To Appear in the ACM Multimedia Journal

# Dynamic Management of Guaranteed Performance Multimedia Connections*

Colin Parris, Hui Zhang† and Domenico Ferrari
parris, hzhang, ferrari@tenet.Berkeley.EDU
Computer Science Division
University of California at Berkeley
Berkeley, CA 94720

*Keywords: Multimedia, Network Management, Quality of Service, High Speed Networks.*

**Abstract**

Most of the solutions proposed to support real-time (i.e. guaranteed performance) communication services in packet-switching networks adopt a connection-oriented and reservation-oriented approach. In such an approach, resource allocation and route selection decisions are made before the start of the communication on the basis of resource availability and real-time network load at that time, and are usually kept for the duration of the communication. This rather *static* resource management approach has certain limitations: it does not take into account (a) the dynamics of the communicating clients; (b) the dynamics of the network state; and (c) the tradeoff between quality of service and network availability, thus affecting the *availability* and *flexibility* of the real-time network services. Availability is the ability of the network to accommodate as many real-time clients as possible, while flexibility is the ability to adapt the real-time services to changing network state and client demands. In this paper, we present the Dynamic Connection Management (DCM) scheme, which addresses these issues by providing the network with the capability to dynamically modify the performance parameters and the routes of any existing real-time connection. With these capabilities, DCM can be used to increase the availability and flexibility of the guaranteed performance service offered to the clients.

## 1 Introduction

With the advent of high-speed networking, there is an increasing demand for new applications that transmit digital video and audio over packet-switched networks. These applications have stringent real-time performance requirements in terms of throughput, delay, delay jitter and loss rate [Fer90], and cannot be supported by the "best-effort" communication services currently offered in data networks. To support these performance requirements, new solutions have been proposed that provide guaranteed performance or *real-time services* in packet-switched networks [FV90, Top90, Zha89, AHS90, CGG91, PT89, LP91, CSZ92].

These solutions are usually connection-oriented, and require fixed routing and resource reservation on a per-connection basis. In most of these solutions, resource allocation and routing decisions are made at the time of connection establishment, based on the resource availability and real-time network load at that time, and are kept for the duration of the connection. However, such a static resource management approach does not reflect the inherent dynamics of client requirements and network state, both of which may change during the lifetime of the connection. Also, it does not address the tradeoff between quality of service and network availability: higher quality of service offered to a fraction of the clients may lower the availability of the network, and cause other communication requests to be rejected. It is desirable to adapt the quality of service offered to the clients based on the load of the network. Of course, in order to stay within the framework of

*This research was supported by the National Science Foundation and the Defense 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., Pacific Bell, 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 authors, and should not be interpreted as representing official policies, either expressed or implied, of the U.S. Government or any of the sponsoring organizations.

†Current address: Lawrence Berkeley Laboratory, 1 Cyclotron Road, MS: 50B-2239, Berkeley, CA 94720

guaranteed performance communication, the adaptation should be graceful (i.e., to be done with minimal or no disruption to the clients) [PVZ93]. The proposed resource management algorithms, in their current state, cannot support such graceful adaptations.

In this paper, we present the Dynamic Connection Management (DCM) scheme, which permits the modification of the *traffic* and *performance* parameters, and of the *route* of a connection during the lifetime of that connection. These modifications are subject to a *DCM modification contract* that constrains the degree of disruption that a connection will experience during parameter or route modification. Based on the modification contract and the semantics of the applications, modification can be done by the network without any client involvement. With these capabilities, DCM can be used to increase the availability and flexibility of the guaranteed performance service offered to the clients.

The DCM scheme is based on the framework of the guaranteed performance communication service (i.e., the *Tenet* framework) referred to in [FV90] . In our view, a guaranteed performance communication service is a performance contract specified between one or more clients and the network. The DCM scheme enriches the semantics of this contract. Without DCM, the contract is static: after the client specifies the traffic and performance parameters of the connection and the connection is established, the network guarantees that the performance requirements of the client will be met as long as the client sends data according to the traffic specification. With the introduction of DCM, the contract will be extended to include the possible modification of the traffic or performance parameters and the level of service disruption the client is willing to tolerate during modification.

