Guaranteed Performance Communication in High Speed Networks

by

Dinesh Chandra Verma

Dedicated ....

To my mother,

who always urged me forward

Chapter 1: Problem Statement

Chapter 2: The Traffic Contract-I: Specification and Mechanism

Acknowledgments

I would like to express my heart-felt thanks for Professor Domenico Ferrari, my research advisor at UC Berkeley. Domenico is a model research advisor. He introduced me to the field of computer networks and laid the foundations of the theory of guaranteed performance communication, upon which this thesis is based. He was encouraging and sympathetic even during times when I was confused and unsure about my research.

I am obliged to Professor Radu Popescu-Zeletin and Petia Todorova for introducing me to ATM networks. Prof. Zeletin suggested that I apply the concepts of real-time communication to ATM networks, which led to my research efforts in this area. I am indebted to Prof. Pravin Varaiya, Prof. Ronald Wolff, Prof. Zeletin, and Dr. Luis-Felipe Cabrera for reading my thesis. David Anderson, who was kind enough to serve on my quals committee, was the originator of the idea of real-time message streams, which developed into this thesis.

A number of ideas in writing the thesis were clarified by discussions with Prof. Bernd Wolfinger. His precision was extremely useful in developing the overall organization of this thesis. Many members of the Tenet Group were very helpful provided valuable support and advice. I would specially like to thank S. Keshav and Riccardo Gusella for their useful comments regarding my research. Stuart Sechrest, Peter Danzig, Pratap Khedkar, Shashi Shekhar, Ramesh Govindan and Venkat Rangan made my initial years at Berkeley very enjoyable. I would like to thank all members of the Tenet Group for their advice and support, specially Hui Zhang, Colin Parris, Amit Gupta and other members working on real-time

communication.

I thank Kathryn Crabtree for her help and advice in all administrative matters.

I would like to express my regards for my parents, who provided me with constant moral support and for giving me an upbringing and education which prepared me for the doctoral program.

This research was supported by the National Science Foundation and the Defence Advanced Research Projects Agency (DARPA) under Cooperative Agreement NCR-8919038 with the Corporation for National Research Initiatives, by AT&T Bell Laboratories, Hitachi Ltd., Hitachi America Ltd., the University of California under a MICRO grant, and the International Computer Science Institute. The views and conclusions contained in this document are those of the author, and should not be interpreted as representing official policies, either expressed or implied, of the U.S. Government or any of the sponsoring organizations.

Chapter 1: Problem Statement

1.1 Introduction

As computer communication evolves into the twenty-first century, high bandwidth low delay communication is becoming the norm, rather than the exception. This is leading to data networks being used by applications which previously had restricted themselves to specialized networks. These applications include digital television, digital audio and facsimile transmission, and in general multimedia (including continuous-media) applications. The quality of service (QOS) [Leiner 88] expected from the network by these applications varies over a wide range: some are sensitive to delays experienced in the communication network, others are sensitive to loss rates, while yet others are sensitive to delay variations. An integrated network, which aims to support all these services, must attempt to meet the needs of all the applications.

The network may attempt to *meet these diverse requirements* in one of *two ways*. It can try to provide a service that is (1) *good enough to meet the most stringent requirements*, or, it can attempt to utilize the information it has about the different requirements of the applications, and use this information to (2) provide *different qualities of service to different client applications*. The first approach, providing the best possible quality of service to all the applications, allows for simpler networks which can operate at high speeds, and becomes especially attrac-

tive if the requirements of most applications do not differ much from each other. However, the second approach will result in a better usage of network resources, and probably cheaper operation, if the diversity in requirements is high.

In its most general form, the second approach would ask the client to specify its performance requirements by means of a number of parameters and attempt to use this information. By specifying the values of the parameters, the client is allowed to choose a particular QOS. Let us call a value assignment of these parameters a *selection*. If some of the parameters can have any numerical value, the total number of possible selections can be infinite. The first approach is a case in which only one selection is possible, the one being offered by the network.The choice between the two approaches depends upon technology and the diversity in the requirements of the applications. Intermediate approaches, which offer a somewhat restricted menu of services, can also be used.

Assuming that a choice for the menu of the possible performance guarantees being offered by the network has been made, two types of resource management schemes can be used to support this menu: a *reactive* scheme or a *predictive* scheme. In a reactive solution, the state of the network is monitored continuously, and, if an undesirable situation (one in which the performance guarantees can be violated) is detected or expected to occur in the near future, suitable corrective measures (such as requesting some applications to reduce their traffic) are taken. In a predictive solution, applications are only allowed to use the network when they are considered safe.[1] The applications ask the network to

1. By “safe” we mean that the QOS guarantees made to applications already using the network are not violated. Of course, the network may decide to make it “safe” for some high-priority application by refusing to service some existing low-priority applications.

reserve appropriate resources, and the network allows them to communicate only if sufficient resources are available. Predictive solutions assume that applications can predict (to a certain extent) their traffic characteristics and performance requirements.

In high-speed networks where propagation delays dominate other components of the total delay, reaction times can be prohibitively large, and reactive solutions become less attractive. Furthermore, reactive solutions have been known to cause oscillations in network performance indices over time [Zhang 89]. Thus, predictive solutions seem better in high-speed wide-area networks.

The general goal of this thesis is to study the problem of predictive resource allocation for providing performance guarantees in future networks. In the thesis, we identify the different sub-areas of the problem, explore the issues that arise in each of the sub-areas, and provide a scheme that can be used to address the relevant issues in that sub-area. In the next sections, we shall define the problem in more precise terms and explore the different solutions proposed in the literature.

A study of some reactive schemes for high-speed networks can be found in a companion thesis by a fellow-member of Tenet research group at Berkeley [Keshav 91].

1.2 Problem Definition

The problem definition consists of two parts. In the first part, we explain the terms and concepts we will be using in this thesis. In the second part, we describe

the assumptions we make regarding the communication network.

1.2.1 Terminology

There are two principal types of entities assumed in this study: a *network entity* and *client entities*. The client entities are of two kinds: namely, *sending* entities and *receiving* entities. Either of the two entities in a pair can “hire” the network entity to carry information from the sending entity to the receiving entity. Thus, an explicit *contract* is signed between the communicating entities and the network. This contract specifies the kind of traffic that will be accepted by the network for this pair of sending and receiving entities, and the quality of service the pair of clients will receive.

The model is connection-oriented, i.e., an explicit connection between the sender and the receiver is established before the actual transfer of information can begin. The connection-oriented communication paradigm has been accepted as the standard for Broadband Integrated Services Digital Networks (BISDN) [CCITT I.128]. Therefore, in this work, we will be concentrating on providing quality of service to connection-oriented traffic only. We assume that data is transferred in fixed size units called *packets* or *cells*. We shall refer to a connection with guaranteed performance as a *channel*.

We shall be treating the contract as if it is a legaly binding agreement that has to be adhered to by both the client and the network entities. We believe that such a contractual service would need to be provided by a BISDN eventually, when it shall be catering to a large number of (yet unforseen) applications.

The contract has two major components, i.e., a specification of the traffic characteristics, and a description of the performance requirements. These client requirements can be specified in a number of ways [Ferrari 90b]. However, the most interesting ones from a client's viewpoint appear to be the bounds on the throughput, delay, delay variation and packet loss rate of a connection. Depending on the state of technology, and the traffic requirements, different QOS menus can be offered to the clients.

For the purposes of this study, we will be assuming that the network is based on the asynchronous transfer mode(ATM), and will consider a simplified model of such a network. In the next subsection, we describe the network model in more detail.

1.2.2 Network environment

In an ATM network, all information to be transferred from the source to the destination is packed into fixed size cells, which are identified and switched by means of a label in the cell header. The term *asynchronous* refers to the fact that cells allocated to the same connection may exhibit an irregular recurrence pattern, as cells are filled according to the actual demand. All communication in an ATM network is *connection-oriented*, i.e., a connection needs to be established before data transmission can begin. While connections can be duplex in general, we will confine our attention in this thesis to simplex (one-way) connections. A duplex connection can be obtained by creating two simplex connections in opposite directions. The asynchronous transfer mode is a more flexible alternative to the *synchronous* transfer mode, in which cells belonging to a connection could

only occupy certain predetermined time-slots on a transmission link. The difference between the two modes is illustrated by Figure 1-1, which shows some possible patterns of slot usage in the two modes.

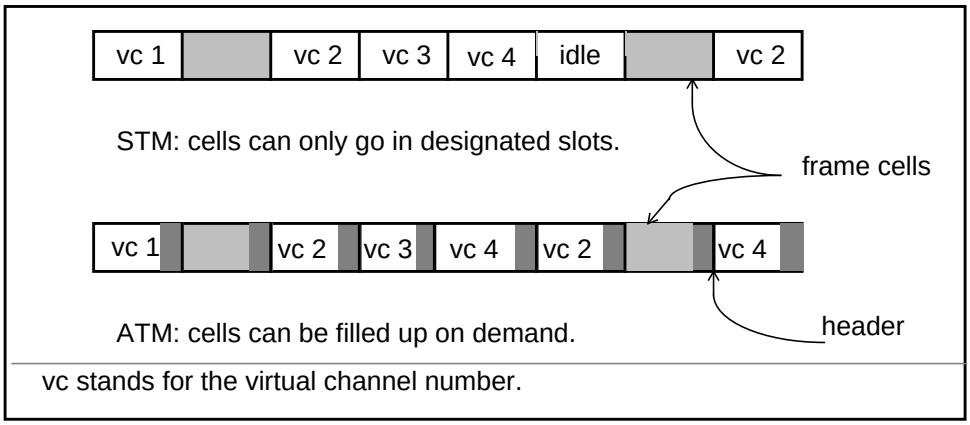

Figure 1-1:This figure illustrates the difference between the synchronous and the asynchronous transfer modes. Four connections are assumed to exist. In STM, time is divided into frames (marked by the shaded cells in the figure), and each connection gets one or more slots in a frame to transfer its cells. Thus connection 2 can only transfer cells in the first slot of a frame, connection 3 in the second slot, connection 4 in the 3rd slot and connection 1 in the fourth slot. In the central frame shown at the top of the figure, the slot for connection 1 is wasted because there is no cell to use it. In ATM, cells are allowed to use the slots as soon as they are available, and so the cell belonging to connection 2 can go ahead in the empty slot.

An ATM network consists of a number of ATM *switches* connected in an arbitrary mesh. We assume that each switch is accompanied by a *switch manager*, which is responsible for accepting or rejecting channels through the switch. The switch manager should be running the algorithms required to establish connections and perform any admission control tests required to ensure the quality of service needed by individual connections. When a channel is established, the switching fabric maps cells arriving along that channel from the input link to the appropriate output link. We are not concerned here with the exact details of this mapping mechanism. Any of the common switching techniques may be used for

the mapping operation. These techniques include shared-memory switching [Condruse 87], shared medium switching [Gopal 87] or space division switching fabrics (like banyan switches) [Hui 87]. A survey of switching schemes can be found in [Ahmadi 89] or [Tobagi 90]. In this thesis, we make no assumptions regarding the switching scheme used, as long as the switch model proposed below is valid.

An ATM switch can be modeled as a set of input queues terminating the incoming links, connected through an interconnection network to a set of output queues (outgoing links). Usually the only purpose of the input queues is cell/bit synchronization and clock recovery; so the input queue size is small. Nevertheless, it should be noted that statistical switching techniques will necessitate large input queue sizes in some cases. The output queues are needed since cells from different incoming links that are to be routed towards the same outgoing link may arrive concurrently. Each switch can be represented as in Figure 1-2.

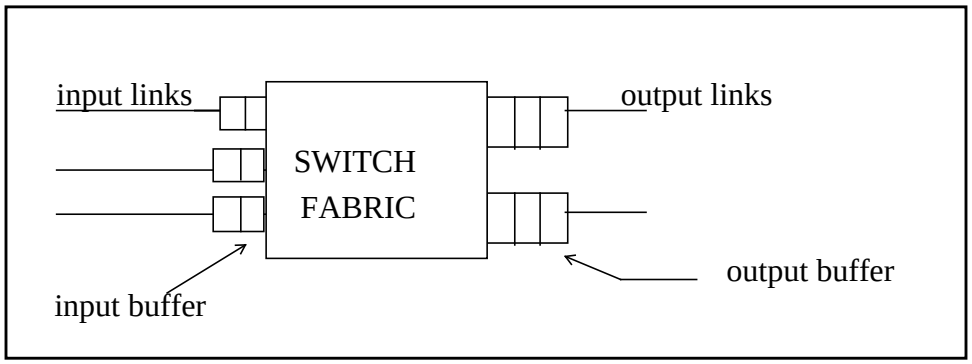

Figure 1-2: Structure of a typical ATM switch.

A description of the different switch architectures and a justification of the above model have been provided by Karol [Karol 87]. For ease of presentation, we will assume that (a) the ATM switch is internally non-blocking, (b) cells arriving

on different input links and destined for different output links do not interfere with each other, (c) input queues are non-existent, and (d) framing details can be ignored. Thus, we are assuming a *perfect switch* in our thesis, which will allow us to concentrate on one output link and devise a scheme to bound the queueing delay at that output link.

An ATM network consists of a number of switches connected in an arbitrary topology. The ATM clients are outside the network and may request channels from the network. The network deals with routing and resource allocation issues.The general structure of the network is shown in Figure 1-3.

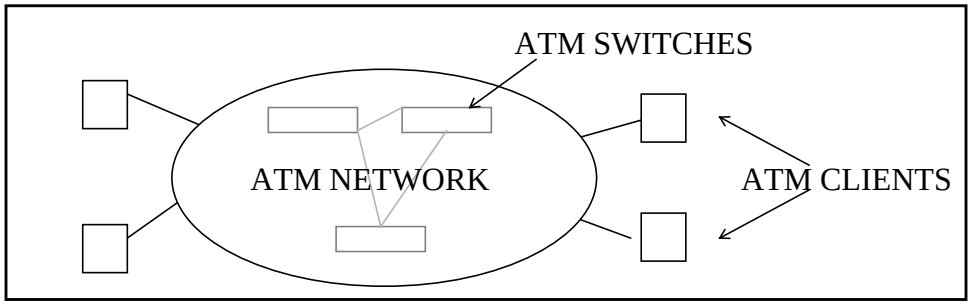

Figure 1-3: An ATM network consists of a number of switches connected in an arbitrary topology. The ATM clients are outside the network and may request channels from the network. The network deals with routing and resource allocation issues.

Each channel in the above environment can be modeled as passing through a number of queueing servers, each server modeling the output link of an ATM switch. Let us take a look at the possible cause of delays, delay variations, and cell loss in the network described above.

The delay in the network can be seen to consist of three components:(1)

the *propagation delay*, which depends on the distance between the communicating clients, is independent of the type of switching node used and, given the route of the connection, is difficult to reduce; (2) the *switching delay,* which depends on the implementation scheme, but is usually very low [Yeh 87] and (3) a *queueing delay* at each queueing server. The queueing delay can be controlled by restricting the number of connections through the switch or by using an appropriate queueing discipline. The queueing delay is the most important variable component of overall delay and the most significant component responsible for delay variations.

We are not considering the delays involved in fragmentation of data to be transmitted into cells or reassembly of this data from the cells. The focus of this thesis is only on performance (delays, delay variations and loss rates) within the network.

The cells that are lost in an ATM network can be divided into three categories depending on the cause for cell loss. The three possible causes of cell losses are: *errors in transmission*; *insufficient buffers* at an ATM switch; and *lateness* of cell arrival *at the destination*. Since errors in transmission cannot be controlled, and the cells lost because of delays can be limited in number by bounding delays, we restrict our attention to guaranteeing a bound on the rate of the losses due to buffer overflows in switches.

Thus, the problem of guaranteed performance communication can be solved if we can design a scheme that bounds the queueing delays and the number of cells lost due to buffer overflows in the switches. A brief description of the

work previously done in this area is given below.

1.3 Previous Work

Because of the long propagation delays involved in BISDN networks, predictive solutions, which reserve resources at connection establishment, are most suitable. A number of solutions have been proposed that can be used to provide these QOS guarantees. The list given below, while by no means exhaustive, covers most of the important ideas that have been proposed so far. Some of the work has been done in the realm of conventional packet-switching networks, but we have included it since it can be easily adapted to the ATM world as well.

1.3.1 Virtual clock

The virtual clock paradigm, proposed by Lixia Zhang [Zhang 89], attempts to reserve an average bandwidth for connections in a packet switching environment. For each connection, a *virtual clock* counter is maintained which progresses so as to allow one packet to be transmitted by the switch at every virtual clock tick. Any ticks not used to send a packet are wasted; thus a connection cannot remain silent for long periods and attempt to use a large amount of bandwidth subsequently. Channels with different bandwidths have their virtual clocks ticking at different rates. However, the solution does not provide for traffic with throughput requirements that can vary with time, and does not deal with any performance parameters except for throughput. Although *delay bounds* could be provided, they would generally *tend to be rather loose*.[2]

2. A slight modification of the virtual clock scheme leads to the scheme of fair queueing [Demers 89].

1.3.2 Stop-and-go queueing

Stop-and-go queueing [Golestani 90] deals with essentially smooth traffic, that is traffic restricted to sending only a fixed number of packets in a network wide fixed interval of time called a frame[3]. At each node in the network, packets arriving during a frame are stored (*stopped*) until the start of the next frame and sent (allowed to *go*) whenever the next frame starts. Connections requiring different bandwidths declare different number of packets to be sent in a frame. At each node, the frame duration serves as an upper bound on the delay that can be experienced by the packet in that node. The scheme can be extended to one with multiple frames in the same network.

Stop and go queueing has the nice property that the end-to-end delay variation of a packet is never more than the frame size a connection chooses. However, a connection requiring a smaller delay or delay variation will have to be allocated higher bandwidth. This *relationship between bandwidth and delay* makes the scheme unattractive for low-delay low-bandwidth applications such as sound. The same frame durations must be used in the entire network; so the frame durations cannot be changed. It is not obvious how to make a good choice of frame durations.

1.3.3 Hierarchical round robin

Hierarchical round robin (HRR) [Kalmanek 90] is a hybrid between fair queueing and stop-and-go queueing. Like stop-and-go queueing, it has a number of frames with different durations which serve two purposes: (1) rate allocation

3. The frame in Stop-and-go queueing differs somewhat from the frame in Section 1.2.1

and (2) delay bounding. Connections at each level (i.e., with the same frame duration) are served by a round robin scheduler. Although the scheme suffers from the *same* disadvantages *as stop-and-go queueing*, the multi-level implementation is claimed to be simpler.

1.3.4 The DASH resource model

The DASH project [Anderson 90] attempts to provide performance guarantees in a distributed operating system by resource reservation (assuming the network can support such guarantees), and can be extended naturally to do the same in a network. The characterization of traffic adopted by the DASH approach can be used to provide for limited amounts of burstiness, but the network allocates only the average bandwidth in a fashion similar to the virtual clock mechanism. Delay guarantees are also made which take into account the burstiness of the incoming traffic streams.

The DASH resource model can be considered as a step in the right direction, but *fails to address many important issues* such as the variation in the traffic's burstiness from node to node, the accumulation of delay variation, and the provision of loss-rate guarantees.

1.3.5 Real-time channels

A real-time channel, as defined in [Ferrari 90], is a simplex connection between a sender and a receiver. It has traffic and performance attributes associated with it. The traffic characterization consists of a minimum and average inter-packet interval, while the performance attributes may specify a bound on end-to-end delays[4] [Ferrari 90a], a bound on end-to-end delay variation

[Verma 91], or a bound on the maximum packet loss rate [Ferrari 90c]. To the best of our knowledge, real-time channel is the only abstraction which provides all the important performance parameters, accounts for the change in the traffic characteristics from node to node, and does not involve fixing any network-wide frames.

The only drawback of real-time channels (in their original formulation) is that they require a deadline-based scheduling policy in the switches. It is not clear if this deadline based scheduling policy can be implemented at high speeds in practical networks. *This thesis will attempt to provide guarantees similar to those offered by real-time channels*, extend upon the real-time channel work, and explore the guarantees that can be provided in the absence of a complex deadline-based scheduler.

1.3.6 The class related rule

The class related rule solution [Gallassi 89] assumes that traffic can be specified or categorized into a number of homogeneous traffic types, connections of each type having similar bandwidth and burstiness characteristics. The maximum number of connections that can be supported in this manner for a given quality of service can be determined by simulation (or analysis in some cases). In order to deal with heterogeneous traffic [Gallassi 90], an interpolation method is used which can determine when it is safe to accept a new connection.

The approach suffers from the problems involved in *determining the interesting classes of service* and selecting the appropriate *heuristic to interpolate*

---

4. The delay bounds may be deterministic or statistical. In the latter case, the delay bound is met with a certain probability greater than a given value.

between different classes.

1.3.7 Other works

There are a number of other papers that have addressed the problem of resource allocation in ATM networks. A solution similar to the class related rule uses the concept of virtual bandwidth [Akhtar 87]. Resource allocation at multiple levels, namely the connection level, the burst level, and the cell level have also been proposed [Hui 88]. A simple admission control rule for admitting bursty traffic is mentioned in [Esaki 90] and [Todorova 90b]. In the realm of packet-switching networks, the concepts of flows [Comer 89] and the Magnet-II network architecture proposed in [Lazar 90] deserve to be mentioned. However, none of these works address the problem of verifying the contract made at the channel establishment time, nor do they propose a way to take into account the changes in the burstiness of traffic within the network. A complete solution to the problem of resource allocation must address these issues as well.

1.4 The Sub-areas of the Problem

The sequence of operations that are to be carried out by the client and network entities in order to obtain guaranteed performance communication can be broken down into the following sub-areas: (1) *signing of a contract* between the client and the network; (2) *translating* the client-network contract into a contract for each node along the path; (3) developing a scheme to *enforce* the respective portions of the contract by the client and the network; and (4) developing a scheme that the client or the network can use to *verify* if its counterpart is obeying the contract. Within each category, we can break the problem down into three dis-

tinct components, a component dealing with conventional network communication, a component describing the obligations of the communicating clients, and a component describing the obligations of the network. In the next sub-sections, we describe each of the four sub-areas, and finally present a summary of our approach to the problem in Table 1-1.

1.4.1 Specification of contract

Before the actual communication begins, the network entity and the client entity *sign a contract*. The contract has three major components:

(C1) *Destination*: The client entity declares which of the other client entities it wishes to communicate with.

(C2) *Traffic Specification*: This part of the contract states the characteristics of the traffic that will be carried on the channel to be established. The sending client entity has the onus of ensuring that this part of the contract is kept.

(C3) *Performance Specification*: This part of the contract states the network performance desired by the client entity for the channel being established. It is the onus of the network entity to ensure that this part of the contract is kept for all established channels.

1.4.2 Mapping the contract to the network

The contract signed between the client entity and the network entity is global or end-to-end in the sense that it does not depend on the topology or properties of the actual network. The contract is global, since the client is only worried

about getting data across to its peer at the other end of the network, is making promises about his data traffic only at the client-network interface, and is only interested in the end-to-end performance that it will obtain. In order to provide guaranteed performance communication, we need to map the client-network contract into network-dependent node-specific contracts. For the three different areas of the contract, the following operations have to be performed.

(C1) The client-network contract only specifies the end-points of the communication. The mapping process involves *selecting a route* through the network. Thus the “global” contract, which requires carrying data between the two end-points, is mapped into a series of “local” contracts for carrying data on a particular link along a selected route or path.

(C2) The client-network contract only states the traffic characteristics at the point where traffic enters the network. At other points in the network, for example at a node downstream along the path of a channel, the *traffic characteristics may be different* from the original traffic characteristics. The mapping operation has to account for these changes, either by devising a new characterization or by taking actions that restore the original traffic characteristics at all nodes along the path of a connection.

(C3) The client-network contract only states the *end-to-end performance requirements* of the client. This contractual obligation is *mapped into per-node* performance requirements along the path.

Notice that this mapping operation does not require any interaction between the client and the network. Thus, the network entity may decide to

change this mapping anytime as long as it is able to meet its obligations towards the clients.

1.4.3 Mechanism to satisfy contract

Both the client and the network need a mechanism ensuring that they can meet the obligations of the contract they have signed. An algorithm is needed to ensure that all the three areas of the contract are adhered to.

(C1) A scheme is required which ensures that all data from the sending client is addressed properly to the destination and is handled appropriately by all the nodes along the path.

(C2) The client needs to have a *regulatory algorithm* which ensures that it does not send traffic into the network at a rate higher than what it promised in the contract.

(C3) The node needs to have a scheme which allows it to provide the performance guarantees it committed to when the client-network contract was made. The scheme would consist of at least two components:

I. A set of *admission control* rules, which enable any node in the network to decide whether it can accept and meet the requirements of a new contract.

II. A *scheduling* mechanism, which enables each node in the network to meet the required performance bounds of all the clients whose contracts have been accepted.

1.4.4 Verification of the contract

In a general network environment, the client and the network entities may not trust one another. Thus, we also need the following algorithms:

(C1) A scheme that the sending client may use to verify that the receiver is indeed receiving the data transmitted by the sender. Similarly, the network needs a scheme to verify that the client is not trying to clandestinely send to some destination not mentioned in the contract. [5]

(C2) A scheme that the network may use to verify that the client is not cheating on the traffic contract and sending data into the network at rates higher than what it promised.

(C3) A scheme that the sending and receiving clients may use to verify that the network is actually providing the performance that was agreed upon in the contract.

The reader will have noticed that these four sub-areas contain problems that need to be solved for the three components of the contract. Component (C1) corresponds to the problem of providing a connection oriented service in a general network. The problems in this area can be solved in a number of ways (for example using the abstraction of virtual circuits [Tanenbaum 88]). In this thesis, we assume that a solution to this problem already exists and will not discuss it any further.

---

5. In non real-time communication (that is, communication that does not need performance guarantees), an explicit contract is not required. In these cases, we assume that an implicit contract with very weak conditions has been signed and is being used for such communication.

Component (C2) deals with the problem of traffic specification, while component (C3) deals with the problem of performance guarantees. This thesis will attempt to provide a solution to the problems in all the four sub-areas [6] in these two components of the contract. We first look at the issues in traffic specification and then deal with analogous problems for performance guarantees.

1.5 Thesis Organization

Having identified the different sub-areas in the resource allocation problem, we now present the organization of this thesis, and how we address the problems in each of the sub-areas.

The thesis consists of 5 chapters. The first chapter introduces the problem and the last one summarizes the results of our investigation. Chapters 2-4 address the different areas that we have sketched in Section 1.4. Our approach to the problem of predictive resource allocation and the different chapters of the thesis that address specific sub-areas are shown in Table 1-1.

Chapter 1 (this chapter) introduces the problem, specifies the network environment we assume and explores related work in the area. It also presents an overview of the different sub-areas that need to be explored, and describes the organization of the thesis.

Chapter 2 discusses various traffic specification models that can be used for predictive resource allocation. These models must allow us to specify a wide variety of traffic types, yet be simple enough to allow the detection of any cheating

6. The four sub-areas are the ones outlined in Sections 1.4.1-1.4.4.

Table 1-1.Sub-areas in the problem of predictive resource allocation

	(C1) **Conventional Contract**	(C2) **Traffic Contract**	(C3) **Performance Contract**
(1) Specification	Addressing Issues **(solutions exist)**	Which traffic model to use? **Chapter 2**	Which menu of QOS should be provided? **Chapter 4**
(2) Mechanism	Communication Abstraction **(solutions exist)**	How to make traffic conform to the model? **Chapter 2**	How can the network provide required QOS? **Chapter 4**
(3) Mapping	Routing Issues. **(solutions exist)**	How do traffic parameters change in network? **Chapter 3**	How to map end-to-end performance to per-node performance? **Chapter 4**
(4) Verification	Authentication, Acknowledgments **(solutions exist)**	How to verify whether client is obeying the model? **Chapter 3**	How to verify whether network has the promised performance? **Chapter 4**

or malfunctioning on the part of the network's clients. Moreover, simpler traffic specifications are easier to analyze, and enable us to (more easily) determine the resource usage of a connection in advance. In this chapter, we survey several alternative models that can be used for traffic specification, narrow the choice down to two models, and compare these two models. For each of these two models, we also present an algorithm which will allow a client entity to adhere to its part of the contract.

Chapter 3 discusses the mapping operation in the area of traffic specification. In general, traffic on a connection may change its characteristics due to network fluctuations along the path of a connection. We determine whether this

change in traffic characteristics is significant, and propose a solution that allows us to preserve the input traffic specification at all nodes along the network. A scheme that can be used by the network entity to verify whether the client is adhering to its part of the traffic contract is also proposed.

Chapter 4 discusses the problem of performance specification. A number of possible QOS menus that can be offered by the network under different circumstances are presented. We determine which of these menus make sense under different network and application assumptions. We examine three different menus of QOS services that can be offered by the network. We also analyze the different schemes that can be used to map the end-to-end performance obligations into per-node performance obligations. Finally, scheduling policies and admission control criteria that are best suited for each QOS menu are proposed and their performance studied by means of analysis and simulation.

Chapter 5 discusses the strengths and weaknesses of out approach. In this chapter, we summarize our conclusions from this investigation, and explore avenues for further research. One area which we have not explored fully in our work is the verification of the performance contract by the client (see Table 1-1). Chapter 5 presents a summary of the different possible approaches that might be taken in this field.

Chapter 2: The Traffic Contract-I: Specification and Mechanism

2.1 Introduction

The notion of traffic contract was introduced in Chapter 1. In this chapter, we look at the description of the traffic by the client, examine the different issues involved in traffic specification, and explore the different possible solutions to this problem.

This chapter is organized as follows. In the next section (Section 2.2), we look at the desirable properties of a good traffic specification model. In Section 2.3, we survey some of the different traffic models that can be used by the clients to characterize their traffic. Having selected the appropriate traffic model, each client needs to determine the parameters of the model that best describes its traffic. Section 2.4 develops an algorithm that can help clients to choose these parameters. The next section, Section 2.5 describes the design of a leaky-bucket that the clients can use to regulate the flow of data into the network (this is the mechanism for the traffic contract). Finally, Section 2.6 develops a method which can be used to compare the different traffic models to be used for guaranteed performance communication. In Section 2.7, this method is applied to two of the models described in Section 2.3, and the choice of a traffic model for the thesis is made.

2.2 Issues in Traffic Specification

Since we are adopting a predictive solution to the QOS problem (see Section 1.1), the specification must describe the traffic as accurately as possible. Moreover, the resources allocated to different channels and the performance guarantees made to them would depend on the specifications declared by all the clients using the network. Some of the clients may be cheating, and can potentially jeopardize the performance guarantees made to other clients. Thus any traffic specification should have the following desirable properties:

(1) *Representativity:* the specification must be representative of the traffic. Thus, it should represent reasonably well the behavior of the traffic over an extended period of time. If the traffic specification is not representative, it can lead us to one of two undesirable situations: either we reserve more resources than the channel can possibly use, or the resources allocated are not sufficient to provide the performance required by the client. The latter is, of course, worse than the former.

(2) *Verifiability:* the network must be able to verify whether a client is obeying its promised traffic specification or not. This is essential because a misbehaving client can potentially ruin the performance of the network for everyone. The network must be able to detect violations and take appropriate measures in these situations.

(3) *Preservability:* the traffic characteristics of any channel may change as we move along its path. Thus, the amount of resources allocated to a channel may be different at different nodes along the path. The network must be able to

either preserve the same traffic characteristics from node to node or account for this change in traffic characteristics.

(4) *Usability:* the specification must be usable in admission control tests and resource allocation computations without a large overhead. A description which satisfies all the other criteria mentioned in this section, but does not allow simple resource allocation schemes, is useless for our purposes. Usability also requires that clients be able to easily state their traffic requirements by using the specification.

(5) *Efficiency*: For a given set of channels, the use of different traffic specification models can result in predicting different bounds for their performance. Given two traffic specifications, we would prefer to use one which allows a larger number of channels to be established, while still satisfying the same set of performance requirements. As a corollary, the specification should allow for *statistical multiplexing*, if permitted by the performance requirements of the clients.

Theoretically, a full characterization of the traffic could be given by the client by specifying the exact times (measured from some reference point like the time the channel will be established) of cell arrivals on its channel. Equivalently, the client could specify the inter arrival times between consecutive cells. This would imply specifying a series of variables $X_1,... X_i... X_N$, where N is the total number of cells transmitted on the channel, X_i (for $i > 1$) is the interval between the i^{th} cell and the $i\text{-}1^{th}$ cell, and X_1 is the interval between the channel establishment time and the sending of the first cell. In a long-lasting channel, N can be a very large number, and a full specification, while representative, verifiable and

preservable, is not usable. This is so because many applications will not be able to predict the exact inter arrival times for the exact sequence of cells they will be generating, and, even if they could, specifying the whole stream would be impractical.Thus, we need a more concise description of the traffic, one that could be used by the clients.

A concise description may allow us to predict some global properties of the *X*-series, for instance, a bound on its mean and variance. At the same time, it may specify the correlation between consecutive members of the *X*-series as well. In the next section, we look at some simple models that have been proposed in the literature and examine their suitability with respect to the criteria mentioned above.

2.3 Different Traffic Models

In this section, we will look at some of the common traffic specification models that are found in the literature, and evaluate how they meet the criteria mentioned in Section 2.2.

2.3.1 Probabilistic models

A large number of papers, especially among those taking a queueing analysis approach, assume that the traffic pattern can be described by means of a probability distribution function. In other words, the cell inter arrival times are independent, identically distributed random variables drawn from some probability distribution. The simplest and most common of such distributions is the Poisson distribution, which has a number of “nice” properties that permit closed-form solu-

tions to many problems. However, the Poisson model may not be very representative of traffic on a single channel [Gusella 90].

Other, more sophisticated models, based on the theory of Markov chains or of renewal processes [Wolff 90] can also be used. The more sophisticated models are somewhat more difficult to analyze, but in general can better approximate real-world situations. An example of this approach can be found in [Ferrandiz 91]. With a sufficiently complex model, one can mimic the real traffic pattern of any client sufficiently well.

Among the criteria listed in Section 2.2, these models satisfy those of efficiency, representativity and usability. However, they do not seem to satisfy the criteria of verifiability and preservability; that is, we are not aware of an easy way for a node to verify whether the clients are actually obeying their promises. Since some of the clients may be misbehaving, either with malicious intentions, or because of a failure in their equipment, these models cannot be used in our approach, where the guarantees to a law-abiding client must be maintained even if other clients are not fulfilling their promises.

2.3.2 Average rate model

The average rate model [Golestani 90] [1]specifies a bound on the average bandwidth required by a channel over an averaging interval. This could be looked upon as a window-based mechanism, where at most *N* cells can be sent in an averaging interval of length *I*. Alternatively, we can look upon it as a rate-based

1. This model is also called the *(r,T)* model, *r* being the rate of a connection over a time period of length *T*. Our notation with different symbols is intended to show the relationship between this model and the other models introduced later.

specification, when the average spacing between any two cells is at least a certain value x_{ave} (= *I*/*N*) in every interval of length *I*. Thus, this model provides a bound on the average throughput required by the client over some averaging interval.

The average-rate model is verifiable, preservable and usable. It is able to represent certain kinds of traffic very well. An example would be uncompressed digital video where *I* could be the time to generate a frame of video data, and where roughly the same number of cells are generated every time-period of length *I*. Similarly for digital audio. However, for applications which do not have a similar characteristic period, the number of cells sent in every period of length *I* may have a large variation. In these cases, the model would have to pick the maximum of these numbers, and this may result in inefficiencies. Thus, this model would not be representative of traffic with a highly variable bit-rate, and must be augmented by models which are representative for traffic patterns with variable rates or bursts.

2.3.3 Linear bounded model

The linear bounded model [Cruz 87] specifies that the number of cells transmitted during any time interval on a channel be bounded by a linear function of the length of the time interval involved. In every time interval of length *T*, the number *N(T*) of cells on the channel must be bounded by

$$N(T) \leq \left\lfloor \frac{T}{x_{ave}} \right\rfloor + B, \qquad \textit{(2-1)}$$

where x_{ave} is the average spacing between cells and *B* is the burst-size.

An intuitive explanation of the model is that it assumes that at most *B* cells can be transmitted in rapid succession, while a long-term average rate of $1/x_{ave}$ is maintained.[2]

This model satisfies most of the criteria mentioned in Section 2.2. It is preservable, verifiable and usable. The model may be representative for continuous-media applications [Anderson 90], and we examine its efficiency in Section 2.6 of this chapter.

2.3.4 The x_{min}-x_{ave}-I model

This model [Ferrari 90] characterizes the traffic on a channel by means of two parameters. The minimum spacing between any two cells must be larger than or equal to a certain value x_{min} at all times (this corresponds to the peak rate), while the average cell spacing in any interval of length I must be larger than or equal to x_{ave}. In this model, burstiness may be expressed as the ratio of x_{min} to x_{ave}. This model can be seen as a generalization of the peak-rate average-rate model used for traffic characterization in ATM networks, for example in [Choi 89].

This model seems to satisfy most of the criteria in Section 2.2. It is simple, usable, preservable and efficient (in that it allows statistical multiplexing; see Chapter 4 for details). It can represent certain classes of applications very well, for example digital voice with long on and off periods. However, the simple formulation may not be able to characterize traffic with a very high degree of variability in a representative manner.

---

2. The actual specification by Cruz uses different terms, but is functionally equivalent to our description. We have attempted to use the same terms and symbols in the specification of all the three models we study in this chapter.

2.3.5 Sophisticated burstiness models

We now look at some models that can represent burstiness in a more sophisticated fashion. One such model is the characterization of a process by means of correlations between the inter arrival times at different lags. The whole process is characterized by means of these correlations, which are normalized to produce the indices of dispersion [Gusella 90].

Another such model looks at the maximum amount of buffering required to prevent any losses to the arrival process at a constant-rate-first-come-first-served (FCFS) server [Low 91]. In this scheme, the traffic is described by means of a curve that shows the maximum amount of buffer space required at different service rates for the FCFS server.

Since both these characterizations are based on continuous curves (which also means they can potentially be more representative of the traffic stream than the simpler models mentioned in previous sections) rather than a few discrete parameters, their specification and use are not very easy. Thus, they are not verifiable or maintainable, but if a predictive resource allocation method using these specifications can be obtained, such a method would likely be more efficient than the ones using simpler models.

Among the three models that meet most of the criteria mentioned in Section 2.2, the average rate model is a special case of the x_{min}-x_{ave}-I model (with the parameter x_{min} being zero, and x_{ave} being $\frac{I}{N}$). Therefore, we will discuss only the x_{min}-x_{ave}-I model and the linear bounded model in the rest of the chapter.

