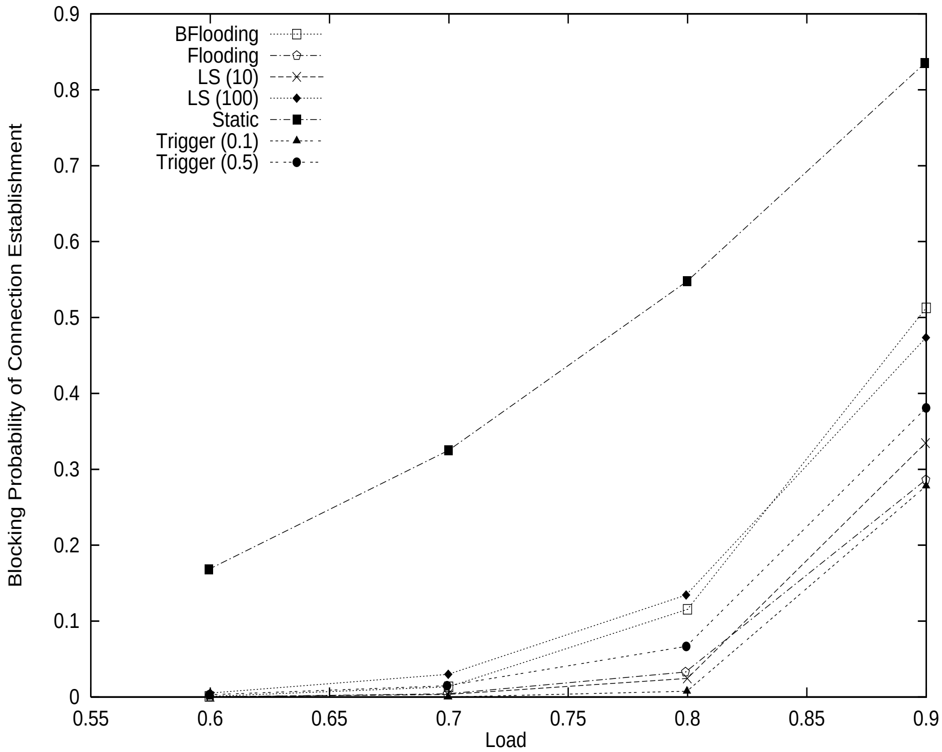


(d) MBONE

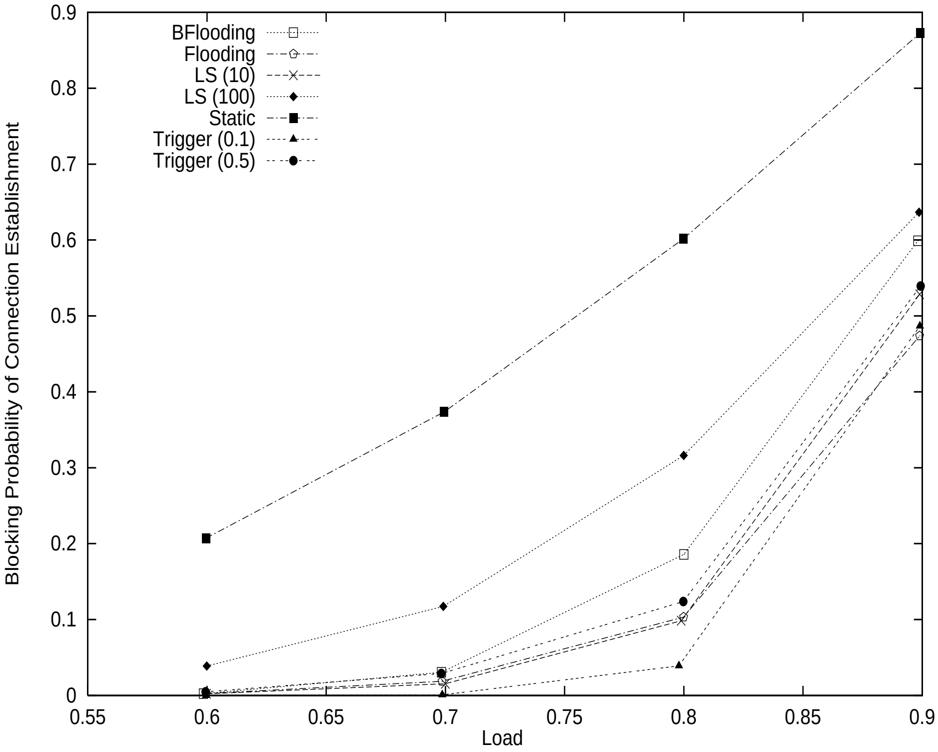
Figure 6.7: Connection-blocking probability, $b \sim (0, 10\%]$