With the additional flexibility provided by DCM, a wider range of network support is possible for multimedia applications. This range includes support for variable speed visualization applications, mobile real-time communications, and multicast connection management. Examples of the support that we propose to provide for these applications will be discussed in more detail in the paper.

The paper is organized as follows: in Section 2, we motivate the need for Dynamic Connection Management by giving various examples and showing the lack of support in the current static resource management algorithms; Section 3 summarizes related work; in Section 4, we briefly review the Tenet scheme, of which the DCM scheme is an extension ; in Section 5, we describe the DCM scheme in detail; in Section 6, we present results from simulation experiments; finally, in Section 7 we conclude by summarizing the paper.

## 2 Motivation

As mentioned in Section 1, current resource management algorithms are mostly *static*, however, both the client's requirements and the network's state may change during the lifetime of the connection due to the *dynamic* nature of both the client requirements and the network state. More importantly, clients' requirements and network state may be interdependent — there is a tradeoff between the quality of service offered to clients and the availability of the real-time service. In this section, we will give examples to motivate the need for Dynamic Connection Management algorithms. We will divide our examples into three categories: a) dynamics of client requirements, b) dynamics of network state, and c) tradeoff between quality of service and network availability.

### 2.1 Dynamics of Client Requirements

Clients may need different amounts of resources during the lifetime of a conversation. For example, in the case of a still image browser, where degradation of quality is not allowed [Sto92], variation of the browsing speed corresponds to different requirements for network bandwidth. If the network does not provide a mechanism to dynamically adjust the bandwidth allocated to the connection, the visualization program will have to reserve resources according to the maximum requirements of the client at the beginning of the connection, which will result in wasted resources. However, if the client is allowed to change the amount of resources reserved for the connection dynamically during the lifetime of the connection, the client would have the option of reserving just enough resources for playback at the beginning of the program, and of acquiring more resources later if browsing at higher speed is needed.

Allowing the clients to dynamically adjust the traffic or performance parameters will also reduce the burden on the clients of setting the parameters correctly the first time. Since the future integrated services networks will have to support applications with diverse traffic characteristics and performance objectives, the interface the network exports to the communication clients must be *general* and *parameterized* [FBZ92]. Since the interface is general, it may be difficult for the clients to estimate the parameters according to the model. This problem is worsened by the fact that, in the current proposed resource management solutions, once the parameters are specified, they remain fixed during the entire lifetime of

the connection. This will force the clients to act conservatively, and mostly likely, reserve more resources than needed. If there is a mechanism to dynamically change the parameters of the connection, another mechanism can be added to estimate the appropriate parameters for the connection during data transfer.

Multicast connections also provide a rich environment for dynamic management. With multicast connections it is often the case that the addition of a new member to a multicast session already in progress produces a multicast tree that is not minimum-cost. This situation may occur even if the previous multicast tree was minimum-cost and the branch connecting the new member to the tree is also minimum-cost. In these cases, the main branches of the tree may be dynamically and transparently rerouted to produce the minimum-cost tree that includes the new member.

With the current demand for wireless real-time connections [SCB92], dynamic connection management provides the ideal primitives for wireless network support. If we assume a model in which the portable clients are intelligent [TYT91], the movement of the client among base stations can be thought of as the rerouting of the connections, which can be supported without disruption by DCM.

### 2.2 Dynamics of Network State

In addition to the dynamics of client requirements, the network state also changes during the lifetime of a connection. Examples are failures or exceeded error thresholds in links and nodes in the network. In connectionless networks, packets can be dynamically rerouted when there are failures inside the network. We would like to port this flexibility of dynamic rerouting to connection-oriented networks. This flexibility is important in that a connection that traverses a failed link or whose error thresholds have been exceeded across a link can be quickly rerouted (this could be achieved by rerouting that portion of the connection that traverses the offending link) to another link, thereby reducing the number of lost or error-ed packets occurring on this connection. These error thresholds are quantifications of error tolerances appropriate to each connection, for instance, in video connections, the number of consecutive packets or cells lost may be the error tolerance of choice rather than the total number of packets or cells lost.