2.4 The Selection of Model Parameters

Given that we have chosen one of the simple models in the previous section as the one to be used for traffic specification, the client has to specify the parameters of the model to the network. However, if we select the parameters in a simple-minded manner, we may end up with values that are unduly influenced by outliers.

For example, let us consider a brief excerpt of a sample traffic trace. The following are the inter-cell spacings in milliseconds seen on a network link: 324, 10, 66, 7267, 102, 124. If we were using the x_{min}-x_{ave}-I model, we would be forced to set the value of x_{min} to the lowest possible value, i.e. 10 milliseconds, and the network will have to allocate enough resources to handle a cell every 10 milliseconds, even though most of the other inter-cell spacings are much longer. The same problem can be seen in determining the values of the averaging interval. We would like to specify a value of x_{min} (or x_{ave} and I) which is more representative of the traffic stream. How should these values be determined?

If we plot the histogram of inter-cell spacings of a channel, we may obtain a shape such as shown in Figure 2-1. In general, there may be some small probability of having an inter-cell spacing close to zero. Since a small inter-cell spacing may require reserving lot more resources in the network, we would like to impose a somewhat higher bound on the minimum inter-cell spacing. However, in this process, we have distorted the input inter-cell interval distribution, ending up with a new inter-cell interval distribution. Notice also that the choice of the model parameters does not uniquely determine the new inter-cell interval distribution. More than one distribution may have the same value of model parameters, but

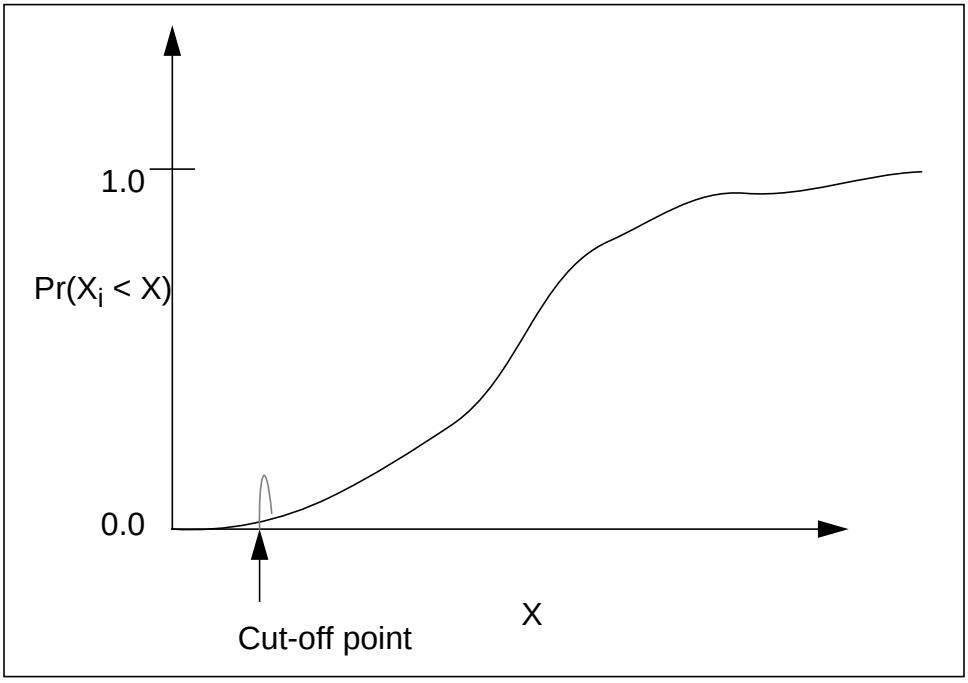

Figure 2-1 This figure shows the choice of a suitable parameter for a *unfriendly* traffic pattern. While the minimum inter-cell spacing in the original distribution is close to 0, one may decide to hold cells that come too close together and space them out by a certain amount. The cut-off point is chosen so that the probability of this holding is reasonably small. Notice that this process changes the probability distribution function (shown by the dotted curve in the figure). In effect, we are converting an old distribution of inter-cell intervals into a new distribution

the network would reserve the same amount of resources for all of them. Figure 2-2 shows two distributions that have the same value of x_{min} and x_{ave} parameters according to the x_{min}-x_{ave}-I model. Since they have the same parameters, the network will reserve the same amount of resources for them.

What would be the ideal new inter-cell interval distribution? From the network's point of view, the best input distribution would be the one that allows it to accept the maximum possible number of channels. This would be determined by the parameters that have been chosen by the clients to characterize their traffic.

In order to understand how the distribution may change, let us take a look

at Figure 2-3. The original (or unregulated) traffic stream (with possibly some cells with very small inter-cell spacing) is fed into a black-box (called the leaky bucket) which produces the “regulated” traffic conforming to the chosen model with different model parameters.[3] The distribution of inter-cell intervals on the regulated traffic stream is likely to be different from that on the unregulated traffic stream.

From the client’s point of view, the best output distribution would be the one that provides the client with the least delays or losses in the conversion process. Notice that there are many possible distributions that correspond to the same choice of the parameters, and that the client would obviously be interested in one that minimizes the delay in the process required to convert the old distribution to the new distribution.

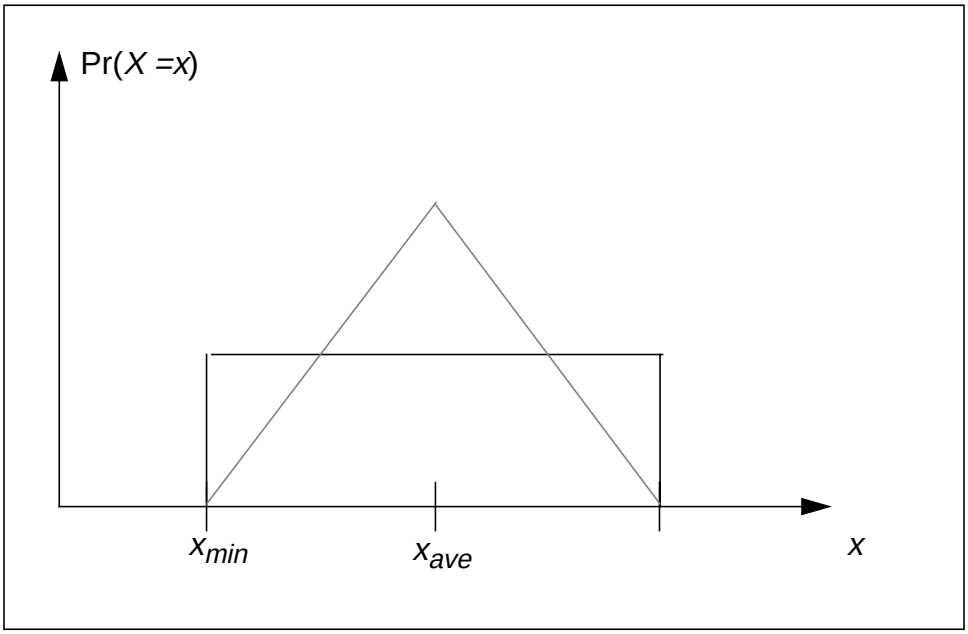

Figure 2-2 The same set of parameters can correspond to several possible distributions of the inter-cell intervals. The two curves shown in the figure have the same peak and average values but still differ considerably.

3. We refer to the original traffic stream as the *unregulated* traffic stream and the output of the leaky bucket as the *regulated* traffic stream.

There are some properties that must hold true for the conversion process. Since cells can only be sent out after they are received, the conversion process is restricted to holding cells. The holding of cell *i* will increase the spacing between the i^{th} and the $i\text{-}1^{th}$ cell, but will tend to decrease the spacing between the i^{th} and the $i+1^{th}$ cells. Furthermore, we will confine our attention to on-line conversion processes. This means that the holding time of the i^{th} cell will be determined only by the past inter arrival times (cells *1* to *i - 1),* and not by the inter arrival times of future cells.

Given certain values of parameters, and a history of inter-cell intervals, the

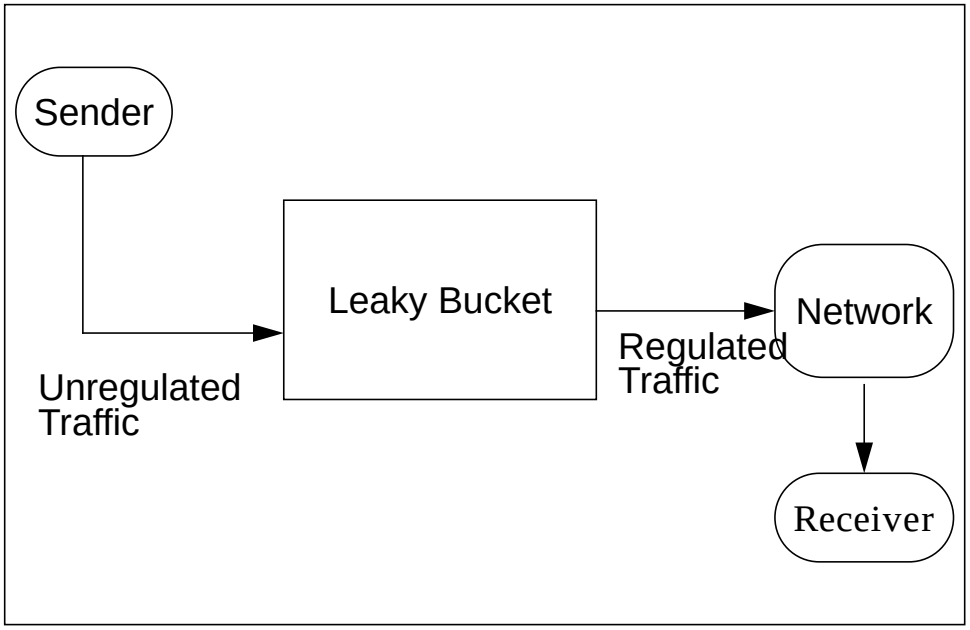

Figure 2-3 :Configuration of the leaky-bucket experiments.

traffic models impose a constraint on the time when the next cell can be sent. If cells have been sent out at inter-cell spacings of $X_1,\ldots X_n$, then the models enforce the rule that the next cell can only be sent out after an interval $X_{k+1} = f_{k+1}(X_1,\ldots X_k)$. The function f_{k+1} for the x_{min}-x_{ave}-*I* model is given by equa-

tion (2-2), and for the linear bounded model by equation (2-3).[4]

$$f_{k+1}(X_1, \ldots X_k) = \max\left[x_{min}, \left\{ I - \sum_{i=k}^{k-\left\lceil \frac{I}{x_{ave}} \right\rceil} x_i \right\} \right] \qquad (2\text{-}2)$$

$$f_{k+1}(X_1, \ldots X_k) = \max_j \left[(j+1-B_{max})\, x_{ave} - \sum_{i=k}^{i=k-j} x_i \right]^+ \qquad (2\text{-}3)$$

Both these functions[5] depend on the past history of the inter-cell intervals, through the sum of past inter arrival times. Furthermore, these functions decrease as the sum of the past inter arrival times increases.

A *greedy conversion process* is one that will transmit a cell as soon as the model parameters allow it to do so. Thus, a greedy algorithm converts the original inter-cell interval series $X_1 \ldots X_n$ into a new series $Y_1, \ldots Y_n$ which is defined by

$$Y_1 = X_1 \qquad (2\text{-}4)$$

$$Y_i = \max\left\{ f_i(Y_1, \ldots, Y_{i-1}), \left(\sum_{j=1}^{i} X_j - \sum_{j=1}^{i-1} Y_j \right) \right\} \qquad (2\text{-}5)$$

Equation (2-5) has two components, one making sure that the output stream $Y_1, \ldots Y_n$ obeys the model, and the other component ensuring that a cell is never transmitted before it is generated in the original inter-cell interval distribution.

We now prove:

---

4. For a proof of correctness of these equations, refer to Section 2.5 .
5. $x^+ = \max(x, 0)$.

Theorem 1: Given either of the two models described above and the values of its parameters, the greedy conversion algorithm minimizes the maximum delay experienced by any cell during an on-line conversion process.

Proof: Suppose a better on-line conversion algorithm exists which results in a lower delay for the n^{th} cell. Let the incoming cell stream be characterized by $X_1 \ldots X_n$. Let the greedy algorithm convert the input stream into the sequence $Y_1, \ldots Y_n$. Since we are looking at on-line algorithms, only these values are of interest. Let the new algorithm (supposed to be better than the greedy one) convert the input sequence into $Z_1, \ldots Z_n$ with a smaller maximum delay. Let k be the smallest index where the greedy algorithm's sequence and the new algorithm's sequence differ. In this case, we clearly will have $Z_k \geq Y_k$, since we cannot transfer a cell to the output earlier than allowed by the greedy algorithm. Now consider an algorithm which can transfer the cells according to the sequence $Y_1, \ldots . Y_k, Z_{k+1} + (Z_k - Y_k), \ldots Z_n$.

Let us see whether this sequence is allowed by our model (that is, we have to show that for the x_{min}-x_{ave}-I model no cell in a sequence is sent closer than the minimum permitted spacing and that the average spacing is also maintained; similarly for the linear bounded model). This can be done by verifying that equations (2-2) and (2-3) are valid for the new sequence. There is no problem with the first k inter-cell intervals. Consider the possibility of a violation of the model parameters for higher indices. Noticing that the two functions mentioned in equations (2-2) and (2-3) depend only on the sum of inter arrival times, we can end up in one of two cases, i.e., when the sum (that is the term $\sum Y_j$ in equation (2-5)) involves the k^{th} term and when it does not. When it does, the value of the right-hand-side

of the equation remains unchanged since we are increasing the next inter-cell interval by exactly the same amount that we decreased the k^{th} inter-cell interval. Thus, the new sequence is allowed by the model. Otherwise, we have increased the value of the sum, and both equations (2-2) and (2-3) allow us to transfer cells at a smaller spacing, so the new sequence is still allowed by our model. The process can be repeated for higher values of *k* till we obtain the same sequence as in the greedy algorithm, which will have a delay not larger than the optimal one.

By induction on *n*, we can show that a better on-line algorithm cannot exist for any finite *n,* and thus the greedy algorithm is the best one (in the sense that no algorithm better than the greedy one exists) for minimizing the delays in the conversion process. ∎[6]

2.5 Traffic Contract Mechanism: The Leaky Bucket

Given that we know the optimal on-line algorithm (in the sense that it minimizes the delays experienced by the cells of the original traffic stream) for converting any arbitrary traffic stream into a traffic stream obeying the x_{min}-x_{ave}-I model (or the linear bounded model), we now provide a detailed description of the optimal on-line algorithm for both of the models.

We shall refer to this operation as the *leaky-bucket conversion*, since it is very similar in principle to the concept of leaky bucket as used in the literature, for example [Turner 86]. We would like to draw the reader's attention to the fact that leaky bucket is the mechanism that the client can use to ensure that its traffic flow-

6. The theorem holds for both the models since both equations (2-2) and (2-3) depend only on the sum of inter arrival times. This allows us to decrease the inter arrival times of consecutive packets to the extent permitted by the greedy algorithm without violating the model parameters.

ing into the network conforms to the terms of the traffic contract.

2.5.1 Leaky bucket for the x_{min}-x_{ave}-I model

We first provide the algorithm to perform the leaky-bucket conversion for the x_{min}-x_{ave}-I model. According to this model, no two cells can be sent closer than the minimum spacing x_{min}, and the average rate over any interval of length I must be at least x_{ave}. Equivalently, any consecutive $\left\lceil \frac{I}{x_{ave}} \right\rceil$ cells must together span a total time interval no less than I:

$$(\forall k) \left(\sum_{i=k}^{k+\left\lceil \frac{I}{x_{ave}} \right\rceil} X_i \geq I \right) \qquad (2\text{-}6)$$

From equation (2-6) and the x_{min} constraint, it follows that we cannot send cells more frequently than at the intervals specified by equation (2-2).

If our original stream is $X_1, X_2, \ldots$, and we wish to convert it into a regulated stream $Y_1, Y_2, \ldots$(using the leaky bucket configuration described in Figure 2-3) which obeys the x_{min}-x_{ave}-I model for some value of these parameters, we can execute the algorithm described in Figure 2-4. The algorithm consists of two routines, `initialization` and `per_cell_operation`. The former is used to initialize the state of a newly created channel, while the latter is called whenever a new cell is received. The important term being computed by `per_cell_operation` is `cell_eligib_time`, the instant when the cell can be shipped safely. The history mechanism keeps the sum of the most recent $\left\lceil \frac{I}{x_{ave}} \right\rceil$ inter-cell intervals, and is needed to ensure that the average-rate guarantees are not violated at any time. The number of operations to be performed per cell is a constant.

```
int b = fllor(I/x_ave);
TIME history[b+1];
int hist_start;
int hist_end;
TIME hist_sum;
TIME y_clock, cell_eligib_time;

initialization()
{
   int i;


   for (i=0;i<=b;i++){
      history[i] = infinity;
   }
   hist_sum = b*infinity;
   hist_start = 0;
   hist_end = b;
   y_clock = clock;
}

per_cell_operation(clock)
{
   TIME y_i, this_time;

   this_time = max((y_clock - clock), 0.0);
   y_i = max(y_clock,clock) - prev_dep;

   hist_sum = y_i + hist_sum - history[hist_start];
   hist_start = (hist_start + 1)% (b+1);
   history[hist_end] = y_i;
   hist_end = (hist_end + 1)% (b+1);
   y_clock = max(clock,y_clock) + max(I-hist_sum,x_min);
   cell_eligib_time = clock + this_time;
}
```

Figure 2-4 The algorithm for converting a unregulated stream into a stream obeying the x_{min}-x_{ave}-I model. `per_cell_operation` is called whenever a new cell is received and it returns `pkt_eligib_time`, the instant when the cell can be shipped safely. Note that the conversion process assumes that the model parameters are given at initialization time. The history mechanism is needed to ensure that the average-rate guarantees are not violated at any time. The variable `y_clock` stores the smallest possible time when the next cell can be sent out, and is updated every time a new cell is received. The algorithm is presented for a single channel.

As far as the client is concerned, it may use any values of the parameters (i.e., any values of I, x_{ave} and x_{min}) that result in a low conversion delay. However, since the client might be charged for the resources it would be consuming, it would be better for the client to choose the smallest possible value of I and the largest possible values of x_{ave} and x_{min} that will not make conversion delays intolerably large. The delay actually involved in the conversion process will depend on the original traffic stream and on the value of the parameters chosen for the regulated process. Let us define *representative values* for the original traffic stream as those values chosen for the parameters of the x_{min}-x_{ave}-I model that result in a "small delay" in the greedy conversion process described by Figure 2-4. Let us also define [7] small delay as an average conversion delay less than the average inter-cell interval in the original stream.

Section 2.7 presents the representative values of the parameters of the x_{min}-x_{ave}-I model for a Markov Modulated Poisson Process (MMPP process). [8]

2.5.2 Leaky bucket for linear bounded model

To construct a leaky bucket for the linear bounded model, we can adopt a strategy similar to that adopted in the previous section. According to this model, the number of cells sent during an interval of time T is bounded by equation (2-1). Suppose that T is the time period between the arrival of the $k+1^{th}$ and the $k-j^{th}$ cells. In that case,

---

7. The motivation for choosing this definition of small delay is that, in this case, we would only expect one buffer to be used on the average for the leaky bucket algorithm.
8. The details of the MMPP process are described in that section.

$$T = \sum_{i=k-j}^{i=k+1} X_i, \qquad (2\text{-}7)$$

and, since *j+1* cells have arrived in this period, we see that condition (2-1) is equivalent to:

$$(\forall k)\ (\forall j)\left(j+1 \le \left\lfloor \frac{\sum_{i=k-j}^{i=k+1} X_i}{x_{ave}} \right\rfloor + B \right). \qquad (2\text{-}8)$$

A sufficient condition for this to hold always is

$$(\forall k)\ (\forall j)\left(j+1 \le B + \frac{\sum_{i=k-j}^{i=k+1} X_i}{x_{ave}} \right), \qquad (2\text{-}9)$$

where we have eliminated the floor operator. A simple transformation shows that this is equivalent to

$$(\forall k)\ (\forall j)\left((j+1-B)\, x_{ave} - \sum_{i=k-j}^{i=k} X_i \le X_{k+1} \right). \qquad (2\text{-}10)$$

Equation (2-10) provides a lower bound on the value of X_{k+1}. This leads to equation (2-3), and we can devise a leaky bucket similar to the one described in the previous section. This leaky-bucket algorithm is shown in Figure 2-5. The algorithm is structured in the same manner as that of Figure 2-4, with a routine for initialization and a routine to be invoked on the arrival of every cell. However, the reader will notice that, in this algorithm, the value of the lower bound on the time of the next expected cell depends on the entire history of the cell stream (because

```
TIME history[];
int count;

initialization()
{
   y_clock = clock;
}

per_cell_operation(i,x_i)
int i;
TIME x_i;
{
   this_time = max(y_clock - clock, 0.0);
   y_i = max(y_clock,clock) - prev_dep;
   history[i] = y_i;

   max_val = sum = 0.0;
   for(j=0;j<i; j++) {
      max_val = max(max_val, (j + 1 -b) * x_ave - sum);
      sum += history[i - j];
   }
   history[i] = y_clock - prev_dep;
   prev_dep = y_clock;
   y_clock = max(y_clock, clock) + max_val;
   pkt_eligib_time = clock + this_time;
}
```

Figure 2-5 : The algorithm for converting an unregulated stream into a stream obeying the linear bounded model. `per_cell_operation` is called whenever a new cell is received and it returns `pkt_eligib_time`, the instant when the cell can be shipped safely. The conversion process assumes that the model parameters are given at initialization time. The whole history of the stream is needed to ensure that the average-rate guarantees are not violated at any time. The variable `y_clock` stores the smallest possible time when the next cell can be shipped out, and is updated every time a new cell is received

the lower bound on X_{k+1} in this model depends on the maximum value of equation (2-10) for all $j < k$, which requires keeping the summation of inter arrival times for all values of j), and it may not be economically possible to store the entire history

of long-lasting cell streams.

Thus, we must devise a separate leaky-bucket mechanism for this traffic model. We may decrease the complexity in implementation by using slightly higher bandwidth for the traffic, and use a credit-based scheme to regulate the traffic according to this model.

In the credit-based scheme, we start with a certain number of credits in a credit pool (see Figure 2-6). The client is permitted to send cells whenever there

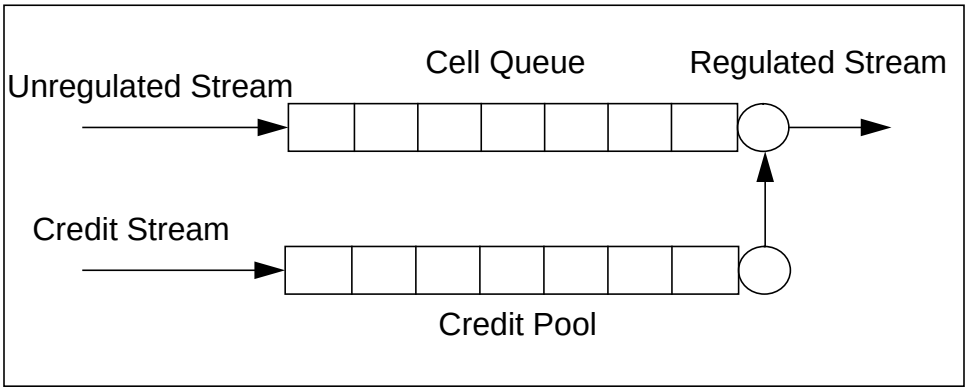

Figure 2-6 : The modified leaky bucket for the linear bounded model. In this leaky bucket scheme, a number of credits are kept in a credit pool. A cell may be transferred if there is a credit present in the credit pool. The transfer of a cell destroys one credit. Credits are generated at regular intervals, but the total number of credits in the queue cannot exceed the burst-size parameter of the linear bounded model.

is a credit in the pool. Credits increase at an average rate of one per x_{ave} time units, but at most B credits are allowed to be present in the pool.

If we use this credit-based scheme, the average rate of credit generation has to be slightly higher than the long-term average rate of the original traffic stream. However, the scheme does not require to store all the history, and thus can be used in practical networks. We can define representative values of x_{ave}

and *B* for the original traffic stream in a manner analogous to that in Section 2.5.1.

Section 2.7 presents the conversion delays involved in the case of a simple Markov Modulated Poisson Process (MMPP) and the linear bounded model.

2.6 A Comparison of The Two Models

We compare the two models, the x_{min}-x_{ave}-I model and the linear bounded model, on the basis of their efficiency, that is the amount of network resources that are required by identical traffic patterns with identical performance constraints using the two different models. [9]Since statistical multiplexing is not allowed (to the best of our knowledge) by the linear bounded model, we consider only the case of deterministic[10] performance guarantees.

The experiments we will conduct are described in Figure 2-7. An unregulated traffic stream is fed into a leaky bucket which converts it into one of the two models. It is then fed into a simple network consisting of *N* queueing nodes in series.

In order to compare the two different traffic models, we consider a set of channel establishment requests. All the requests are identical, in the sense that the traffic pattern on all channels are identical, and all channels have the same

---

9. The utilization of the network resources depends on the traffic model being used and the admission control rules that accept channels in the network. The reader will notice that there is a violation of the modularity of thesis organization in this section, in the sense that the admission control rules form part of the performance contract mechanism which is only presented later, in Chapter 4. However, a comparison of the utilization of resources is not possible without the admission control rules, hence we are using very simple versions of these rules, and present the generalized admission control rules in Chapter 4.

10. By a deterministic guarantee, we mean a guarantee which is to be met without exceptions for all cells on a channel.

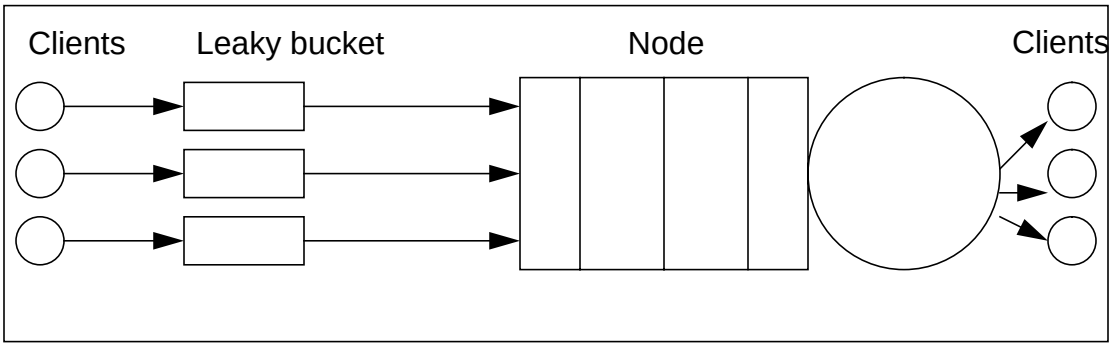

Figure 2-7 : The experiment for comparing the two traffic models. A number of clients generating unregulated traffic streams tried to establish as many channels as possible through a node by using the different traffic models. The original traffic stream was converted into the regulated traffic stream by the use of leaky buckets, and simple admission control tests used to determine how many channels to accept. The network consisted of *N* queueing nodes in cascade; the figure has been drawn for *N = 1.*

end-to-end delay requirements. The network tries to accept as many channels as possible using simple admission control rules for both models.

The end-to-end delay requirement D for a channel can be broken down into two components; the first (D_c) deals with the conversion delay involved in transforming the unregulated traffic stream into a regulated traffic stream obeying either of the two models, and the other component (D_n) is the delay to be experienced in the network. The number of channels that can be accepted by the network depends on the model parameters chosen and on the network component of the delay bound. For a channel that traverses N nodes in a network, the network component of the delay can be broken into a delay at each node (d_n), which may be obtained by the relationships

$$D_n = D - D_c, \qquad (2\text{-}11)$$

$$d_n = \frac{D_n}{N}. \qquad (2\text{-}12)$$

Instead of fixing the conversion delays arbitrarily, we considered the number of channels that the use of each model could successfully establish for a given delay bound D. We took, as the performance index for a model, the maximum number of channels that can be established with the choice of any value of D_C. This number serves as an upper bound for the number of channels that the model can support with a given input traffic and a given delay bound.

The simple admission control rules used to limit channel establishment are described below. We assumed that the service time of a cell in a node and for all channels in the network is a fixed constant t.

For the x_{min}-x_{ave}-I model, channels through a node were accepted subject to the constraints:

$$\sum_i \frac{t}{x_{min,\,i}} \le 1, \qquad \textit{(2-13)}$$

$$\sum_i t \le d_n. \qquad \textit{(2-14)}$$

For the linear bounded model, channels were accepted subject to the following constraints:

$$\sum_i \frac{t}{x_{ave,\,i}} \le 1, \qquad \textit{(2-15)}$$

$$\sum_i B_i t \le d_n. \qquad \textit{(2-16)}$$

The admission control tests are simplified versions of the tests using admission control schemes described in [Ferrari 90a] and [Andrews 89]. The simplifications are a result of the fact that all channels are identical, in the sense that

they have the same traffic and performance parameters at every node.

Simulation results obtained for the purpose of comparing the two models are described in Section 2.7. The results will show that the x_{min}-x_{ave}-I model accepts more channels than the linear bounded model when the delay constraints are relatively tight, while the two models accept almost the same number of channels in other cases.

2.7 Simulation Results

In this section, we describe the results of the simulation experiments performed to study the conversion delays for the two models and the comparison as described in Section 2.6. The results depend on the distribution of the unregulated traffic chosen for simulations. In our study, we used a number of different distributions, but will only describe the results of a simple MMPP traffic model in detail. The results using other traffic distributions are similar in nature to those that will be presented.

2.7.1 Workload

The unregulated traffic for our simulations was generated from a 2-state MMPP model. In order to generate a process with a mean inter-cell interval equal to x_{ave}, the first state had inter-cell intervals distributed exponentially with mean $0.5x_{ave}$, while the second state had inter-cell intervals distributed exponentially with mean $1.5x_{ave}$. The probability of transition from one state to the other was 0.2 in both states (see Figure 2-8).

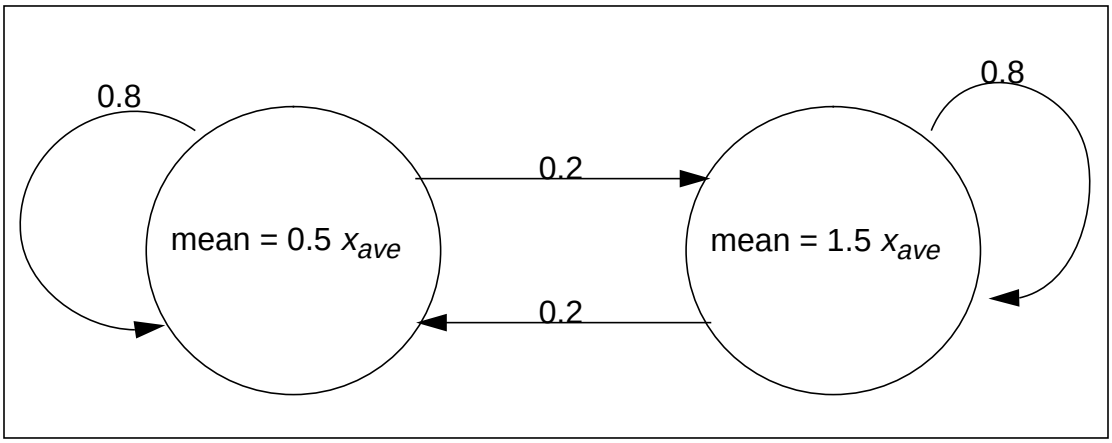

Figure 2-8 The model used for generating unregulated traffic on a channel.

2.7.2 Leaky-bucket delays

We now present the leaky-bucket delays for the MMPP process described in Figure 2-8 according to the x_{min}-x_{ave}-I model and the linear bounded model, respectively. By leaky-bucket delays we mean the delays experienced by an unregulated MMPP process during its conversion to a traffic stream regulated according to the x_{min}-x_{ave}-I model using the algorithm described in Figure 2-4, or to the linear bounded model using the algorithm in Figure 2-6.

For the x_{min}-x_{ave}-I model, the delays in the conversion process depend on the values of the three parameters chosen. The average delays involved in the conversion process are shown in Figure 2-9. The first graph shows the delays involved when the x_{ave} parameter was chosen to be 40% less than that of the original stream[11], while the second graph shows the same for a factor of 60%. Different values of x_{min} were used for both the values of x_{ave}. Notice that the conversion delays can be reduced further by increasing the bandwidth (reducing

11. The parameter of a regulated traffic must be lower than that of the original stream. Otherwise, the delays in the conversion process can not be bounded. Because of the high burstiness of the MMPP process chosen, we have to trade-off delays with bandwidth.

x_{ave}) or else by increasing the value of I. Recall from Section 2.5.1 that a representative value assignment of the parameters for the MMPP process would be any combination of x_{min}, x_{ave} and I that satisfies the constraint that the average delay be less than one average inter-cell interval of the original MMPP process. In Figure 2-9, we can see that, for an original average inter-cell interval of 100 time units, a representative set of parameter values could be x_{ave} = 71 time units, x_{min} = 24 time units, and I = 32 * 100 = 3200 time units.

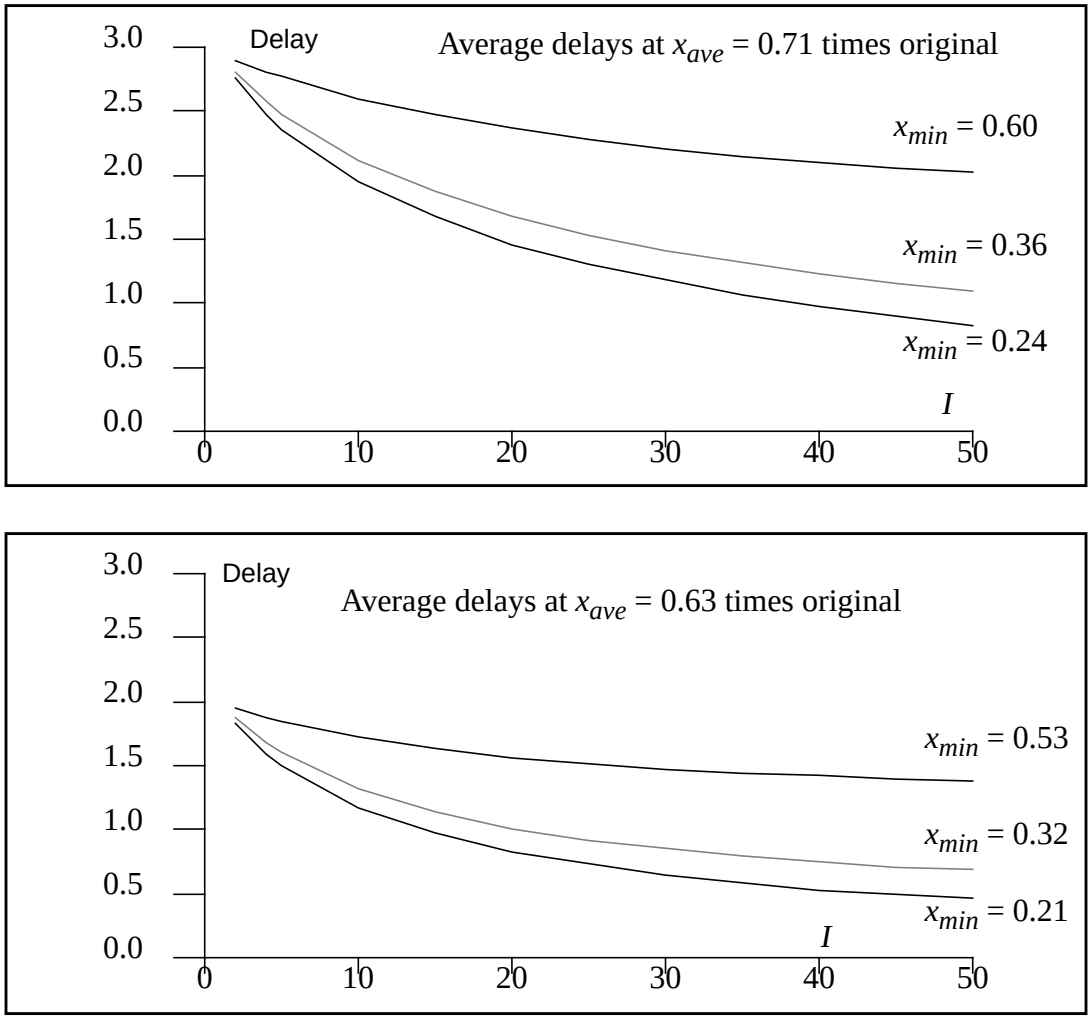

Figure 2-9 Average delays involved in regulating the MMPP process of Figure 2-8 into the x_{min}-x_{ave}-I model. Both axes are normalized with respect to the average inter-cell interval of the unregulated stream.

For the linear bounded model, the average conversion delays are shown in Figure 2-10. Three curves are shown, the ones for which the value of x_{ave} used was 40%, 60% and 80% lower than that of the original stream. The horizontal axis shows the burst-size parameter used for the conversion process. In this figure, it is obvious that the average conversion delays are usually smaller for the linear bounded model than for the x_{min}-x_{ave}-I model, although there is no direct one-to-one correspondence between the parameters of the two models. This phenomenon can be explained by the fact that the linear bounded model allows multiple back-to-back cells, while the x_{min}-x_{ave}-I model forces a minimum spacing equal to x_{min} between these cells, thus resulting in a larger delay in the conversion process. Defining a representative value of the parameters as one which results in

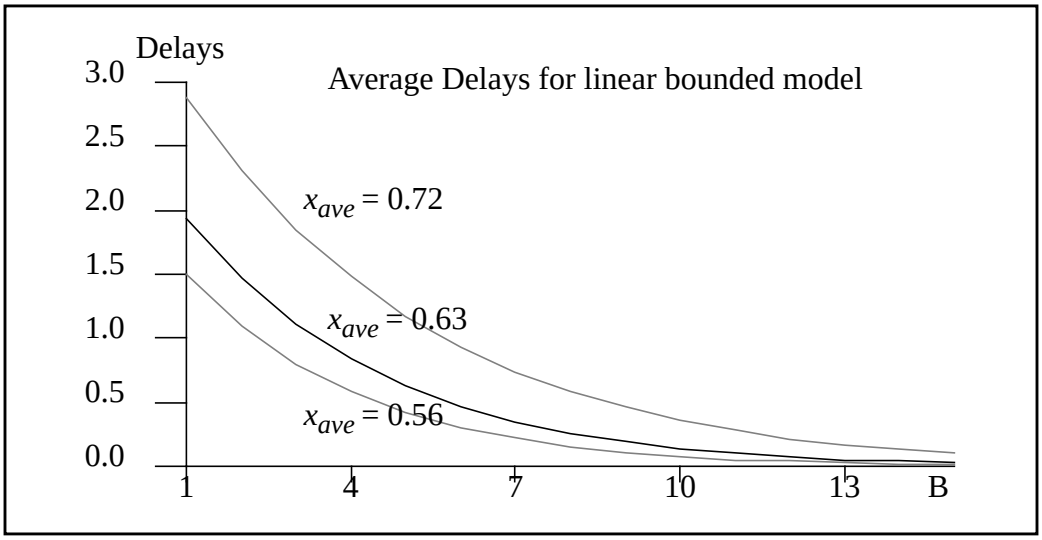

Figure 2-10 . Average conversion delays for the MMPP process using the linear bounded model. The vertical axis has been normalized by dividing with the original average inter-cell interval.

an average delay of less than one inter-cell interval, choosing B=8 appears to be sufficient with an average output bandwidth 40% higher than the original. Thus, a stream with mean inter-packet time of 100 units would require x_{ave} = 71 units and

B = 8.