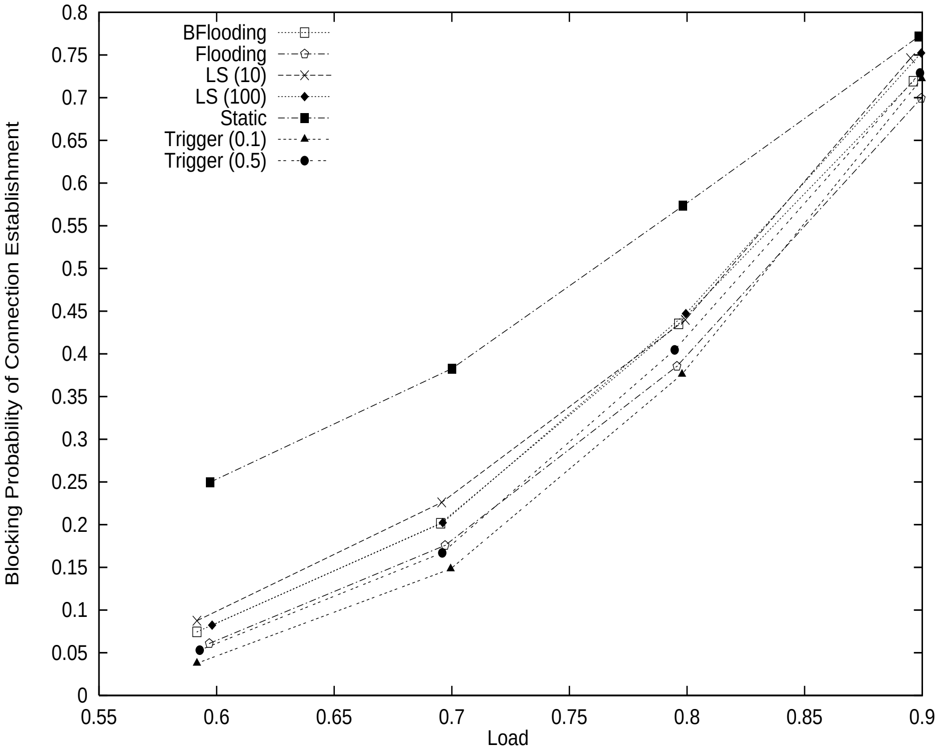
(a) 5-ary 2-cube



(b) 5-ary 3-cube

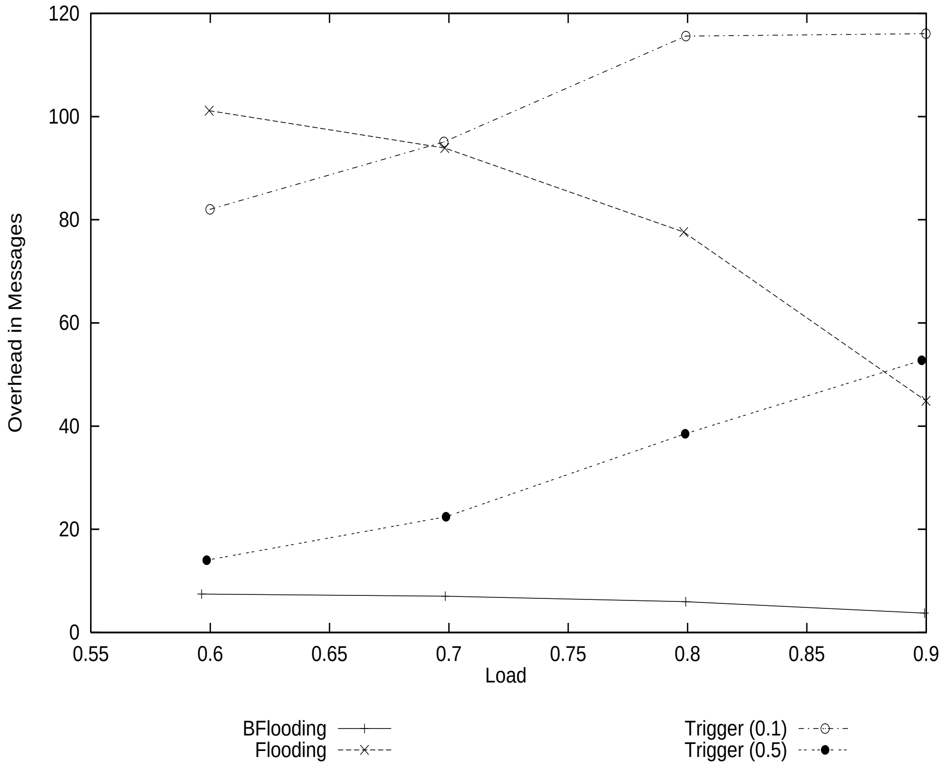


(c) 10-ary 2-cube

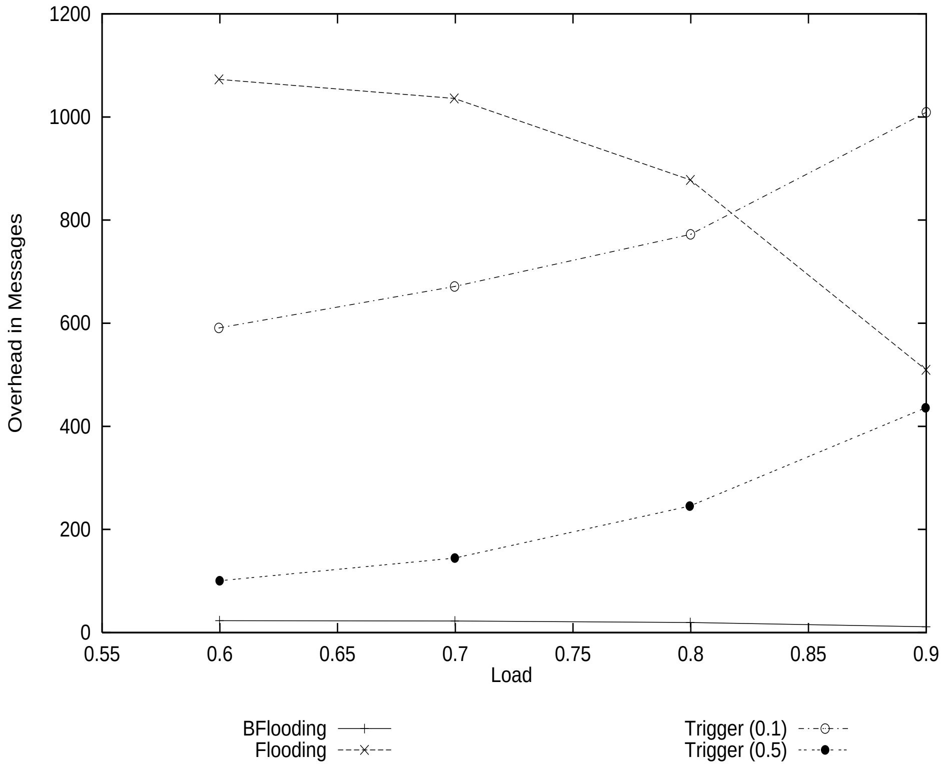


(d) MBONE

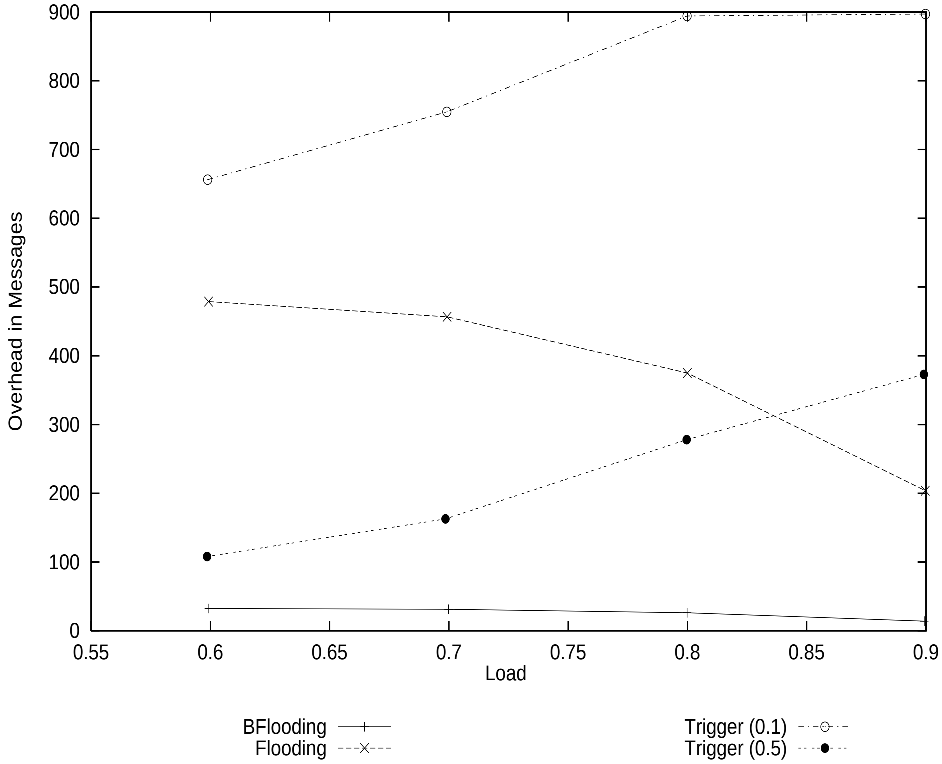
Figure 6.8: Connection-blocking probability, $b \sim (0, 15\%]$