Dynamic Connection Management can also be used to support network load balancing by transparently rerouting connections in accordance with an appropriate load balancing algorithm.

### 2.3 Quality of Service vs. Network Availability

One of the criticisms against reservation-based algorithms is that they do not address the tradeoff between quality and availability of service. Although the network will guarantee the quality of service to the connections already established, it may have to block other connections due to lack of resources. This is necessary if the quality of service of the established connections cannot be compromised. However, there are certain applications that have the ability to adapt to different qualities of service, and may be willing to reduce their quality requirements in cases of network saturation. For example, some video coders are designed with tunable parameters so that the compression ratio can be adjusted to output streams with different bit rates [DB89, KV88, Gha89]. When the network is less loaded, we would like the compression ratio to be small so that we can have higher video quality; however, when the network is close to saturation, we would like to increase the compression ratio so that we can admit more connections. Such adaptation, also known as *media-scaling* [DHH+93], has been proposed in the context of datagram networks [YH91, GG91], but without providing any guarantees before or after the change. By using Dynamic Connection Management on the real-time network connections, we can better address the tradeoff between quality of service and availability of service: parameter adjustments will only be applied with the consent of the application [1].

### 2.4 Possible Alternative Approaches

Our approach to the problem discussed previously is to introduce mechanisms that allow modification of traffic and performance parameters and the route of a guaranteed performance connection at the networking layer. There are two other possible alternative approaches.

In the first alternative approach, when there is a desire to change parameters, the end system establishes another connection with the new parameters, switches the traffic from the old connection to the new connection, then tear downs

[1] If network service is free, all clients will ask for the highest quality of service. Quality of service only makes sense when there is an adequate pricing structure [PKF92, CESZ91]. We assume that the pricing policy takes into account the traffic and performance parameters as well as the duration of a connection, thus providing incentive for clients to adjust their quality of service on a voluntary basis.

the old connection. In such an approach, the network does not know the relationship between the old and new connections, thus cannot share resources between them. Without the sharing of resources network, availability will be adversely affected. Also, mechanisms like rerouting a channel locally[2], which is used by the DCM algorithm to accelerate the transition time, can not be used in such an approach.

The second alternative approach is to establish additional connections to transmit the excess data when more bandwidth is needed.This second alternative can easily result in resource fragmentation. In resource fragmentation, numerous channels, with low performance requirements, are scattered throughout the network in such a manner that availability of the network to high performance channels is limited while there is obviously enough aggregate resources to support these channels. This also requires the end-system to support multiple connections for one logical stream and provide synchronization among these connections. Also, it is unclear how to degrade QOS for a connection in such an approach.

## 3 Related Work

Besides the Tenet approach, which has been implemented in the Tenet Real-Time Protocol Suite, there are several other guaranteed performance service schemes of which the authors are aware; the guarantees provided by these schemes span a wide range of performance parameters. The more widely recognized schemes are the Flow Protocol [Zha89], the Heidelberg Resource and Administration Technique (HeiRAT) [VHN92], the Session Reservation Protocol [AHS90], the Asynchronous Time Sharing (ATS) approach [LP91], the Multipoint Congram-oriented High-performance Internet Protocol [PT89], the extended Capacity-Based Session Reservation (CBSRP) protocol [TTCM92], and the Integrated Service Packet Network (ISPN) architecture [CSZ92].