2.7.3 Comparison results

In order to compare the two models using the method outlined in Section 2.6, we need to bound the maximum delays in the conversion process. Unfortunately, this delay cannot be bounded absolutely since the MMPP process can have very rapid arrivals one after another. Therefore, we define the notion of the 99th percentile delay, d_{99}, which defines the least possible bound on the delays experienced by no more than 1% of all the cells in our simulations.

In order to compare the two models, we examined the number of channels with identical traffic and performance parameters that can be established using the two models. We first evaluated the d_{99} curves for converting the MMPP model into one of the regulated models by simulating the algorithms described in Figure 2-4 and Figure 2-6. These curves were subsequently used to obtain the conversion delays (D_c)for the MMPP process using either of the two models, and the available delay bound per node (d_n) using equation (2-12). The admission control rules outlined in Section 2.6 were used to determine the number of channels that can be established under these conditions.

In the first series of experiments of this comparison, the conversion delays with different sets of parameter values were obtained by means of simulation experiments. One such curve, showing the variation of d_{99} with I and x_{min} (when using the x_{min}-x_{ave}-I model) is shown in Figure 2-11,[12] while a similar curve for

12. The delays experienced in the conversion process tend to be high due to the burstiness of the MMPP process.

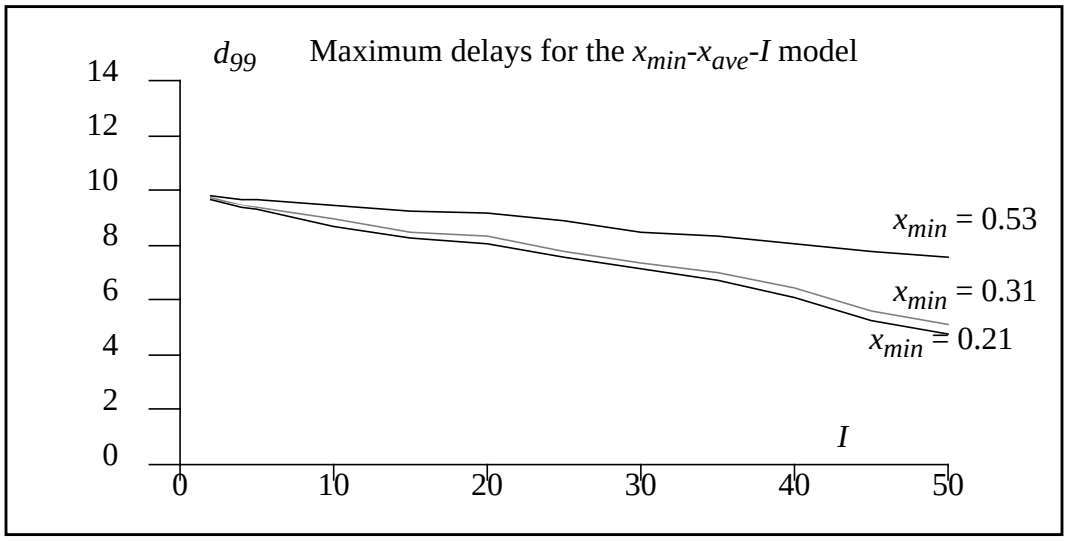

Figure 2-11 The maximum conversion delays for the x_{min}-x_{ave}-I model. This curve is drawn for value of x_{ave}. which is 0.71 times that of the original unregulated traffic.

the linear bounded model is shown in Figure 2-12. The curves in Figure 2-11 have been drawn for a value of x_{ave} which is 0.71 times that of the unregulated stream. Similar curves were also obtained for the cases where x_{ave} was 0.625, 0.55 and 0.5 times that of the original inter-cell interval.[13]

For a MMPP process requiring a certain delay bound D, all possible value of the conversion delays D_c (less than D) were tried. Corresponding to each of these values of the conversion delays, we determined the parameters that would result in these conversion delays using the results of the first series of experiments. As an example, the parameters for the x_{min}-x_{ave}-I model may be obtained from Figure 2-11, and the parameters for the linear bounded model may be determined from Figure 2-12). These parameters were used to establish as many channels as possible using the simple admission control tests mentioned in Section 2.6 and a network delay d_n given by equation (2-12). The maximum number

13. The value of x_{ave} for the regulated stream has to be lower than that of the original stream. Otherwise, the values of d_{99} would depend on the length of the simulation runs.

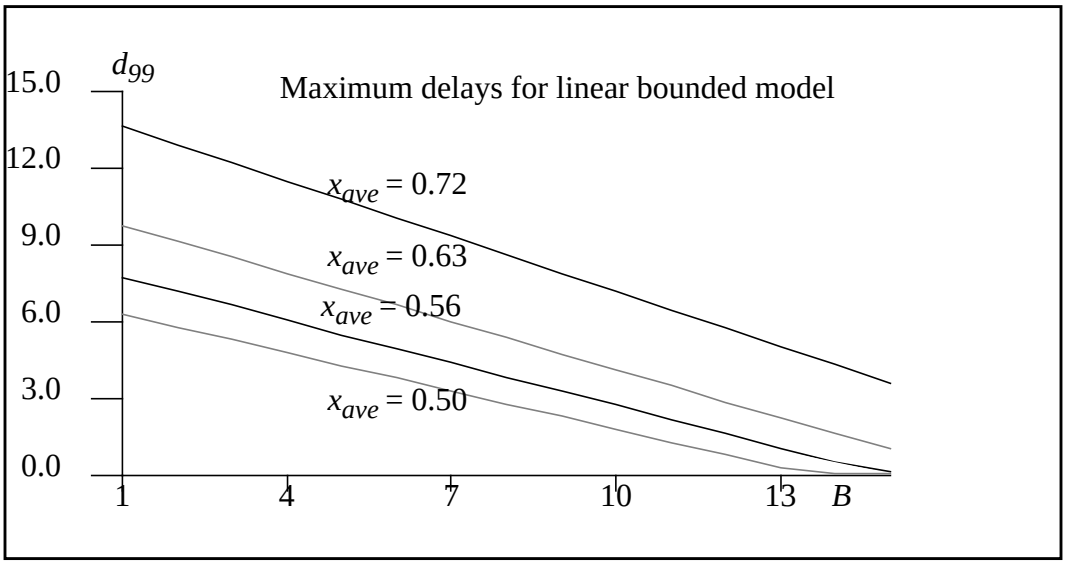

Figure 2-12 The maximum delays for conversion of the MMPP process into the regulated stream using the linear bounded model.

of channels that could be established for any value of the conversion delays was thus determined for both the models.

Maximum delay curves for both models were stored in tabular form for different parameter values instead of as continuous curves. The parameters for the x_{min}-x_{ave}-I model that corresponded to a particular value of conversion delay were obtained from the x_{min}-x_{ave}-I table. A similar table was used for the linear bounded model.

The maximum numbers of channels that can be established in this manner using the two models are shown in Figure 2-13. All channels were assumed to have an original unregulated average inter-cell interval of 100 units, and the service time in the nodes was taken to be one time unit. Thus, at most 100 channels could be established by any scheme whatsoever. Because of delay constraints, and the use of higher bandwidths in the regulated streams, the actual number of accepted channels will be less than 100.

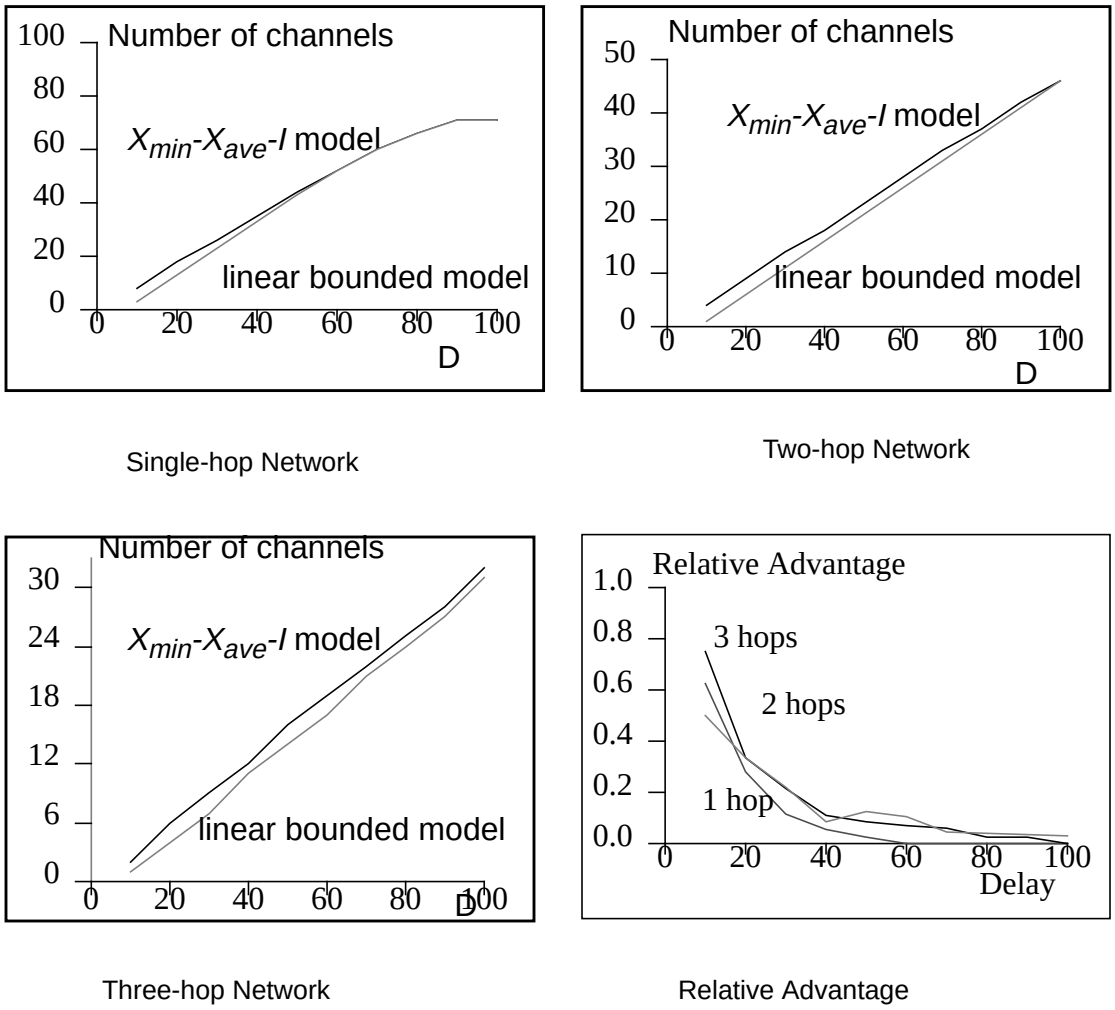

Figure 2-13 Number of channels accepted by both models for the MMPP process with different delay requirements. At low delay requirements, spacing out the cells in the x_{min}-x_{ave}-I model yields some advantage over the linear bounded model. At more relaxed delay requirements, both models perform almost identically.

At low delay requirements, the x_{min}-x_{ave}-I model has some advantage over the linear bounded model. For example, at an end-to-end delay requirement of 10 units, it accepts 8 channels as opposed to only 3 for the linear bounded model; and at a delay of 20, it accepts 18 channels as opposed to 13. If we can accept N_1 channels using the x_{min}-x_{ave}-I model and N_2 channels using the linear bounded model, the relative advantage of the x_{min}-x_{ave}-I model over the linear bounded model is given by:

$$\text{Relative Advantage} = \left(\frac{N_1 - N_2}{N_1}\right). \qquad \textit{(2-17)}$$

In Figure 2-13, the relative advantage of the x_{min}-x_{ave}-I model increases slightly as the path length increases. However, for a single-hop network, there is no significant difference in performance between the two models at delay requirements in excess of 40 time units.

Notice also that, for a single-hop network, both models can only accept about 70 channels overall; thus, any predictive scheme using these models with a MMPP process will use only 70% of the network's bandwidth. The rest of the bandwidth can be used by traffic not requiring performance guarantees.

Qualitatively similar results were also observed with other kinds of arrival processes; for example, one in which inter-cell intervals were chosen according to a Poisson distribution, or one in which the inter-cell intervals were distributed uniformly between 0 and twice the mean inter-cell interval.

Thus, the x_{min}-x_{ave}-I model seems to have a slight advantage over the linear bounded model for delay-sensitive traffic.[14] Moreover, this model can be used to provide statistical guarantees as well. We are not aware of a simple scheme to extend the linear bounded model to the statistical case. Therefore, we will be using the x_{min}-x_{ave}-I model in the rest of the thesis for the purpose of providing performance guarantees.

14. While simulation studies of this nature can not prove that one model is better than the other, the better performance of the x_{min}-x_{ave}-I model in the two or three types of distributions that we tried suggests that it might perform better in other cases as well.

Chapter 3: The Traffic Contract-II: Mapping and Verification

3.1 Introduction

Although we have selected a specific traffic model for the purpose of resource allocation, which the client will use to specify its traffic characteristics at channel establishment time, two traffic-related problems remain. The first problem is the detection of contract violations (either deliberate or unintentional) by a client (the *verification* problem). The second problem is that the traffic characteristics of a channel as declared in the contract are valid only at the interface between the sender and the network, and may change along the path of the channel (the *mapping* problem). We call it the mapping problem since the variation in the traffic characteristics is dependant on network topology, and we are attempting to map the traffic contract that is valid at one point in the network (at the sender-network interface) to other points in the network.

In general, the traffic characteristics at the beginning of a path through a network may not be the same as those at the end of the path. Furthermore, dependencies may develop among different channels as they share parts of the same network. This may create problems for admission control schemes that assume independence among channels.

In this chapter, we will study the effect of the network on traffic patterns, and

propose a simple way to deal with the problem of changes in traffic characteristics because of network fluctuations. The scheme can also be used to detect violations by a client of the traffic-related aspects of a contract.

The chapter is structured in the following fashion: in the next section (Section 3.2), we explain why the traffic characteristics of a channel might change at a node. We then attempt to obtain an estimate of how important these changes in the characteristics might be by means of simulation experiments described in Section 3.3. Three different approaches to alleviating the problems due to a change in traffic characteristics are examined in the next section (Section 3.4), and one of them is described in detail in Section 3.5. Then, in Section 3.6, we take a look at the possible dependencies that may arise among channels with a common route in a network. Finally, we present a verification scheme for the traffic contract in Section 3.7.

3.2 The Problem of Traffic Mapping

If the traffic pattern of a channel changes along its path, the channel would require different amount of resources to be reserved in different nodes. Any change in the parameters of the traffic model we are using needs to be taken into account if we want to provide the legal guarantees we have talked about in Chapter 1.

In general, the change in the traffic pattern at a node is caused by delay fluctuations. Let us consider a node with the first come first served service discipline. If two cells happen to arrive at a node at a time when the node is relatively busy, they may be sent out with a greater inter-cell spacing than the spacing they had at their arrival; if, on the other hand, there is a drop in the load between the two arriv-

als, the inter-cell spacing will be reduced. Thus, the distributions of inter-cell intervals may change in an unpredictable fashion.

The traffic pattern on a channel may change in a very complex fashion along the path of the channel. For the purpose of resource allocation, we can confine ourselves to the changes in the values of the traffic parameters. Thus, since x_{min} is one of the parameters in our traffic model, we will study the changes in the value of x_{min} along the channel's path. Similarly, we are interested in the changes of x_{ave} and I.

The change in traffic pattern depends on the service discipline used at the switches in the network. It is possible to imagine a malicious server, which buffers up cells up to a certain limit and then spews them out in rapid succession so as to distort the distribution of inter-cell intervals. It is also possible to imagine servers that preserve the same inter-cell intervals (or maybe the distribution) by maintaining enormous amounts of state information[1]. However, instead of analyzing a large number of service disciplines, we will concentrate on analyzing the behavior of some typical service disciplines like First Come First Served (FCFS) or Earliest Deadline First (EDF). These disciplines have been selected because they can be used to provide the required QOS guarantees (see Chapter 4 for a detailed description of the scheduling disciplines).

For the FCFS service disciplines, small inter-cell intervals would put the cells in the same busy period at a node, and thus they will be spaced out depending on the total workload that has arrived (between the arrival of two cells from the

1. One way to maintain the same inter-cell intervals would be to use a distributed jitter control scheme such as the one described in [Verma 91].

channel under observation) on the other channels sharing the same output link. If that is a period in which a lot of channels are active, then they would be spaced by a relatively large interval, but, if that is a period in which only a few channels are active, the inter-cell spacing will tend to be reduced. Large inter arrival times are likely to put cells in different busy periods, and thus add a random noise with zero mean to the inter arrival time distribution (assuming that the delays experienced by two cells in different busy periods will be independent of each other). For the case of EDF scheduling, the total amount of cross-traffic load is limited by the delay bounds and rates assigned to different channels, and so the spacing variations are limited. However, the cell spacing may still be reduced. The net effect can in general be quite complex. A quantitative estimate of the distortion caused by the network was obtained by means of the simulation experiments described below.

3.3 The Simulation Experiments

The simulation experiments were based on the configuration shown in Figure 3-1.A single queueing server was studied while being traversed by a number of channels. One of the channels was selected as the tagged channel. The change in the traffic pattern of the tagged channel was monitored and compared with the original traffic pattern. The goal of the simulation experiments was to discover whether the distortion of traffic parameters is likely to be a problem.

The factors affecting the distortion in the traffic pattern of the tagged channel are (1) the queueing discipline used at the server; (2) the intensity and the arrival pattern of the cross-traffic and (3) the arrival pattern on the tagged channel itself. The cross-traffic and tagged traffic patterns were independent of each other.

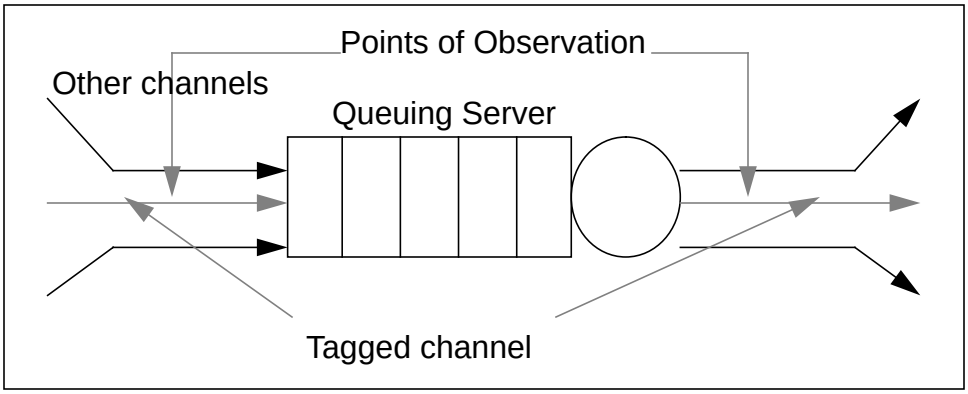

Figure 3-1 : The simulation configuration to observe changes in traffic characteristics. A number of channels feed into the same queueing server with a given scheduling discipline. The traffic patterns on the tagged channel at the input and output are observed and compared.

3.3.1 Factors

In our simulation studies, the following values of the different factors were considered.

Service Discipline: FCFS or EDF. With the FCFS discipline, cells were serviced in first-come first-served order. With the EDF discipline, each channel was assigned a delay bound, and cells arriving on the channel were assigned a deadline obtained by adding the corresponding delay bound to the instant of arrival. Cells were serviced in the order of earliest deadline. The delay bound of each channel was generated randomly from a uniform distribution between 200 and 400 time units. [2]

Cross-traffic: Exactly 200 channels were assumed to exist at the node. The service time of a cell on any channel was taken to be one time unit. All channels were assumed to have the same traffic parameters (that is, they obeyed the same

2. The specific values of delay bounds and other parameters have been chosen so as to ensure that the set of channels present at the node satisfy the admission control rules stated in Chapter 4.

values of x_{min}, x_{ave} and I, and were generated according to the same distribution, but the arrivals on different channels were independent of one another).

On each of the channels that were characterized by the x_{min}, x_{ave}, and I parameters, traffic was generated according to one of the following arrival processes: periodic, uniform, geometric, and bunch. These processes are described below:

A periodic process has arrivals at regular intervals of duration x_{ave} time units.

A uniform process has inter-cell intervals drawn from a uniform distribution between x_{min} and $2x_{ave} - x_{min}$.

In a geometric process, cells arrive in bursts, with an inter-cell spacing equal to x_{min} within each bursts. The burst lengths are geometrically distributed. The inter-burst intervals are also geometrically distributed with a mean that causes the x_{ave} for the entire process to match the value of that parameter for the channel. A filter was used to ensure that the average rate over any period of I did not exceed x_{ave}.

In a bunch process, a single burst of I/x_{ave} cells separated by intervals of x_{min} is followed by a silence $\left(I - \left\lceil \frac{I}{x_{ave}} \right\rceil x_{min}\right)$ time units long. This brings the average rate in an interval I to x_{ave}.

Utilization: The following utilizations of the node were considered: 0.4, 0.5, 0.6, 0.7 and 0.8. When the node utilization was u, x_{ave} on each of the 200 channels was $200/u$ time units. I was taken to be 20 times x_{ave}, and x_{min} was chosen to be

50 time units. Only these values of I and x_{min} were used in order to reduce the number of simulation runs required for this study.

Tagged traffic: The traffic on the tagged channel consisted of a simple stream with cells arriving at regularly spaced intervals. Three average inter-cell spacings were considered: 10, 20 and 30 time units respectively. Thus, the histogram of inter-cell intervals at the input to the tagged channel was as shown in Figure 3-2. In terms of the x_{min}-x_{ave}-I model, the values of x_{min} and x_{ave} are equal and

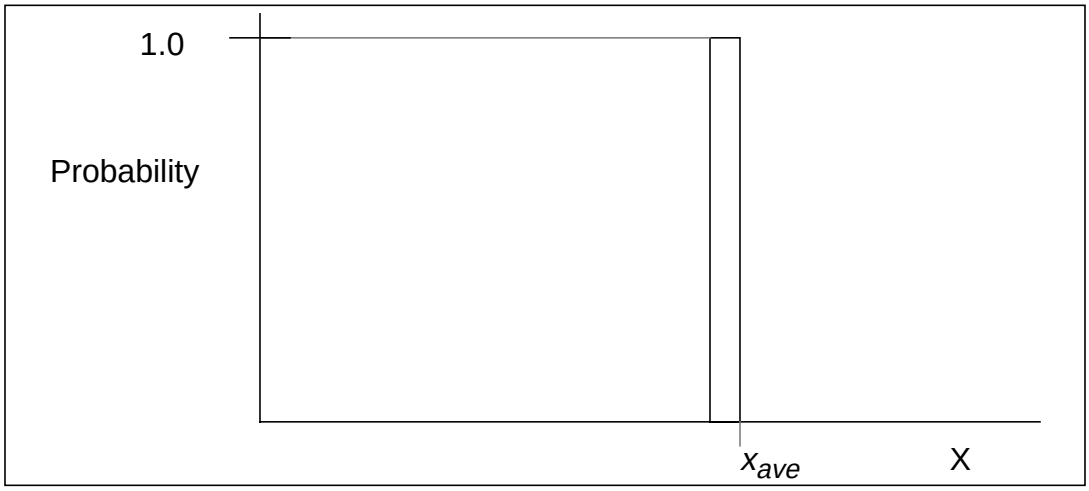

Figure 3-2 The histogram of the original inter-cell inter arrival times on the tagged channel. The distribution consists of cells arriving at regular intervals of x_{ave} time units each. In terms of the x_{min}-x_{ave}-I model, the values of x_{min} and x_{ave} are both equal and I can be any integral multiple of x_{ave}.The values of x_{ave} used were 10, 20 and 30 time units respectively.

I can be any integral multiple of x_{ave}.

Although the tagged traffic stream is periodic, there is no synchronization effect in the simulator which may distort the results. This is because the traffic on all other channels is generated randomly.

3.3.2 Changes in the value of x_{min}

To characterize the output traffic pattern on the tagged channel, we measured the minimum inter-cell spacing, and its 1st percentile. The 1stpercentile (x_1) is the largest interval lower than 99% of all the inter-cell intervals of the output traffic pattern. While the measurement of minimum inter-cell spacing may be subject to distortion due to the occurrence of low-probability events in our simulations, x_1 is not subject to the same drawback.

The results of the simulations for the case of EDF scheduling are shown in Figure 3-3. There are three sets of curves in each of the four graphs shown in this figure, each set corresponding to a fixed value of x_{min} (30, 20 or 10) on the tagged channel. Within each set corresponding to a fixed x_{min}, three curves are drawn, one showing the original value of x_{min}, one showing the value of x_{min} at the output of the node, and one showing the value of x_1 at the output of the node. The curve for the output value of x_{min} is, as expected, quite irregular. The reader will notice that the minimum output inter-cell spacing is often much smaller than the minimum spacing at the input of the tagged channel. The same effect can be observed for the value of x_1. The reduction in x_1 is not as dramatic as that for the minimum value, but the tendency of cells to come closer together is obvious. An interesting observation is that arrival patterns that are more random tend to produce a larger reduction in the values of x_{min} and x_1 than arrival patterns in which cells arrive in bunches of fixed size. A plausible explanation is that, with bunches of fixed sizes, most of the cells in the bunch experience almost the same delays, and it is only at the start or at the end of a bunch of cells that we are likely to see a difference. The expected traffic on a real channel is more likely to consist of bunches with random lengths

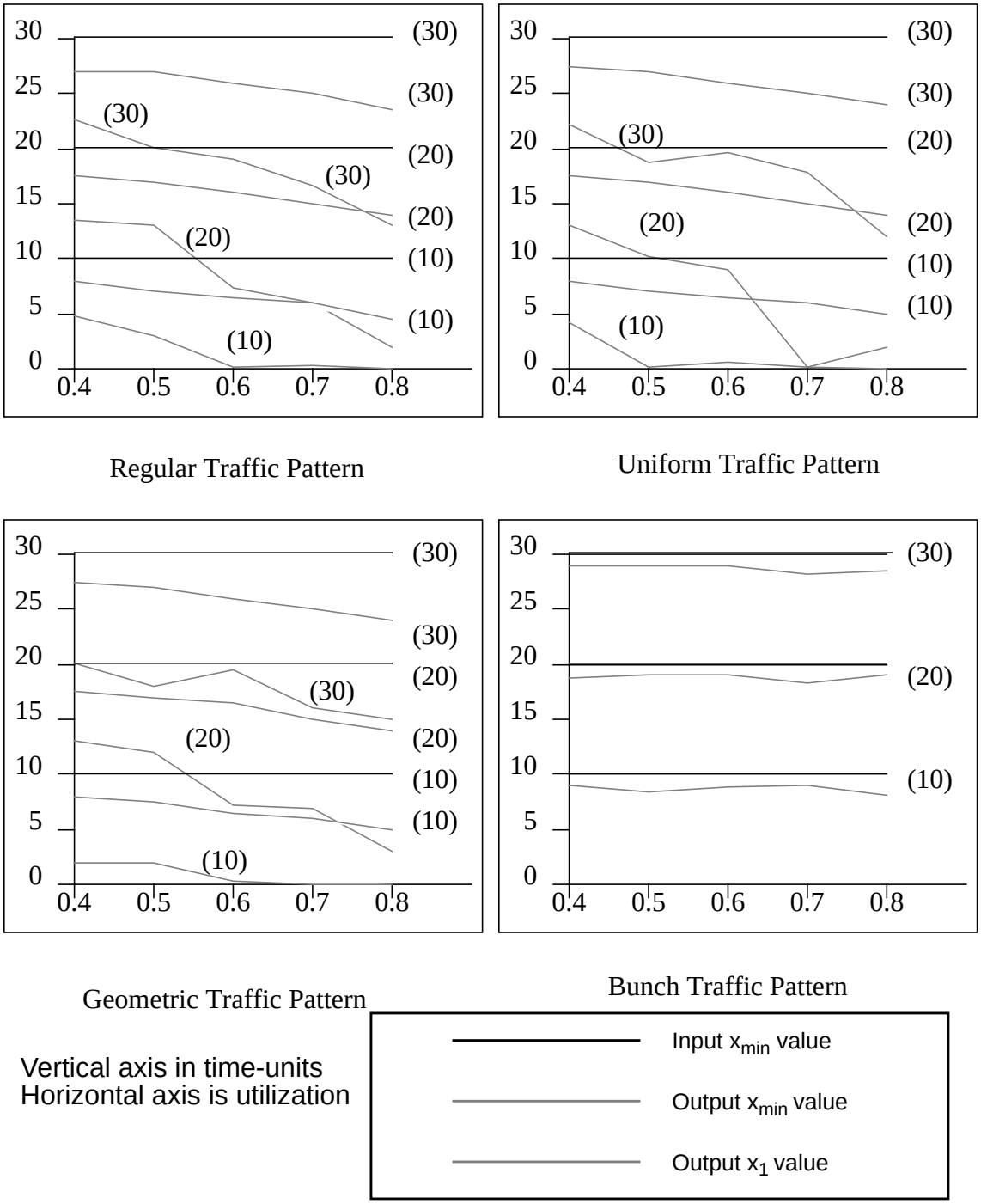

Figure 3-3 Variations in the values of x_{min} and x_1 for different arrival patterns at a single queueing server using the earliest-deadline-first scheduling discipline. The three curves in each set show the input value of x_{min}, the output value of x_{min}, and the output value of x_1 for different server utilizations. There is an appreciable decrease in x_{min} for the regular and uniform traffic patterns. Curves in each set are marked by the input x_{min} value in parentheses. So, the dotted line with caption (20) corresponds to the output x_1 for an input x_{min} of 20 time units. The x_1 curve for bunch traffic pattern is the same as that for the input x_{min} value.

(as in the geometric traffic pattern), and an appreciable change in traffic parameters cannot be ruled out.

Similar results for the case of the FCFS scheduler are shown in Figure 3-4. The effects over here are similar to those of the EDF scheduler. Thus, both scheduling policies seem to have very similar influences on the traffic parameters.

Although the 1st percentile and the minimum inter-cell spacing tend to become smaller after traversing a node, this effect may be either enhanced or weakened by nodes further downstream. Cells which may have been brought close in the first node may be spread apart in the second node, and thus, the effect may be weakened. In order to study the effect of multiple nodes in series, we examined the variations in traffic on a tagged channel which traversed a number of nodes in succession. The workload on each of the nodes was kept the same as in the previous simulations. Although we experimented with all kinds of arrival processes for cross-traffic at different node utilizations, we only present the results of a geometric traffic pattern with a utilization of 60% in all the nodes. The original traffic pattern on the tagged connection was as described in Figure 3-2. The simulation configuration can be seen in Figure 3-5.

The resulting variations in x_{min} are shown in Figure 3-6. In general, the values of x_{min} and x_1 tend to decrease as the path length increases. There is a visible tendency for the value of x_1 to decrease as the path-length increases. This implies that the number of buffers allocated to provide a loss probability less than 1% at any node should account for the variation in the traffic pattern, and should grow in downstream nodes. [3]

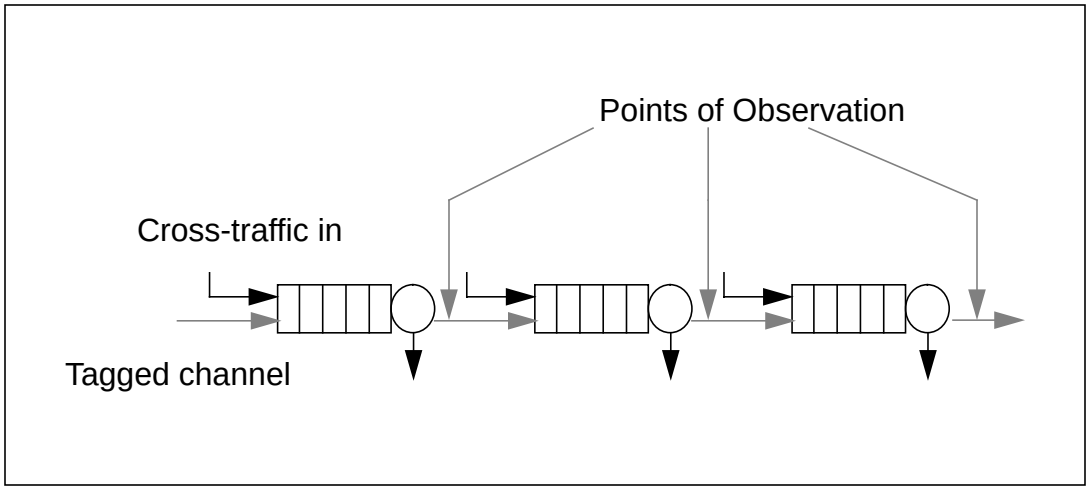

Figure 3-5 The simulation configuration to determine whether the change in traffic parameters is worsened or improved by a number of nodes in series. The tagged channel, which originally had the traffic pattern described in Figure 3-2, passed through a number of nodes, each with the EDF scheduling and a utilization of 60%. The arrival pattern on each of the 200 channels in the cross-traffic was geometric. The parameters of the resulting stream at the points of observation were measured.

Although the graph shows the variation for the geometric traffic pattern only, similar results were obtained with other cross-traffic patterns as well.

3.3.3 Changes in the value of x_{ave}

Until now we have only considered variations in the value of x_{min}. Let us now examine the variations in the value of x_{ave}. An advantage of the simple traffic pattern on the tagged channel is that the original traffic pattern remains the same irrespective of the value of I chosen for the tagged channel. Thus, a single simulation experiment can be used to measure the variation in x_{ave} for different values of I. With an arrival process like geometric or bunch, the original traffic pattern on the tagged channel would have been dependent on the value of I that we had selected.

---

3. We are assuming here that a change in the value of x_{min} affects the buffer allocation and utilization. In our simulations, only a few cells were present from any connection at a single node at any time, so plotting the buffer distributions did not provide any interesting results. However, in a ATM network, many cells from the same connection may be present at the same time, and the buffer distribution will also change.

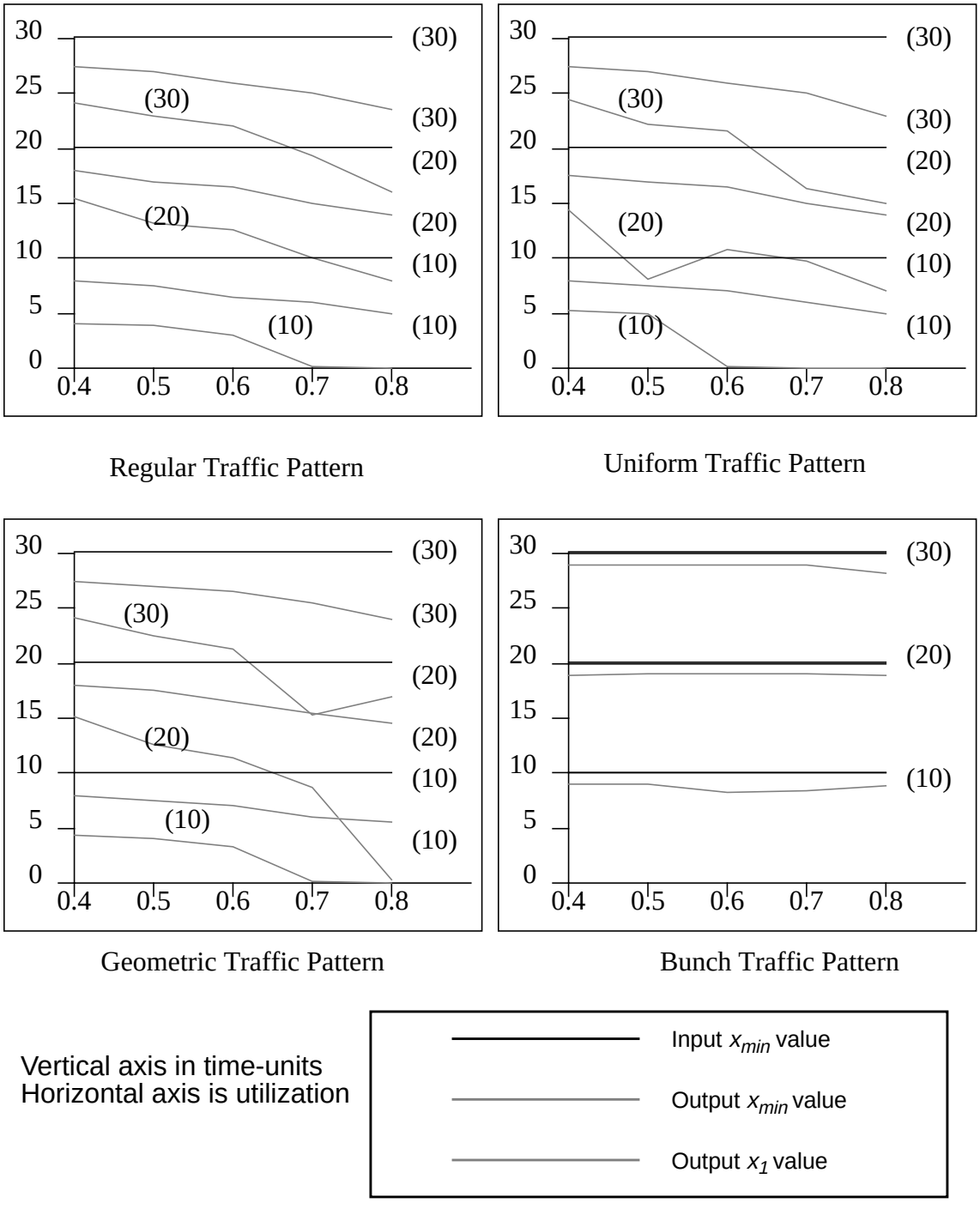

Figure 3-4 Variations in the values of x_{min} and x_1 for different arrival patterns at a single queueing server using the first-come first-served service discipline. The three curves in each set show the original value of x_{min}, the final value of x_{min} and the final value of x_1 for different server utilizations. There is an appreciable decrease in x_{min} for the regular and uniform traffic patterns. The decrease is dependent on the server utilization, but more or less independent of the original x_{min}. The number in parentheses indicate the original value of x_{min} for that curve.