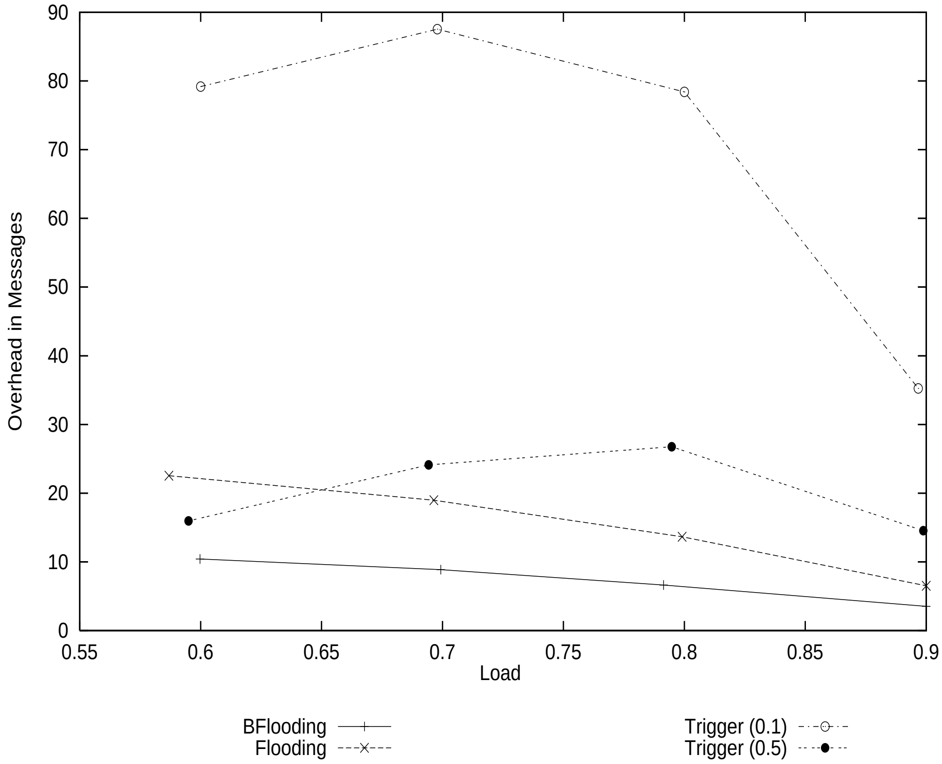
(a) 5-ary 2-cube



(b) 5-ary 3-cube

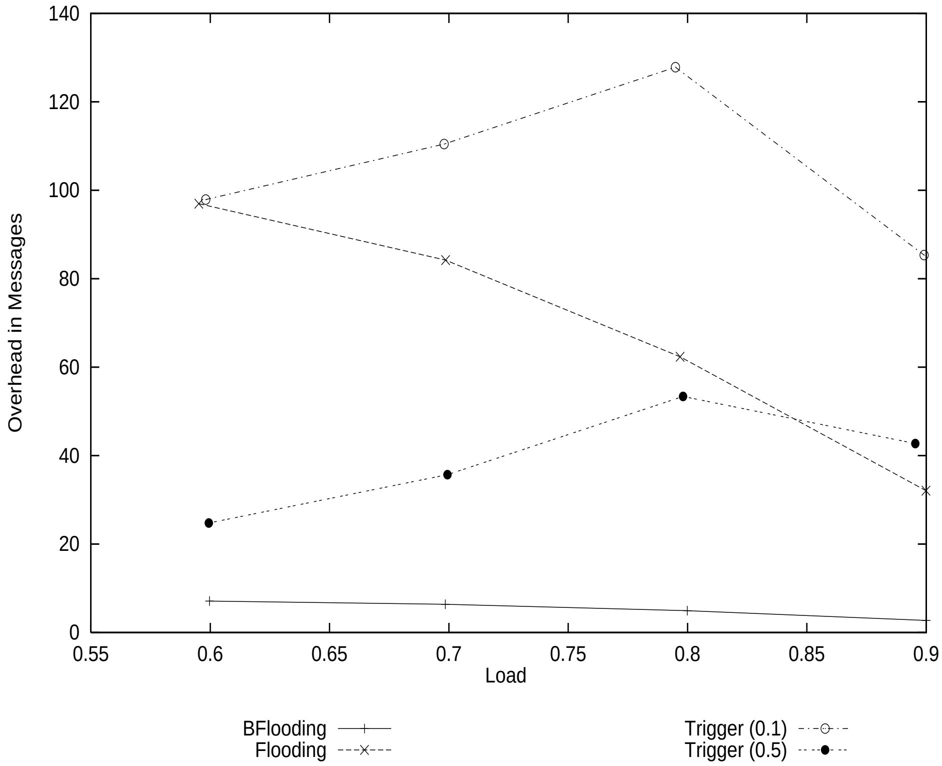


(c) 10-ary 2-cube

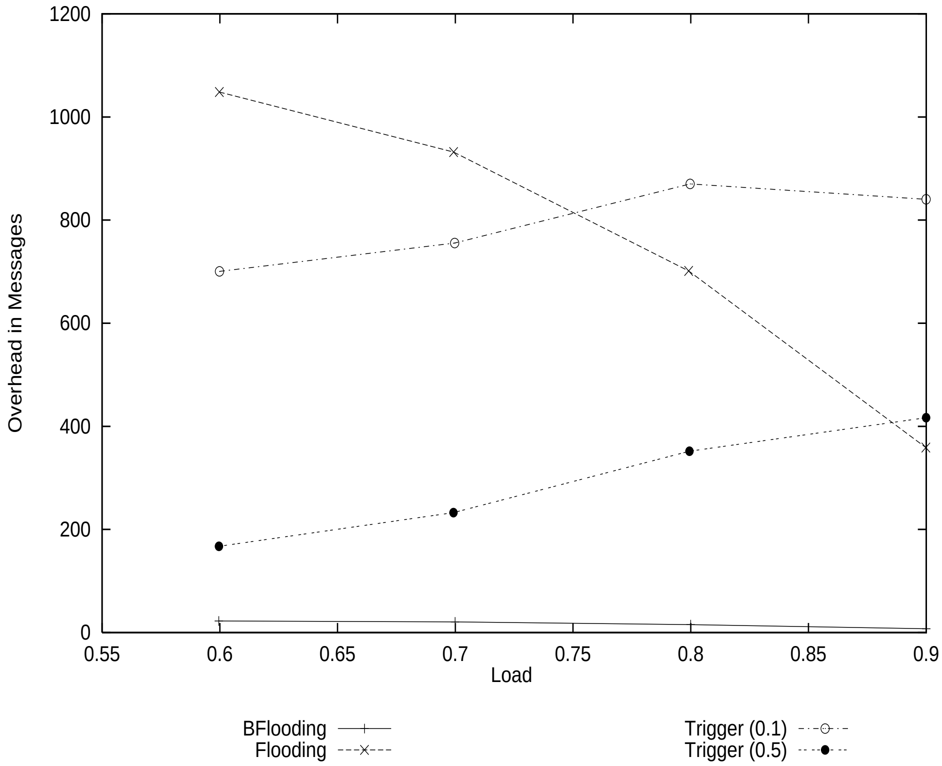


(d) MBONE

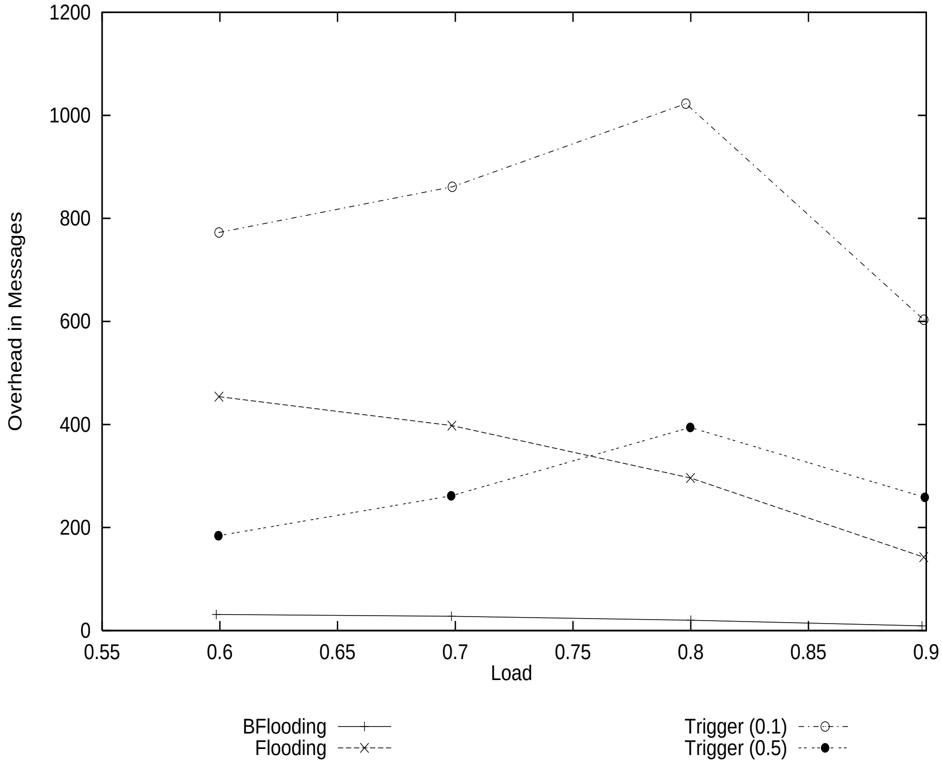
Figure 6.9: Overhead, $b \sim \ (0, 10\%]$