Only two of these schemes permit the modification of the parameters of the communication stream. In the HeiRAT scheme [VHN92], which uses the connection oriented protocol ST-II [Top90]. Connections are established by using the ST-II Control Message Protocol (SCMP). An SCMP establishment message travels to each node along the route and attempts to acquire the needed resources from the *ST-II agents* at these intermediate nodes. If the resources can be acquired, the message is permitted to continue to the next hop. When the message successfully reaches the destination node or encounters an intermediate node whose agent cannot provide the needed resources, the message returns to the sender with the appropriate response. The SCMP specifications include an SCMP **change** message, which can be used to modify parameters, however, the HeiRAT implementation does not currently support modification of parameters or routes dynamically.

In the extended version of the CBSRP protocol [TTCM92], the user can specify the minimum and maximum values for two parameters: the desired temporal and spatial resolutions of the media to be transmitted. The specified values allow the network to assign each client to a particular class of service. When a new client requires the establishment of a session, if the available resources are already saturated, some existing sessions may be forced to reduce their qualities of service (i.e. to modify their parameters), to accommodate the new request. The minimum quality of a session is, however, always guaranteed once the session is established. One limitation of CBSRP is that the quality of service is specified only in terms of inter-packet distance and packet size, while delay, delay jitter, and packet loss bounds are not taken into account. Another limitation is that this feature is only discussed in a local area network environment, while issues associated with a more general internetworking environment are not addressed.

## 4 Background

In this section, we give a brief overview of the current version of the Tenet resource management algorithms [FBZ92]. We describe three aspects of the scheme: the Tenet performance contracts, which define the service abstraction; the Tenet mechanisms, which constitute a distributed connection establishment procedure; and the Tenet algorithms, which consist of the service discipline at the switches and the admission control tests.

### 4.1 The Tenet Performance Contract and Mechanisms

The Tenet algorithms are based on a communication abstraction called a *real-time channel* [FV90, VZF91]. A real-time channel is a network connection associated with traffic and performance parameters. The parameters are provided by the clients to specify their traffic characteristics and performance requirements. The traffic specification consists of the parameters $(Xmin, Xave, I, Smax)$, where $Xmin$ is the minimum packet inter-arrival time, $Xave$ is the average packet inter-arrival time over an an averaging interval, $I$ is the averaging interval, and $Smax$ is the maximum packet size. The

[2] Local rerouting can be used to accommodate a change in the delay bound of a connection.

performance parameters by which clients describe their requirements are: delay bound $\overline{D}$, maximum delay bound violation probability $\overline{Z}$, maximum buffer overflow probability $\overline{W}$, and delay jitter bound $\overline{J}$.

The service abstraction defines a contractual relationship between the network and the client: once the channel is established, the network guarantees, in the absence of network failures, that it will meet the specified performance requirements of the client, provided that the client obeys its traffic specification.

A channel needs to be *established* before data can be transferred. This channel establishment is achieved in the following manner: a real-time client specifies its traffic characteristics and end-to-end performance requirements to the network; the network determines the most suitable route for a channel with these traffic characteristics and performance requirements; it then translates the end-to-end parameters into local parameters at each node, and attempts to reserve resources at these nodes accordingly. This is done in a distributed manner during a round-trip communication.

On the forward pass of the channel establishment round trip, resources are reserved to get the best possible level of local performance so as to ensure that resource deficiencies further along the path can be accommodated. This process continues along each node until the destination node is reached or an intermediate node rejects the channel. At the end of the forward pass, the destination summarizes the information collected along the path and determines if all of the end-to-end performance bounds obtained by reserving resources during the forward trip are better than the corresponding client's requirements. If so, on the reverse pass, the resources reserved during the forward pass are reduced or relaxed so that only the necessary amounts of resources are committed.

## 4.2 Tenet Algorithms

In order to provide performance guarantees, two levels of controls are needed: at the connection level, channel admission control algorithms reserve resources for each of the connections and limit the maximum utilization of the network by real-time traffic; at the packet level, the service discipline at each of the switches determines the multiplexing policy and allocates resources to different connections according to their reservations.

As shown in [CSZ92, Fer92, ZK91, Zha93], many service disciplines can be used to provide real-time service. However, different service disciplines require different admission control algorithms. For the purpose of this paper, we assume that the service discipline used at the switches is Rate-Controlled Static Priority [ZF93, Zha93] or RCSP. In this section, we first briefly summarize the properties of the RCSP service discipline, then give the corresponding admission control tests.