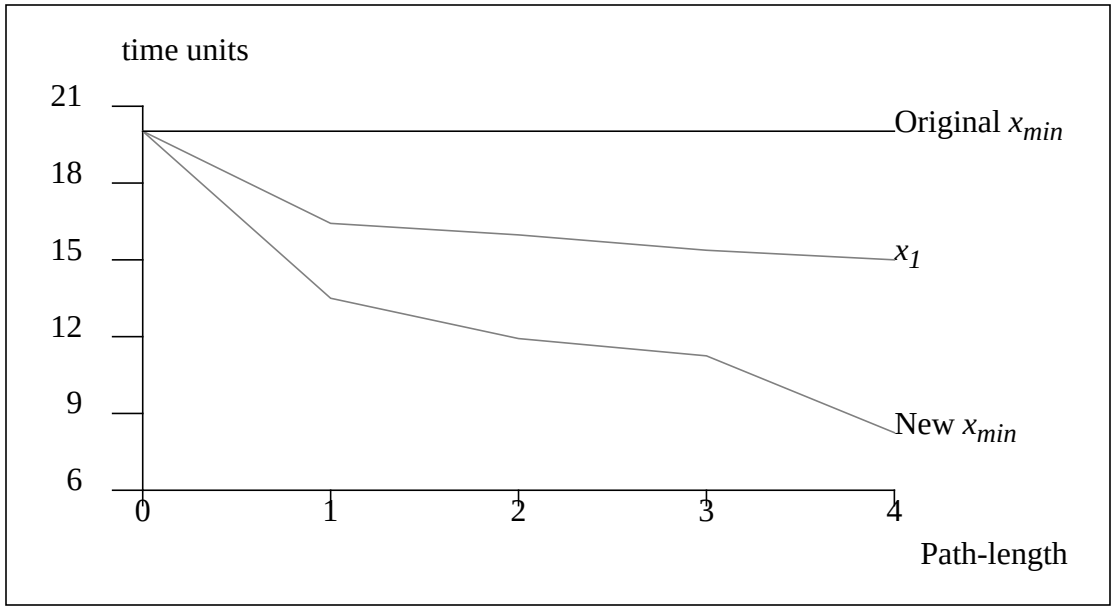

Figure 3-6 Variations in x_1 and x_{min} as a function of path length. Each of the nodes along the path had a utilization of 60%. The tagged channel spanned all the nodes and was observed at the exits of each of the nodes along the path.

In general, the change in x_{ave} should be smaller compared to the change in the value of x_{min}. This is because the averaging interval I is usually chosen large enough that the queueing server becomes idle at least once in every interval of length I. If the node was idle at the beginning of an interval of length I, and it was idle also at the end of the same interval, then all the cells that have arrived in this interval have also left within the same interval, and the average inter-cell spacing at the output equals that at the input for this interval. If the node was idle at the beginning of the interval, but not idle at the end, some of the cells may not have been transmitted in this period, and so the average inter-cell interval will increase. On the other hand, if the node was not idle at the beginning of the interval, then some of the cells from the previous interval would have come together with some cells in the current interval and the average inter-cell interval could decrease. We wanted to discover whether these variations in the value of x_{ave} are significant.

The results are shown in Figure 3-7. The output x_{ave} was computed by taking each successive bunch of I/x_{ave} cells in the cell stream at the point of observation. The values of two parameters are plotted in the figure: the first is the minimum average inter-cell interval for any interval of length I, and the other is the 1st percentile of the average inter-cell spacing. The 99% confidence interval for the average intervals shown in the figure was less than 0.2 time units for each of the curves. There is a very small change in the value of x_{ave}, which becomes even less as one uses larger values of I. Thus, the simulations confirm our expectation that the average rate does not change appreciably due to network load fluctuations. If a client chooses a small value of I, then the change may be more pronounced.

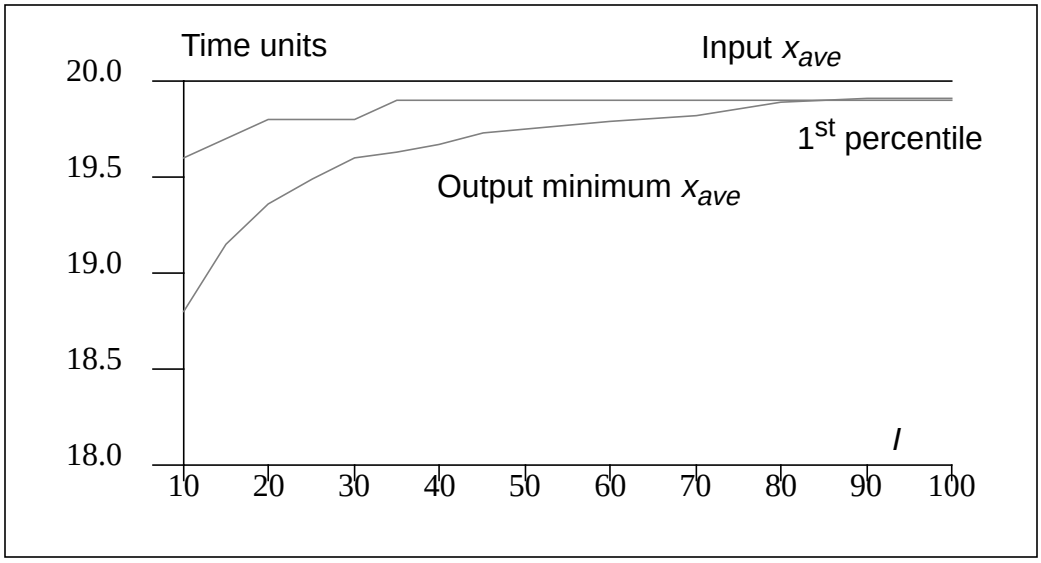

Figure 3-7 The variation in the value of x_{ave} as a function of I. The three curves show the minimum average inter-cell spacing in an interval spanned by I/x_{ave} cells. The curves have a 99% confidence interval of 0.2 time units along the y-axis. The use of longer averaging intervals reduces the variations in x_{ave} to a very small value.

Like the minimum inter-cell intervals, the average inter-cell intervals also show a tendency to decrease as the path length increases. Figure 3-8 displays the variation in the average rate of the tagged channel according to the simulation con-

figuration shown in Figure 3-5. The absolute reduction in the value of x_{ave} is much smaller than the reduction in the value of x_{min}, (as can be seen by comparing Figure 3-6 with Figure 3-8) and it may not be a problem in channels with small path lengths.

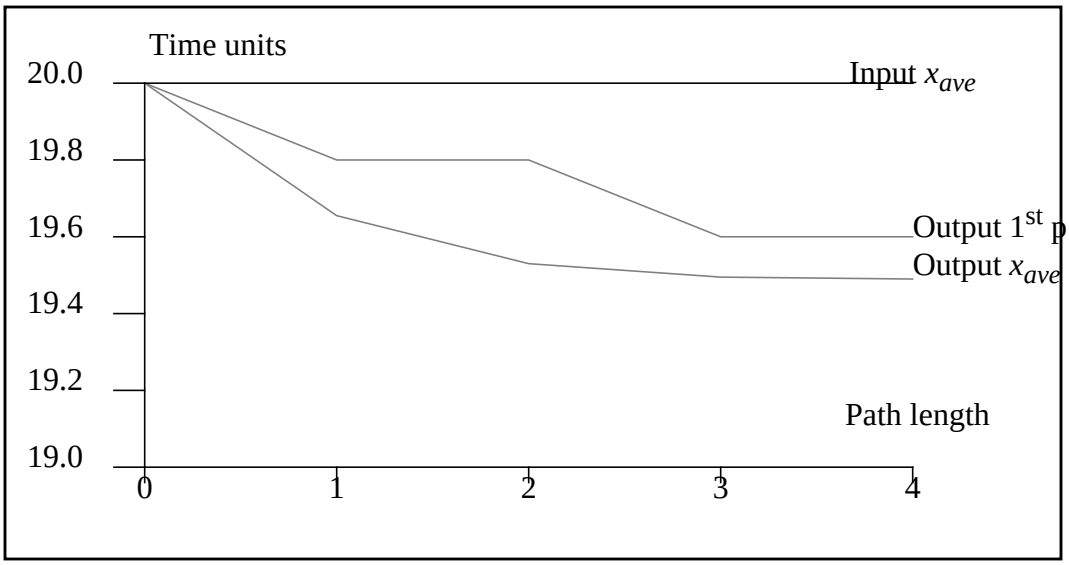

Figure 3-8 The variation in the average rate with path length. Both the minimum average spacing and the 1st percentile decrease slightly as the path length increases. The 99% confidence intervals for the curves are 0.1 time units. The change in the average rate is very small.

3.4 Alternative Approaches to The Mapping Problem

The simulations described in Section 3.3 show that we cannot assume that the model parameters of a channel will remain the same along its path. A channel may span a relatively large number of nodes in a network. We can handle the variations in one of the following three ways:

(1) *Model the change*: If it is possible to characterize the change in the traffic parameters and derive a simple relationship that produces values of the output traffic parameters from those of the input traffic parameters, the output traffic param-

eters can be used for resource allocation in the node downstream. The same process can be repeated for longer path lengths. The problem with this approach is that the modelling process is difficult and tedious. Furthermore, and much more importantly, the change in traffic parameters will vary over time as new channels are established and deleted on the same network.

(2) *Reconstruct the input traffic pattern*: Alternatively, we may try to reconstruct the original traffic pattern (that is, the sequence of inter-cell intervals at the entry point to the network) at each and every node.

(3) *Partially reconstruct the input traffic pattern*: This approach attempts to maintain the same model parameters instead of trying to reconstruct the original traffic pattern fully.

Some schemes that can lead to full or partial reconstruction of the traffic pattern are described in Section 3.5. Those schemes can also be used to verify whether a client is fulfilling its promises concerning the traffic parameters.

3.5 Reconstruction of Traffic Pattern

The reconstruction of a traffic pattern needs to be done at each node in our network model (that is, at each output queue of an ATM switch). Our approach is the following. The node consists of two modules: a *regulator*, which acts as a “policeman” preventing misuse of the network, and a *scheduler*, which is responsible for selecting the cells to be transmitted next on the output link (see Figure 3-9). Rate control requires that we calculate the expected arrival time of each cell along a channel, i.e., the time the cell should have arrived if it had obeyed its traffic

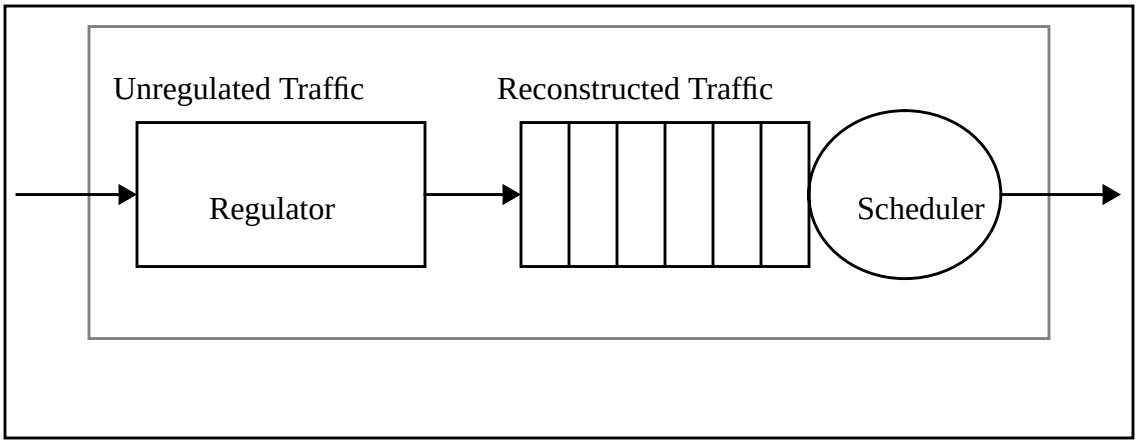

Figure 3-9 The structure of a node in our network model. Each node consists of two modules, a regulator, which acts as a policeman preventing misuse of the network, and a scheduler, which is responsible for selecting the cells to be transferred next on the output link.

parameters. Calculation of this expected arrival time is easy and can be done by using equation (2-3), as illustrated in Figure 3-10. The reader will notice that the algorithm is very similar to the conversion algorithm (Figure 2-6), since we are converting a traffic that does not obey the x_{min}-x_{ave}-I model into a traffic that obeys the model. The only difference is that we know that the traffic originally did obey the x_{min}-x_{ave}-I model, and thus the conversion process is essentially trying to reproduce the original traffic pattern.

On the acceptance of a channel, some state information needs to be kept at each node. This state information allows the node to predict the earliest time the next cell on a channel can arrive. On the arrival of every cell, the procedure `cell_arrival()` is called; the procedure computes the time the cell should actually have arrived at the node if it had obeyed all the traffic specifications. If the cell happens to have arrived before it was expected, it is held in the regulator for a period of time equaling the difference between expected arrival time and the current time, before being handed over to the scheduler

```
int b = floor(I/x_ave);
TIME history[b+1];
int hist_start;
int hist_end;
TIME hist_sum;
TIME y_clock,cell_eligib_time;

initialization()
{
   int i;

   for (i=0;i<=b;i++){
      history[i] = infinity;
   }
   hist_sum = b*infinity;
   hist_start = 0;
   hist_end = b;
   y_clock = clock;
   cell_eligib_time = 0.0;
}

cell_arrival(clock)
{
   TIME y_i, this_time;

   this_time = max((y_clock - clock), 0.0);
   y_i = max(y_clock,clock) - cell_eligib_time;

   hist_sum = y_i + hist_sum - history[hist_start];
   hist_start = hist_start + 1% (b+1);
   history[hist_end] = y_i;
   hist_end = hist_end + 1% (b+1);
   y_clock = max(clock,y_clock) + max(I-hist_sum,x_min);
   cell_eligib_time = clock + this_time;
}
```

Figure 3-10 The algorithm for computing the expected arrival time for a packet according to the x_{min}-x_{ave}-I model. `cell_arrival` is called whenever a new cell is received and it returns `cell_eligib_time`, the instant when the cell was supposed to actually arrive. The history mechanism is needed to ensure that the average-rate guarantees are not violated at any time. The variable `y_clock` stores the smallest possible time when the next cell can arrive, and is updated every time a new cell is received. For partial reconstruction of the input traffic pattern, a cell is kept in the calendar queue till its `cell_eligib_time`. The same scheme can be used to verify if a client is violating its traffic contract. The algorithm is presented for a single channel.

The hardware implementation of this rate control scheme may be done by means of a number of calendar queues [Brown 88]. The scheme is shown in Figure 3-11. For the time being, assume that we have an infinite number of such calendar

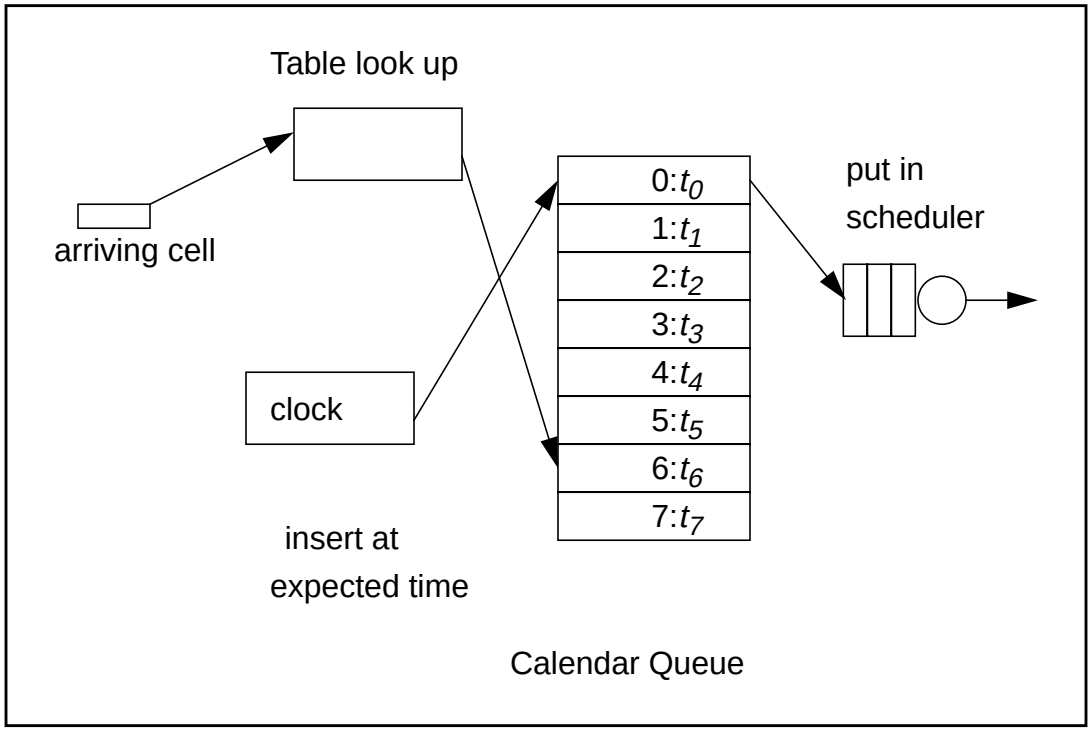

Figure 3-11 One possible hardware implementation of the regulator module. Only one such module is needed for all the channels using the same output link.

queues. Each calendar queue contains a list of the cells that were expected at that time, but have arrived earlier. When the clock reaches that value of time, all these cells are transferred to the scheduler by appending[4] this list to the list of cells in the scheduler queue. The insertion of a cell in the calendar queue as well as the appending of the list can be done in a constant number of steps. Only one such hardware module is needed for all the channels sharing the same output link.

4. The appending can be done in constant number of time if the scheduler implements a first-come-first-served discipline. If the scheduling discipline is earliest deadline first, then the operation is no longer possible in a constant number of steps.

In the case where we have only a finite number of calendar queues, the queues can be rolled over and re-used as their time expires.

Due to the computation of the expected arrival times, the traffic parameters are enforced at every switch along a channel's path. This rate control process has some similarities with a rate-based scheduling mechanism described in [Kalmanek 90].

The reconstruction process achieved by the regulator described in Figure 3-10 is only partial since it can only delay cells. Thus, if two cells were separated by a large amount in a previous node, the regulator cannot bring them closer together since it cannot determine how much extra time the first cell should be held without receiving the second. Thus, a few large inter-cell intervals will remain even after the regulation process. Note that this poses no problems for the resource allocation algorithms.

The histogram of the inter-cell intervals on a tagged channel with periodic traffic traversing a node is shown in Figure 3-12. At the output, only 37.9% of the cells preserve their spacings with respect to the previous cells. The distribution of new inter-cell intervals about the mean is almost symmetrical. The histogram was obtained for earliest deadline first scheduling with 50% server utilization. The traffic pattern on the cross-channels was uniform, and the other details were as described in Section 3.3. As can be seen, the traffic pattern, which at the input was a periodic arrival process, showed a spread about its mean after the first node.

The results of the partial reconstruction of the traffic stream are shown in Figure 3-13. With partial reconstruction, almost 99.9% of the cells were restored to

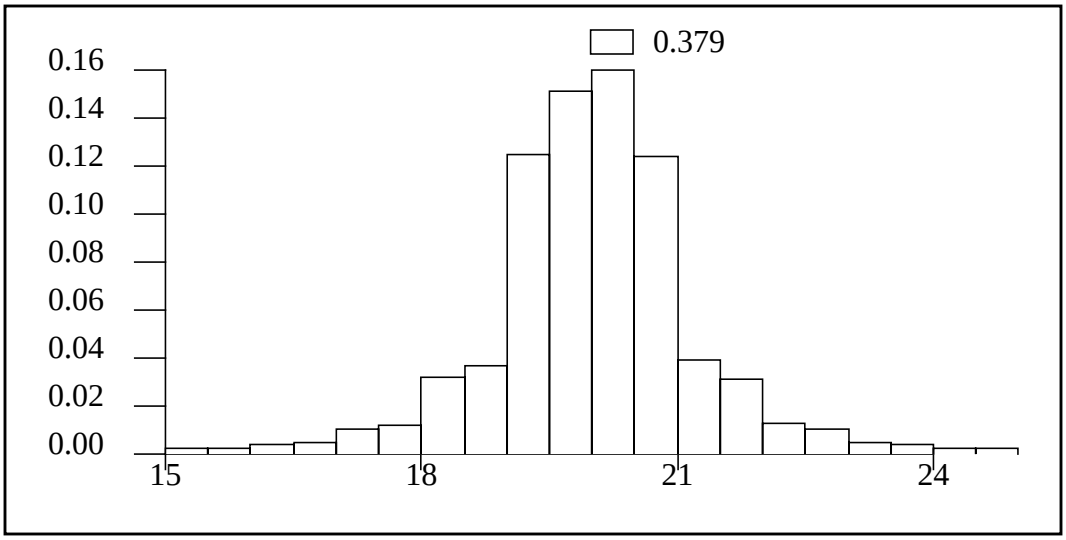

Figure 3-12 The distribution of inter-cell intervals at the output of a node with an EDF scheduling discipline. The original pattern consisted of a regular stream of arrivals at intervals of 20.0 time units. At the output, only 37.9% of the cells preserve their spacing with respect to the previous cell. The distribution about the mean is almost symmetrical.

the original cell distribution but about 0.1% were spaced too far apart. The partial reconstruction process does ensure that no two cells arrive closer than the minimum spacing x_{min}.

A perfect reconstruction of the input traffic pattern can be done by a modification of the regulation process. Since a full reconstruction is required in order to provide performance guarantees about delay variation (or delay jitter), we describe the full reconstruction scheme in Chapter 4.

3.6 Changes in Traffic Dependencies

Statistical multiplexing is a commonly used technique to increase network utilization. It attempts to take advantage of the fact that not all channels send data into the network at the same time. Hence, the network can overbook its resources (to a limited extent) and operate more efficiently. Overbooking of resources results

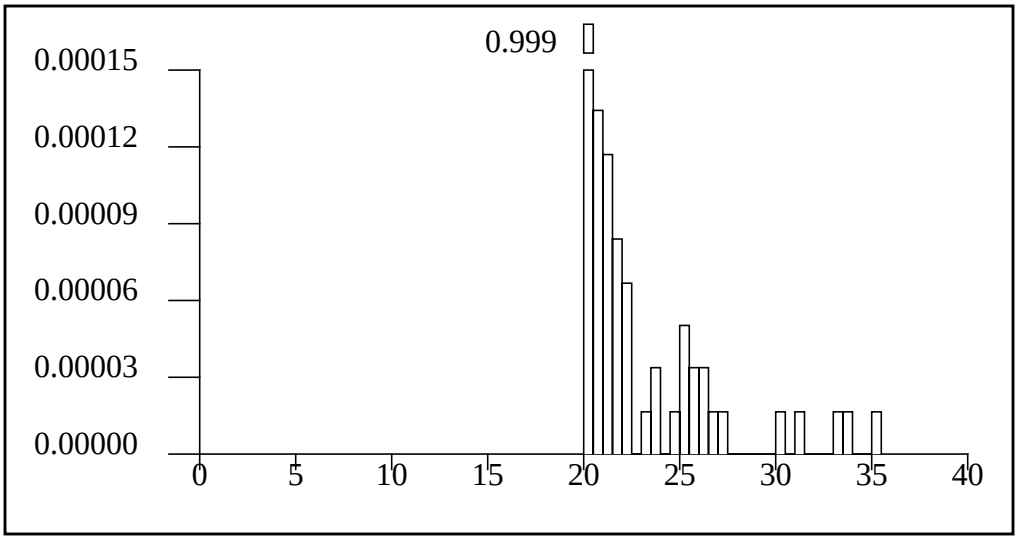

Figure 3-13 The distribution of inter-cell intervals after partial reconstruction. About 99.9% of the cells were restored to the original inter-cell interval. However, a few of the cells were spaced too far apart to be dealt with properly by the partial reconstruction process. Notice that the partial reconstruction process ensures that no cell is closer together than the minimum spacing x_{min}.

in a non-zero probability that too many channels be active simultaneously, and thus the network may not be able to meet their performance requirements. In order to provide a statistical performance guarantee, the network must be capable of computing this probability every time a new channel establishment message is received.

Such probability computations invariably must make some assumptions about the dependencies between the traffics on different channels. A simplifying assumption that is often made is that traffic patterns on any two channels are mutually independent of each other. In other cases, a client may declare the existence of dependencies among a set of channels that it wishes to establish. However, the independence assumption (and the client-specified dependencies) are valid only at the first node along the path of the channel(s). If two channels share one or more nodes along a path, their traffic patterns might become dependent on one another.

This may invalidate the computation of the probability of missing a performance bound.

We attempted to find out whether any significant dependencies can develop in the network by analyzing a setup similar to that described in Section 3.3. The only difference was that we considered two tagged channels, on which traffic was originally independent. Instead of using a periodic traffic on the tagged channel(s), we used the same traffic pattern as on the cross channels. The model of the network we consider is shown in Figure 3-14.

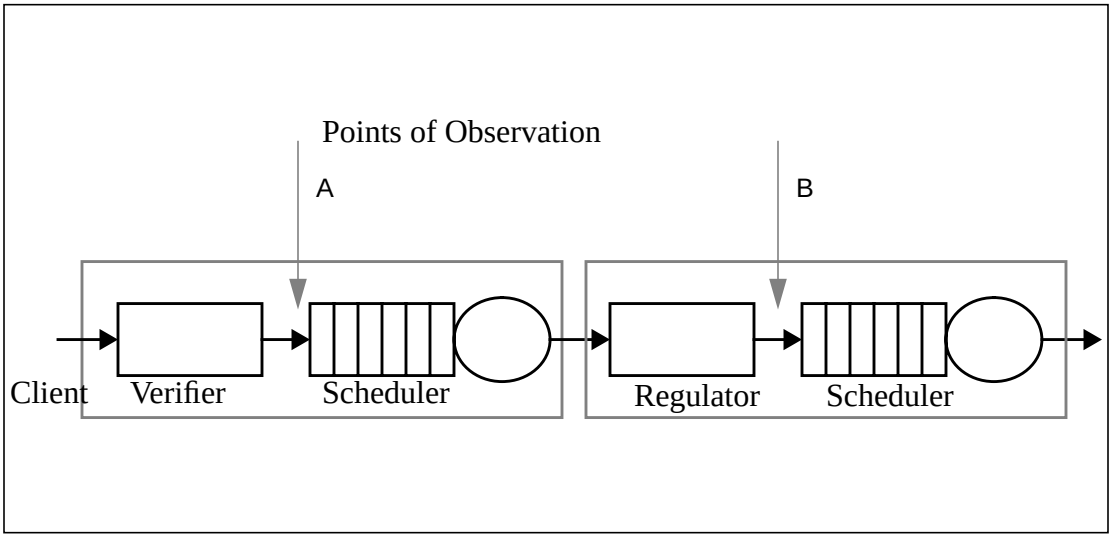

Figure 3-14 The model used for analyzing the development of correlations in a network.Two tagged channels follow the path indicated by the means of thick arrows. All the data shipped by a client passes through a verifier (or regulator) which ensures that the client is obeying the traffic contract. Some correlations may develop because cells along the two channels may interact in the scheduler. The channels share another scheduler along the path, after passing through the regulator in the next node. We want to find out if the correlation assumptions made at point A are valid also at point B.

One metric for measuring the dependencies between the two tagged channels is the cross-covariance of the traffic on the two channels. The cross-covariance between the two streams can be defined in two ways.[5]

Suppose the inter-cell intervals on the first tagged channel are $X_1, X_2, \ldots X_n$, and those on the second tagged channel are $Y_1, Y_2, \ldots Y_n$. Then the covariance at lag *k* between the two streams can be defined as

$$C1_k(X, Y) = Covariance(X_i, Y_{i+k}) \qquad (3\text{-}1)$$

This is the *covariance of intervals* at lag *k* for the two processes.[6]

A similar covariance can be defined by considering a number of cells in some fixed time interval, rather than looking at individual inter-cell intervals. Suppose the time axis were divided into intervals of length *T* each, with $N1_1, N1_2, \ldots$ cells arriving in each successive interval of length *T* along the first tagged channel, and $N2_1, N2_2 \ldots$ cells arriving along the second tagged channel. The correlation between the two streams can be measured by the covariance of the counts of cells in these intervals.

$$C2_k(X, Y) = Covariance(N1_i, N2_{i+k}) \qquad (3\text{-}2)$$

This is the *covariance of counts* at lag *k* for the two processes.

We now examine how these covariances change when the two channels share a link.

Dependencies between different channels may develop because of two major factors: (1) if the traffic on a channel is dependent on network load (for example, if there is some feedback from the network to the client, as in a reactive congestion control scheme), and (2) if different cells on different channels experience

---

5. Other metrics can be defined, but analyzing the changes in the covariance coefficients is sufficient to predict the changes in traffic dependencies.
6. Notice that the *Cov(X,Y)* at lag *k* may be different from *Cov(Y,X)* at lag *k*.

different delays due to the load fluctuations at a shared server. In the predictive resource allocation models that we are examining, there is no feedback, so correlations can only develop because of distortions in the traffic stream caused by the scheduling mechanism.

How much distortion can the scheduler cause? In both the FCFS and EDF scheduling policies, the sheduler passes through periods of activity (busy periods) and inactivity (idle periods). Assuming that the start of a busy period is a regeneration point for the scheduler, we can assume that variations in the inter-cell intervals caused in two distinct busy periods would be independent of one another.

Suppose the inter-cell interval X_i happened to be changed to $\hat{X}_i = X_i + U_k$ in the k^{th} busy period, and that the inter-cell interval Y_j along the second tagged channel were changed to $\hat{Y}_j = Y_j + V_l$ in the l^{th} busy period. Then, we can assume that U_k and V_l are independent. This implies that [7]

$$C(\hat{X}_i, \hat{Y}_j) = C(X_i + U_k, Y_j + V_l) . \qquad (3\text{-}3)$$

Expanding the right-hand side of this equation, [8]we obtain

$$C(\hat{X}_i, \hat{Y}_j) = C(X_i, Y_j) + C(X_i, V_l) + C(U_k, Y_j) + C(U_k, V_l) . \qquad (3\text{-}4)$$

Since the k^{th} and l^{th} busy periods are different, and U_k and V_l are independent, also X_i and V_l are independent, since the effect of any cell can only last till the current busy period ends. Similarly, we can argue that Yj and U_k are independent. As a result, we obtain that

---

7. For ease of notation, we shall use *C(X,Y)* to donate *Covariance(X,Y)* in the rest of this chapter.
8. *C(X,Y)* here may stand for either *C1(X,Y)*, the covariance of intervals or *C2(X,Y)*, the covariance of counts.

$$C(\hat{X}_i, \hat{Y}_j) = C(X_i, Y_j) . \quad (3\text{-}5)$$

Although we have considered inter-cell arrivals, a similar argument can be made for the number of cells that may have been serviced in any given busy period.

3.6.1 Changes in the covariance of intervals

Suppose that the duration of the maximum busy period at the scheduler is β_{max}. The value of β_{max} can be determined by the number and traffic parameters of the channels accepted at the scheduler. This value determines the maximum lag at which the output covariances (that is, the covariances at point B in Figure 3-14) can be different from the input covariances (that is, the covariances at point A in Figure 3-14).

If x_{min} [9] is the minimum inter-cell interval of a tagged channel, then the maximum lag till which the covariances can change is given by β_{max}/x_{min}. The typical value of these maximum lags in our simulations is roughly 5. The value corresponds to the maximum number of cells from that channel that can be present in the same busy period at the node.

Furthermore, due to the partial reconstruction mechanism, the traffic patterns at points A and B in Figure 3-14 are almost identical (only 0.1% of the inter-cell intervals did not have the same value at points A and B in the simulations reported in Figure 3-13). Assuming that the same accuracy of reconstruction holds for most of the cases, any change in the covariance properties of the cell streams would be very small.

9. If the two channels have different x_{min}, then we consider the smaller of the two values.

3.6.2 Changes in the covariance of counts

Suppose we use a time period T for counting cells which is larger than the maximum busy period β_{max} at the node. In this case, any two intervals which are at a lag of 2 or more must have the node becoming idle at least once between the end of the first interval and the start of the second interval. By equation (3-5), the covariance at lags 2 or more should not change.

Thus, the only changes in the covariance of counts can occur at lag 0 or 1. Notice also that, if the covariance at lag 0 increases, the covariance at lag 1 should decrease and vice-versa. A decrease in the correlation at lag 0 means that less cells from the second tagged channel tend to be in an interval if there are more cells arriving on the first channel in that interval. These cells will be leaving in the next interval, thus increasing the covariance at lag 1. A similar argument can be used to show that a increase in the correlation at lag 0 would cause a decrease in the correlation at lag 1.

Thus, the changes in the covariance of counts are very limited.

3.6.3 Simulation verification

In order to assure us of the correctness of the analyses presented above, we looked at the actual changes in the covariances using simulation. The analysis was based on some restrictive assumptions like that of independence between busy periods, and we wanted to explore whether the simulation setup behaved differently because these assumptions were not satisfied.

The simulation setup for the correlation studies is shown in Figure 3-15.

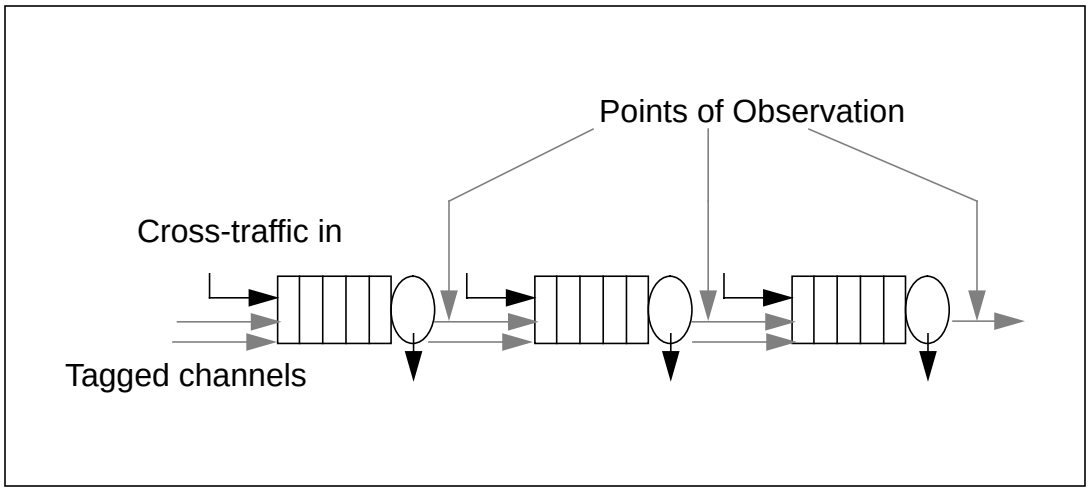

Figure 3-15 The set-up used for correlation studies. Two tagged channels passed through a number of nodes. The traffic on the two tagged channels were uncorrelated at the beginning of the path. The two channels were examined at the points indicated. The relative covariances were found to be the same for a large variety of traffic arrival patterns and network load utilizations.

Two tagged channels passed through a number of nodes, each loaded with cross-traffic bringing the overall node utilization to about 60%. The traffic patterns on the two tagged channels were independent of each other initially. We tried to measure the correlations at a number of observation points.

All the four traffic patterns were experimented with at different node utilizations along the channel's path. We also experimented with variable numbers of cross-traffic channels and with situations in which the busy period was likely to be long. However, in none of the cases we found any significant change in the covariance coefficients (as specified by equations (3-1) and (3-2)) up to a path-length of 5 nodes. Even the changes in covariances at small lags were not significant. This leads us to believe that the correlations between different channels are not likely to change appreciably in a network.

As a result of this observation, it is possible to assume that the dependen-

cies between two channels declared by the clients at the boundary nodes of the network hold for all common nodes along the path of those channels.

3.7 Verification of the Traffic Contract

The verification of traffic characteristics is a must since a malicious user could send cells into the network at a much higher rate than the declared maximum or average value. The same effect might be caused by a failure in the sending host or in a switch. If we do not take appropriate countermeasures, such malicious or faulty behavior can prevent the satisfaction of the delay bounds guaranteed to other clients of the real-time service, thereby damaging the clients and destroying the specified functionality and performance characteristics of the service. The enforcement of the traffic specifications can be termed *rate control*. Rate control can be done by forcing the “offending” cells to wait till the time they were actually supposed to arrive.

When buffer space is limited, cells that arrive too much in advance of their proper time might even be dropped because of buffer overflow. Of course, a sufficient amount of buffer space will have to be allocated to ensure that cells of a well-behaved channel are not dropped because of network load fluctuations.

Rate control can be done by a process identical to the partial reconstruction scheme described in Section 3.5. The verification of the client’s traffic parameters needs to be done only at the first node in a channel’s path. If the client is obeying its contract, then the reconstruction process would be an identity, that is, the holding time of a cell in the calendar queue mechanism would be zero. On the other hand, if the client is cheating, either intentionally or unintentionally, the calendar

queues will hold the cells until they are actually expected to arrive.

Thus, the calendar queue mechanism is required at all nodes in the network. At the boundary nodes of the network, the mechanism is used for contract verification, while at the intermediate nodes it is used for reconstruction of the original traffic pattern.

Chapter 4: The Performance Contract

4.1 Introduction

The performance contract spells out the obligations of the network towards the communicating clients. As mentioned in Chapter 1, we need to address four major issues (specification, mapping, mechanism and verification) in this component of the contract.

By specification of the performance contract, we mean that we need to compile a list of possible grades of QOS (Quality of Service) that clients are allowed to ask from the network. Clients are likely to be interested only in end-to-end performance guarantees, and we need a scheme to map the end-to-end performance guarantees into per-node performance guarantees. Given the performance requirements of the clients, we need a mechanism that the network can use to meet those requirements, and a scheme for the clients to verify that the network is fulfilling its promises.

This chapter discusses these issues and is structured in the following manner. In Section 4.2, we explore some of the possible QOS menus that can be provided to clients by the network. In the next section (Section 4.3), we describe the alternative methods by which end-to-end performance requirements can be broken down into per-node performance requirements. In Sections 4.4, 4.5 and 4.6, we propose three possible QOS menus, and discuss the scheduling and admis-

sion control schemes that can be used to support the requirements. Finally, in Section 4.7, we explore the different schemes that a client can use to verify that the network is indeed performing as required.