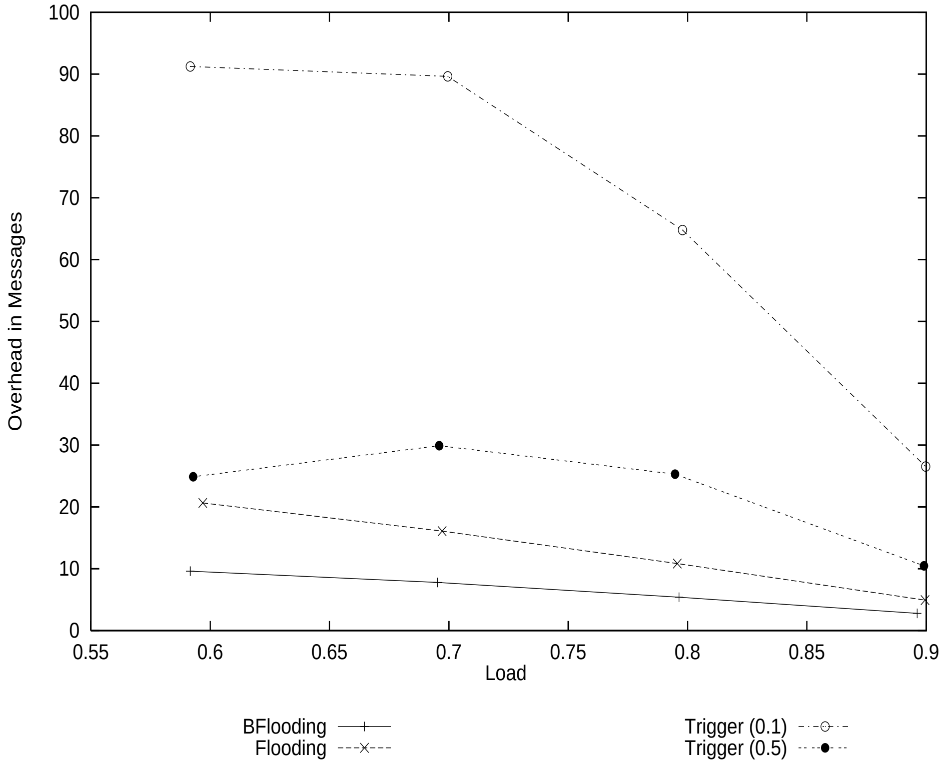
(a) 5-ary 2-cube



(b) 5-ary 3-cube

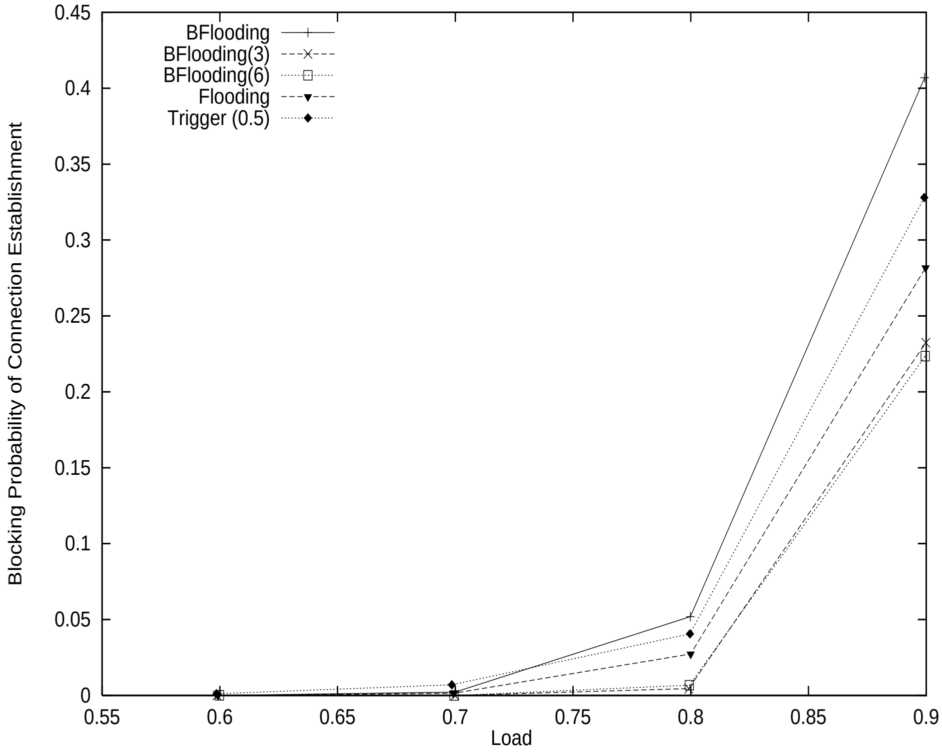


(c) 10-ary 2-cube

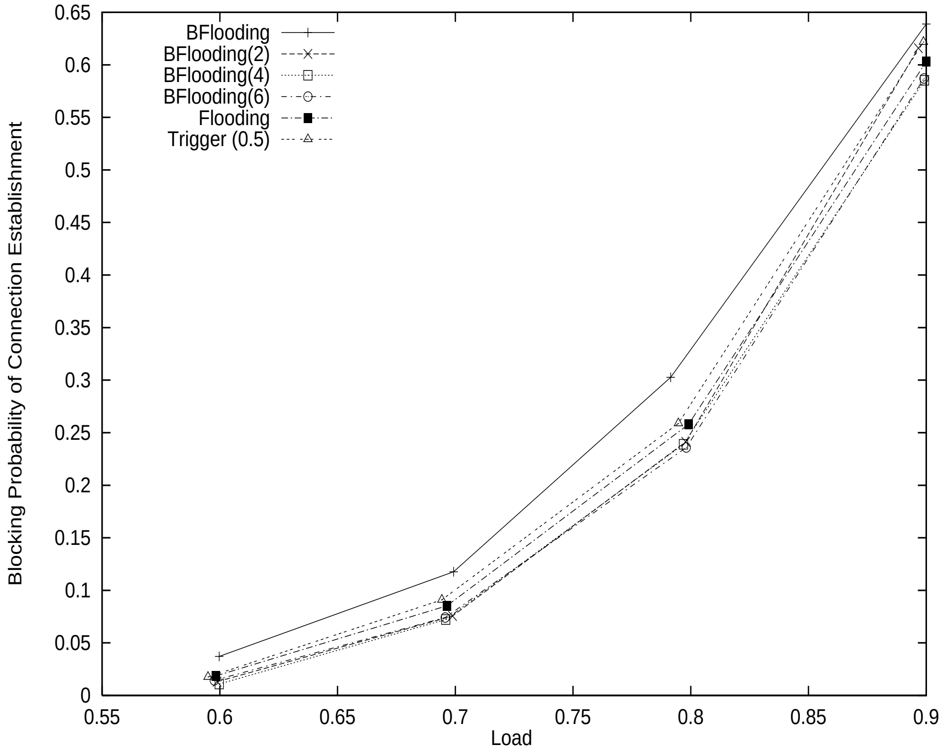


(d) MBONE

Figure 6.10: Overhead, $b \sim (0, 15\%]$

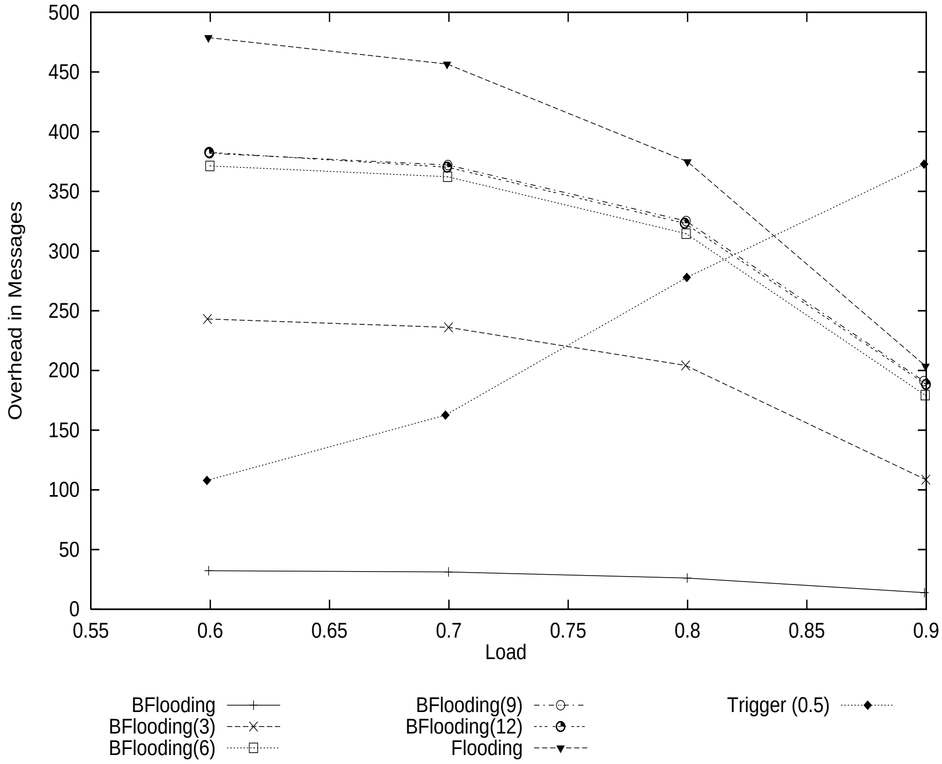


(a) 10-ary 2-cube

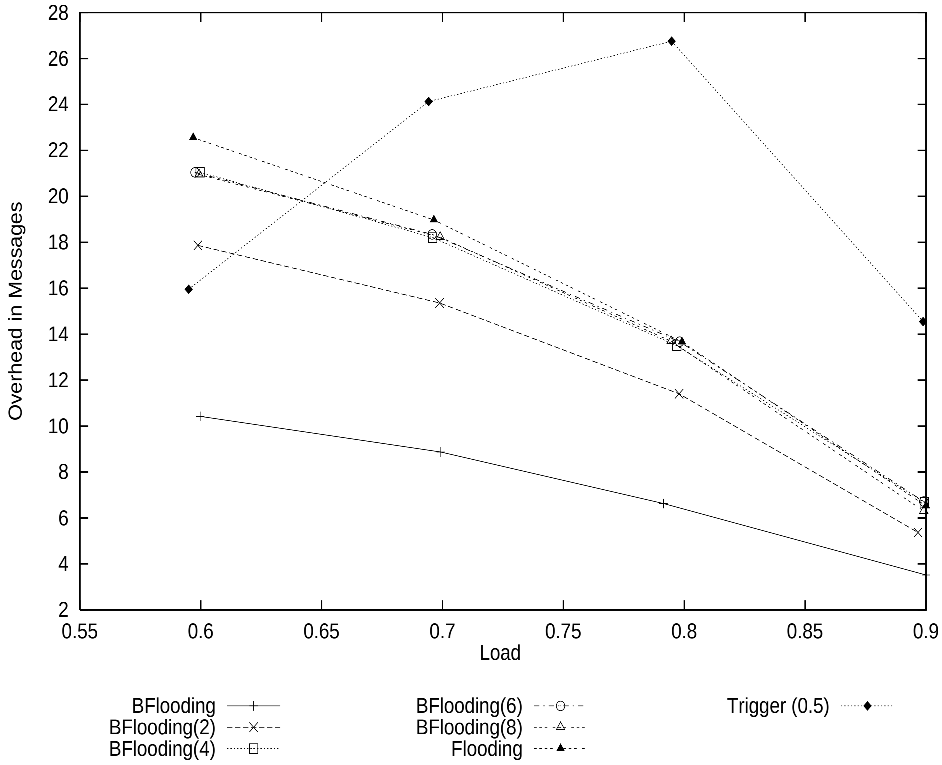


(b) MBONE

Figure 6.11: Connection-blocking probability for different search-scopes



(a) 10-ary 2-cube



(b) MBONE

Figure 6.12: Overhead for different search-scopes

at all. The overhead does not change after the increment gets larger than 6. By setting the search scope to ∞, one can obtain the result of Flooding. BFlooding(3), BFlooding(6), and BFlooding(9) are superior to Flooding in terms of performance and message overhead. The result for the 10-ary 2-cube suggests use of an adaptive approach for determining the search scope, depending on network load. That is, in case of heavy load, a larger search scope needs to be used, which will greatly improve the connection-establishment success probability of our approach. Since our approach is not based on any global network-load information, it must use an indirect approach like prediction based on the connection-establishment success rate or local information on the source node's outgoing links. The result for the MBONE topology verifies the same trend observed in the 10-ary 2-cube.

6.4.3 Discussion

In the simulation study, we assumed a static network topology, and thus, did not include the distribution of topology information in calculating the message overhead. This assumption can be justified when the network is quite stable, i.e., the probability of link/node failure is very low and addition/removal of link/node is rare. In reality, however, networks are quite dynamic since new nodes and links are added to, or removed from, the existing network as seen in the rapidly growing Internet. In addition, as the network gets larger, the number of failed and/or restored nodes and links grows. Thus, we need to consider the effect of topology changes on message overhead. Since addition or removal of links/nodes must be known to the entire network in any QoS routing scheme, we only consider the failure or restoration of links/nodes. In addition, we consider only link connectivity, because the failure or restoration of a node can be considered as those of the entire set of links attached to the node.

Topology changes can be handled by the datagram routing scheme (e.g., OSPF) without any additional overhead. Since we are considering the QoS routing problem in an ISPN environment, we can assume that a datagram routing scheme for best-effort traffic is working concurrently with the proposed QoS routing scheme, and that the datagram routing scheme handles the distribution of topology change information.

However, it may be meaningful to consider message overhead due to topology changes separately from the datagram routing scheme. In order to examine the overhead increase due to the change of link connectivity, we assume that a link toggles between "connected" and "disconnected" state and the lifetime of each state is exponentially distributed with mean $\bar{t}$. Although this model does not capture the exact link connectivity change characteristics in a general network environment, it enables us to quantify the frequency of link connectivity changes. Using this model, we analyze the message overhead of the QoS routing schemes considered in the simulation.

First, let's consider BFlooding. Every link's connectivity changes once, on average, for a period of $\bar{t}$. Thus, if we assume that the new link connectivity information is broadcast via a minimum spanning tree as done for link-state distribution, approximately 40,000 messages per $\bar{t}$ will be generated in a 10-ary 2-cube. If we do not differentiate these link connectivity-change messages from connection-request messages, the sum of link connectivity-change messages and connection-request messages becomes the message overhead. Per-request message overhead depends greatly on $\bar{t}$. For example, if $\bar{t}$ is 1,000 (10,000) times larger than the average connection-request inter-arrival time, per-request