RCSP is a service discipline proposed to achieve both *flexibility* in terms of allocating service priority and bandwidth resources to different connections, and *simplicity* in terms of high speed implementation. As shown in Fig. 1, an RCSP server has two components: a rate controller and a static-priority scheduler.

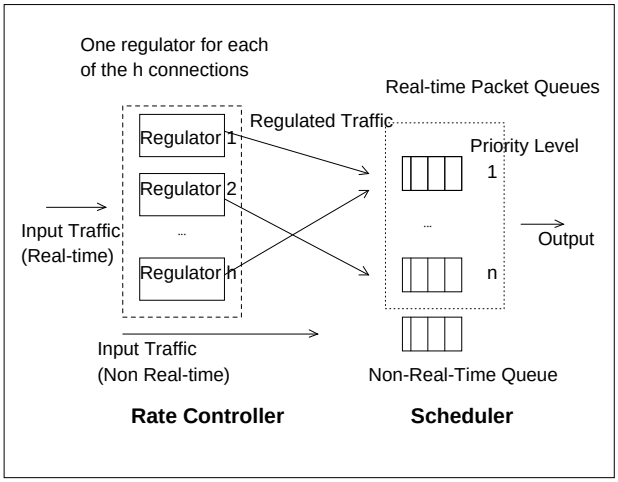



Figure 1: Rate-Controlled Static-Priority Queueing

Each priority level corresponds to a delay bound. Each connection is assigned to a priority level during connection establishment time. Multiple connections can be assigned to the same priority level. During the data transfer phase, all the packets on the connections associated with a priority level are appended to the end of the queue for that priority level. The

server services the packet at the head of the non-empty-queue with the highest priority. Once the transmission of a packet starts, it will not be preempted.

A single RCSP server can guarantee a number of local delay bounds to different connections. When the local node receives an establishment request, it determines if enough bandwidth and schedulability resources can be reserved to ensure the satisfaction of throughput and delay bound guarantees. Resources are reserved according to the results of Test 1 given below.

*Test 1: Let $\overline{d_i^1}, \overline{d_i^2}, \ldots, \overline{d_i^n}$ ($\overline{d_i^1} < \overline{d_i^2} < \cdots < \overline{d_i^n}$) be the delay bounds associated with each of the $n$ priority levels, respectively, at switch $i$. Let $C_q$ be the the set of connections which are established and assigned level $q$ ($1 \leq q \leq n$), and the $j^{th}$ connection within $C_q$ has the traffic specification ($Xmin_j^q, Xave_j^q, I_j^q, Smax_j^q$). Assume that the link speed is $l$, and the size of the largest packet that can be transmitted onto the link is $\overline{Smax}$ . A new connection with the traffic specification ($Xmin_{new}, Xave_{new}, I_{new}, Smax_{new}$) can be assigned to level $m$, or be assigned a local delay bound $d_i^m$, if the following inequality holds:*

$$\sum_{q=1}^{m'} \sum_{j \in C_q} \lceil \frac{\overline{d_i^{m'}}}{Xmin_j^q} \rceil Smax_j^q + \lceil \frac{\overline{d_i^{m'}}}{Xmin_{new}} \rceil Smax_{new} + \overline{Smax} \leq \overline{d_i^{m'}} l \quad m' = m, \cdots, n \tag{1}$$

Intuitively, the longest waiting time in the scheduler for a level-$m'$ packet corresponds to the case in which a lower-priority packet is being transmitted when the packet arrives at the scheduler, and is followed immediately by the longest possible transmission of packets with higher or equal priorities. Test 1 bounds this longest waiting time to be less than $d_i^{m'}$ for $m' = m, \cdots, n$, which are the priority levels that can be affected by placing a new connection in priority level $m$ (level 1 has the highest priority).