4.2 Specification of the Performance Contract

The specification of a performance contract can be done by choosing the QOS (Quality of Service) menu that the network will offer to its clients. The choice of this QOS menu depends on the variability of the requirements of different applications and on the operational characteristics of the network.

From the point of view of an application, the perfect network would look like a zero-delay line. Any unit of data put into the network by the sending side would be received at the other end instantly. The stream of cells generated by the sender would be identical to the stream of cells obtained by the receiver. It would perform as though the sender and the receiver were at the same location.

A real network, however, differs from the perfect network in three major ways: it has a non-zero delay, the delay is subject to some variation or jitter, and some of the cells may be lost in the network. Such a network must be able, however, to control these aspects of its performance to meet the requirements of the client. Thus, the most general specification of a client's performance requirements needs to make use of three metrics: one to measure delay, one to measure delay variation, and the third to measure the loss rate.

As an example, the requirements of a client can be specified by providing a bound on the maximum delay to be experienced by a cell on the channel, a bound

on the difference between the maximum and the minimum delays experienced by a cell on the channel, and a bound on the probability that a cell is lost in the network.

While a client may require a bound on any of these three metrics, some of the requirements may be trivial to satisfy. For example, if the network never introduces any significant delay variation or delay jitter, then it is unnecessary for the client to ask for a bound on delay jitter. In the network model proposed in Chapter 1, one or more of these performance bounds may be unnecessary under different conditions. In the remainder of this section, we attempt a back-of-the-envelope calculation to determine the conditions under which each of the three bounds would be significant.

Let us recall from Chapter 1 that our model of the network consists of a mesh of queueing servers with arbitrary topology. Assume that each queueing server is identical and has a constant service time μ (μ depends on the link speed). Assume also that B buffers are available at every queueing server. If we further assume (for the sake of simplicity) that the loss rate of a channel can be approximated by the probability that more than B buffers are occupied in an M/M/1 model with infinite buffers, the loss rate at utilization ρ equals ρ^B at every queueing server [Wolff 90].

Following this model, about 100 buffers are needed to ensure a loss rate of less than 10^{-15} for a utilization of approximately 70%, and roughly 65 buffers are needed to ensure a loss rate of 10^{-10}. Keeping in mind that traffic may be burstier than the Poisson process, thereby requiring more buffer space than in the M/M/1

model, B must be at least 100 to ensure a loss rate of 10^{-10} or less at every node. If we want to have static buffers allocations for different channels (the previous calculations apply when buffer space is dynamically shared between channels), B would need to be much higher.

With a constant service time equal to $1/\mu$ for every cell, and at most B cells present in the queue at any time, the maximum delay of a cell at a queueing server is B/μ. With a cell-size of 53 bytes (424 bits), the maximum delay for a link bandwidth of r Mbps is $(424B)/r$ milliseconds.

Let us consider a channel that has a route about 6000 km long, i.e., with a propagation delay of approximately 30 ms, and with 3 switches along its path. Assume that delays need to be controlled only when they exceed more than 20%[1] of the propagation delays. Otherwise, only the loss rates are of interest. In this case, delays need to be controlled if they exceed 2 ms at every switch. If each switch has a thousand cell-sized buffers (B = 1000), our back-of-the envelope calculations in previous paragraphs suggests that we need to control delay and delay variation for link speeds up to approximately 850 Mbps. If the channel were only 1000 Km long, with only one switch along its path, delays would need to be controlled for speeds up to 1.7 Gbps.[2] At higher speeds, only the loss rates need to be controlled. Since queueing delays are the main cause of delay variations, delay variations need to be controlled whenever delays need to be controlled.

---

1. A number pulled out of the hat for the sake of back-of-the-envelope calculations.
2. For channels with smaller propagation delays, a reactive scheme can also be developed to guarantee performance. However, in networks where channels with both small and large propagation delays may coexist, a proactive scheme provides a uniform framework to guarantee performance for both types of channels.

Since current research networks have an operational speed of 45 Mbps or 600 Mbps, we conclude that delay and delay variations are likely to be significant in the short or medium-term future, even when networks operate at a few Gbps speeds. For very high speed networks in the long-term future, operating at hundreds of Gbps speeds, only the loss-rates would be significant.

Depending on the operational characteristics of the network, different QOS menus should be chosen. Since it is unlikely that a menu appropriate for all types of networks can be found, we will discuss only three menus in this chapter.

QOS menu 1 takes into consideration all the three performance metrics, providing channels with bounded delays, delay variations, and loss rates. This menu is the most general case we consider.

QOS menu 2 deals with only the loss rate metric. This is a much simplified version of QOS menu 1, and is likely to be more efficient for the cases in which we have channels with long routes (and large propagation delays) and very high link speeds. We can say that this menu is best-suited for the case when bandwidth is abundant, and buffers are scarce.

QOS menu 3 deals with a case which is in between the first two menus. It offers some limited classes of service. Each class of service has a delay bound, a delay variation bound and loss rate bound associated with it.

QOS menus 2 and 3 can be considered as simpler versions of QOS menu 1. The reader may wonder why we are considering these menus instead of merely solving the more general problem of providing QOS menu 1. The main advantage

in considering the two restricted menus is that the mechanism to provide them is simpler than that required for QOS menu 1.

4.3 Mapping the Performance Contract to the Network

In the contract based model assumed in this thesis, clients need to specify their performance requirement at channel establishment time. Clients are likely to be interested only in the global performance they will be getting, i.e., in the end-to-end delays or loss rates the network has to offer. They will generally not be concerned with the local performance guarantees at the switches. However, the switches need to operate on the basis of their local performance guarantees. Thus, some scheme is needed to convert the global performance guarantees into local performance guarantees.

A number of different approaches can be used for this conversion. They are summarized below.

(A1) The simplest possible approach would be to *predetermine the local performance bounds* at each node depending on the QOS menu entry chosen by the client. Thus, one option in the performance menu (say, QOS type 0) would ensure that no cell on the channel is to have a delay larger than 0.5 ms at any node on the way. Another option (QOS type 1) may specify that the loss rate at any switch should be no more than 10^{-9}. This would work very well in the case where the menu options are limited and a simple direct mapping can be made. This scheme also allows for fast establishment for a channel, where the data cells can follow the establishment message immediately, each node being prepared to accept these cells as soon as it processes an establishment request (assuming

optimistically that the establishment will be successful in most of the cases).

The disadvantage of predetermining the local performance bounds is that clients are given performance guarantees depending on the route selected for the channel, and thus the size of the network is not transparent to them.

(A2) Another approach would be to use a *centralized performance server,* one per administrative domain, to perform this division of global performance bounds into local performance bounds. The advantage of a centralized performance server is that it is easier to manage information about the global state of the network centrally, provided the performance server is not overloaded and does not crash. The client has to use a 2-step channel establishment procedure. First, it calls the performance server to assign a path to the channel, and to break up its global performance requirements into local performance requirements for nodes along that path. Then, the client can use this information for a fast channel establishment along the selected path. Alternatively, the performance server can establish the channel on the client's behalf. After the channel is established, the client is informed about the first node on the path and is given a channel identifier, which it can use for communication on the channel. The two schemes are shown in Figure 4-1.

(A3) The third method is the most general one, and does not require a centralized server. In this *distributed scheme*, the call establishment message passes along the nodes as the route for the channel is selected by means of a suitable routing algorithm. At each node, the local state is examined, a value for the local performance bound is proposed and included in the establishment message, and

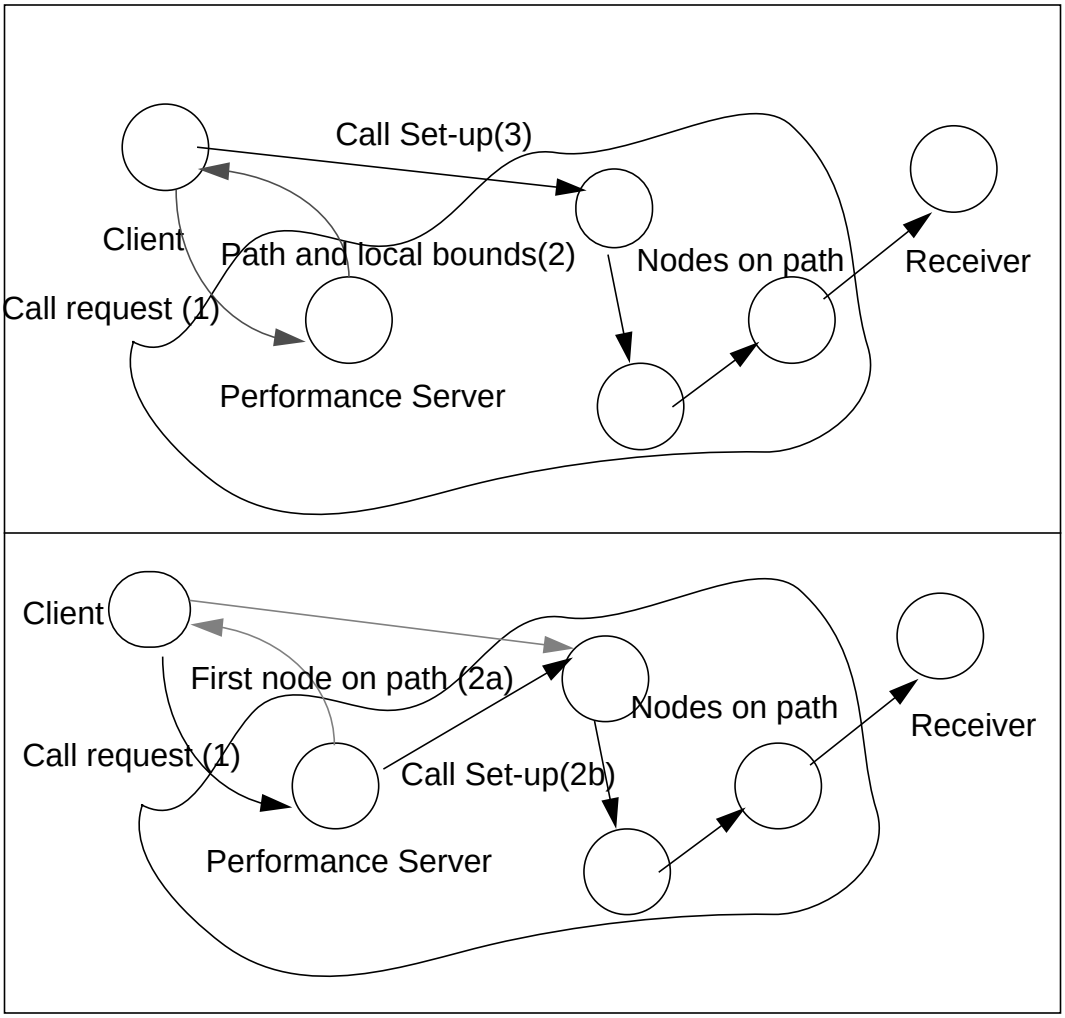

Figure 4-1 The division of global performance bounds into local performance bounds using a centralized performance server. The upper part of the figure shows an approach in which establishment occurs in two phases (involving the client twice), while the lower part shows a method in which the client is involved only once. The dotted lines indicate the operations that may take place after the call-set has been done.

the next node along the route is selected[3]. The destination node examines all the proposed local performance bounds and determines whether they are adequate to satisfy the global performance bound. If so, the destination may relax the performance bounds of some or all of the intermediate nodes along the path of the channel. On the return trip of the establishment message, these (possibly) modified

3. The scheme can also be used with source routing, where the entire route of the channel is determined by the source node.

performance bounds are committed at the local nodes. This solution is distributed, and can span multiple administrative domains. This solution is used in the Berkeley approach to real-time channel establishment [Ferrari 90a], and is illustrated in Figure 4-2.

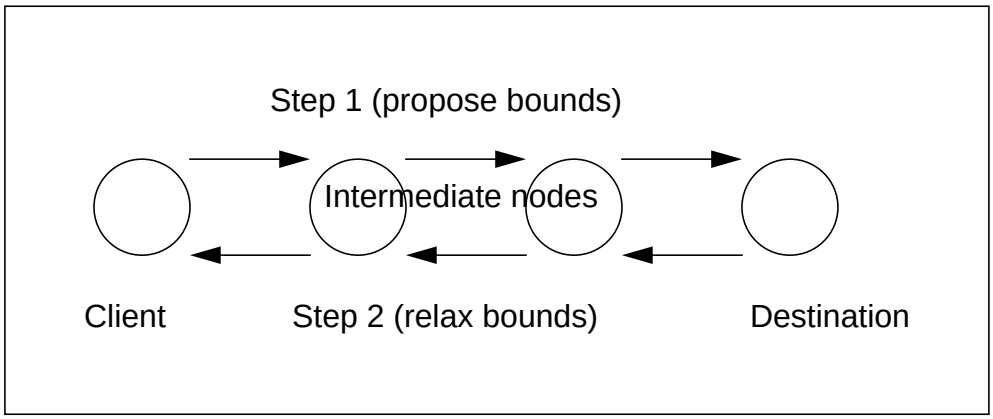

Figure 4-2 The distributed approach to divide performance bounds. In the forward pass, nodes propose a value for their performance bounds, which may be relaxed by the destination. On the reverse path, the (possibly modified) performance bounds are committed at the intermediate nodes.

The mechanism used for this division of global performance bounds (delay, delay variation, and loss rate) produces the local performance bounds at each of the intermediate nodes.

The end-to-end performance bounds can be divided among the intermediate nodes according to a number of policies. As an example, consider an end-to-end delay bound of 100 time units for a channel that passes through 10 queueing servers. One policy could be to assign local delay bounds equally, that is, allocate a local delay bound of 10 units at each of the three queueing servers. Another policy could be to assign larger delay bounds to more heavily loaded or more congested queueing servers. The efficiency of the latter policy depends upon the network topology and the nature of channel establishment requests. Since these

properties are not yet known for future high-speed networks, we prefer for the moment, the simpler policy that divides the performance bounds equally among all nodes along the path.

Performance bounds involving delays are additive, while performance bounds involving loss rates are multiplicative. Therefore, an end-to-end delay bound of 100 time units, in our example, would translate to a local delay bound of 10 units at each of the 10 queueing servers along the path of the channel, while an end-to-end loss rate of 10^{-9} for the channel would translate to a loss rate of 10^{-10} at each of the queueing servers.

In the rest of the chapter, we will assume that the local performance bounds have been determined, and will derive equations that tell us when it is safe to accept a new channel.

4.4 QOS Menu 1: Specification and Mechanism

The first menu is the most general one among those we will consider. In this case, the clients are offered the choice of specifying their performance requirements by means of three parameters, *D*, *J* and *W*.

D is the bound on the end-to-end delay a cell may experience on the channel. The bound is deterministic in the sense that it is to be met under all circumstances, except in the case of a network failure.[4] Since deterministic guarantees tend to be expensive in terms of network resources, some clients may be willing to settle for a statistical guarantee. In this case the delay is characterized by two

4. By a network failure, we mean a node crash or a link failure, whether due to hardware or software failures or both.

parameters, *D* and *Z*, the new parameter *Z* being the probability that the delay will be below the bound *D*. Notice that a value of 1 for *Z* implies that the delay guarantee is deterministic, while a value of 0 for *Z* implies that no guarantees are being given regarding delays.

J is the parameter describing delay jitter, defined as the maximum difference between the delays experienced by any two cells on the channel. Since there is a delay parameter associated with the channel as well, the delay of a cell is constrained to be within the interval (*D-J*, *D*). Thus, both the minimum and the maximum delay for the channel are being specified. The *J* parameter can also come with a statistical flavor, i.e., with a probability parameter *U*. In this case, the probability that the delay of a cell is within *D-J* and *D* is at least *U*.

W is the cell loss parameter. It determines the acceptable loss rate for the channel. The probability that a cell sent on the channel is delivered successfully to the receiver must be at least *W*.

The menu consists of any combination of (*D,Z*), (*J,U*) and *W*. This is the same menu that is offered by the real-time services described in [Ferrari 90c]. Some simple conditions must be satisfied for the service description to be well-formed. Assuming that D^p is the propagation delay for the channel, we must have:

$$D \geq D^P, \qquad (4\text{-}1)$$

$$0 \leq W \leq 1, \qquad (4\text{-}2)$$

$$0 \leq Z \leq 1, \qquad (4\text{-}3)$$

$$0 \leq U \leq 1, \qquad (4\text{-}4)$$

$$J \leq D - D^P, \qquad (4\text{-}5)$$

[5]

and

$$W \geq \varepsilon . \tag{4-6}$$

where ε is the line error rate.

These conditions do not impose any significant restrictions on the client. In fact, these are the natural constraints that must hold in any network. These parameters, by means of one of the procedures described in Section 4.3, are divided into local performance bounds for every intermediate node *i,* obtaining the parameters d^i, z^i, j^i, w^i, u^i which are self-explanatory[6]. These bounds have to be chosen so as to allow for the end-to-end performance parameters to be satisfied. Some very simple rules that ensure this are:

$$D \geq \sum_i d^i + D^p, \tag{4-7}$$

$$J \geq j^n, \tag{4-8}$$

where n is the last node, and

$$W = \prod_i w^i, \tag{4-9}$$

$$Z = \prod_i z^i, \tag{4-10}$$

$$U = \prod_i u^i. \tag{4-11}$$

All of the conditions are based on the most pessimistic assumptions. Thus,

---

5. A higher bound on delay jitter can be specified by the client, but is unnecessary.
6. In our notation a capital letter denotes an end-to-end performance bound, while a lower case letter denotes a per-node performance bound. A superscript is used as an index for the node, and a subscript is used for a channel. Thus D_i is the end-to-end delay of channel i, while $d^n{}_i$ is the local delay bound of channel i at node n. When there is no ambiguity, we will drop either the subscript or the superscript.

a cell is considered delayed even if it is delayed at only one node. The condition is pessimistic, and thus safe.

An explanation for equation (4-8) is in order. In this case, we are assuming that the full reconstruction policy referred to in Section 3.4 is being used. The full reconstruction policy and a detailed explanation for equation (4-8) are discussed in the next subsection.

4.4.1 Reconstruction policy

As observed in Section 3.5, we model an output ATM link as a regulator-scheduler pair. The regulator has the responsibility for reconstructing the original traffic pattern of a channel. A partial reconstruction scheme was described in Section 3.5, and we now describe a full reconstruction scheme.

We first explain why a full reconstruction scheme may be needed. If we want to approximate the constant delay line abstraction, the input traffic pattern on a channel (that is, the traffic pattern at the entrance to the network) must be maintained at the point where it leaves the network. The only deviation allowed for it is the delay jitter bound specified by the client.

One way to achieve the constant delay line abstraction is to reconstruct the input pattern at each node. In order to do so, the end-to-end delay bound is broken into a set of local delay bounds. The scheduler in each node and the regulator in the next node together provide a constant delay. The value of this local constant delay is fixed by the mapping operation described in Section 4.3 so that the local delays along the path of a channel satisfy equation (4-7).

If the scheduler in a node is able to tell the exact value of the delay experienced in it by a cell to the next node, the regulator downstream can take the following action: if the delay experienced is less than the delay bound, the cell is held for the remaining amount of time (the delay bound minus the actual delay), otherwise it is delivered to the scheduler immediately. If the delay experienced in the scheduler is less than the delay bound, then the holding in the next regulator ensures that each cell experiences a constant amount of delay in each node until the last scheduler along the path of the channel.

With this reconstruction scheme, the end-to-end delay variation of cells along a channel equals the delay variation of cells in the last scheduler. This provides the justification for equation (4-8). Since the scheme works only when each scheduler is able to meet its delay bound, the delay jitter guarantees are violated whenever the delay guarantees are violated, and we must have:

$$U \leq Z; \qquad (4\text{-}12)$$

if the n^{th} node is the last one, we have

$$d^n = j^n \leq J.^7 \qquad (4\text{-}13)$$

Equations (4-12) and (4-13) state that the end-to-end jitter is limited by the delay bound at the last node. In order to obtain a scheme that provides bounds on delay and delay jitter (*D,Z,J,U*), it would be sufficient to develop a scheme that provides only delay bounds (*D*, *Z*). In view of the preceding, we will not talk of the delay jitter parameter *J* or the statistical delay jitter parameter *U* in the subsequent sections. In those cases where equation (4-12) is not satisfied, we take *Z* to be the

7. To be more precise, the delay jitter should not include the constant service time for an ATM cell at a queueing server, which is part of the delay bound. This makes no difference to our arguments.

maximum of U and Z. This increase in the value of Z is sufficient to meet the required performance bounds.

Notice that the full reconstruction scheme assumes the availability of sufficient buffer space to reconstruct the input traffic pattern. Since the maximum delay of a cell in a regulator is the same as the maximum delay of a cell in the scheduler at the previous node, the reconstruction process requires the same amount of buffer space as the one allocated for the channel at the previous node. In a homogeneous network, with identical nodes and (most probably) identical delay bounds at all nodes, the reconstruction process requires doubling the buffer space at each switch.

4.4.2 Scheduling

In order to provide for any possible delays and loss rates, the scheduler at each node must give proper priority to cells that have more stringent delay or loss requirements. We can identify three distinct classes of channels in this case: the channels with z equal to zero can be categorized into one class, the channels with z equal to one can be categorized into another class and those with other values of z can be put into yet another class. These three classes of service are the best-effort, deterministic and statistical service categories. It is possible to break the statistical service category into a number of different classes, each class corresponding to a different range of z values. Furthermore, the network may decide to put channels with z greater than some particular threshold into the deterministic category, and to demote the channels with z less than some other threshold into the best-effort category.

A good scheduling policy for this menu should take into account the different delay and loss requirements of the cells present in the queue. Thus, it should take into account all the three different performance parameters dealing with delays or loss rates, namely *d*, *z* and *w.* Thus, the scheduling policy must give higher priority to cells with more stringent delay requirements. Similarly, cells from a channel with a higher value of *z* must be given a higher priority than those with a lower value, at least in those cases in which the two cells have the same delay requirements. Cells from channels with a lower value of *w* must be discarded before cells with a higher value of *w* are. Thus, all the three different parameters affect our choice of the scheduling policy.

Taking into account the fact that the *w* parameter can be used effectively to determine whether or not to allow a cell into a node's buffers, we can simplify our approach by subdividing the problem into two components: (1) a scheme to admit cells into the node, which depends only on the buffers available for the channel and on its *w* parameter; and (2) a scheme to service cells already in the buffer, which takes into account the delay parameters *d* and *z*.

We take the delay parameter into account by using a variant of deadline-based scheduling [Liu 73]. A cell from channel *i* which enters the node at time *t* is assigned a deadline equal to $t+d_i$, the latter being the delay bound associated with that channel at the node. When the choice of the next cell to be shipped is made, the cell with the earliest deadline is serviced. In this way, the cells from delay-sensitive channels get priority over the cells from other channels.

In order to account for *z*, the other delay parameter, the deadline given to

the cells must also depend on the z value of the channel. However, there appears to be no simple way to determine what this priority should be. Thus, we propose the simple rule that, whenever a cell from a channel with a lower *z* value can cause a cell from a channel with a higher *z* value to be delayed, the latter should be shipped in preference to the former. In other words, a cell with a larger value of *z* should not miss its deadline because of a lower priority cell (one with a smaller *z* value). We also have to ensure that no cells that arrive later may miss a deadline because a lower priority cell was allowed to go through.

Let us assume that there are *M* levels of queues for the scheduler. At level *i*, the channels are guaranteed to have a *z* parameter higher than a certain value ζ_i. In other words, all channels that required their *z* values to lie between ζ_{i-1} and ζ_i are put at level *i.* This discretization is to ensure an efficient implementation of the scheduling policy.

For each of the *M* queues, a list of the cells to be shipped on the output link at that level is maintained. The list is kept sorted in the order of increasing deadlines. Whenever a cell is to be shipped, the deadlines of the cells at the heads of all the queues are compared, and the cell at the lowest possible level which can be transmitted without affecting the delay guarantees of channels at the higher levels is allowed to go. Within each sorted queue, no two cells are allowed to have the same deadline. The deadline of one of the cells is decreased (repeatedly if necessary) to ensure this condition.The admission control policy ensures that this decrease is always possible.

Buffer space may be shared dynamically among all the channels, or allo-

cated statically to individual channels, or managed according to some combination of the previous two schemes. For the sake of simplicity, we will consider only a static allocation of the buffer space. Each channel is assigned a certain number of buffers that it is allowed to use. Buffers not used by a channel are wasted.

Let us examine the admission control policy that must be followed with this scheduling policy and buffer management scheme.

4.4.3 Admission control

The admission control policy for this menu consists of a set of tests which can be used to determine whether the delay constraints of a given set of channels can be met. However, with the distributed method of delay allocation (see Section 4.3), the testing procedure must also be able to determine, during the forward trip of the establishment message, the values of the performance bounds to be offered. Thus, it must be able to determine the lower bounds on delays and loss rates that can be offered to the new channel. In this section, we attempt to find the set of admission control rules that will tell us when the performance guarantees of a set of channels are safe.

The admission control problem on a per-node basis can be stated as follows. Suppose there are N channels accepted at a node. The performance parameters of the i^{th} channel are d_i, z_i and w_i, and the traffic parameters are $x_{min,i}$, $x_{ave,i}$ and I_i. Can we develop a set of sufficient (perhaps not necessary) conditions which ensure that the performance requirements of all the channels will be satisfied?

Solution: Let us first attempt to develop a set of conditions which are suffi-

cient to ensure that delays experienced by a cell from channel *i* will be within its delay bound d_i with a probability higher than z_i.

Following the paradigm used in the Berkeley approach to real-time channel establishment [Ferrari 90a], we simplify the problem by identifying two potential causes for missing a deadline: (1) an arrival rate higher than the maximum possible service rate for an extended period, and (2) the presence of cells with conflicting delay requirements. We ensure that condition (1) does not occur with a probability exceeding $(1-z_i)$, and that condition (2) is never present in the absence of condition (1).

Condition (1) can be avoided by ensuring that

$$\sum_{i=1}^{N} \frac{t}{x_{min,i}} \leq 1, \qquad (4\text{-}14)$$

where t is the time required to transmit a cell on the output line, $x_{min,i}$ is the minimum inter-cell arrival time for channel i, and N is the number of channels present at the node (including the new one). This imposes a situation in which the worst-case arrival rate can never exceed the service rate.

In order to avoid condition (2), i.e., to avoid conflicts due to the presence of cells with different deadlines, we will attempt to find a bound on the maximum delay a cell can experience in a node with the deadline-based scheduling policy described in Section 4.3, provided equation (4-14) is satisfied. The delay bound offered to a channel is safe if it is larger than this maximum delay. (The maximum delays are computed by using Theorem 4-1 described below).

If condition (1) and (2) are both satisfied, the delay bound is always met. In those cases where we are allowed to miss a few deadlines ($z_i < 1$), we ensure that condition (1) holds with a probability higher than z_i, and that condition (2) holds whenever condition (1) holds. The computation of this probability is done by means of equations (4-20) and (4-21) to be presented later in this section.

In order to meet the loss rate performance guarantee, we ensure that we allocate sufficient buffers to each channel in the node. ∎[8]

We now prove the theorems required in order to complete this solution.

Theorem 4-1 Let channel i in a node have a local delay bound d_i, let K be the set of all channels in the node with local delay bounds smaller or equal to d_i, and let L be the set of channels with higher delay bounds. If the channels in sets K and L taken together satisfy equation (4-14), then $d_{max,i}$, the maximum delay of a cell on channel i with the deadline-based scheduling policy, satisfies the relation:

$$d_{max,\,i} \le \left((|K| + 1)\, t + \sum_{k \in K} \frac{d_i - d_k}{x_{min,\,k}} t \right), \qquad (4\text{-}15)$$

where $|K|$ is the number of channels in set K.

Proof: Let us define a busy period of a node as a continuous interval during which there is at least one cell present in the scheduling queue from any channel. Without loss of generality, assume that a given busy period started at time *0*. Let us mark a cell from channel i which arrived at time T and refer to it as the *tagged cell*. Recalling that the deadline-based scheduling scheme assigns a deadline

8. A ∎ marks the end of a solution or proof.

equal to $x + d_i$ to a cell arriving on channel i at time x, the deadline assigned to the tagged cell was $T+d_i$.

The delay experienced by the tagged cell in the queue is due to the servicing of cells which arrived during the same busy period and had a deadline smaller than that of the tagged cell. Set K consists of channels whose delay bounds are equal to[9] or smaller than that of channel i, and therefore a cell that arrived on channel $k \in K$ would have a deadline smaller than that of the tagged cell if it arrived in the interval $(0, T + d_i - d_k)$.

Suppose the tagged cell's delay in the scheduling queue is d, that is, it is serviced at time $T+d$. The delay of the tagged cell is the difference between the sum of the service times of cells that were serviced in the period $(0,T+d)$ and the length of the period $(0,T)$. The length of the period $(0,T)$ is obviously T.

Let us try to determine the maximum number of cells that can be serviced at the node in the interval $(0,T+d)$. This can be done by computing the maximum number of cells that can be serviced in that period on any of the other channels, belonging either to set K (those with lower delay bounds) or to set L (those with higher delay bounds).

We know that the tagged cell arrived at time T and had a deadline of $T+d_i$. Therefore, it may have had to wait for all the cells that arrived on a channel $k \in K$ in the interval $(0, T + d_i - d_k)$. In the worst case, there could be $\lceil (T + d_i - d_k) / x_{min,k} \rceil$ such cells. The maximum time spent servicing cells from channels in set K in the period $(0, T+d)$ is at most $\sum_{k \in K} \lceil (T + d_i - d_k) / x_{min,k} \rceil t$.

9. It follows that set K includes channel i as well.

Cells arriving on a channel $m \in L$ in the time-interval $(0, T+d_i-d_m)$ will be assigned a deadline lower than that of the tagged cell and serviced before it. However, some of the cells that arrive in the interval $(T+d_i-d_m, T)$ may also be serviced before the tagged cell. This situation may arise if such a cell has the lowest deadline among all the cells in the scheduling queue at a time when the tagged cell did not arrive. As a pessimistic assumption, we assume that all cells on a channel $m \in L$ that arrive in the time-interval $(0,T)$ are serviced before the tagged cell. The number of such cells on channel m could at most be $\lfloor T / x_{min,\,m} \rfloor$. We are using the floor operator here since cells that arrived on a channel in set L at the same time as the tagged cell will be serviced after the tagged cell, and therefore will not be serviced in the period $(0,T+d)$.

The floor operator ignores the situation when a cell with a larger deadline happens to arrive just before the tagged cell and is serviced promptly. However, there can be at most one cell of this type. Therefore, the maximum time spent in servicing cells from channels in set L in the period $(0, T+d)$ is at most $\sum_{m \in L} \lfloor T / x_{min,\,m} \rfloor t + t$.

Therefore, $d_{max,i}$, the worst-case delay of the tagged cell, satisfies the inequality:

$$d_{max,\,i} \le \sum_{k \in K} \left\lceil \frac{T + d_i - d_k}{x_{min,\,k}} \right\rceil t + \sum_{l \in L} \left\lfloor \frac{T}{x_{min,\,l}} \right\rfloor t + t - T, \qquad (4\text{-}16)$$

where t is the service time for one cell. Since $\lceil x \rceil \le x+1$ and $\lfloor x \rfloor \le x$, we find that:

$$d_{max,\,i} \le \left\lfloor \sum_{k \in K} \left(\frac{T + d_i - d_k}{x_{min,\,k}} + 1 \right) + \sum_{l \in L} \frac{T}{x_{min,\,l}} + 1 \right\rfloor t - T, \qquad (4\text{-}17)$$

which, after some simple algebraic manipulation, can be found to be equivalent to:

$$d_{max,\,i} \leq T\left(\sum_{m \in K \cup L} \frac{t}{x_{min,\,m}} - 1\right) + \sum_{k \in K} \frac{d_i - d_k}{x_{min,\,k}} t + (|K| + 1)\, t. \qquad (4\text{-}18)$$

If equation (4-14) is enforced, the first term in equation (4-18) is always negative, and we obtain equation (4-15). ∎

Theorem 4-2 If condition (4-14) holds at a node, the delay requirements of channel *i* will be satisfied if:

$$d_i \geq (|K| + 1)\, t + \sum_{k \in K} \frac{d_i - d_k}{x_{min,\,k}} t. \qquad (4\text{-}19)$$

Proof: follows from Theorem 4-1 and the fact that the maximum delay should be less than d_i. ∎

The preceding two theorems give us a scheme to guarantee deterministic delay bounds. Our approach to providing statistical delay bounds was to allow equation (4-14) to be violated sometimes. Furthermore, the scheduling policy described in Section 4.3 had *M* scheduling levels with level *l* having a probability of deadline overflow less than ζ_l.

Let us assume that the worst case for violating condition (4-14) is when all the channels are either transmitting at their peak rates or not transmitting at all. The probability p_i that a channel *i* is transmitting is given by $x_{min,\,i} / x_{ave,\,i}$, and the probability that a given combination C of channels is transmitting at any time is given by

$$Pr(C) = \left(\prod_{i \in C} p_i\right) \prod_{k \notin C} (1 - p_k) \qquad (4\text{-}20)$$

Let us consider the channels that are accepted at the scheduling level *j,* enumerate the combinations which do not satisfy condition (4-14), and add up all these probabilities. This gives the probability of deadline overflow at scheduling level *j* for the statistical guarantees. Let the probability of overflow (that is the value of equation (4-20)) at level *j* be given by P_j. The admission control test would consist of verifying that at level *l:*

$$\zeta_l \leq \sum_{j \leq l} P_j \qquad (4\text{-}21)$$

Although the evaluation of the probability as described by equation (4-20) may appear to take exponential time, accurate fast approximations can be obtained, such as the one described in Appendix 1.

We must ensure that the delay bounds are met whenever condition (4-14) is satisfied. Thus, we need to verify that equation (4-19) holds for all combinations of channels in levels (*0... l-1*) which satisfy condition (4-14).

To provide channels with loss rate guarantees, we need to determine the sufficient amount of buffer space to allocate to each channel.

Theorem 4-3 If a cell on channel *i* does not spend more than *T* time units at a node, and the channel has a minimum inter-cell interval $x_{min,i}$, then at most $\lceil T / x_{min,i} \rceil$ cells from that channel can be present at the node at any time.

Proof: Suppose *m* cells are present at the node in the worst case. The first cell must have experienced a delay of at least $mx_{min,i}$, since no two cells arrive closer than $x_{min,i}$. Since this time has to be less than *T,* it follows that $m \leq \left\lceil \frac{T}{x_{min,i}} \right\rceil$. ∎

In order to obtain the maximum possible buffer space required by a channel, we notice that a cell on a deterministic channel i (with z_i =1) never stays at a node longer than its delay bound d_i, and at most $\lceil d_i / x_{min,i} \rceil$ cells from that channel may be present at the node at any time. Also, suppose I_{max} is the maximum value of I among all the channels. Let us enforce the condition

$$\sum_i \frac{t}{x_{ave,i}} < 1, \qquad \textit{(4-22)}$$

where i is any channel at the node. Since the average inter-cell interval $x_{ave,i}$ is maintained by channel i over every time interval of length I_i or more, we know that the average rate is also maintained in the period of I_{max}. Thus, the node will always become idle once in every interval of length I_{max}, and channel i will not have more than $\lceil I_{max} / x_{ave,i} \rceil$ cells present in the node at any time.

Notice that condition (4-22) needs to be enforced only if there is at least one channel i such that $x_{ave,i} > x_{min,i}$. In those situations where the two parameters are equal for all the channels, condition (4-14) implies condition (4-22).

It follows that buffer overflow will not occur for channel i if we allocate B_i buffers indicated by the following relation:

$$B_i = \min\left(\left\lceil \frac{d_i}{x_{min,i}} \right\rceil, \frac{I_{max}}{x_{ave,i}}\right). \qquad \textit{(4-23)}$$

When we have to provide statistical channels with a loss rate guarantee, we can distinguish the following two cases. When w < z, we can simply drop all the cells that are delayed beyond their required delay bound. Thus, these channels can be assigned buffer space as if they had a deterministic delay bound and a loss rate

of w/z. On the other hand, if $w > z$, no cells will be lost if we allocate B_i buffers, with

$$B_i = \frac{I_{max}}{x_{ave,\,i}}. \qquad (4\text{-}24)$$

In case the bound offered by equation (4-24) is too large, we can reduce the allocation for statistical channels in the following manner. The probability that a cell on channel i that has been placed in the scheduling queue at level I will miss its deadline is at most ζ_I, so an allocation of buffers as given by equation (4-23) is not going to cause a loss rate of more than $1-\zeta_I$ (which is less than the promised loss rate of z_i to the channel, and therefore safe). Thus, the expected length of an overload (i.e. a period when cells will miss their deadlines) should not exceed $I(1-\zeta_i)$. Thus, a more reasonable estimate of the buffer space required to avoid cell losses is

$$B_i = \left\lceil \frac{I_{max}(1-\zeta_I)}{x_{min,\,i}} \right\rceil. \qquad (4\text{-}25)$$

In order to reduce the buffer space allocation for channels that can tolerate losses ($w < 1$), we can reduce the allocation to b_i buffers, with

$$b_i = B_i w_i. \qquad (4\text{-}26)$$

As a summary of this sub-section, we assert:

Theorem 4-4 Given a set of channels, with the i^{th} channel having traffic parameters $x_{min,i}$, x_{ave},I_i and performance parameters (d_i,z_i) and w_i, the performance of each channel in the set can be guaranteed with the scheduling and reconstruction policies described in Section 4.4.1 and 4.3 if the inequalities (4-14), (4-19), (4-21) and (4-22) are satisfied, and buffer space satisfying equation (4-

26) is available.

Proof: follows from the discussion above. ∎

4.4.4 Performance

In this section, we evaluate the admission control policy presented in Section 4.3? We are interested in the following questions:

(1) Given a set of standard channels, how many can be accepted by the admission control strategy described in Section 4.3?

(2) What are the characteristics that determine the number of channels accepted by this admission control policy? For what spectrum of traffic characteristics is this strategy good and for what spectrum is it bad?

(3) What are the typical real-time traffic utilizations of a node permitted by this admission control strategy?

(4) How close are the actual delay bounds to the predicted delay bounds? Are we being too pessimistic in adopting this admission control policy?

(5) What are the buffer utilizations produced by this admission control policy? How do they compare to the allocated amounts of buffer space?

We obtained answers to these questions by means of a simulator written in CSIM [Schwetman 90]. Recalling that the performance distribution mechanism allows us to concentrate on the per-node admission control policy, we can limit ourselves to simulating the behavior of a node under loads of different kinds. In order

to answer questions (1) and (2), we used homogeneous loads, i.e., channels where all the parameters have identical traffic and performance parameters. Later on in this section, we shall examine the performance of the scheme under the case of heterogeneous load, where different channels have different traffic characteristics and performance requirements.