link connectivity change messages will be 40 (4). In that case, the total message overhead is given as 65–73 (29–37) from Table 6.2. The overhead of BFlooding is still much smaller than that of Flooding, LS(10), and LS(100) shown in Table 6.2. In contrast, when $\bar{t}$ is small, e.g., 10 (100) times larger than the average connection-request inter-arrival time, per-request link connectivity-change messages will be 4,000 (400). In this case, BFlooding loses its own merit, low message overhead, relative to Flooding, LS(10), and LS(100). Thus, BFlooding is not a good candidate for QoS routing for such an unstable network. In fact, neither LS(10) nor LS(100) is good. This is because the link-state update period is similar to the link connectivity-change interval, and thus, it degrades the accuracy of link-state information. For such an unstable network, pure flooding-based QoS routing is the best in terms of performance and overhead. It does not require global topology information in its routing/signaling, and thus is not affected by inaccurate link-state information or the frequency of topology changes. Furthermore, it does not require any shortest path calculation, thus lowering the operational cost.

In the link-state routing with triggered link-state distribution, the number of topology-change messages generated must be the same as that of BFlooding, because the change of link connectivity triggers link-state distribution. According to the definition of link-state routing with triggered link-state distribution, link-state information is distributed when the available link bandwidth changes by more than the trigger level, and the change of link connectivity satisfies this condition. Thus, the relation between Trigger(0.1) or Trigger(0.5) and BFlooding does not change regardless whether the network topology is static or dynamic.

In summary, the proposed QoS routing has very low overhead and operational cost, yet providing good performance if the network is reasonably stable.

6.5 Conclusion

In this chapter, we have proposed and evaluated a cost-effective QoS-routing scheme that incurs small overhead and operational cost, but provides reasonably good performance. Unlike link-state routing, the proposed scheme does not require distribution and maintenance of link-state information, nor expensive on-line path computation. Instead, every node is required to keep a distance table which can be obtained off-line using almost static network-topology information. A qualified route for each requested real-time connection is searched by flooding request messages with a limited hop count.

Using a simulation study, we have comparatively evaluated the performances and overheads of ours and others'. In terms of overhead, our scheme outperforms the others in most cases, although some of the other schemes provide performance slightly better than, or comparable to, our scheme. Although our scheme has a lower connection-establishment probability than brute-force flooding or link-state routing, it still provides reasonable performance at much lower overhead and cost.

CHAPTER 7

SUMMARY AND FUTURE WORK

7.1 Summary of Contributions

We summarize the contributions of this dissertation, and explore possible extensions of the work presented here.

Traffic-controlled rate-monotonic priority scheduling: To provide per-connection delay guarantees over ATM networks, we execute traffic shaping at the UNI and employ a link scheduler called TCRM at the intermediate switches. TCRM requires only a small buffer space due to its non-work-conserving service policy. It is shown to emulate circuit-switching in the cell level, so one can provide CBR services in ATM networks while keeping the statistical multiplexing gain. We have developed a simple admission-test algorithm and also presented an implementation of TCRM using a systolic array priority queue. This implementation scales well (important for large-scale high-speed networks) and requires far less memory than the implementation of PGPS. TCRM is similar to RCSP and RTC in terms of implementation complexity, but can achieve high channel admissibility similar to that of PGPS which is much more complex to implement than TCRM.