RCSP servers also hold packets to ensure traffic smoothness and bounded delay-jitter properties in a network of switches. The following gives the delay property for a connection traversing a tandem of RCSP switches [3].

*Delay Property: Let $\overline{d}_{1,j}, \cdots, \overline{d}_{i,j}$ be the local delay bounds for the first $i$ switches along the path traversed by connection $j$, $\pi_{i-1,i}$ be the propagation delay from switch $i-1$ to switch $i$, and $D_{i,j,k}$ be the delay experienced by the $k^{th}$ packet on connection $j$ from switch 1 to switch $i$, the following properties holds for any $k$,*

$$\sum_{i'=1}^{i-1} (\overline{d}_{i',j} + \pi_{i',i'+1}) \leq D_{i,j,k} \leq \sum_{i'=1}^{i-1} (\overline{d}_{i',j} + \pi_{i',i'+1}) + \overline{d}_{i,j} \tag{2}$$

The property gives the upper and lower bounds on the delay for any packets traversing a path of RCSP switches. The end-to-end delay jitter is bounded by the local delay bound of the last switch along the path.

To ensure enough buffers are reserved so that the performance guarantees are not violated, the following local condition must be met at switch $i$.

*Test2:*

$$R_{bu} + \lceil \frac{\overline{d}_{i-1,j}}{Xmin} \rceil \times Smax + \lceil \frac{\overline{d}_{i,j}}{Xmin} \rceil \times Smax \leq B \quad i = 1, \cdots, n \tag{3}$$

*where $\overline{d}_{i-1,j}$ and $\overline{d}_{i-1,j}$ are the local delay bounds for the connection at the $(i-1)_{th}$ and $i_{th}$ switches along the path respectively, $R_{bu}$ is the current buffer space occupied (in bits) and* B *is the maximum buffer space (in bits) allotted.*

Note that the buffer space depends on the delay bound in the previous switch. This is due to fact that RCSP switch also holds packets, and the longest time a packet from connection $j$ is held in switch $i$ is $d_{i-1,j}$ where $\overline{d}_{0,j} = 0$.

## 5 Dynamic Connection Management (DCM)

Dynamic Connection Management extends the Tenet scheme by enabling

- the modification of traffic parameters, performance parameters, and routes under global or local control subject to a modification contract that specifies the extent of disruption to be experienced by the client during modification, and

[3] There are two types of RCSP servers: delay-jitter controlling RCSP server and rate-jitter controlling RCSP server. Only delay-jitter controlling RCSP server has this delay property [Zha93]. We assume in the paper that delay-jitter controlling RCSP servers are used.

- the reduction of real-time channel establishment and modification times.

DCM is comprised of the *DCM scheme* and the *DCM policies*. The DCM scheme is the collection of algorithms and mechanisms that permit the network to dynamically modify channel parameters and routes. The modification of a channel is a procedural abstraction whereby a real-time channel with the new performance and traffic parameters (referred to as the *alternate* channel) is established, the client's traffic is moved from the current real-time channel (referred to as the *primary* channel) to the alternate channel, and then the primary channel is removed. The movement of traffic from the primary to the alternate channel is referred to as the *transition* from the primary to the alternate channel. Modifications can also be applied to change the routes of a channel.

The DCM policies are *rules* that determine if a real-time channel is to be modified, and the new values of its parameters. These rules may examine the network's or client's state data to determine if modification should take place. The DCM policies are usually implemented as management applications, and will not be discussed in this paper.

In a manner similar to the Tenet scheme, the DCM scheme can be described from three aspects: the DCM modification contract, the DCM algorithms, and the DCM mechanisms. The DCM mechanism are enhanced features of the Tenet mechanism in that they provide procedures for faster establishment and modification and greater granularity of control.