In the homogeneous load case, suppose we can accept *N* real-time channels. Let the node have *B* buffers available for real-time channels, and let these channels be all deterministic. We know that we are constrained by the tests in equation(4-14), (4-19) and (4-23). By simplifying these equations, we obtain:

$$N_{det} = \min\left(\left\lfloor \frac{d}{t} \right\rfloor, \left\lfloor \frac{x_{min}}{t} \right\rfloor, g_{det}(B, d, w, l, x_{min}, x_{ave})\right), \qquad (4\text{-}27)$$

where g_{det} is the function that tells how many channels can be accepted when buffers are the bottleneck resource. This function can be expressed by means of the following equation:

$$g_{det}(B, d, w, l, x_{min}, x_{ave}) = \left\lfloor B \Big/ \left(\min\left(\left\lceil \frac{d}{x_{min}} \right\rceil, \left\lceil \frac{l}{x_{ave}} \right\rceil \right) w \right) \right\rfloor \qquad (4\text{-}28)$$

Equation (4-27) identifies the three possible cases for the rejection of a channel, namely the lack of buffer resources, the lack of bandwidth resources, and the inability to provide the required delay bound. While buffers and bandwidth are usually looked upon as network resources, equation (4-27) shows that delay constraints should also be viewed as another network resource. The three resources impose different kinds of constraints on the number of channels accepted, which may lie in the shaded region in Figure 4-3.

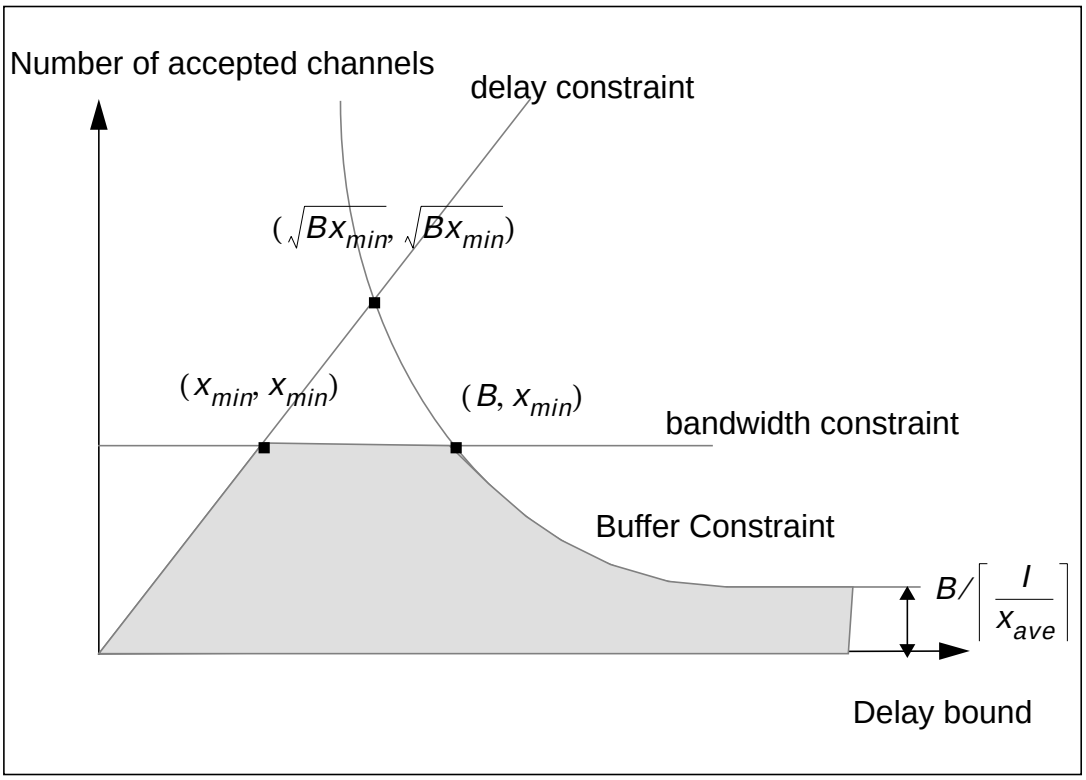

Figure 4-3 The effect of different resources on the number of deterministic channels that can be accepted at a node under homogeneous load. Buffer space, bandwidth and delay requirements impose three constraints on the number of channels that can be accepted. Any number of channels which satisfies all the three constraints can be accepted. In this figure, the safe region is shown by means of shading. In real networks, the constraints imposed by buffers are usually less restrictive.

In order to understand the behavior of the admission control scheme, let us examine a simple switch with buffer space for B cells (B = 1000). On the basis of the results of Chapter 2, we can conclude that I/x_{ave} can be approximately set at 20. Let us further assume that all channels have $w = 1$. For the sake of simplicity, let us also assume that there is no burstiness, that is, $x_{min} = x_{ave}$. Let us see how many channels can be accepted by our scheme in this situation when x_{min} is 100 time units.

Figure 4-3 tells us that, for a delay bound d less than 100 time units, we can only accept d channels. For a delay bound between 100 and 1000 time units, we

can accept 100 channels, while for delay bounds larger than a 1000 time units the number of accepted channels gradually decreases, and asymptotically we can accept 50 channels. Thus, the best performance of the admission control scheme is in the range where delay constraints are between 100 and 1000 time units.

The number of successfully established channels will be higher in the case of statistical channels. The additional parameter introduced here is the statistical delay parameter z, which can affect the test in equation (4-21). The buffer allocation for the statistical channels is also different from that of the deterministic channels. (see equations (4-23) and (4-25).

The number of statistical channels accepted in the homogeneous statistical case is given by equation (4-29):

$$N_{stat} = min\left(\left\lfloor \frac{d}{t} \right\rfloor, f_{stat}\left(\left\lfloor \frac{x_{min}}{t} \right\rfloor, z, \frac{x_{min}}{x_{ave}} \right), g_{stat}(B, z, d, w, I, x_{min}, x_{ave}) \right) \quad (4\text{-}29)$$

The function g_{stat} is defined by the relationship:

$$g_{stat}(B, z, d, w, I, x_{min}, x_{ave}) = \left\lfloor \frac{B}{\left(min\left(\left\lceil \frac{d + I(1-z)}{x_{min}} \right\rceil, \left\lceil \frac{I}{x_{ave}} \right\rceil \right) \right) w} \right\rfloor . \quad (4\text{-}30)$$

The value of f_{stat} is the cardinality of the largest set of channels that satisfies equation (4-21). In the case of homogeneous channels, the values of f_{stat} is the largest integer N that satisfies equation (4-31), where $k_d = \left\lfloor \frac{x_{min}}{t} \right\rfloor$ and $p = \frac{x_{min}}{x_{ave}}$:

$$\sum_{k=k_d}^{N} \binom{N}{k} p^k (1-p)^{N-k} \leq z. \quad (4\text{-}31)$$

Function f_{stat} is plotted in Figure 4-4. The dependence of f_{stat} on z is influ-

enced by the ratio of the minimum to the average inter-cell spacings. It can be seen that even a small change in the value of z can cause a significant increase in the value of f_{stat} specially at lower values of the ratio x_{min}/x_{ave}. A corollary of this fact is that a statistical channel is going to be much cheaper than a deterministic channel when bandwidth is the bottleneck resource.

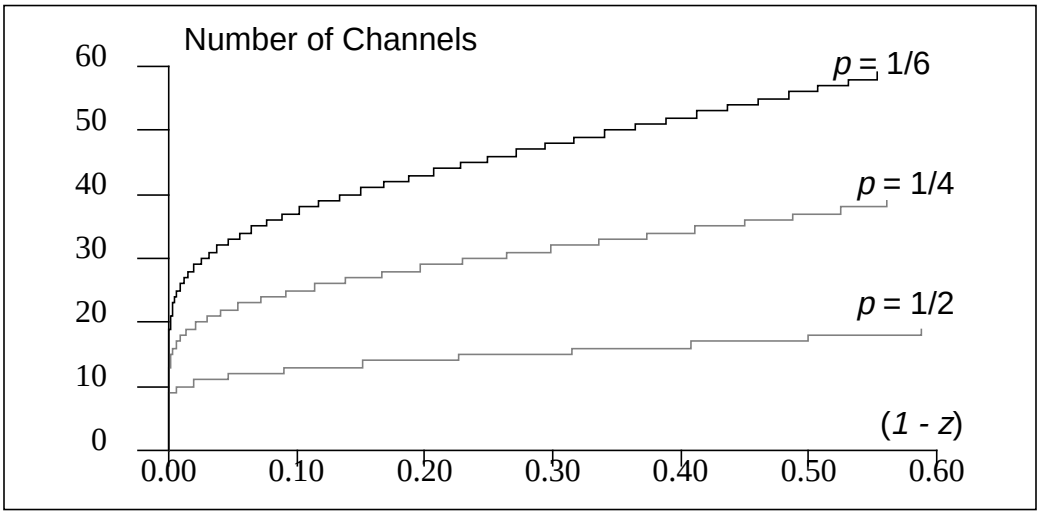

Figure 4-4 Variation of f_{stat} according to equation (4-31). The two major factors influencing the shape of this curve are z and p. p is the ratio of the minimum to the average inter-cell intervals. ($1-z$) is the probability of missing a delay bound.

Our analysis until now has provided answers to the first two questions in the beginning of this section. Let us now look at the third question, and examine how the utilization of a node depends on the different traffic parameters considered so far.

In order to understand how the utilization of a node depends on the different parameters mentioned so far, we first look at the case of a deterministic channel only. We are still looking at the homogeneous load case, where all the channels have identical properties. In order to further simplify our presentation, let us assume that the parameters w and I are identical for all channels. For a determin-

istic channel, bandwidth is reserved on basis of the parameter x_{min} and not on the basis of x_{ave}. Let us assume that the two parameters have equal values.

We can isolate two cases here. When there are sufficient buffers available, the utilization of the node depends mainly on the delay requirement of the channels. If d, the local delay bound of a channel, is less than x_{min}, its minimum inter-cell spacing, the utilization is constrained to be d / x_{min}, but, if the delay requirement is larger, a 100% node utilization can be achieved.

At least $B_{req} = (\min(d, x_{min})) / (tw)$ buffers are required per channel in order to achieve 100% utilization. If those many buffers are not available, the utilization is constrained by the buffer space available at the node. These dependencies of node utilization can be seen in Figure 4-5.

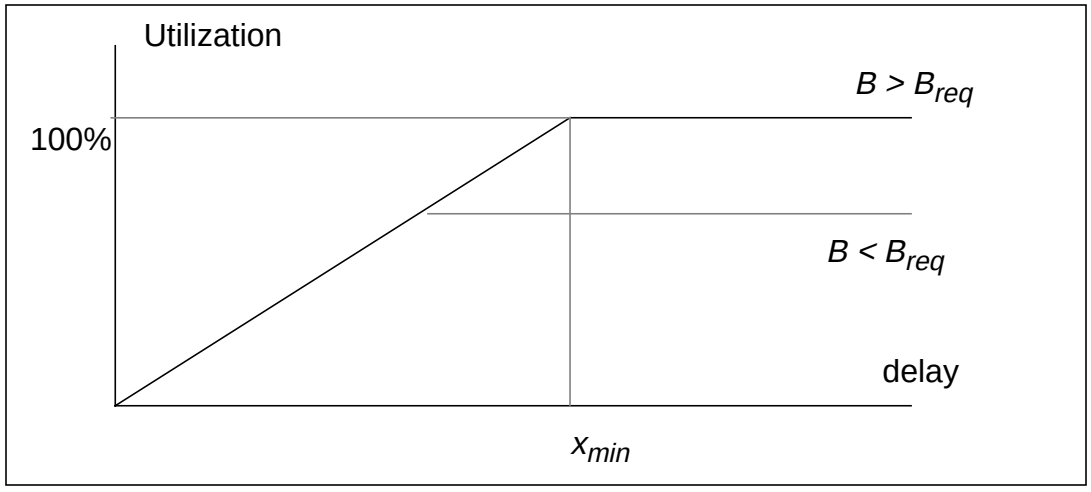

Figure 4-5 Variation of deterministic node utilization with channel parameters, assuming x_{min} and x_{ave} are the same. B_{req} is the minimum number of buffers required to obtain a 100% node utilization.

The utilizations shown in Figure 4-5 assume that the average and minimum inter-cell spacings are the same. The maximum possible utilization (100%) can be

reached if the delay bound is large and enough buffers are available. In those cases where they are different (which is likely in real traffic patterns), the utilization will be further reduced by the ratio between x_{min} and x_{ave}.

The study of utilization for statistical channels can also be broken into two cases. In the first case, where delay bounds are tighter than the minimum inter-cell interval ($d < x_{min}$), the utilizations for deterministic and statistical channels are identical. In the second case, where the delay bounds are larger than the minimum inter-cell interval, the utilization depends on the function f_{stat}.

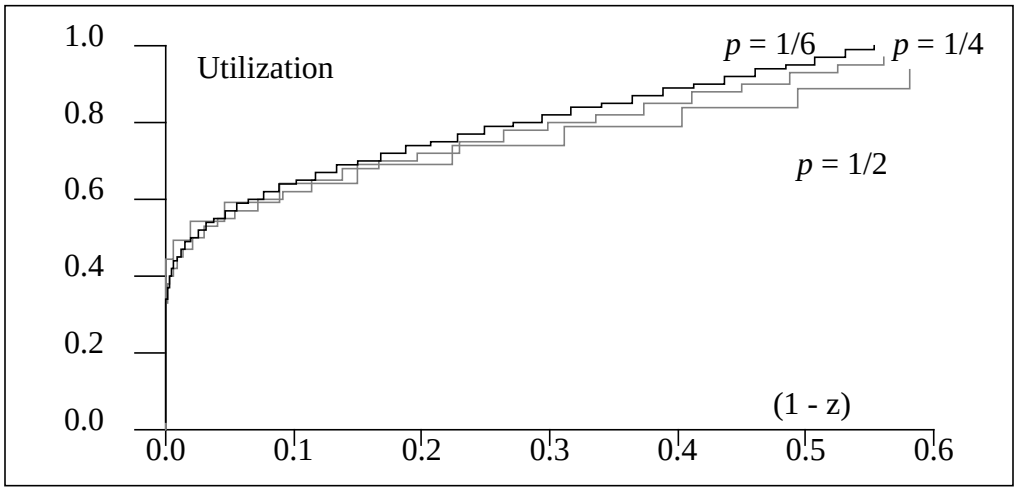

Figure 4-6Variationofutilizationforstatisticalchannels.*p*istheratiooftheminimumtotheaverage inter-cell intervals. (*1-z)* is the probability of missing a delay bound.

Figure 4-6 shows the node utilization for the same type of channels as in Figure 4-4. While the number of channels accepted depends on the ratio of the average to the minimum inter-cell spacing, the effective node utilization is more or less independent of this burstiness ratio. The figure shows that statistical channels can achieve an appreciable gain in maximum possible node utilization. Let p be the ratio between x_{min} and x_{ave}. For deterministic channels, the maximum utilization of

the node is bound to be less than *p*. However, for statistical channels, utilization can be much larger. Notice that even a small change in the value of *z* can result in a fairly substantial increase in the value of node utilization.

Up till now, we have only considered the homogeneous case, where all the channels are identical. Let us consider a situation where channels may belong to any type and can have any values for the performance requirements. In this case, the utilization of a node depends on the characteristics of the set of channels that have been established through the node.

In order to study heterogeneity, we considered channels with different delay bounds and different peak and average bandwidths. A mixture of channel types, with per-node delay bounds uniformly distributed between 10 and 100 time units, was chosen. A similar bound for x_{min} was also chosen. The burstiness of each channel was uniformly chosen between 2 and 10 (in other words, x_{ave} was 2 to 10 times x_{min}). Under these assumptions, we measured the expected average utilization of a typical node. All the channels had identical *z* parameters, and the scheduler was assumed to have only one level of scheduling *(M=1)*. The maximum possible utilizations of the node produced by these simulations are plotted in Figure 4-7. The figure also shows the expected utilization when the traffic is more bursty, with a burstiness ratio uniformly chosen between 2 and 20, and between 2 and 50. For very stringent values of *z* (greater than $1\text{-}10^{-7}$), the utilization is low (roughly 30%). However, for less stringent values of *z* (e.g., $1\text{-}10^{-4}$), the utilization improves (to about 48%). Thus, the algorithm seems to perform satisfactorily when the burstiness is between 2 and 10. At higher burstiness of traffic, the utilization is somewhat lower. The 99% confidence intervals for the expected utilization in this

figure are about 0.02 for each of the curves.

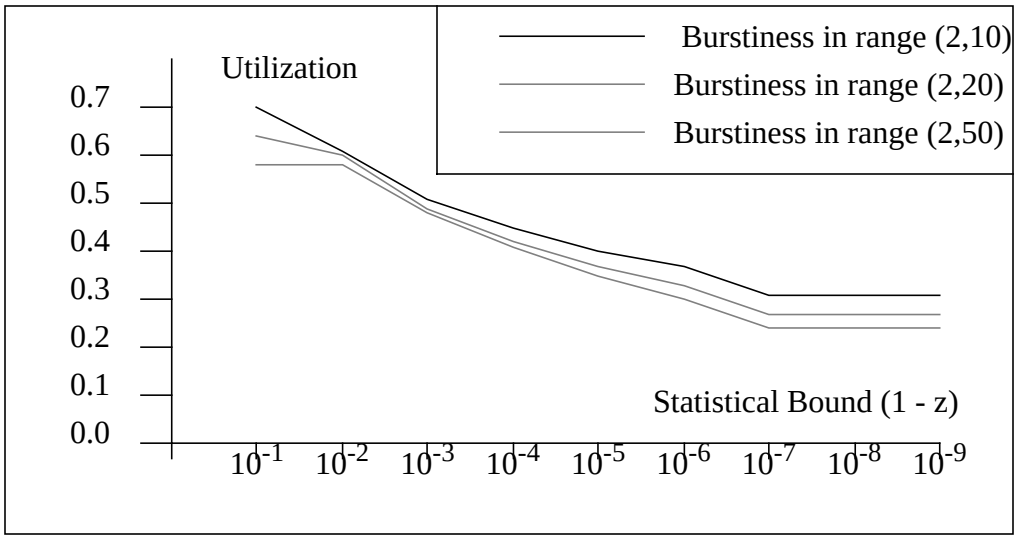

Figure 4-7 The average utilization of a node subject to admission control for QOS menu 1 under a mixed traffic load. The 99% confidence limits for all the three curves are approximately 0.02.

In order to determine how realistic our bounds are for the actual delays and loss rates (questions (4) and (5) at the beginning of this section), we measured the buffer and delay distributions for a channel in our scheme. Our bounds are pessimistic, so we expected a large difference between the average delays and our bounds. The distributions of buffer spaces occupied and delays are shown in Figure 4-8 for a channel selected from the simulations just referred to. The channel had a delay bound of 62 time units, a peak arrival rate of one cell every 12 time units and a burstiness ratio of about 4. The arrival pattern of cells on all channels sharing the link was a bunch process (as described in Section 1.3.1), and the node utilization was about 60%. The probability of missing the channel's deadline in our scheme was 0.05. However, out of about 10^5 cells transmitted over the channel in our simulation experiment, no cell missed its deadline. The bounds that are computed for admission control are valid for any kind of traffic pattern, and thus tend to

be much larger than the maximum delays we actually observed in simulations. If we were able to make further assumptions about the traffic arrival patterns, a better bound on cell delay and buffer occupancy could be obtained.

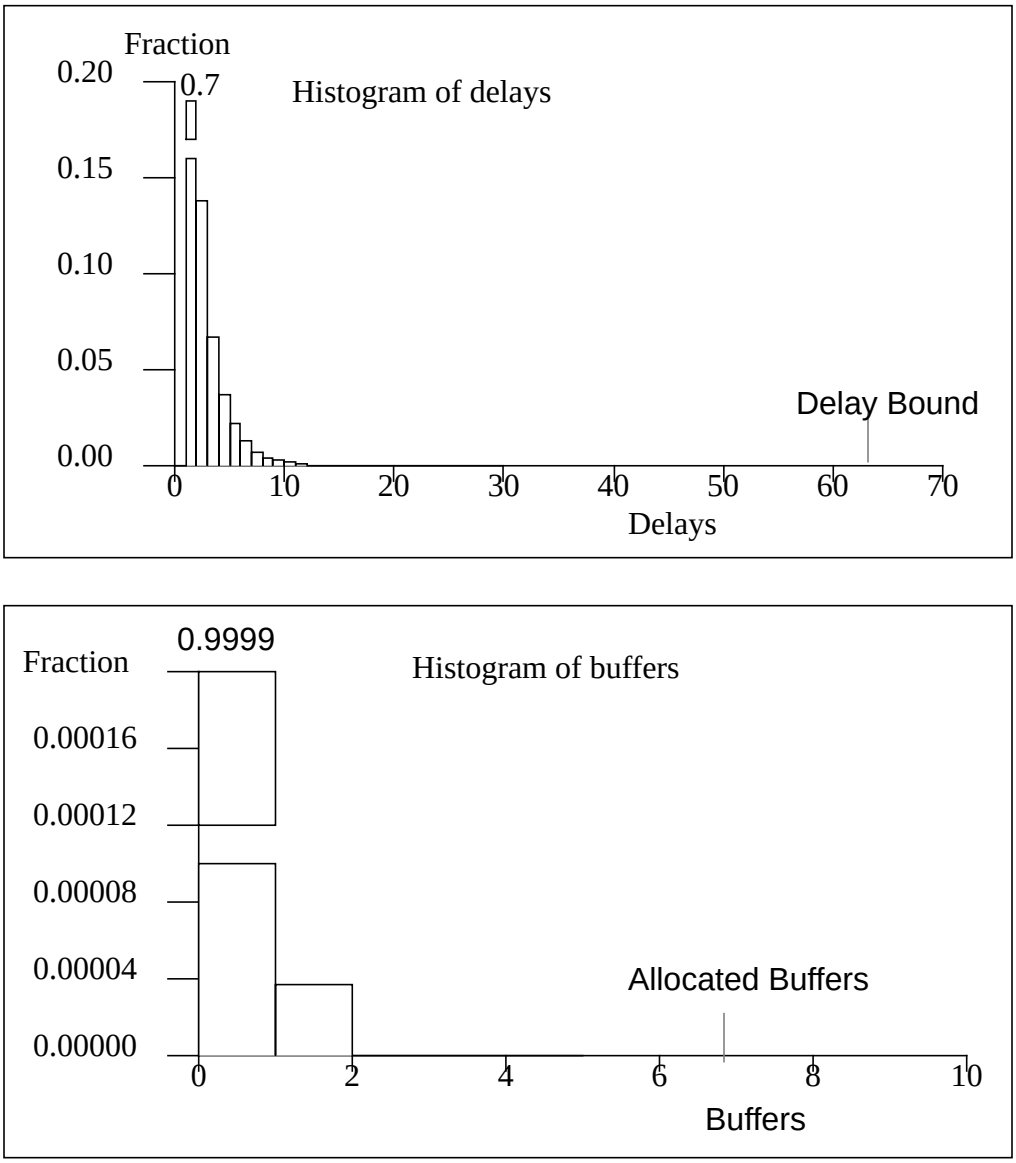

Figure 4-8 The distribution of delays and buffers required on a channel in our admission control scheme. The allocated delay bound and buffer space are marked on the curves. The curves show a break in the first bin, to accommodate large values of the fraction in the graph. Thus 70% of the cells had a delay of one time unit while about 99.99% of the cells found that there was no her cell from the same channel present when they arrived at the node.

4.5 QOS Menu 2: Specification and Mechanism

Let us now shift our focus to a network which is extremely fast and where buffers are scarcer than bandwidth. In these cases, a client specifies its QOS requirements by declaring a bound on the maximum acceptable loss rate *W*. As in the discussion of the previous menu, we assume that the end-to-end value of the bound has been broken down into per-node bounds. Let the value of this bound for channel *i* at a node be w_i. Thus, the problem is that of determining when a request for a new channel can be accepted by the node.

The specification is well-formed if the following condition is satisfied:

$$0 \leq w \leq 1, \qquad (4\text{-}32)$$

which simply says that *w* is a probability and so cannot lie outside the (*0...1*) range.

4.5.1 Scheduling and reconstruction policy

Although it is possible to design a scheduling scheme which discriminates in terms of priority between channels with even slightly different values of the loss bound, such a scheme would typically not be useful. In a practical network, we can break the QOS menu down into a small number of classes. The number of classes can be anywhere from, say 2 to 16, depending on the network characteristics and traffic requirements. Let us assume that there are *M* such levels.

Since buffer space is much more important than the actual scheduling of cells in this case, the buffer space management policy deserves our attention. The scheduling scheme can consist of a number of levels, with a simple priority rule governing them. Low loss channels will be given preference over channels with a

higher loss rate bound. The idea behind this scheduling scheme is that cells from a channel with higher priority will tend to occupy less buffer space since they spend less time in the node. Thus, the same amount of buffers would result in a lower loss-rate for higher priority traffic than for lower priority traffic. Within each priority class, cells are serviced on a first-come-first-served basis. This scheduling policy is simple enough to be implemented in high-speed networks.[10]

As far as buffer space is concerned, the following policies are possible:

(1) All the channels sharing an output link share the same buffer space. Thus, the buffer space is shared dynamically between low priority and high priority channels. The advantage of the sharing is that buffers are used more efficiently. However, a scheme that requires channels with higher priorities to have preferential access to the buffers is desired.

(2) On the other hand, we could try a policy in which buffers are statically allocated to individual channels. This architecture protects channels from one another, so that a misbehaving client cannot invade the buffer space allocated to other clients. However, the efficiency of buffer usage is much lower.

(3) Somewhere in the middle, we have a policy where the buffer space is divided statically among different levels and dynamically among channels at the same level.

Recalling the fact that the regulator in our scheme is able to protect chan-

10. The simple first-come-first-served disciplines dangerous because a misbehaving channel can ruin the performance for all the clients. However, as explained later in this section, our rate-control module prevents that from happening.

nels from one another, static buffer allocation loses its main appeal. We could, therefore, opt for a dynamic sharing of buffers, but, since sharing of buffers by channels with different loss rate requirements is complex, we propose that only channels with the same loss rates share buffers. Thus, the third architecture appears to be the best of the above options.

We shall now describe the buffer architecture for the regulator. The regulator will have buffers shared dynamically by all channels. Thus, it needs a scheme to protect channels from one another. One such scheme would be to admit cells into the node on the basis of their expected arrival times. If buffer space were at a premium, a cell which was expected further in future should be dropped in preference to a cell which was expected in the near future.

The regulator mechanism (implemented using calendar queues) described in Section 3.5 provides us with an efficient way to implement this scheme. A calendar queue is maintained for each clock tick, and contains a list of cells that are expected at that time.

Suppose N_0 is the total number of calendar queues implemented in the regulator and t_0 is the value of one clock tick, then cells that have a difference larger than $N_0 t_0$ between the actual arrival time and the expected arrival time should be dropped. The role of the regulator is very important since, in the absence of its rate control functionality, the simple priority scheduler we have proposed would not be able to provide different grades of service.

The regulator can implement either a full or a partial reconstruction policy. We recommend the use of a partial reconstruction policy since bounds on delay jit-

ter are not important in the context of this QOS menu. A description of the partial reconstruction policy can be found in Section 3.5.

4.5.2 Admission Control

Suppose there are *M* classes of service, with level *i* offering a minimum quality of service marked by a buffer overflow probability not greater than $1-\eta_i$. In this case, all channels with a value of *w* between η_i and η_{i-1} are put into level *i*. We assume that levels are counted starting from 1 and that η_0 is 0.

In order to bound the loss rate at a level of the scheduler, we use the same approach as that which yielded equation (4-20). The assumption here is that we allow cells to be lost whenever there is an overflow. An overflow for a level is defined as a situation where channels at that level do not satisfy condition (4-14), and can be evaluated in a manner analogous to that in equation (4-20). Suppose the probability of an overflow at level *j* is Φ_j. Cells may be lost at a level if there is an overflow in any of the higher priority levels. Thus, we must enforce the condition that

$$\eta_i \geq \sum_{j \leq i} \Phi_j \qquad (4\text{-}33)$$

at all levels (i.e., for all values of *i*).

Theorem 4-5 Let *K* be the set of channels existing at scheduling level *j* and let *L* be the set of channels existing at higher levels (that is, levels *1...j*). Let there be M_j channels in set *L*, and N_j channels in set *K*. If there is no overflow at any of the levels *1...j*, then no cells will be lost if we allocate M_j+N_j buffers to level *j*.

Proof: In this case, if all these channels are simultaneously active, let the number of cells present at time T after the start of a busy cycle be N_T. By this time, some of the cells that arrived on the M_j higher channels are serviced and whenever there were no cells present from the these channels, that time was used to service cells from the channels at level *j.* Let us make a conservative estimate and say that all the cells that arrived on the M_j channels by the time T have been serviced and that all the cells at the j^{th} level remain in the queue. Hence,

$$N_T \le \sum_{k \in K} \left\lceil \frac{T}{x_{\min,k}} \right\rceil - \left(\left\lfloor \frac{T}{t} \right\rfloor - \sum_{l \in L} \left\lceil \frac{T}{x_{min,\,l}} \right\rceil \right). \qquad (4\text{-}34)$$

Since $\lceil x \rceil$ can be bounded above by $(x+1)$, and $\lfloor x \rfloor$ can be bounded above by x, we can simplify the above expression to obtain that

$$N_T \le N_j + M_j - \frac{T}{t}\left(1 - \sum_{k \in (K \cup L)} \frac{t}{x_{min,\,k}}\right). \qquad (4\text{-}35)$$

If the sets K and L have no overflow, we know that $\sum_{k \in (K \cup L)} \frac{t}{x_{min,\,k}} \le 1$, and can further obtain:

$$N_T \le N_j + M_j. \qquad (4\text{-}36)$$

This suffices to prove the theorem. ∎

Theorem 4-6 Suppose there is an overflow in one of the levels *1...j*, that is equation (4-33) is violated for one of the layers *1...j*. Suppose we want to select the largest set of channels from all the channels accepted at layers *1...j* without causing an overflow (that is, equation (4-33) is not violated if we only consider channels from this set). One such set can be obtained by choosing channels with the largest values of x_{min} until we just reach the condition $\sum \frac{t}{x_{min}} = 1$.

Proof: The selected set satisfies $\sum \frac{t}{x_{min}} = 1$. Since replacing any channel in the selected set would mean replacing it with a channel with a higher value of x_{min}, this condition would be violated. ∎

Theorem 4-7 Suppose we determine the size of the largest set which satisfies $\sum \frac{t}{x_{min}} = 1$ according to the scheme described in Theorem 4-6, and allocate those many buffers to be dynamically shared between all channels at level *j*; the loss-rate of a channel at level *i* is less than η_i as given by equation (4-33).

Proof: Whenever $\sum \frac{t}{x_{min}} = 1$ at all levels 1..i,Theorem 4-5 implies that the maximum number of buffers occupied by all the channels at those levels can not exceed the one provided by Theorem 4-6, therefore losses can only occur if this condition is violated. The probability of this occuring is bounded by η_i as given by equation (4-33).

Satisfying equations (4-33) ensures that the probability of overflow (which can now be seen as the probability of cell loss) is low enough to meet the loss requirements of all channels at that level. Notice that, if an extra channel is added to the set *L* of Theorem 4-5(which means adding a channel at a higher priority level than level *j),* an extra buffer needs to be assigned to level *j*. This implies that ensuring a lower loss rate to a channel requires more buffers, one at each level below the one in which the channel is added. As expected, channels without stringent loss requirements are less expensive for the network.

The reader has certainly noticed that the number of cells given by equation (4-36) is a bound on the total number of cells that can be present in the node from all levels *1...j* and not merely on the cells at level *j*. Thus, we are examining the

worst possible situation, one where no cell from the j^{th} level is serviced until all cells from the higher priority levels are serviced.

Let us consider the case where no overflows are allowed at any level. At this time, adding a channel to a higher priority level requires allocating a buffer for all levels below that priority level. However, when a channel is added after an overflow situation is present, we only need to add a buffer if the largest set of channels in which there is no overflow changes, i.e., if the sets K and L considered in the derivation of equation (4-36) change and the value of $N_j + M_j$ changes. Thus, there may be cases when extra buffers may not be needed to be added at any level on the acceptance of a channel.

In the next section, we will study the performance of the admission control policy described here.

4.5.3 Performance

Studying the performance of menu 2 means answering the following questions:

(1) How do the actual loss rates compare to the predicted loss rates for a channel?

(2) How do the buffer requirements of channels at different priority levels compare?

We answered these questions by means of a simulator written in CSIM [Schwetman 90]. In order to determine the expected utilization of the node, we sim-

ulated an ATM output queue with 4 levels of priority. The loss requirements at these priority levels were fixed at 10^{-2}, 10^{-4}, 10^{-6} and 10^{-8} respectively. We assumed that a channel establishment request could fall into one of these ranges with equal probability, that the minimum inter-cell spacing (x_{min}) was uniformly distributed between 10 and 100 time units[11] (one time unit being the time to transmit one ATM cell), and that a channel's peak rate was between 2 and 10 times its average rate. The value of *I* was 500 time units for all the channels. A channel was equally likely to belong to any of the four QOS classes, and was accepted subject to the admission control tests outlined in the previous section.

Figure 4-9 shows the histogram of buffer occupancy for the four different types of channels in a typical simulation run with a node utilization of about 70%. The simulation was run for a duration of about 10^9 simulated time units. While this duration is sufficient to estimate the number of buffers required up to loss rates of 10^{-4} with a reasonable confidence interval,[12] it is inadequate to get a good estimate of it for a loss rate of 10^{-6} or 10^{-8}. In order to keep our simulations runs to a reasonable length, we adopted a curve-fitting method to obtain the buffer space required at these low probabilities of loss. We first obtained the number of buffers required for loss rates from 10^{-1} to 10^{-4}. Assuming that low loss rates can be approximated reasonably well by an exponential distribution curve, we obtained the least-square best fit to these points, and extrapolated it to a loss rate of less

---

11. The workload chosen for the simulation may appear to be arbitrary. However, since good validated models of traffic in future high speed networks are not available, a uniform choice of characteristics seems to capture the expected heterogeneity in traffic requirements quite well.

12. To get an estimate of a loss rate of 10^{-x}, we need to have at least 10^{-x+2} cells for just one measurement of the buffer space required for this loss rate; assuming that we need 100 measurements to get a good estimate of the buffer space required, we need to simulate at least 10^{-x+4} cells. With a utilization of 70% in our simulations, this translates to simulating for a time-period of at least 10^{-x+5} time units.

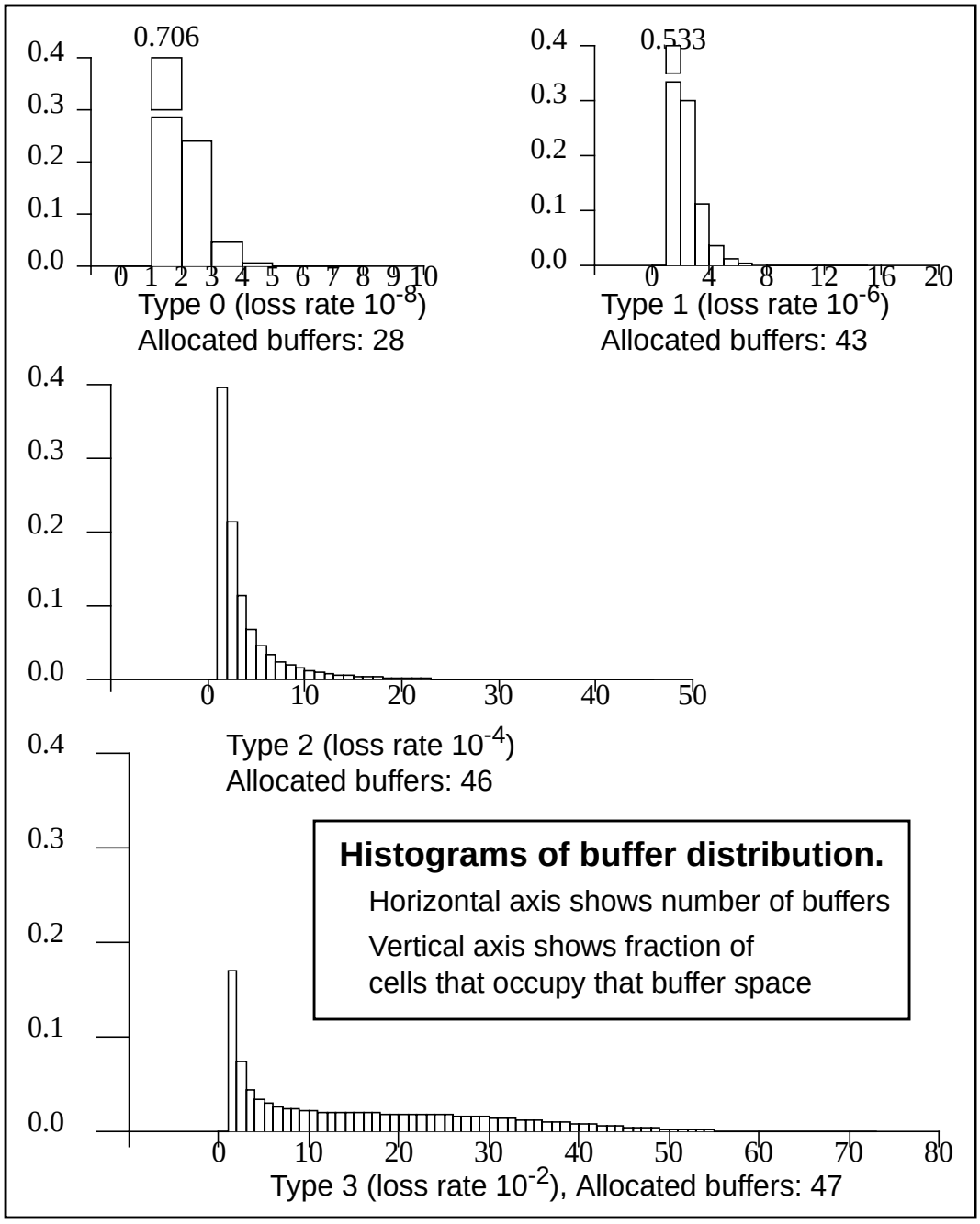

Figure 4-9 The histogram of buffer distributions for a typical simulation experiment for QOS menu 2.

than 10^{-9}. In this manner, we were able to avoid excessively long simulations.