Statistical real-time communication over ATM networks: Based on the histogram-based source traffic model, we investigated the cell loss behavior of aggregate video streams which are multiplexed onto a common channel serviced by a modified TCRM. Assuming a constant cell-arrival rate, we have modeled the arriving traffic of the aggregate video streams as a Poisson process and derived the cell-loss ratio from the $M/D/1/N$ analysis. Then, the cell-loss ratio of the aggregate video streams is given as the weighted sum of the derived cell-loss ratios, and the weight was given by the rate-histogram. Through a simulation study using MPEG-coded video data, we showed the effectiveness of our framework for statistical real-time communication.

Semi-real-time communication: The semi-real-time class was introduced to service VoD-like applications more efficiently than the real-time or best-effort class. We defined and derived a statistical traffic envelope to characterize the semi-real-time class traffic using the Central Limit Theorem. Trace-driven simulations have shown that the derived statistical traffic envelope performs as intended as a worst-case traffic envelope in a statistical sense. Through a numerical evaluation, we showed that VoD-like applications can be better serviced using the semi-real-time communication service in terms of consumed network resources.

Statistical real-time communication over Ethernet: We modeled the 1-persistent CSMA/CD with BEB as an SMP process and derived conditional success probability of each packet in its transmission, depending on the number of trials for transmission. From this conditional success probability, we obtained the tail distribution of packet transmission delay. Based on this analysis, we derived a connection admission control for realizing statistical real-time communication on the Ethernet.

Distributed QoS routing using bounded flooding: QoS routing based on bounded flooding is a cost-effective QoS-routing scheme that incurs small overhead and operational cost, but provides reasonably good performance. Unlike link-state routing, the proposed scheme does not require distribution and maintenance of link-state information, nor expensive on-line path computation. Instead, every node is required to keep a distance table which can be obtained off-line using almost static network-topology information. In the routing/signaling phase, signaling messages are flooded through routes to the destination node with hop counts smaller than a pre-specified limit in order to search and establish a QoS route. Through a simulation study, we showed that although our scheme has a lower connection-establishment probability than brute-force flooding or link-state routing, it still achieves reasonable performance at much lower overhead and cost.

7.2 Future Directions

In this dissertation, we have addressed the problem of providing real-time communication over ISPNs. However, there still remain many unsolved problems and issues associated with QoS guarantees in ISPNs. The problems listed below are some of them.

Implementation of high-speed scheduler: Implementation of an efficient and scalable high-speed cell scheduler is crucial for providing per-connection QoS in large-scale high-speed networks. Although we presented an implementation of TCRM, it is very important to implement a cell scheduler in hardware and test it in a high-speed network environment.

Statistical modeling of WWW traffic: World Wide Web (WWW) is one of the most popular applications in the current Internet. Presently, WWW traffic is treated as best-effort traffic. However, one may enhance its QoS by using the semi-real-time class considered in this dissertation. To derive the statistical traffic envelope of WWW traffic, we need to model the session arrival behavior as well as the traffic arrival behavior within a session. The statistical traffic envelope of WWW traffic is expected to show a distinct feature because of its extremely bursty characteristics and long range dependency.

Hierarchical approach on QoS routing: To achieve scalability, it is very important that a QoS routing algorithm have a hierarchical structure for signaling and for exchanging topology change information. It would be interesting to extend our QoS routing algorithm to a hierarchical form. The hierarchical algorithm must be designed to minimize the effect of inaccurate topology information observed in hierarchical approaches.

Implementation of QoS routing algorithm: In this dissertation, we have presented the QoS routing algorithm in the form of a flowchart and provided many simulation results. However, in order to prove the usefulness of our approach in QoS routing, it is very important to investigate the performance of our algorithm through experiments in a real network.

Resource reservation scheme: We did not address the resource reservation protocol in this dissertation. However, this issue is crucial to QoS networking. In particular, it would be interesting to investigate the relationship among our cell scheduling scheme, QoS routing, and RSVP [101] which is one of prototypical resource reservation protocols.

BIBLIOGRAPHY

BIBLIOGRAPHY

[1] C. M. Aras, J. Kurose, D. S. Reeves, and H. Schulzrinne, "Real-time communication in packet-switched networks," *Proceedings of the IEEE*, vol. 82, no. 1, pp. 122–139, January 1994.

[2] R. R. Bahadur and R. Rao, "On deviations of the sample mean," *Ann. Math. Statis*, vol. 31, pp. 1015–1027, 1960.

[3] J. Beran, R. Sherman, M. S. Taqqu, and W. Willinger, "Long-range dependence in variable bit-rate video traffic," *IEEE Trans. on Commun.*, vol. 43, pp. 1566–1579, 1995.

[4] D. Bertsekas and R. Gallager, *Data Networks*, Prentice-Hall International, Englewood Cliffs, New Jersey, second edition, 1992.

[5] S. L. Beuerman and E. Coyle, "The delay characteristics of CSMA/CD networks," *IEEE Trans. on Commun.*, vol. 36, no. 5, pp. 553–563, May 1988.

[6] R. Breyer, *Switched and fast Ethernet : how it works and how to use it*, Ziff-Davis Press, Emeryville, Calif., 1995.

[7] C. Chang, "Stability, queue length, and delay of deterministic and stochastic queueing networks," *IEEE Trans. on Automatic Control*, vol. 39, no. 5, pp. 913–931, May 1994.

[8] H. J. Chao, "Architecture design for regulating and scheduling user's traffic in ATM networks," in *Proc. of ACM SIGCOMM*, pp. 77–87, 1992.

[9] H. J. Chao and N. Uzun, "A VLSI sequencer chip for ATM traffic shaper and queue management," *IEEE Journal of Solid-State Circuits*, vol. 27, no. 11, pp. 1634–1643, November 1992.

[10] J. Chao, "A novel architecture for queue management in the ATM network," *IEEE J. Select. Areas Commun.*, vol. 9, no. 7, pp. 1110–1118, September 1991.

[11] C. C. Chou and K. G. Shin, "Statistical real-time video channels over a multiaccess network," in *Proc. of High-Speed Networking and Multimedia Computing*, pp. 86–96, February 1994.

[12] D. D. Clark, S. Shenker, and L. Zhang, "Supporting real-time applications in an integrated services packet network: architecture and mechanism," in *Proc. of ACM SIGCOMM*, pp. 14–26, 1992.

[13] R. B. Cooper, *Introduction to Queueing Theory*, North-Holland Publishing Company, New York, second edition, 1981.

[14] R. L. Cruz, "A calculus for network delay, part I: network elements in isolation," *IEEE Trans. on Information Theory*, vol. 37, no. 1, pp. 114–131, January 1991.

[15] R. L. Cruz, "A calculus for network delay, part II: network analysis," *IEEE Trans. on Information Theory*, vol. 37, no. 1, pp. 132–142, January 1991.

[16] M. de Prycker, *Asynchronous Transfer Mode: Solution for Broadband ISDN*, Ellis Horwood, London, 1991.

[17] A. Demers, S. Keshav, and S. Shenker, "Analysis and simulation of a fair queueing algorithm," in *Proc. of ACM SIGCOMM*, pp. 1–12, 1989.

[18] A. Elwalid, D. Mitra, and R. Wentworth, "A new approach for allocating buffers and bandwidth to heterogeneous, regulated traffic in an ATM node," *IEEE J. Select. Areas Commun.*, vol. 13, no. 6, pp. 1115–1127, August 1995.

[19] D. Ferrari and D. C. Verma, "A scheme for real-time channel establishment in wide-area networks," *IEEE J. Select. Areas Commun.*, vol. 8, no. 3, pp. 368–379, April 1990.

[20] M. R. Frater, J. F. Arnold, and P. Tan, "A new statistical model for traffic generated by VBR coders for television on the Broadband ISDN," *IEEE Trans. on Circuits and Systems for Video Technology*, vol. 4, no. 6, pp. 521–526, December 1994.

[21] M. W. Garrett and M. Vetterli, "Joint source/channel coding of statistically multiplexed real-time services on packet networks," *IEEE/ACM Trans. on Networking*, vol. 1, no. 1, pp. 71–80, February 1993.

[22] M. W. Garrett and W. Willinger, "Analysis, modeling, and generation of self-similar VBR video traffic," in *Proc. of ACM SIGCOMM*, pp. 269–280, September 1994.

[23] E. Gelenbe and G. Pujolle, *Introduction to Queueing Networks*, John Wiley and Sons, New York, 1987.

[24] L. Georgiadis, R. Guérin, V. Peris, and R. Rajan, "Efficient support of delay and rate guarantees in an internet," in *Proc. of ACM SIGCOMM*, pp. 106–116, 1996.