It can be seen from the histogram of buffer occupancy that our allocation is still conservative.The histogram has been plotted for about 200 million cells for

each type of traffic. While it may appear counterintuitive that cells belonging to more lossy channels occupy more buffers, it is a result of the scheduling policy which gives higher priority to low loss channels.The total allocated buffer space for the four different types of channels were generally higher than the ones actually used. The only loss observed was for type 3 traffic where 0.001% of cells were lost.

Figure 4-10 shows the amount of buffer space required by each type of traffic at different loads. All four types of channels require almost the same amount of buffer space on the average, but the confidence intervals for higher loss traffic are higher. This is because, for that traffic, buffer allocation is more dependent on the number of accepted channels at higher levels, and thus more variable.The total amount of buffer space, even at utilizations of about 80% is less than 1000 buffers. Thus, the scheme appears to be well within the bounds of current technology.

4.6 QOS Menu 3: Specification and Mechanism

While the first menu applied to the more general case where both bandwidth and buffers were in balanced supply, the second menu applied to the case where bandwidth was plentiful and buffers were the only scarce commodity. In between these two extremes, there may be a situation where bandwidth may not be that cheap, and channels may still require some delay guarantees. However, QOS menus would not offer a continuous range of services, but would offer only a small number of delay and loss rate choices. In this section, we present one such QOS menu and show how it can be implemented by a suitable scheduling and admission control mechanism.

In QOS menu 3, clients can declare their performance requirements to

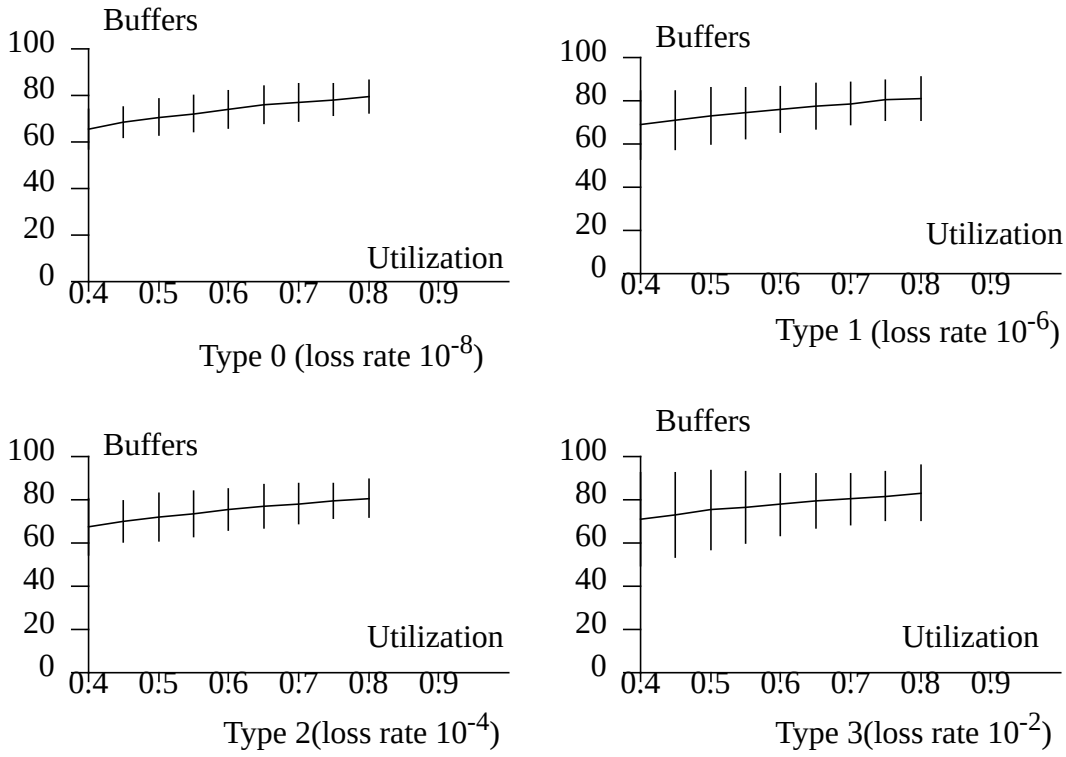

Figure 4-10 Amount of buffer space required for different types of channels at a node for QOS menu 2. Vertical lines represent 99% confidence intervals.

belong to any of the following three types at channel establishment time:[13]

(1) Class 0 Traffic: this traffic is typically smooth ($x_{min} \approx x_{ave}$) , and requires a delay smaller than d_0 per switching node and no cell losses due to buffer overflows.

(2) Class 1 Traffic: this traffic is bursty, and requires a delay smaller than d_1 per switching node. Cell loss probabilities due to a missed delay bound or due to buffer overflows are less than a certain probability bound w_1.

(3) Class 2 Traffic: this traffic is bursty, requires no delay bounds, but must

13. These traffic types are based somewhat loosely on one of the CCITT recommendations [CCITT G.142]

have no cell losses due to buffer overflows.

Like QOS menu 2, this menu is a special case of QOS menu 1. If the performance requirements of a channel are specified by means of three parameters, (its delay bound *d*, the probability *z* of not missing a deadline, and the probability *w* of not losing a cell), then class 0 traffic has ($d=d_0, z = 1, w =1$); class 1 traffic can be treated as ($d = d_1, z = w_1, w = w_1$), and class 2 traffic has a very large value of the delay bound, but $w = 1$. In this special case, we can use a simpler scheduling policy than the deadline-based scheduling scheme described in Section 4.3.

4.6.1 Scheduling and reconstruction policy

Keeping in mind that the delay requirements of class 0 channels are more stringent than the delay requirements of class 1 channels, the first choice of a scheduling scheme appears to be one that gives priority to class 0 traffic over class 1. Such a scheme is studied in [Murase 89], where the authors show that it minimizes the influence of class 1 traffic on class 0 traffic.

Class 1 traffic is delay sensitive, and so must be given priority over class 2 traffic. In each ATM switch, three queues are maintained at each output port: one for cells belonging to each class. Each queue is managed on a first come first served basis. When the node must choose the next cell to be transmitted, the scheduler picks a class 0 cell, if one is present. Otherwise any existing class 1 cell is transmitted. Class 2 cells are transmitted only when no cells from any of the upper classes are present.This scheduling mechanism results in good delay characteristics for class 1 traffic [Todorova 90a].

This is a simple priority scheduling mechanism, which can be implemented at the high speeds required for B-ISDN networks. Certainly, this is not the optimum scheduling scheme for an ATM network; complex algorithms will in general perform much better [Fitzpatrick 89][Chen 89]. However, these algorithms may not be feasible at high speeds.

Like QOS menu 2, QOS menu 3 can also operate only in conjunction with the partial reconstruction (and rate control policy) described in Chapter 3. The reconstruction policy is needed to protect channels from one another.

4.6.2 Admission Control

When a switch receives an establishment request message, it performs some or all of the following tests:

(1) the class 0 bandwidth test, required when the channel to be established belongs to class 0, and involving the existing class 0 channels sharing the same output link;

(2) the class 1 bandwidth test, to be performed for the establishment of both class 0 and class 1 channels if the new channel belongs to class 1, or if at least one class 1 channel shares the same output link; this test involves all class 0 and class 1 channels on the output link;

(3) the class 2 bandwidth test, to be performed in all cases;

(4) the delay bound tests for classes 0 and 1 to see whether the delay constraints of class 0 and class 1 traffic are being met;

(5) the buffer space test, which is needed for all types of channels in order to determine whether sufficient buffer space is present in the switch to accommodate the new channel.

If the request passes the tests, then the switch manager sends the establishment message on to the next switch. Otherwise the request is rejected.

The class 0 bandwidth test consists of verifying that the outgoing link is fast enough to accommodate the additional class 0 channel without impairing the guarantees given to the others. Keeping in mind that class 0 channels have a constant bandwidth requirement, we simply need to check that

$$\sum_i \frac{t}{x_{min,\,i}} \le 1, \qquad \textit{(4-37)}$$

where the sum is taken over all the class 0 channels sharing the output link with the incoming request.[14]

The class 1 bandwidth test verifies whether it is safe to statistically multiplex a number of channels to achieve better switch utilization. It is responsible for determining whether the probabilistic delay guarantees for class 1 channels already accepted by the switch are safe (within the z_1 bound) as required by the QOS guarantees. The test consists of evaluating the probability of overflow, as defined by equation (4-20), and verifying that it is less than $1\text{-}z_1$. Both class 0 and class 1 channels are included in this test.

The delay bound tests verify that the delay guarantees of class 0 and class

14. We would like to remind the reader that t is the time required to transmit a cell on the output link.

1 channels can be met whenever the switch is not overloaded. A bound on the maximum delays experienced by these (delay sensitive) cells is derived below.

If there are N_0 class 0 channels, equation (4-37)implies that the maximum delay of a class 0 cell can not exceed $(N_0 + 1)\,t$ where t is the basic cell transmission time on the output link. The delay bound test for class 0 traffic consists of verifying that $(N_0 + 1)\,t \le d_0$

Theorem 4-8 If there are N_0 class 0 channels, equation (4-37)implies that the maximum delay of a class 0 cell can not exceed $(N_0 + 1)\,t$.

Proof: The proof is based on a result in [Cidon 88]. Let N_T be the number of class 0 cells present T time units after the start of a busy period. Since class 0 cells are always serviced with the highest priority, we must have

$$N_T \le \sum_{i \in K} \left\lceil \frac{T}{x_{min,\,i}} \right\rceil - T, \qquad \textit{(4-38)}$$

where K is the set of class 0 connections at the node. On replacing $\lceil x \rceil$ with x, we find that $N_T \le T\left(\sum_{i \in K} \frac{t}{x_{min,\,i}} - 1\right) + |K|$. Equation (4-37) implies that the first term is negative, and hence the number of class 0 cells can not be more than $|K|$, which is bounded by N_0. ∎

Theorem 4-9 Let us consider the largest combination of class 0 and class 1 channels that satisfies condition (4-37). This set will consist of all the N_0 class 0 channels on the output link (let this set be called K) and N_1 class 1 channels, selected in the order of decreasing values of x_{min} (let this set be called L). The delay of a class 1 cell can not exceed $((N_0 + N_1)\,t) \,/\, \left(1 - \sum_{i \in K} \frac{t}{x_{min,\,i}}\right) + t$.

Proof: Consider a cell on a class 1 channel that arrives at time instant T after the start of a busy period and is serviced at time $T+d$. Between the start of the busy period and the time when this cell is serviced, the scheduling policy ensures that all cells that arrived on a class 0 channel are serviced, and that all cells which arrived on a class 1 channel before time T are serviced as well. We must have:

$$d \leq \left(\sum_{i \in K} \left\lceil \frac{T+d}{x_{min,\,i}} \right\rceil \right) t + \left(\sum_{i \in L} \left\lceil \frac{T}{x_{min,\,i}} \right\rceil \right) t - T. \qquad (4\text{-}39)$$

Since $\lceil x \rceil \leq x + 1$, we obtain

$$d \leq \sum_{i \in K} \left(\frac{T+d}{x_{min,\,i}} + 1 \right) t + \sum_{i \in L} \left(\frac{T}{x_{min,\,i}} + 1 \right) t - T, \qquad (4\text{-}40)$$

or equivalently,

$$d \leq (N_0 + N_1)\, t + T \left(\sum_{i \in K \cup L} \frac{t}{x_{min,\,i}} - 1 \right) + \left(\sum_{i \in K} \frac{d}{x_{min,\,i}} \right) t, \qquad (4\text{-}41)$$

which can be further simplified on the basis of equation (4-37), this implies that the coefficient of T in equation (4-41) is negative and can be replaced by 0. Accounting for an extra cell service time due to a lower level cell, we obtain:

$$d \leq \frac{(N_0 + N_1)\, t}{\left(1 - \sum_{i \in K} \frac{t}{x_{min,\,i}}\right)} + t. \qquad (4\text{-}42)$$

This completes our proof. ∎

In order to ensure that no class 1 cell gets delayed beyond its required delay bound d_1, it is sufficient to ensure that

$$\frac{(N_0 + N_1)\, t}{(1 - \sum_{i \in K} \frac{t}{x_{min,\, i}})} + t \leq d_1 . \qquad \textit{(4-43)}$$

The class 1 delay bound test consists of verifying that equation (4-43) holds at the node.

Buffer allocation is required to support channels with a bounded loss rate. A shared buffer architecture results in a much better buffer utilization than an architecture in which each channel is statically allocated a fixed number of buffers. Notice that the rate control scheme prevents greedy clients from usurping the whole of the shared buffer space. Thus, it is safe to have channels share a common buffer pool.

Allocation of buffer space for class 0 and class 1 channels is easy, and can be done in the manner of QOS menu 2. Given the sets *K* and *L* in the class 1 delay bound test, we know that the total buffer space required by all class 0 channels is at most N_0, and that the total buffer space required by all the class 1 channels (to ensure that the loss rate is less than z_1) is at most N_0+N_1.[15] However, class 2 channels are not supposed to lose any cells, even though they do not have an explicit delay bound. Thus, we need a separate method to determine the buffer space to be allocated to class 2 channels.

What is the total buffer requirement of all class 2 channels sharing the same communication link? Let us examine the state of the scheduler at time *T* (measured in units equal to the cell transmission time) after it makes a transition from an idle

15. The reader may have noted that we are assuming that all cells that are received at the node when condition (4-37) does not hold are lost. It is a conservative assumption, and hence safe.

to a busy state; the count of cells present in the scheduler queue can be bounded above by the amount C_T given by

$$C_T \leq \left(\sum_i \min\left(\frac{I}{x_{ave,i}}, \frac{T}{x_{min,i}}\right)\right) - T, \qquad (4\text{-}44)$$

where i ranges over all the channels sharing the output link including the channel being established. Let C_{max} be the maximum value of C_T.

Theorem 4-10 No class 2 cells will be lost if we allocate C_{max} buffers to hold class 2 cells.

Proof: Trivial. ∎

The maximum value of C_T will occur either at $T = 0$ or at $T = \lceil I / x_{ave,i} \rceil$, where i could be any of the channels sharing the output link. Thus, with N channels sharing an output link, the maximum buffer requirement (in number of cells) can be computed by obtaining the maximum value of equation (4-44), evaluated at the $N+1$ different values of T. The maximum value can be evaluated numerically.

Thus, the buffer space test consists of evaluating the maximum number of the buffers required at the values of T mentioned in the previous paragraph, doubling this value to account for the regulator, and then verifying that the buffer space in the switch exceeds this computed amount. Of course, this test assumes that the regulator and the scheduler share the same buffer space.

Keeping in mind the fact that the maximum number of cells a channel can have in the regulator is the maximum number of cells of that channel that are present in the scheduler in the previous node on the channel's route, it follows that,

in a network of homogeneous switches, the buffer requirement of the regulator is the same as the buffer requirement of the scheduler.

These admission control criteria are simple and easy to enforce at channel establishment time with limited amounts of processing power. However, the reader will have noted that both the delay and the buffer computations are based on worst-case assumptions, assuming that all channels may be sending cells at the same time. There is a cost to worst-case design in terms of poor network utilization; however, it is the only safe way if we are to ensure that the network will always provide the required quality of service. In the next section we examine how pessimistic our assumptions are.

4.6.3 Performance

In this section, we present the results of the simulation experiments we ran to study the performance of our admission control algorithms. We were interested in the following issues:

(1) How close are the predicted delay bounds and buffer requirements to the actual delays experienced and buffers utilized in our simulation experiments? The scheduling policy here is somewhat simpler, and thus, the bounds of QOS menu 1 are no longer applicable.

(2) What are the maximum possible bandwidth utilizations in the case of class 1 traffic? Are these values of utilization acceptable?

We obtained answers to these questions by means of a simulator written in CSIM [Schwetman 87].

We measured the actual delays and buffer requirements observed in the simulations in the presence of all three kinds of traffic, and compared them against the predicted upper bounds given by our admission control rules. We were not able to discover any cases in which the desired quality of service was not met. While simulation results cannot be cited as a proof of correctness of our admission control scheme, they lend confidence to the assumptions made in Section 4.6.3 to derive the admission control rule, and provide a cross-check.

In order to answer question (1), we present the results of a typical run of the simulator. Channels were created with a peak bandwidth 1/20 of the output link bandwidth, and bursty channels had a burstiness of 4. A channel was equally likely to be of any class. An on/off model was used for generating class 1 traffic, and the network wide I was assumed to be 400 time units. In order to keep the simulation runs to reasonable lengths, we observed the amount of delay we had in our simulations at different probabilities up to a probability of 10^{-3} and extrapolated it up to 10^{-9} using the extrapolation method described in Section 4.5.3.

Figure 4-11 compares the simulated delays with the predicted maximum delays, and Figure 4-12 compares the buffer allocation with the actual buffer utilization for class 2 traffic. Our predicted bounds are always met, confirming the correctness of the admission control schemes. The bounds appear to be reasonably close to the observed values, yet far enough to be safe.

In order to test the behavior of our admission control policy in the face of bursty traffic, we assumed the worst possible situation for our admission control policies. Thus, all channels were assumed to be of class 1, which would lead to the

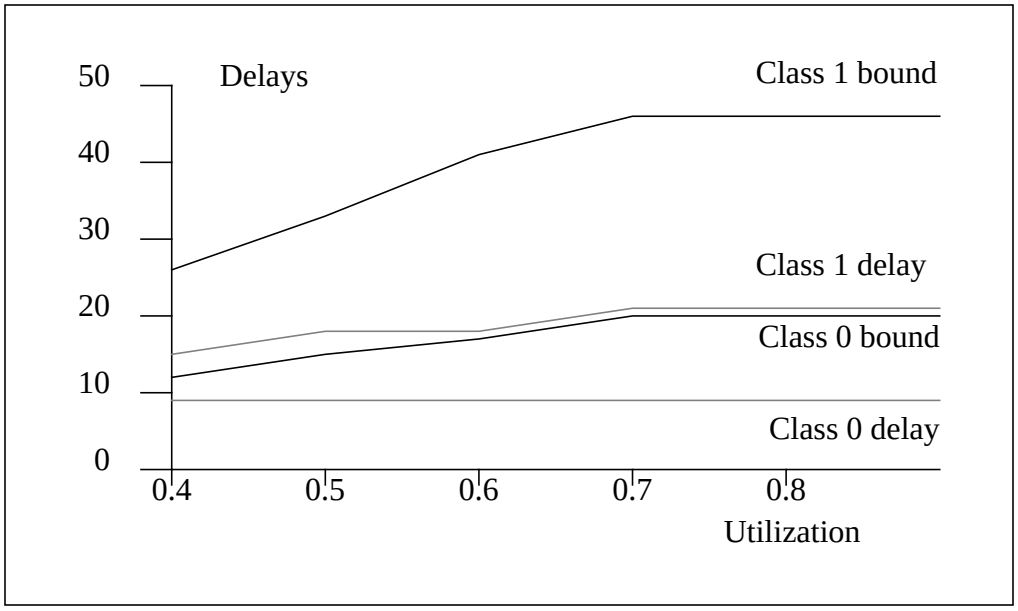

Figure 4-11 Comparison between allocated delay bounds and maximum observed delays for class 0 and class 1 traffic in QOS menu 3.

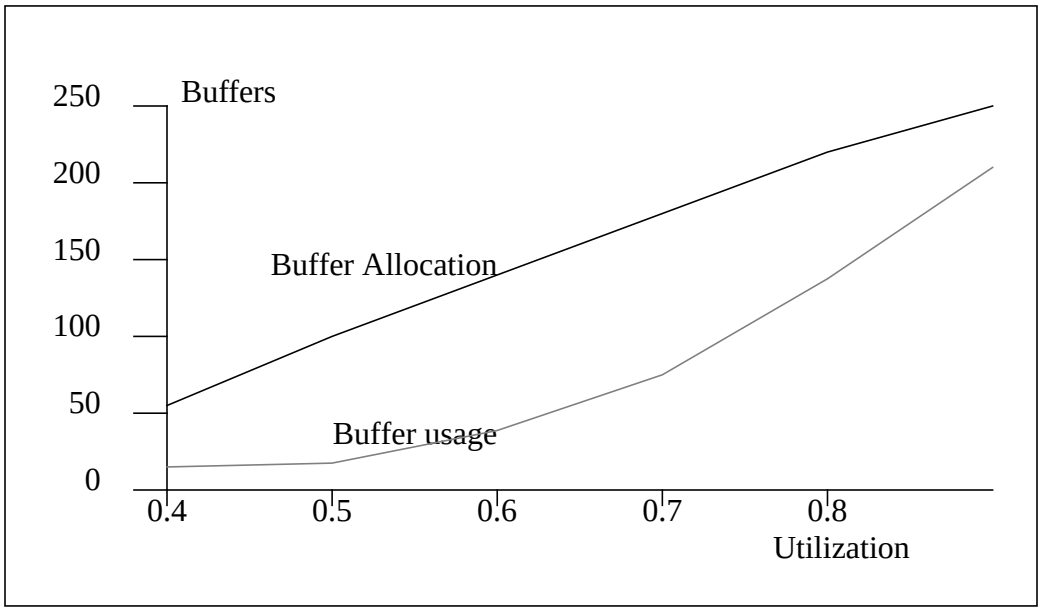

Figure 4-12 Buffers allocated and buffers actually used for loss-sensitive class 2 traffic.

lowest possible utilization of the network. The delay bound d_1 was set to be equal to 100 times the time required to transmit a cell on the output link; this roughly corresponds to a delay of 1 ms on a 45 Mbps network.

Figure 4-13 shows the maximum possible utilization of the output link in the

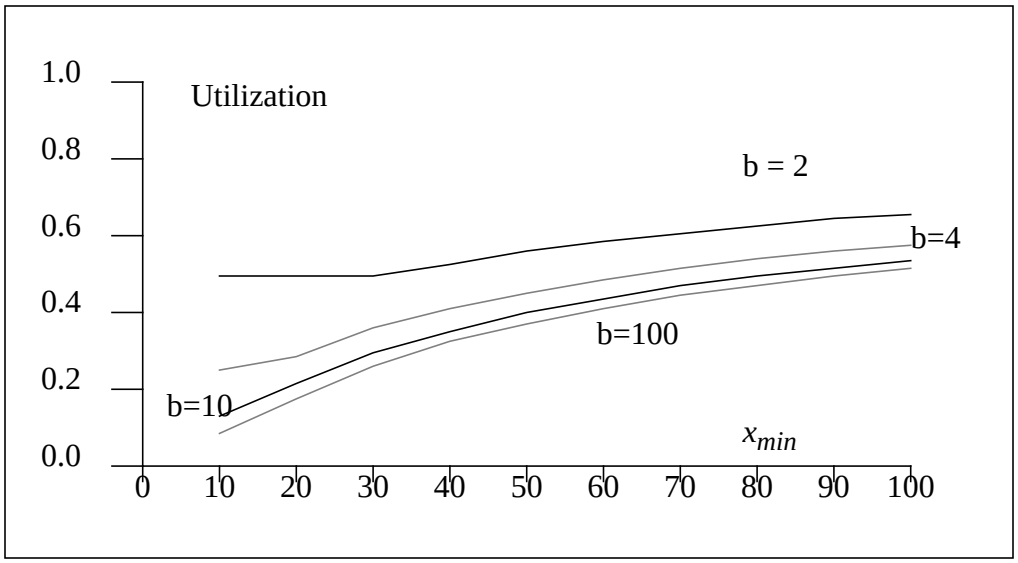

Figure 4-13 The maximum possible utilization for class 1 traffic at different values of burstiness. The utilization depends more on the peak rate of the channel than on the burstiness. fairly bursty traffic (b = 100) does not cause a collapse in maximum possible utilization.

presence of only class 1 channels, with the probability of a cell missing the delay bound of d_1 equal to 10^{-9}. On the horizontal axis, we have the minimum inter-arrival time of a single channel, that is, x_{min}. The figure shows that utilization is poor for bursty channels with high peak bandwidths, but fair for channels with lower peak bandwidths.

From Figure 4-13, we see that utilization, when the peak bandwidth of an individual channel is 1/50 times that of the output link, is about 40% with a burstiness of 5. Thus, a 45 Mbps output link can support traffic with burstiness of 5 at a utilization of 40% if the maximum peak bandwidth is about 1 Mbps. Even in this case, we can support higher peak bandwidths efficiently if we permit a higher overflow probability. It is also obvious that the maximum possible utilization depends

more on the peak bandwidth of a connection than on the burstiness of the traffic.

4.7 Verification of the Performance Contract

The performance contract consists of an agreement between the network and the client entities whereby the network promises to transport cells on the channel with some performance constraints. The verification of the performance contract by a client consists of a scheme by which the sender and the receiver can verify that data is being delivered within the contractual performance bounds.

The scheme to verify performance would depend on the QOS menu that the network has chosen to support. However, we can solve the problem for the most general case, that is, QOS menu 1, and design a verification scheme for the other two QOS menus as special cases. This can be done, because QOS menu 2 is a special case of QOS menu 1, where only the loss rate parameter is significant, and QOS menu 3 is a special case of QOS menu 1, that is, menu 1 with only three types of services. Knowing the performance bounds for each QOS type at each switch, and the number of switches along the path of a channel, we can compute the end-to-end performance bounds, and reduce the problem of performance verification for menus 2 and 3 to that of performance verification for QOS menu 1.

In QOS menu 1, we have three performance parameters to verify. *(D, Z)* specifies the delay bound constraints, *(J,U)* specifies the delay jitter bound, and *W* specifies the maximum loss rate. The probability that a cell had a delay in the range (*D-J,D*) is greater than *U*, the probability that a cell has a delay less than *D* is greater than *Z*, and the probability that a cell is not lost is greater than *W.*

There is an obvious problem for the client if it attempts to verify a probabilistic guarantee such as the value of *U, Z* or *W*. No matter how many cells are lost or delayed beyond the required limits (even if all the cells on a channel are lost or delayed), the network could always claim that the situation would have improved if the client had sent some more cells. The network could claim that the probabilistic bounds are valid only in the limit, when an infinite (or sufficiently large) number of cells are sent on a channel, and thus anything might happen to a finite number of cells on the channel. A performance verification scheme must address the issue of statistical performance guarantees.

Another issue in the design of a performance verification scheme is whether the client needs a *sufficient verification scheme* or a *necessary verification scheme.* A verification scheme is a program which is asked the question “Is the network performing according to the contract?” to which it provides a binary answer, “Yes” or “No”. A sufficient verification scheme would be a scheme which would say “Yes” only when it is certain that the network is unable to live up to its contractual obligations, thus it may allow the network to miss a performance bound sometimes. A necessary scheme would be one which would say “No” only when it is certain that the network is performing according to the contract, thus it does not allow the network to default on its obligations anytime.

Ideally, we would like a verification scheme that is both sufficient and necessary. However, in the cases where such a verification scheme is difficult to arrive at, we would opt for a sufficient verification scheme. The reason for choosing a sufficient, rather than a necessary, verification scheme is that the client would like to have sufficient proof before it can claim that the network is violating the contract.

Let us consider an easy case, namely that of verification of the delay, delay variation and loss rate bounds when all guarantees are deterministic, by which we mean that *Z=W=U=1*. In this case, no cell should experience an end-to-end delay outside the range (*D-J, D*), and no cell should be lost. This condition is both necessary and sufficient for performance verification in this case.

Detection of lost cells can be done by means of a sequence number field in each cell. By verifying that the sequence numbers at the receiver do not have any gaps, the fraction of cells that are lost in a network can be detected by the receiver. In the deterministic case that we are considering, this fraction should be zero.

We need to develop a scheme to obtain the fraction of cells that were delayed beyond *D*, or experienced a delay not in the range (*D-J,D*). If we have some way of synchronizing clocks at the sender and the receiver, or of accessing a universal time server, the absolute delay of a cell in the network can be measured. The sender can put a time-stamp in a cell indicating the time that the cell was handed over to the network, and the receiver can measure the time when the cell was received. The difference of the two provides the actual delay experienced by the cell and can be used to verify both delay and delay jitter guarantees.

Even if clocks are not synchronized, it is possible to design a sufficient verification scheme for delay jitter. If the clock rates at the sender and receiver are roughly the same (although the actual readings of the clocks may be different), the sender can put *relative time-stamps* in each cell. A relative time-stamp would be the difference in time between the sending of the current cell and that of the first cell on the connection. The difference in the delays of two cells would be the differ-

ence in the intervals at which they were received at the sender, minus the difference in their time-stamps. For any pair of cells, this difference in the delays should not exceed the maximum allowable delay jitter. Notice that the detection of a delay difference larger than *J* is sufficient proof that the network is not behaving properly. However, there may be cases where the network may not provide delay within the range (*D-J,D*) and the scheme would fail to notice it, for example, the network may be providing delays within the range (*2D-J,2D*) and this scheme would not be able to detect the difference. [16]

However, this approach is susceptible to cell losses, in the sense that it would break down if the first cell was lost in the network. In order to make it more robust to cell losses, some cells (marked especially) can be sent with a relative time-stamp equal to zero.

As an example of this approach, suppose we have an 8-bit sequence number for the cells, that is, every 256^{th} cell has sequence number 0. The relative time-stamps for a cell with sequence number 0 is always set to be 0. The relative time-stamp for a cell with a non-zero sequence number would be the difference between the time it was sent and the time that the most recent cell with sequence number 0 was sent. A cell loss only makes the relative time-stamps of a group of 256 cells difficult to use at the receiver. A dummy cell with a sequence number of 0 can be inserted x_{min} time units after a cell with sequence number 255 to ensure that the delay jitter between these two cells is also taken into account.

---

16. A similar scheme would be to send the time-difference between a cell and the time the previous cell on the connection was sent out. This scheme would be weaker in the sense that it would only detect the difference in delays between consecutive cells. It may also be used to detect the difference in delays among any two members of a group of consecutive cells, none of which is lost in the network.

At this stage, we do not have a satisfactory scheme that determines if a delay bound has been violated in the absence of synchronized clocks. This area is a subject for future study.

Let us now consider a channel for which $U < 1$, $Z < 1$ or $W < 1$. The first step in the verification scheme for such channels is to determine the fraction of cells that have been lost on the channel, and the fraction of cells that have been delayed outside of the range (D-J,D), and the fraction of cells that had a delay beyond the bound D. The fraction of lost cells can be obtained by checking sequence numbers at the receiver, while the fraction of cells outside the appropriate delay or delay jitter range, can be obtained by means of time-stamps in the presence of synchronized clocks. If a *sufficiently large* number of cells is transferred, then the fraction of cells received successfully at the receiver must be greater than W, the fraction of cells received within the required delay bound D must be greater than Z, and the fraction of cells received within the delay range (D-J,D) must be greater than U.

How large is sufficiently large? The exact number is hard to determine because cell delay violations or cell losses tend to occur in bursts. The probability that a cell is lost (or delayed beyond the required bound) is correlated with the probability that other cells arriving soon thereafter are lost (or delayed). However, if the network were able to specify a time-interval T_I such that the delay or loss probability of two cells that arrive at a spacing larger than T_I are not correlated, then it is possible for the client to determine whether the fraction of cells that were successfully received (or received within the required delay or delay jitter bound) approximates the expected fraction W (or Z or U).

Is it possible for the network to suggest a value of T_I? Because of the admission control equation (4-22), the total average rate of cell arrivals at any node in the network is always less than the possible service rate at the node. If $I_{max,n}$ is the maximum value of I among all the channels[17] at node n, the utilization of node n, when averaged over any duration of length $I_{max,n}$ or more should be less than1. If two cells arrive at node n at an interval longer than $I_{max,n}$, or depart from node n at an interval longer than $I_{max,n}$, the probability that one would be lost (or delayed beyond the required bound) is independent of that for the other. When the channel's route covers a number of nodes, T_I can be selected as the maximum value of $I_{max,n}$ along all nodes along the path. An upper bound on the value of T_I needs to be specified by the network to the client at channel establishment time. $T_{I,}$ may however, change over time, but the network needs to ensure, perhaps by refusing some connections, that it remains lower than the declared bound.

Suppose we have sent and received N cells on a channel. If I is the averaging interval for the channel, and T_{Im} is the bound on T_I declared by the network at channel establishment time, we can determine the maximum dependency lag, b_{max}, as:

$$b_{max} = \lceil T_{Im} / x_{ave} \rceil . \qquad (4\text{-}45)$$

The delays of the i^{th} cell and the $i+b_{max}{}^{th}$ cell are independent, as are the probabilities that they will be lost. Consider any of *the* b_{max} series for cell delays, the i^{th} series consisting of the delays of the i^{th}, $i+b_{max}{}^{th}$, $i+2b_{max}{}^{th}$,... cells, for i in the range $1...b_{max}$. We know that there are no dependencies among any two mem-

17. In the x_{min}-x_{ave}-I model, I is the interval over which the average inter-cell spacing x_{ave} of the channel is maintained.

bers of this series. The probability that a cell was delayed beyond the delay bound of *D* is less than *1-Z*. Thus, for a sufficiently large number of cells, the fraction of cells that are delayed beyond the bound in any of the series should be lower than (*1-Z*).

The determination of a sufficiently large number in this case is easy, because of the independence properties among each member of the series. Let us consider the cell losses for one of the series defined in the preceding paragraphs. Let us define X_i as a random variable that takes the value 0 whenever a cell in the i^{th} series is delivered outside the bound of *D* and the value1 otherwise. Then $\{X_i\}$ is a Bernoulli sequence with probability $p \geq Z$. [18] Suppose we measure the fraction of packets that were delayed beyond *D*, that is, compute the fraction of $\{X_i\}$ which is 1 by taking *n* samples. Let $\hat{x}$ be the observed value of this fraction. The law of large numbers for a Bernoulli sequence [Thomasian 69] states that:

$$\Pr\left(\left|\hat{x} - p \geq \varepsilon\right|\right) \leq \frac{p(1-p)}{n\varepsilon^2}. \qquad (4\text{-}46)$$

It follows that

$$\Pr\left(\hat{x} < Z - \varepsilon\right) \leq \Pr\left(\hat{x} < p - \varepsilon\right) \leq \Pr\left|\hat{x} - p \geq \varepsilon\right| \leq \frac{p(1-p)}{n\varepsilon^2} \leq \frac{(1-Z)}{n\varepsilon^2}, \qquad (4\text{-}47)$$

The client can choose its own value of ε, according to how strictly it wishes to verify that its delay or loss rate guarantees are being met. Suppose ε was chosen to be *(1-Z),* a number which ensures that sufficient accurate results would be obtained even for *Z* higher than $1\text{-}10^{-3}$. The number of samples that we would need

18. A Bernoulli sequence with probability *p* is obtained as a result of a number of independent tosses of a coin which turns up a head with probability *p*.

to collect before asserting that the network is not meeting the contract with a certain probability (say P) can be determined to be

$$n \geq \frac{1}{(1 - Z)\,P}. \qquad \textit{(4-48)}$$

Thus, If we want to make sure that the probability of falsely accusing the network is less than 10^{-2} for a Z of 10^{-3}, we need to examine 10^5 elements of the X series. With a b_{max} of about 20 (as a rough estimate from Chapter 3), this translates to a need for observing 2 million cells on the connection.

If we were considering cells delayed outside the range (*D-J,D),* then the random variable X would be 1 whenever a cell was delivered within this range, otherwise it would be 0, and we would use $p \geq U$. If we were considering cell losses, $p \geq W$ and the random variable X would be 1 whenever a cell is delivered correctly to the receiver, and 0 when it is lost. The number of cells to be examined can be derived as in the case of verifying the Z constraint.

4.8 Conclusions

In this chapter, we have examined mechanisms that can be used by a network to provide the required quality of service. We have proposed three different menus of QOS guarantees that may be adopted depending on the network's characteristics, and shown how they can be provided by simple, yet efficient schemes.

This chapter completes the description of our approach to guaranteed performance communication. In the next chapter, we discuss the strengths and weaknesses of our approach, and propose countermeasures to some of the drawbacks of this approach.

Chapter 5: Conclusions

5.1 Thesis Summary

In this section, we present a brief summary of our investigation, and describe our solution to the problem of predictive resource allocation, which was introduced in Chapter 1.

The problem of guaranteed performance communication was viewed as equivalent to signing a contract between the communicating client and the network. The contract was looked upon as consisting of three components: a conventional contract, which dealt with the problem of providing a connection-oriented service in the network; a traffic contract which dealt with the obligations of the clients towards the network, and a performance contract that dealt with the obligations of the network towards the client. Within each of these components, we had to solve four problems, the specification of the contract, a mechanism for meeting the contract, a mechanism for mapping the client-network contract into a set of local contracts at different nodes in the network, and a mechanism for verifying if the contract is being met

We did not examine the conventional contract in any detail since most of the problems and issues in this area have already been studied extensively, and solutions to these problems exist.

In Chapter 2, we examined different traffic models that can be used for the purpose of resource allocation in high speed networks. We concluded that the x_{min}-x_{ave}-I model appears to be the best of all the three models examined from the perspectives of simplicity, usability, and efficiency. We also presented algorithms that the network can use to verify if a client is obeying the model, and a leaky-bucket scheme that allows clients to keep their promises. Thus, Chapter 2 addressed the issues of specification and mechanism for the traffic contract.

In Chapter 3, we studied the problem of the variations affecting model parameters at different points in a network. For connections requiring a loss rate smaller than 1%, the change in the value of x_{min} can not be ignored. We proposed a reconstruction policy that can be used to correct for the variation in model parameters. We also found that the changes in the value of x_{ave} were not significant, and that no major correlations can develop along different connections, even in a fairly heavily loaded network. That chapter dealt with the issue of mapping and verification for the performance contract.

In Chapter 4, we presented schemes that the network can use to provide performance guarantees to communicating clients. We identified three different QOS menus, and discussed the environments where their use would not be most appropriate. We provided simple admission control rules, scheduling policies and buffer allocation schemes for all the three menus, and studied their performance by means of simulation. The schemes usually resulted in a reasonably high bandwidth utilization (in the region of 50-70%), but buffer utilization was not as high (about 30% for QOS menu 1 or QOS menu 2 class 0, which tolerated few losses). This chapter dealt with all the four issues in the performance contract.

Table 5-1. Components in the solution of predictive resource allocation

	(C1) **Conventional Contract**	(C2) **Traffic Contract**	(C3) **Performance Contract**
(1) Specification	Addressing Issues **(solutions exist)**	x_{min}-x_{ave}-I model **Chapter 2**	Three menus of QOS **Chapter 4**
(2) Mechanism	Communication Abstraction **(solutions exist)**	Leaky bucket scheme **Chapter 2**	Admission control and scheduling for each menu of QOS **Chapter 4**
(3) Mapping	Routing Issues **(solutions exist)**	Full or partial reconstruction **Chapter 3**	Distributed or centralized performance distribution **Chapter 4**
(4) Verification	Authentication, Acknowledgments **(solutions exist)**	Rate-control. **Chapter 3**	Statistics at the receiver **Chapter 4**

In brief, the thesis has identified the different problems that are encountered in the area of resource allocation when performance in communication networks is to be guaranteed, and proposed efficient practical schemes that can be used to solve these problems. Therefore, we hope that the thesis has led to a better understanding of the issues involved in guaranteeing performance in high-speed networks.

Table 1-1 in Chapter 1 indicated the problems that had to be solved in the area of guaranteed performance communication.We reproduce the same table here, but this time we indicate the solutions to different sub-areas in different fields. Table 5-1 summarizes the results of our thesis, and is the counterpart to

Table 1-1.

In the next sections, we discuss the limitations of our approach and contributions to the problem of resource allocation in high-speed networks.

5.2 Limitations of the thesis

We have taken a predictive approach to guaranteeing performance. The predictive approach suffers from some obvious disadvantages.

The first disadvantage is the potentially long time required for the establishment of a connection. The establishment of a guaranteed-performance connection requires at least one round-trip, with a number of tests to be performed at each of the nodes along the path. In an environment where most conversations are short-lived, the connection establishment overhead may be too high. The two possible solutions to this problem are (1) to develop a scheme that can cache some long-lasting guaranteed-performance connections between a source and a destination, and multiplex connections with compatible requirements on these long-lasting connections or (2) to develop a scheme that allows a sender to start transmitting cells without waiting for the completion of the round-trip establishment procedure. The first approach has been explored in [Damaskos 89a] and the second approach in [Damaskos 89b]. The multiplexing scheme views a long-lasting guaranteed-performance connection as a two-node network; multiplexing is achieved by establishing guaranteed performance connections over this network. Cells can be transmitted without waiting for a round-trip establishment time if the intermediate nodes can determine and use local performance bounds that are “safer” than the local performance bounds to be used later on. It is possible

to do so with all the three methods of performance distribution described in Section 4.3.

Another problem of predictive resource management is that this type of management is bound to utilize the network resources less than reactive resource management. By carefully monitoring the state of the network, a reactive resource manager can achieve almost 100% utilization of network resources such as bandwidth. A predictive scheme, which is based on conservative estimates, can never attain that goal. The performance of our resource allocation schemes has been reasonable, but far from the maximum possible utilization of 100%. In chapter 2, we saw that the maximum possible utilization for the network using leaky buckets was about 70%. We would like to point out, however, that the predictive scheme can co-exist with reactive schemes being used for lower priority traffic. Thus, any unused bandwidth can be used for lower priority traffic. Unused buffer space can not, however, be utilized by lower priority traffic since we are either using static allocation (QOS menu 1) or space sharing only among connections with the same priority (QOS menu 2 and 3). Moreover, a reactive scheme may be too slow to use for guaranteeing performance to connections that have large propagation delays. In a network where connections with large propagation delays as well as connections with small propagation delays may co-exist with the same priority, a predictive scheme provides a simple approach to guaranteed-performance communication.

Our resource allocation policies can work only in association with communication protocols that support the notion of quality of service according to the traffic model and performance parameters that we have been using. Thus, a change

in the traffic model or in performance parameters would require changes in those protocols. Some of these changes may be easy; for example, modification of the admission control tests does not require any major change in the protocol suite; while some other changes, for example, a change in the performance distribution scheme may be more difficult to make.

The reconstruction process is based on a strict adherence to the contract. Thus, a channel which signed a contract for a certain bandwidth can not get a larger bandwidth even if no other channel is present. The notion of marked packets, where a channel asks for a certain bandwidth for high priority packets but is also allowed, if sufficient bandwidth is available, to transfer low priority packets (these packets are marked as droppable if bandwidth is not available [Gallassi 89]) is not supported by our scheme.

We would like to mention that, while the list of limitations is large, it is possible to extend our work to remove some of these limitations. Section 5.4 examines some possible steps that can be taken in this direction.

5.3 Contributions of the thesis

In this section, we summarize the contributions of our thesis. We tried to understand the problem of predictive pessimistic resource allocation in high-speed networks, and offered solutions in previous chapters.

The approach outlined above offers many advantages. The scheme can be used in local-area networks as well as in wide-area networks with large propagation delays. The scheme is valid for a wide range of network bandwidths and

buffer spaces available in the ATM nodes and switches. The real-time traffic utilization achieved in a typical node is in the (20-50%) range for all the three QOS menus even in the presence of bursty traffic, this is a reasonable range of values for a reservation scheme. The scheme is well-defined, and robust in the sense that it can protect clients from one another. The admission control tests are easy to perform, and channels can be added or deleted without a lot of computation.

The approach in our thesis has been to separate the ideas of traffic and performance contracts. One advantage of this separation is that connections with any type of performance and traffic requirements can be supported. Many other proposals in the area ([Golestani 90] [Kalmanek 90]) suffer from an implicit dependency between the bandwidth and the delay of a connection, with the result that low-bandwidth low-delay connections cannot be supported efficiently.

Since our scheme has identified and solved the problems in different subareas of predictive resource management, our approach scales to networks of any size. Moreover, the scheme is general enough that it can be used for a diverse number of applications. The different QOS menus that have been presented in this thesis span a wide spectrum of network bandwidths and buffer-space availabilities.

To the best of our knowledge, this thesis is the only work in the area of resource allocation that has studied the problem of changes in traffic parameters inside a network, and proposed a scheme that takes this variation into account. Legalistic contractual performance guarantees are also a unique characteristic of the Berkeley approach to real-time communication, which forms the basis of this

thesis.

5.4 Future Work

While the thesis has offered a basic solution to the problem of guaranteed performance communication, a number of issues still remain to be explored. Some of these issues are being explored by other researchers in the author's group, while others are still waiting for attention.

One of the critical policies that can affect network performance is the routing policy to be used for guaranteed performance communication. It is not clear that the routing policies used for conventional virtual circuit routing [Tanenbaum 88] would be efficient for real-time communication as well. We need to explore the routing issues in more detail.

Guaranteed performance communication algorithms can only operate with a protocol suite that supports the notions of traffic and performance contracts.As mentioned in Section 1.3.5, the thesis is a contribution to the concept of real-time channels, which is part of the Berkeley approach to real-time networking. A protocol suite within the Berkeley approach is being designed by the Tenet group at U.C. Berkeley, and can also be used with the ideas proposed in this thesis.

In order to increase the utilization of the network resources by real-time traffic, we can examine more sophisticated traffic models and smarter admission control rules. While our schemes result in a reasonable utilization of bandwidth, we are not performing that well regarding buffer utilization because of our use of static buffer allocation. It should be noted, however, that high buffer utilizations

are unreasonable to expect in any design since the primary purpose of buffers is to prevent the loss of cells during the occurence of low-probability exceptional conditions.

If the use of the network were free, all clients would ask for the best quality of service available even if they do not need it, thus allowing very few channels to be established, with a very poor utilization of network resources. A pricing scheme needs to be explored to prevent this from occurring.

In the thesis, we have taken a very legalistic view of the contract. Thus, we are not able to support concepts like marked packets [Gallassi 89], where clients are allowed to use extra bandwidth for low priority (marked droppable) packets if the network has enough resources available. This limitation can be removed by having a rate-control/reconstruction module which is more flexible than those that we propose.

Finally, we need to find a scheme which would allow the communicating clients to determine if the network is meeting their required delay bound in the absence of synchronized clocks.

The above list is only indicative and not an exhaustive enumeration of all the work that can and should be done in the area. As more experience is gained in the realm of guaranteed performance communication, we will undoubtedly discover new areas of research.

Appendix 1: Computation of Overflow Probability

A1.1 Fast Evaluation of Overflow Probability

In Sections 4.3, 4.5.2 and 4.6.2 of the thesis, we needed to compute an overflow probability. The computation of the overflow probability can be reduced to the following problem.

Given N independent variables $X_1, X_2 \ldots X_N$, X_i taking the value w_i $(0 \le w_i \le 1)$ with probability p_i, and the value 0 with probability $(1 - p_i)$, compute the probability that $X_1 + X_2 \ldots + X_N > 1$.

For the case of admission control in Sections 4.3, 4.5.2 and 4.6.2, the variable X_i would be associated with the i^{th} channel at a node, with the value of p_i given by the ratio of the minimum inter-cell interval x_{min} to that of the average inter-cell interval x_{ave}; and the variable w_i defined as the ratio of the service time for one cell to that of x_{min}.

For the case of channel multiplexing referred to in Chapter 5, the variable X_i can be defined for the i^{th} virtual channel to be multiplexed on a physical channel with the value of p_i given by the ratio of the minimum inter-cell interval on a virtual channel to the average inter-cell interval on the virtual channel; and the variable w_i defined as the ratio of the minimum inter-cell interval on the physical channel

to the minimum inter-cell interval on the virtual channel. The details can be found in [Damaskos 89b].

Although we are required to find only one value in the probability distribution of the variable $Y = X_1 + ... + X_N$, namely the probability that Y exceeds 1, we will determine the whole distribution. Whenever we add a new variable, say X_{N+1}, finding the probability that $X_1 + ... + X_{N+1}$ exceeds 1 requires knowing the probability that $X_1 + ... + X_N$ exceeds x at values other than 1. Therefore, keeping the whole distribution allows us to add (and delete) channels in an easy fashion.

The plot of $Pr(Y \geq x)$ would look like a series of steps as shown in Figure 1. The dotted lines show where the steps begin. Each step begins at a

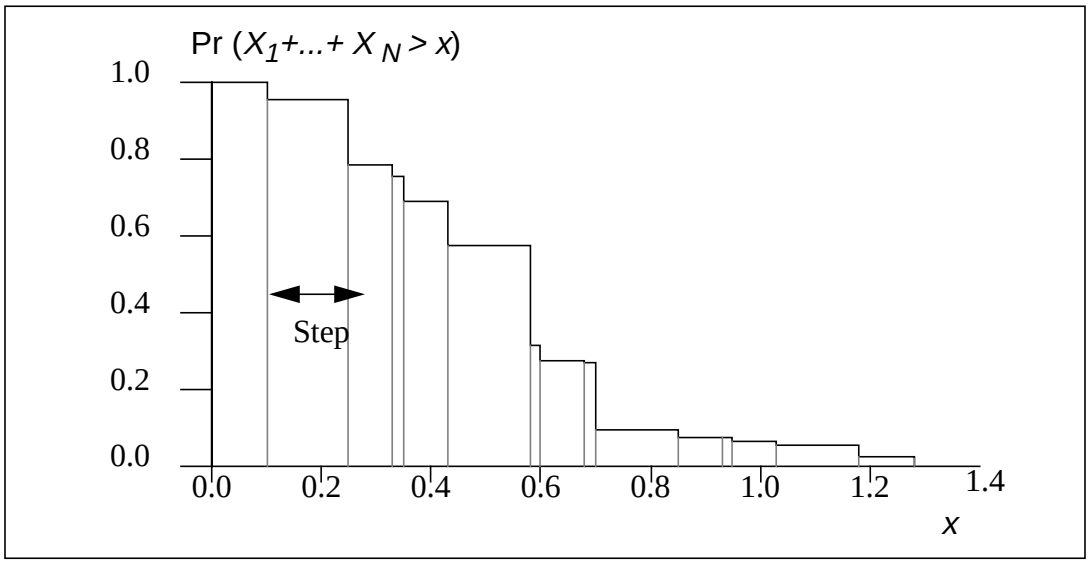

Figure 1 This figure shows the typical shape of the tail distribution of the sum of independent random variables. It consists of a number of steps of different sizes, beginning at arbitrary locations. The dotted lines show the points where the steps begin. If there are N variables in the sum, we may have 2^N steps in the worst case.

point where x equals the sum of w_i for some subset of the N variables $X_1 ... X_N$. In general, the (horizontal) width of the steps can be arbitrary, even very small. In the worst case, there may be as many as 2^N steps in this function, and maintain-

ing the distribution will take $O(2^N)$ time, when adding or deleting any channel.

Figure 2 shows an approximation of the same function if we restrict ourselves to some fixed number of constant-width steps, In order to ensure that the

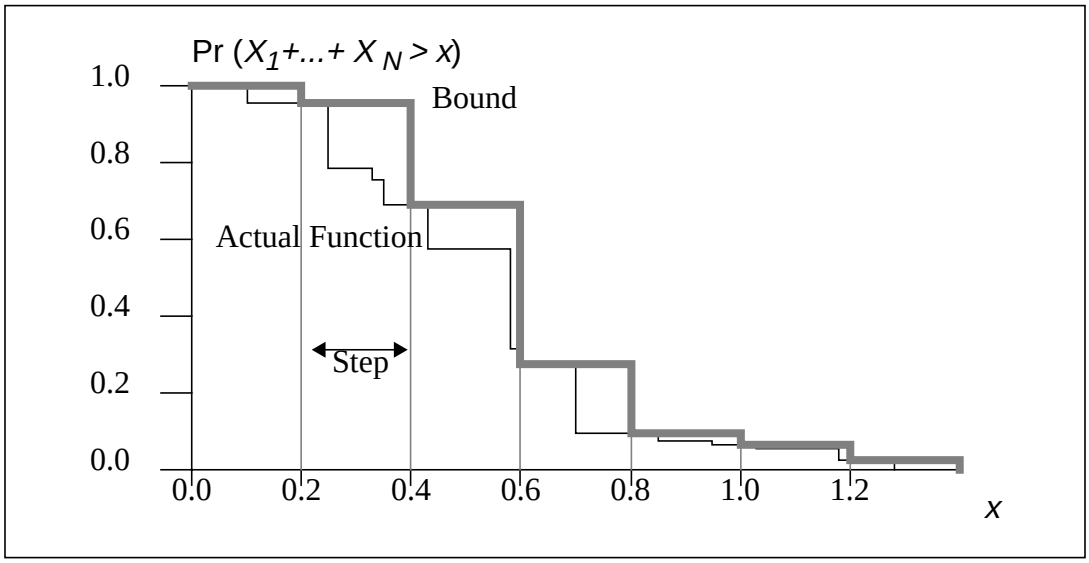

Figure 2 This figure shows the idea behind obtaining an upper bound without exponential computation. We consider steps of fixed width, and in each step obtain an upper bound on the tail distribution. The solid line shows the actual tail distribution, with the thick hatched line marking the approximation and bound. Vertical dotted lines indicate the start of a constant-width step.

approximation thus obtained is an upper bound on the actual distribution, the approximation always assumes that the maximum value of the probability in the entire step is the probability throughout the width of that step. The new constant-width steps are shown by the dotted lines. It is clear if all the actual function steps (that is, steps marked by the vertical dotted lines in Figure 1) started at a location that was also the start of an approximated step (the vertical dotted lines in Figure 2), the approximation would have been the same as the actual probability distribution. If each step in the approximation is δ units wide, and each of the values w_i be an integral multiple of δ, each of the steps in Figure 1 corresponds to

some set C of connections and begins at $x = \sum_{i \in C} w_i$, which is an integral multiple of δ. In this case, the approximate would have been exact. Clearly, as δ becomes smaller, the approximation approaches the actual distribution, even if the w_i are not integral multiples of δ.

The idea behind the fast approximation is to maintain a constant number of steps. Let us define the approximate function *g(N,x)* as the value of the bound, that is, $g(N, x) = Pr\left(\sum_{i=i}^{N} X_i \geq 1\right)$. Also assume that each of the steps in our approximation is δ units wide. On adding a new variable *N+1*, we round off its weight to the next higher multiple of δ and try to add this channel to the existing bound. Conditioning on the new channel,

$$g(N+1, \delta m) = (1 - p_{N+1})\, g(N, \delta m) + g(N, \delta (m - \lceil w_{N+1}/\delta \rceil))p_{N+1} \quad . \textit{(A1-1)}$$

Similarly, deletion of a variable (say the *N+1*th variable) can be done by noticing that:

$$g(N, \delta m) = \left(\frac{1.0}{1 - p_{N+1}}\right) [g(N+1, \delta m) - g(N, \delta (m - \lceil w_{N+1}/\delta \rceil))p_{N+1}] \,. \textit{(A1-2)}$$

Expanding the second factor on the right hand side repeatedly, and recalling that *g(N, x)* for negative or zero *x* is *1.0*, we obtain that

$$g(N, \delta m) = \sum_{k=0}^{\lceil m/h \rceil} \frac{(-1)^k p_{N+1}^k}{(1 - p_{N+1})^{k+1}} g(N+1, \delta (m - kh)), \qquad \textit{(A1-3)}$$

where *h i*s defined as $\lceil w_{N+1}/\delta \rceil$.

We now illustrate the approximation process by means of an example. Let us assume that there are 3 channels that are requested at the node. Let them

have the properties shown in Table 1.

Table 1.

Channel	**p**	**w**
1	0.2	0.2
2	0.1	0.2
3	0.3	0.4
4	0.1	0.3

For the sake of illustration, we choose δ to be 0.2. This would require that we have six columns in our table. In the beginning the table is initialized so that the first entry is 1.0 and all the rest all zero. The various entries in the table as channels are added are shown in Table 2.

Table 2.

	Pr(S > x)					
event	x = 0.0	x = 0.2	x = 0.4	x = 0.6	x = 0.8	x = 1.0
add ch1	1.0	0.2	0.0	0.0	0.0	0.0
add ch 2	1.0	0.28	0.02	0.0	0.0	0.0
add ch 3	1.0	0.352	0.118	0.028	0.002	0.0
add ch 4	1.0	0.5464	0.3826	0.1252	0.0368	0.0084
del ch 3	1.0	0.496	0.314	0.084	0.006	0.0

The deletion of channel 3 has been done without considering the order in which it was inserted. One may verify that this is exactly the result that we would get if channels 1, 2 and 4 were added respectively, and channel 3 request had not come at all. This can be seen in Table 3.

How does the scheme compare with the actual probabilities? One may

Table 3.

	Pr(S > x)					
event	x = 0.0	x = 0.2	x = 0.4	x = 0.6	x = 0.8	x = 1.0
add ch1	1.0	0.2	0.0	0.0	0.0	0.0
add ch 2	1.0	0.28	0.02	0.0	0.0	0.0
add ch 4	1.0	0.5464	0.3826	0.1252	0.0368	0.0084

verify that it is indeed exact when channels 1, 2 and 3 are added. The only source of error is channel 4, since its *w* of 0.3 is not an exact multiple of 0.2. The exact probability is 0.0006 while the bound is 0.0084. The high error need not discourage us, since a smaller value of δ produces results that are fairly accurate.

A1.2 The Algorithms

In this section, we explicitly give the algorithms that need to be executed whenever a channel needs to be added to or deleted from a node (or for the purpose of multiplexing).

The algorithm attempts to compute the array `prob_tab` which is an array of size `NUM_ELEM`. The value of δ is given as `1.0/(NUM_ELEM-1)`. The probability table is initialized by the routine `prob_tab_init`.

```
prob_tab_init(prob_tab)
float prob_tab[];
{
   int i;
   prob_tab[0] = 1.0;
   for(i=1;i<NUM_ELEM;i++) {
      prob_tab[i] = 0.0;
   }
}
```

The probability that a non-negative random variable will exceed zero is always one. Thus, the first entry of `prob_tab` is initialized to one, while the other entries are initialized to zero.

Whenever, a new channel is to be added, we can check what the new probability of overflow is going to be by means of a few simple additions. Given the probability `prob` of activity of the new channel, and the weight `wt` of the new channel, the routine `new_prob` returns the value of the new overflow probability.

```
new_prob(prob,wt)
float prob, wt;
{
   int skip = ⌈wt/δ⌉;
   int i;
   float answer;

   answer = prob*prob_tab[NUM_ELEM-skip]
      + (1.0 - prob_tab) * prob_tab[NUM_ELEM];

   return(answer);
}
```

If a channel passes all the tests and is to be added to the set of existing channels, then the routine `add_channel` is invoked. It uses the same parameters of `prob` and `wt`. The differentiation between `new_prob` and `add_channel` allows us to test if a channel can be safely added without updating the probability table. Thus, the admission control tests can be performed very rapidly.

When we have to delete a channel, we need a temporary working array of the same size as `prob_tab`. The temporary array is used to store the previous value of `prob_tab` as we update the new values. The procedure is described by the routine `delete_chan`.

```
 add_channel(prob,wt)
 float prob, wt;
 {
    int skip = ⌈wt/δ⌉;
    int i;

    for(i=NUM_ELEM;i>skip;i--) {
       prob_tab[i] = prob*prob_tab[i-skip]
           + (1.0 -prob) * prob_tab[i];
    }
    for(i=skip;i>=0;i--) {
       prob_tab[i] = prob + (1.0 - prob) *prob_tab[i];
    }
 }


delete_chan(prob,wt)
float prob, wt;
{
   int skip = ⌈wt/δ⌉;
   int i;
   float tmp_arr[NUM_ELEM+1];

   for(i=0;i<skip;i++) {
      tmp_arr[i] = (prob_tab[i] - prob)/(1.0 - prob);
   }

   for(i=skip;i<NUM_ELEM;i++) {
      prod = 1.0/(1.0 - prob);
      tmp_arr[i] = 0.0;
      for(j=0;j<⌈i/skip⌉; j++) {
          tmp_arr[i] += prod * prob_tab[i];
          prod = prod * prob/(1.0 - prob) * -1;
      }
   }

   /* Now copy the temporary array as the new prob_tab */

   for(i=0;i<NUM_ELEM;i++) {
      prob_tab[i] = tmp_arr[i];
   }
}
```

The algorithms have been presented in a C-like pseudo-code. A similar set

of algorithms can be used in the case of channel multiplexing, or for evaluating the overflow probability in other cases as well.

A1.3 Comparison with Other Approximation Methods

Our approximation method is only one of the many possible ones that can be used to obtain an upper bound on the overflow probability. A number of other upper bounds can be found in literature ([Bennett 62], [Hoeffding 63], [Fuk 71], and [Miyao 91]). All of these bounds are based on a Chernoff-type bound [Chernoff 52] which states that for any random variable *Y*:

$$Pr(Y > x + \mu) \leq E(e^{sY})\, e^{-sx} \qquad \textit{(A1-4)}$$

where μ is the expected value of a random variable *Y* and *s* is any non-negative constant. Taking into account the fact that *Y* is a sum of a *N* independent variables ($Y = X_1 + ... + X_N$), we obtain that

$$Pr\left(Y > x + \sum_{i=1}^{N} p_i w_i\right) \leq \left(\prod_{i=1}^{N} [\, p_i e^{sw_i} + (1 - p_i)\,]\right) e^{-sx}, \qquad \textit{(A1-5)}$$

where the relevant values of μ and $E(e^{sY})$ have been substituted. Since the expected value of Y is the same as the expected utilization of a node, and thus strictly less than 1, equation (A1-5) can be used to obtain a bound on the value of the overflow probability.

The different approximations in the literature ([Bennett 62], [Hoeffding 63], [Fuk 71], and [Miyao 91]) are refinements of this bound, and propose the selection of different values of *s*. [Miyao 91] applies Chernoff's bounds to the problem of admission control in ATM networks, and obtains the value of *s* which minimizes

the probability bound by solving a transcendental minimization equation numerically.

Instead of implementing the numerical methods for solving a complex equation, we tried to compare our approximation method by using the same workload that was used in [Miyao 91]. The workload consisted of a mix of two types of channels, one with a p_i of 0.5 and a w_i of 0.05, and the other one consisted of channels with a p_i of 0.1 and a w_i of 0.05. The results in [Miyao 91] indicate that the number of accepted channels using the approximate bounds differed by about 10% (for example, one could only accept 58 of the first type of channels instead of 70 channels that one would have been able to accept using the exact computation).

How does that compare with our approximation method? Suppose we decided to use 21 bins in our approximation method. In that case, δ (the length of an approximation step) would have been 0.05, same as the value of w_i, and we would have obtained the accurate value of the overflow probability. Thus, there would have been no decrease in the number of accepted connections using our approximation.

How inaccurate are we in the cases where the weights could be random and not an exact multiple of δ; and how does this inaccuracy depend on the number of approximation steps (`NUM_ELEM`)? In order to answer this question, we present the value of the approximate overflow probability computed for different values of `NUM_ELEM`. 10 channels with w_i and p_i randomly distributed between 0 and 1 were considered. The value of the exact overflow probability was 0.02061.

We show the approximate value obtained for different values of NUM_ELEM in Figure 3.:

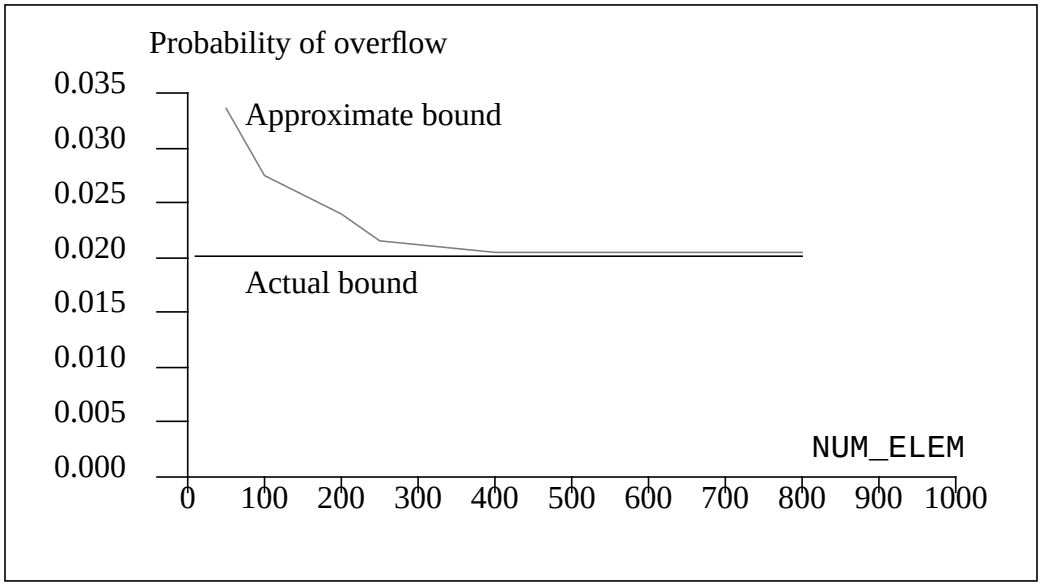

Figure 3 The accuracy of approximate bound for different values of NUM_ELEM.

It is obvious from the figure that the use of a larger value of NUM_ELEM results in a more accurate bound. We found that about 1000 bins are adequate to obtain the approximate overflow probability to a reasonable accuracy in most of our simulations.

References

References

[Ahmadi 89] H. Ahmadi and W. Denzel, “Survey of Modern High Performance Switching Techniques”, *IEEE Journal on Selected Areas in Communications*, Vol 7, no 7,September 1989, pp. 1091-1103.

[Akhtar 87] S. Akhtar, “Congestion Control in a Fast Packet-Switching Network”, *Master’s Thesis*, Washinngton University, December 1987.

[Anderson 90] D. P. Anderson, S. Tzou, R. Wahbe, R. Govindan and M. Andrews, “Support for Continuous Media in the DASH System”, *Proceedings of the 10th International Coference on Distributed Computing Systems*, Paris, France, May 1990.

[Andrews 89]M. Andrews, “Guaranteed performance for Multimedia in a General Purpose Distributed System”, *Master’s Thesis*, Computer science Division, University of California, Berkeley, September 1989.

[Banerjea 91] A. Banerjea and B. Mah, “The Design of a Real-Time Channel Administration Protocol”, *preprint* September 1991.

[Bennett 62] G. Benett, “Probability Inequalities for the Sum of Independent Random Variables”, J*ournal of the American Statistical Association*, vol. 57, pp. 33-45, March 1962.

[Brown 88] R. Brown, “Calendar Queues: A Fast O(1) Priority Queue Implemen-

tation for the Simulation Event Set Problem", *Communications of the ACM*, vol. 31, no. 10, pp. 1220-1227, October 1988.

[CCITT I.128] Recommendation no I.128 of the Intenational Telegraph and Telephone Consultative Committe, January 1989.

[CCITT G.142] Report of the working group 142 of the Intenational Telegraph and Telephone Consultative Committe, 1989.

[Chernoff 52] H. Chernoff, "A Measure of Asymptotic Efficiency for Tests of a Hypothesis Based on the Sum of Observations", *Annals of Mathematical Statistics*, vol. 23. pp. 493-509, 1952.

[Choi 89] T. Y. Choi, "Statistical Multiplexing of Bursty Sources in an ATM Network", GTE Laboratories, Waitham, Massachussets, *Talk* presented at GMD, Berlin.

[Cidon 88] I. Cidon, I. Gopal, G. Grover and M. Sidi, "Real-Time Packet Switching: A Performance Analysis" *IEEE Journal on Selected Areas in Communications*, Vol 6, no 9 , Dec 1988

[Chen 89] D. Chen, J. Walrand and D. Messerschmitt, "Dynamic Priority Protocols for Packet Voice", *IEEE Journal on Selected Areas in Communications.* vol 7, no. 5 , pp 632-643, June 1989.

[Condruse 87] J. P. Condruse and M. Servel, "Prelude: an Asynchronous Time-Division Switching Network", *Proceedings of the IEEE International Conference on Communications 1987*, Seattle, June 1987, pp 769-773.

[Comer 88] D. E. Comer and Ravi Yavatkar, "Flows: Performance Guarantees in

Best Effort Delivery Systems", Rept. No. CSD-TR-791, Computer Science Department, Purdue University, May 1988 also *Proceedings of INFOCOM*, Ottawa, Canada, pp. 100-109, April 1989.

[Cruz 87]R. L. Cruz, "A Calculus for Network Delay and a Note on Topologies of Interconnection Networks", Rept. No. UILU-ENG-87-2246, *Ph. D. Thesis,* University of Illinois, July 1987.

[Damaskos 89a] S. Damaskos and D. C. Verma, "Fast Establishment of Real-Time Channels", TR-89-056, *International Computer Science Institute*, Berkeley, Oct 1989.

[Damsakos 89b]S. Damaskos and D. C. Verma, "Multiplexing Real-Time Channels", TR-89-057, *International Computer Science Institute*, Berkeley , Oct 1989.

[Demers 89] A. Demers, S. Keshav and S. Shenker, "Analysis and Simulation of a Fair Queueing Algorithm", *Proceedings of SIGCOMM*, September 1989, pp. 1-12.

[Esaki 90] H. Esaki, K. Iwamura and T. Kodama, "A Simple and Effective Admission Control Method for an ATM network", *Proceedings of GLOBECOM*, San Diego, California, pp. 300.5.1-300.5.6, December 1990.

[Ferrari 90a] D. Ferrari and D. C. Verma, "A Scheme for Real-Time Channel Establishment in Wide-Area Networks", *IEEE Journal on Selected Areas in Communications*, April 1990, vol. 8 no. 3 pp. 368-379.

[Ferrari 90b] D. Ferrari, "Client Requirements for Real-Time Communication Services", *IEEE Communications Magazine*, November 1990, vol 28, no. 11, pp 65-72.

[Ferrari 90c] D. Ferrari and D. C. Verma, “Real-Time Communication in a Packet-Switching Network”, *Proceedings of he Second International Workshop on Protocols for High-Speeds Networks*, Palo Alto, November 1990.

[Ferrandiz 91], J. M. Ferrandiz and A. A. Lazar, “Admission Control for Real-Time Packet Sessions”, *Proceedings of INFOCOM*, 1991.

[Fitzpatrick 89]G. J. Fitzpatrick and E. A. Munter, “Input Buffered ATM switch traffic Performance”, *Multimedia* ’89 Ottawa, April 20-23, 1989.

[Fuk 71] D. K. Fuk and S. V. Nagaev, “Probability Inequalities for Sums of Independent Random Variables”, *Theory of Probability and Its applications*, vol. 16, no.4, pp 643-660, 1971.

[Gallassi 89] G. Gallassi, G. Rigolio and L. Fratta, “ATM: Bandwidth Assignment and Bandwidth Enforcement Policies”, *Proceedings of GLOBECOM*, Dallas, Texas, pp. 49.6.1-49.6.6, December 1989.

[Gallassi 90] G. Gallassi, G. Rigolio and L. Fratta, “Bandwidth Assignment in Prioritized ATM networks”, *Proceedings of GLOBECOM*, San Diego, California, pp. 505.2.1-505.2.5, December 1990.

[Golestani 90] S. J. Golestani, “Congestion-Free Communication in Broadband Packet Networks”, *Proceedings of the IEEE International Conference on Communications* 90, pp. 308.3.1-308.3.6.

[Gopal 87] I. S. Gopal, I. Cidon and H. Meleis, “PARIS, An Approach to Integrated Private Networks”, in *Proceedings of the IEEE International Conference on Com-*

munications-87, Seattle, WA, June 1987, pp. 764-773.

[Gusella 90] R. Gusella, “A Characterization of the Variability of Packet Arrival Processes in Workstation Networks”, *Ph.D. Thesis*, University of California, Berkeley, also Technical Report No. UCB/CSD 90/612. December 1990.

[Harita 89] B. Harita and I. M. Leslie, “Dynamic Bandwidth Management of primary rate ISDN to support ATM access”, *Proceedings of SIGCOMM*, Septermber 1989, pp 197-211.

[Hoeffding 63] W. Hoeffding, “Probabilty Inequalities for Sums of Bounded Random Variables”, *Journal of the American Statistical Association*, vol. 58, pp. 13-30, March 1963.

[Hui 87] J. Y. Hui and E. Arthurs, “A Broadband Packet Switch for Integrated Transport”, *IEEE Journal on Selected areas in Communuications*, Vol 5, No 8, pp. 1264-1273, October 1987.

[Hui 88] J. Y. Hui, “Resource Allocation in Broadband Networks”, *IEEE Journal on Selected areas in Communuications*, Vol 6, No 9, December 1988.

[Kalmanek 90] C. Kalmanek, H. Kanakia and S. Keshav, “Rate Controlled Servers for Very High-Speed Networks”, *Proceedings of GLOBECOM*, San Diego, December 1990, pp. 300.3.1-300.3.9.

[Karol 87] M. J. Karol, M. G. Hlyuchi and S. P. Morgan, “Input versus output queueing on a space-division packet switch”, *IEEE Transactions on Communications*, COM-35, No 12, Dec. 1987 pp 1347-1356.

[Keshav 91] S. Keshav, “Congestion Control in Computer Networks”, *Ph.D. Thesis*, University of California at Berkeley, August 1991.

[Lazar 90] A. Lazar, A. Temple and R. Gidron, “An Architecture for Integrated Networks that Guarantees Quality of Service”, *International Journal of Digital and Analog Communication Systems*, vol. 3, pp. 329-238.

[Leiner 89] B. Leiner, “Critical Issues in High Bandwidth Networking”, I*nternet RFC 1077*, November 1988.

[Liu 73]C. L. Liu and J. W. Layland, “Scheduling Algorithms for Multiprogramming in a Hard Real Time Environment”, *JACM* Vol. 20 No. 1, Jan. 1973, pp. 46-61.

[Low 91] S. Low and P. Varaiya, “A Simple Theory of Traffic and Resource Allocation in ATM”, *Proceedings of GLOBECOM* , Phoenix, Arizona, December 1991.

[Miyao 91] Y. Miyao, “A Call Admission Control Scheme in ATM Networks”, *Proceedings of the IEEE International Conference on Communications*, Denver, Colorado, 1991.

[Murase 89]. Murase, H. Suzuki, T. Takenchi, “Continuous bit stream oriented services in an ATM network”, *C&C Systems Research Laboratories*, NEC Corporation, Japan.

[Ohni 88] H. Ohnishi, T. Okada and K. Noguchi, “Flow Control Schemes and Delay/Loss Tradeoffs in ATM networks”,*IEEE Journal on Selected areas in Communuications*, Vol 6, No 9, Dec 1988.

[Protonotarios 88] E. N. Protnotarios, G. I. Stassinopoulos, E.D. Sykas and M. E.

Anagnostou, “Resource Allocation and Performance Aspects of ATD Networks”, *R.A.C.E. project* 2023.

[Schwetman 87]H. Schwetman, “CSIM Reference Manual version 12”, *MCC* Tech. Rept. No. ACA-ST-252-87, November 1987.

[Tanenbaum 88] A. S. Tanennbaum, “*Computer Networks*”, Second Edition, Englewood Cliffs, New Jersey: Prentice-Hall, 1988.

[Thomasian 69] A. J. Thomasian, “*The Structure of Probability Theory with Applications*”, McGraw-Hill Book Company, 1969, pp. 153-154.

[Tobagi 90] F A Tobagi, “Fast packet Switch Architectures for Broadband Intergated Services Digital Networks”, *Proceedings of the IEEE*, Januray 1990, vol. 78, No. 1,, pp. 133-167.

[Todorova 90a] P. Todorova and D. C. Verma, “Resource Allocation and Delay Constraints in ATM networks”, *Proceedings of the Second IEEE Workshop on Future Trends of Distributed Computing Systems*, Cairo, Egypt, September 1990.

[Todorova 90b] P. Todorova and D. C. Verma, “Delay Constraints and Admission Control in ATM networks”, *Proceedings pf the GLOBECOM,* San Diego, December 1990, pp. 905.6.1-905.6.4.

[Turner 86] J. S. Turner, “New Directions in Communications (or Which Way to the Information Age?)”, *IEEE Communications Magazine*, vol. 25, no. 10, pp 8-15, October 1986.

[Turner 89] G. M. Parulkar and J. S. Turner, “Towards a Framework for High Speed

Communications in a Heterogenous Networking Environment", *Proceedings of INFOCOM*, Ottawa, Canada, pp. 655-668, April 1989.

[Verma 91] D. C. Verma, H. Zhang and D. Ferrari, "Delay Jitter Control for Real-time Channels in a Packet-Switching Network", Rept. No. TR-91-007, International Computer Science Institute, Berekely, CA-94704, January 1991. Also in *Proceedings of Tricomm*, Chapel Hill, North Carolina, April 1991.

[Wolff 90] R. Wolff, "*Stochastic Modeling and the Theory of Queues*", Englewood Cliffs, New Jersey : Prentice Hall, 1989.

[Yeh 87] Y. S. Yeh, M. G. Hlyuchi and A. S. Acampora, "The knockout switch as a simple modular architecture for high-performance packet switching", *ISS -87*, March 1987.

[Zhang 87] L. Zhang, "Designing a new architecture for packet-switching communication networks", *IEEE Communications Magazine*, vol. 25, no. 9, pp 5-12, September 1987.

[Zhang 89]L. Zhang, "A New Architecture for Packet-Switching Networks", *Ph.D. Dissertation*, Massachusetts Institute of Technology, June 1989.

[Zhang 91] H. Zhang and S. Keshav, "Comparison of Rate-based Service Disciplines", *Proceedings of SIGCOMM*, Zurich, Switzerland, September 1991.