[25] L. Georgiadis, R. Guérin, V. Peris, and K. N. Sivarajan, "Efficient network QoS provisioning based on per Node traffic shaping," in *Proc. of IEEE INFOCOM*, pp. 102–110, 1996.

[26] S. J. Golestani, "A stop-and-go queueing framework for congestion management," in *Proc. of ACM SIGCOMM*, pp. 8–18, 1990.

[27] S. J. Golestani, "A self-clocked fair queueing scheme for broadband applications," in *Proc. of IEEE INFOCOM*, pp. 636–646, 1994.

[28] Guérin and A. Orda, "QoS-based routing in networks with inaccurate information: Theory and algorithms," in *INFOCOM' 97*, apr 1997.

[29] R. Guérin, H. Ahmadi, and M. Naghshineh, "Equivalent capacity and its application to bandwidth allocation in high-speed networks," *IEEE J. Select. Areas Commun.*, vol. 9, no. 7, pp. 968–981, September 1991.

[30] R. Guérin, S. Kamat, and A. Orda. *QoS routing mechanisms and OSPF extensions.* Internet Draft, March 1997. To appear in *Proceedings of IEEE GLOBECOM*, November 1997.

[31] D. Heyman, A. Tabatabai, and T. Lakshman, "Statistical analysis of MPEG2-coded VBR video traffic," in *Proc. of the Sixth International Workshop on Packet video*, 1994.

[32] D. P. Heyman, A. Tabatabai, and T. Lakshman, "Statistical analysis and simulation study of video teleconferencing in ATM networks," *IEEE Trans. on Circuits and Systems for Video Technology*, vol. 2, no. 1, pp. 49–59, March 1992.

[33] C.-J. Hou, "Routing virtual circuits with timing requirements in virtual path based ATM networks," in *Proc. of IEEE INFOCOM*, pp. 320–328, April 1996.

[34] C. Huang, M. Devetsikiotis, L. Lambadaris, and A. Kaye, "Modeling and simulation of self-similar variable bit rate compressed video: A unified approach," in *Proc. of ACM SIGCOMM*, 1995.

[35] B. Jabbari, F. Yegengolu, Y. Kuo, S. Zafar, and Y.-Q. Zhang, "Statistical characterization and block-based modeling of motion-adaptive coded video," *IEEE Trans. on Circuits and Systems for Video Technology*, vol. 3, no. 3, pp. 199–207, June 1993.

[36] C. Kalmanek, H. Kanakia, and S. Keshav, "Rate controlled servers for very high-speed networks," in *Proc. of IEEE GLOBECOM 90*, pp. 12–20, 1990.

[37] D. D. Kandlur, K. G. Shin, and D. Ferrari, "Real-time communication in multi-hop networks," in *Proc. 11-th Int'l Conf. Distributed Comput. Systems*, pp. 300–307, May 1991.

[38] D. D. Kandlur, K. G. Shin, and D. Ferrari, "Real-time communication in multi-hop networks," *IEEE Trans. on Parallel and Distributed Systems*, vol. 5, no. 10, pp. 1044–1056, October 1994.

[39] G. Kesidis, J. Walrand, and C.-S. Chang, "Effective bandwidths for multiclass markov fluids and other ATM sources," *IEEE/ACM Trans. on Networking*, vol. 1, no. 4, pp. 424–428, August 1993.

[40] L. Kleinrock, *Communication Nets: stochastic message flow and delay*, McGraw-Hill, New York, 1964.

[41] L. Kleinrock, "Packet Switching in radio channels: part I-Carrier Sense Multiple-Access Modes and their throughput and delay characteristics," *IEEE Trans. on Communications*, vol. 23, no. 12, pp. 1400–1416, December 1975.

[42] L. Kleinrock, *Queueing Systems*, volume 1, John Wiley and Sons, New York, 1976.

[43] E. Knightly, "H-BIND: A new approach to providing statistical performance guarantees to vbr traffic," in *Proc. of IEEE INFOCOM*, pp. 1091–1099, 1996.

[44] E. Knightly, D. Wrege, J.Liebeherr, and H. Zhang, "Fundamental limits and tradeoffs of providing deterministic guarantees to VBR video traffic," in *Proc. of ACM SIGMETRICS*, pp. 98–107, 1995.

[45] E. Knightly and H. Zhang, "Traffic characterization and switch utilization using a deterministic bounding interval dependent traffic model," in *Proc. of IEEE INFOCOM*, pp. 1137–1145, 1995.

[46] M. Krunz, R. Sass, and H. Hughes, "Statistical characteristics and multiplexing of MPEG streams," in *Proc. of IEEE INFOCOM*, pp. 455–462, 1995.

[47] M. Krunz and H. Tripathi, "Impact of video scheduling on bandwidth allocation for multiplexed MPEG streams," *Multimedia Systems Journal*, vol. 5, no. 6, pp. 47–57, December 1997.

[48] M. Krunz and H. Tripathi, "On the characterization of VBR MPEG streams," in *Proc. of ACM SIGMETRICS*, pp. 192–202, June 1997.

[49] J. Kurose, "On computing per-session performance bounds in high-speed multi-hop computer networks," in *Proc. of ACM SIGMETRICS*, pp. 128–139, 1992.

[50] S. Kweon and K. G. Shin. *Statistical performance guarantees in ATM networks.* Real-Time Computing Laboratory Technical Report, October 1996.

[51] S. Kweon and K. G. Shin, "Traffic-controlled rate-monotonic priority scheduling," in *Proc. of IEEE INFOCOM*, pp. 655–662, 1996.

[52] P. Lavoie and Y. Savaria, "A systolic architecture for fast stack sequential decoders," *IEEE Trans. on Commun.*, vol. 42, no. 2/3/4, pp. 324–334, February/March/April 1994.

[53] C. Leiserson, "Systolic priority queues," in *Caltech Conference on VLSI*, pp. 200–214, January 1979.

[54] C. Liu and J. W. Layland, "Scheduling algorithms for multiprogramming in a hard real-time environment," *Journal of ACM*, vol. 20, no. 1, pp. 44–61, January 1973.

[55] Q. Ma and P. Steenkiste, "Quality-of-service routing with performance guarantees," in *Proc. of the 4th International IFIP Workshop on Quality of Service*, March 1996.

[56] Q. Ma and P. Steenkiste, "On path selection for traffic with bandwidth guarantees," in *Proc. of IEEE International Conference on Network Protocols*, October 1997.

[57] Q. Ma, P. Steenkiste, and H. Zhang, "Routing high-bandwidth traffic in max-min fair share networks," in *Proc. of ACM SIGCOMM*, pp. 206–217, August 1996.

[58] B. Maglaris, D. Anastassiou, P. Sen, G. Karlsson, and J. D. Robbins, "Performance models of statistical multiplexing in packet video communications," *IEEE Trans. on Communications*, vol. 36, no. 7, pp. 834–844, July 1988.

[59] J. M. McManus and K. W. Ross, "Video-on-Demand over ATM: constant-rate transmission and transport," *IEEE J. Select. Areas Commun.*, vol. 14, no. 6, pp. 1087–1098, August 1996.

[60] B. Melamed and D. Pendarakis, "A TES-based model for compressed Star Wars video," in *Proc. of IEEE GLOBECOM*, pp. 120–126, 1994.

[61] B. Melamed, D. Raychaudhuri, B. Sengupta, and J. Zdepski, "TES-based video source modeling for performance evaluation of integrated networks," *IEEE Trans. on Communications*, vol. 42, no. 10, pp. 2773–2777, October 1994.

[62] S.-W. Moon, J. Rexford, and K. G. Shin, "Scalable hardware priority queue architectures for high-speed packet switches," in *Proceedings of IEEE Real-time Technology and Applications Symposium*, 1997.

[63] P. Pancha and M. E. Zarki, "Bandwidth-allocation schemes for variable-bit-rate MPEG sources in ATM networks," *IEEE Trans. on Circuits and Systems for Video Technology*, vol. 2, no. 1, pp. 49–59, March 1992.

[64] A. K. Parekh, *A generalized processor sharing approach to flow control in integrated services networks*, PhD thesis, Department of Electrical Engineering and Computer Science, Massachusetts Institute of Technology, February 1992.

[65] A. K. Parekh and R. G. Gallager, "A generalized processor sharing approach to flow control in integrated services networks: the multiple node case," in *Proceedings of IEEE INFOCOM*, pp. 521–530, 1993.

[66] A. K. Parekh and R. G. Gallager, "A generalized processor sharing approach to flow control in integrated services networks: the single node case," *IEEE/ACM Transaction on Networking*, vol. 1, no. 3, pp. 344–357, June 1993.

[67] C. Parris, H. Zhang, and D. Ferrari, "Dynamic management of guaranteed performance multimedia connections," *Multimedia Systems Journal*, vol. 1, pp. 267–283, 1994.

[68] D. Pickers and R. Fellman, "A VLSI priority packet queue with inheritance and overwrite," *IEEE Trans. on Very Large Scale Integrated Systems*, vol. 3, no. 2, pp. 245–252, June 1995.

[69] PNNI Working Group. *ATM Forum 94-0471R13 PNNI Draft Specification.* Document available at ftp://ftp.atmforum.com/pub/contributions.

[70] G. Ramamurthy and B. Sengupta, "Modeling and analysis of a variable bit rate video multiplexer," in *Proc. of IEEE INFOCOM*, pp. 812–827, 1992.

[71] M. Reisslein and K. W. Ross, "Call admission for prerecorded sources with packet loss," *IEEE J. Select. Areas Commun.*, vol. 15, no. 6, pp. 1167–1180, August 1997.

[72] J. Rexford, A. G. Greenberg, and F. Bonomi, "Hardware-efficient fair queueing architectures for high-speed networks," in *Proc. of IEEE INFOCOM*, pp. 638–646, March 1996.

[73] J. Rexford, J. Hall, and K. G. Shin, "A router architecture for real-time point-to-point networks," in *Proc. of International Symposium on Computer Architecture*, pp. 237–246, May 1996.

[74] O. Rose, "Statistical properties of MPEG video traffic and their impact on traffic modeling in ATM systems," Technical Report 101, University of Wuerzburg Institute of Computer Science, February 1995. (Many MPEG-1 traces are available via FTP from ftp-info3.informatik.uni-wuerzburg.de in pub/MP).

[75] D. Saha, S. Mukherjee, and S. K. Tripathi, "Multirate scheduling for guaranteed and predictive services in ATM networks," in *Proc. of IEEE RTSS*, December 1996.

[76] H. F. Salama, D. Reeves, and Y. Viniotis, "A distributed algorithm for delay-constrained unicasting routing," in *Proc. of the 4th International IFIP Workshop on Quality of Service*, March 1996.

[77] J. D. Salehi, Z.-L. Zhang, J. Kurose, and D. Towsley, "Supporting stored video: reducing rate variability and end-to-end resource requirements through optimal smoothing," in *Proc. of ACM SIGMETRICS*, pp. 222–231, 1996.

[78] A. Shaikh, J. Rexford, and K. Shin, "Dynamics of quality-of-service routing with inaccurate link-state information," Technical Report CSE-TR-350-97, Dept. of Electrical Engineering and Computer Science, the University of Michigan, Ann Arbor, MI, November 1997.

[79] S. Shenker, D. Clark, and L. Zhang. *A service model for an integrated services internet.* IETF working draft, October 1993.

[80] K. G. Shin and C.-C. Chou, "A distributed route-selection scheme for establishing real-time channels," in *Proc. 6-th IFIP Int'l Conf. on High Performance Networking Conf. (HPN'95)*, pp. 319–329, September 1995.

[81] K. G. Shin and Q. Zheng, "FDDI-M: a scheme to double FDDI's ability of supporting synchronous traffic," *IEEE Trans. on Parallel and Distributed Systems*, vol. 6, no. 11, pp. 1125–1131, November 1995.

[82] K. G. Shin and C.-C. Chou, "Design and evaluation of real-time communication for FieldBus-based manufacturing systems," *IEEE Trans. on Robotics and Automation*, vol. 12, no. 3, pp. 357–367, June 1996.

[83] N. Shroff and M. Schwartz, "Video modeling within networks using deterministic smoothing at the source," in *Proc. of IEEE INFOCOM*, pp. 342–349, 1994.

[84] K.-Y. Siu and R. Jain, "A brief overview of ATM: protocol layers, LAN emulation, and traffic management," *Computer Communication Review*, vol. 25, no. 2, pp. 6–20, April 1995.

[85] P. Skelly, S. Dixit, and M. Schwartz, "A histogram-based for video traffic behavior in an ATM network node with an application to congestion control," in *Proc. of IEEE INFOCOM*, pp. 95–104, 1992.

[86] H. Stark and J. Woods, *Probability, random processes, and estimation theory for engineers*, Prentice Hall, Englewood Cliffs, New Jersey, second edition, 1994.

[87] Q. Sun and H. Langendorfer, "A new distributed routing algorithm with end-to-end delay guarantee," in *Proc. of the 4th International IFIP Workshop on Quality of Service*, March 1996.

[88] K. Toda, K. Nishida, E. Takahashi, N. Michell, and Y. Yamaguchi, "Design and implementation of a priority forwarding router chip for real-time interconnection networks," *International Journal of Mini and Microcomputer*, vol. 17, no. 1, pp. 42–51, 1995.

[89] C. Venkatramani and T. Chiueh, "Supporting real-time traffic on Ethernet," in *Proc. of Real-Time Systems Symposium*, pp. 282–286, December 1994.

[90] T. Vo-Dai, "Steady-state analysis of CSMA-CD," in *Performance*, pp. 243–265, 1984.

[91] Z. Wang and J. Crowcroft, "Quality-of-service routing for supporting multimedia applications," *IEEE J. Select. Areas Commun.*, vol. 14, no. 7, pp. 1228–1234, September 1996.

[92] D. E. Wrege, E. W. Knightly, H. Zhang, and J. Liebeherr, "Deterministic delay bounds for VBR video in packet-switching networks: fundamental limits and practical trade-offs," *IEEE/ACM Trans. on Networking*, vol. 4, no. 3, pp. 352–362, June 1996.

[93] G. Xie and S. Lam, "Delay guarantee of virtual clock server," Technical Report TR-94-24, Department of Computer Sciences, the University of Texas at Austin, Austin, TX, October 1994.

[94] O. Yaron and M. Sidi, "Performance and stability of communication networks via robust exponential bounds," *IEEE/ACM Trans. on Networking*, vol. 1, no. 3, pp. 372–385, June 1993.

[95] H. Zhang and D. Ferrari, "Rate-controlled static-priority queueing," in *Proc. of IEEE INFOCOM*, pp. 227–236, 1993.

[96] H. Zhang and S. Keshav, "Comparison of rate-based service disciplines," in *Proc. of ACM SIGCOMM*, pp. 113–121, 1991.

[97] H. Zhang and E. Knightly, "Providing end-to-end statistical guarantees using bounding interval dependent stochastic models," in *Proc. of ACM SIGMETRICS 94*, pp. 211–220, 1994.

[98] H. Zhang and E. Knightly, "Providing end-to-end statistical guarantees using bounding interval dependent stochastic models," in *Proc. of ACM SIGMETRICS*, pp. 211–220, 1994.

[99] H. Zhang, *Service disciplines for packet-switching integrated-services networks*, PhD thesis, Department of Computer Science, The University of California, Berkeley, CA, 1993.

[100] L. Zhang, "Virtual clock: a new traffic control algorithm for packet switching networks," in *Proc. of ACM SIGCOMM*, pp. 19–29, 1990.

[101] L. Zhang, S. Deering, D. Estrin, S. Shenker, and D. Zappala, "RSVP: a new resource ReSerVation protocol," *IEEE network*, pp. 8–18, September 1993.

[102] Z. Zhang, C. Sanchez, B. Salkewicz, and E. S. Crawley. *Quality of service extensions to OSPF or quality of service path first routing (QOSPF).* Internet Draft (draft-zhang-qos-ospf-01.txt), September 1997.

[103] Z. Zhang, D. Towsley, and J. Kurose, "Statistical analysis of the generalized processor sharing scheduling discipline," *IEEE J. Select. Areas Commun.*, vol. 13, no. 6, pp. 1071–1080, August 1995.

[104] Q. Zheng, *Real-time fault-tolerant communication in computer network*, PhD thesis, Department of Electrical Engineering and Computer Science, The University of Michigan, Ann Arbor, MI, 1993.

[105] Q. Zheng and K. G. Shin, "Synchronous bandwidth allocation in FDDI networks," *IEEE Trans. on Parallel and Distributed Systems*, vol. 6, no. 12, pp. 1332–1338, December 1995.