### 5.1 DCM Modification Contract

A client's request for modification is governed by the usual request/response paradigm: the client makes a modification request (or the DCM policy determines that a route modification is needed and submits a request) to the network, and the network returns a response, *accepted* or *denied*, based on the request and the current real-time network load. In the case of a client request , if the request has been accepted, the client can begin sending with the new traffic characteristics and expecting that the performances guaranteed for packets on both the primary and alternate channels are met[4]. It should be noted that the primary and alternate channels exist simultaneously for a short interval of time, *the transition interval*, and then the primary is removed. After this interval only the alternate channel will exist.

In DCM there are contractual obligations made to the client that determine the extent of the disruption that will be experienced by the client due to the transition from the primary to the alternate channel. There are two types of DCM modification contracts:

1. *No performance guarantees will be violated* during the transition from the primary to the alternate channel (The *No-Violation contract* ).

2. *A bounded number of performance violations can occur* during the transition from the primary to the alternate channel (The *Bounded-Violation contract*).

As we are only considering deterministic services in this paper, there are three types of performance violations that may occur: a *delay bound* violation occurs when a client, sending traffic within the specified bounds has at least one packet that exceeds the delay bound $\overline{D}$ at the destination node; a *delay jitter bound* violation occurs when a client, sending traffic within the specified bounds, has at least one packet that exceeds its delay jitter bound $\overline{J}$ at the destination node; and a *packet ordering* violation occurs when a client, sending traffic within the specified bounds, receives packets out of sequence.

These violations may occur singly, or multiple violations can occur simultaneously. The first type of DCM modification contract ensures that none of the three performance violations will occur during a transition to an alternate channel. This contract may be explicitly requested by the client before channel parameter modification or implicitly demanded by the network before channel route modification. Route modification is usually done by the network and must be totally transparent to the client; hence there must be no performance violations. Reroutings may be done directly for network administrative or management purposes, or indirectly due to a client's performance parameter modification request.

There is an intrinsic condition that must be satisfied to avoid performance violations. To see this, let us consider the following case: Assume the $k^{th}$ packet to be the last packet transmitted on the primary channel. Let $\overline{D^p}$, $\overline{D^a}$ be the end-to-end delay bounds of the rerouted connection on the primary and alternate paths, respectively. Also, let $s_k$, $r_k$, $D_k$ and $s_{k+1}$, $r_{k+1}$, $D_{k+1}$ be the sending time, receiving time, end-to-end delay, for the $k^{th}$ and $(k+1)^{th}$ packets, respectively. We have $s_k + D_k = r_k$, and $s_{k+1} + D_{k+1} = r_{k+1}$. To ensure in-order delivery, we need to have $r_k < r_{k+1}$, i.e. $s_k + D_k < s_{k+1} + D_{k+1}$. Re-arranging the terms we have

$$D_k < D_{k+1} + s_{k+1} - s_k \tag{4}$$

[4] This is to say that packets traversing the primary channel will meet the performance guarantees corresponding to the primary channel, and packets traversing the alternate channel will have their alternate channel performance guarantees met, subject to the modification contract explained below.

Notice that the traffic has to satisfy the traffic constraint, or $s_{k+1} - s_k \geq Xmin^a$. Also the $k^{th}$ packet traverses the primary route and $(k+1)^{th}$ packet traverses the alternate route, we have $D_k \leq \overline{D^p}$ and $D_{k+1} \geq \overline{D^a} - \overline{J^a_{path}}$, where $\overline{J^a_{path}}$ is the maximum delay jitter in the alternate route [5]. Since (4) has to hold for any values of $D_k$ and $D_{k+1}$ that satisfies delay and jitter bounds, we pick the largest value of $D_k$, the smallest value of $D_{k+1}$, and get the following,

$$\overline{D^p} < \overline{D^a} - \overline{J^a_{path}} + Xmin^a \tag{5}$$

Notice that $\overline{J^a_{path}}$ can be reduced by buffering at the destination. By introducing the transition buffer as described in section 5.2.2, $\overline{J^a_{path}}$ can be reduced to 0. So the necessary condition for a No-Violation contract is

$$\overline{D^p} < \overline{D^a} + Xmin^a \tag{6}$$

While the No-Violation contract constrains the ranges of parameter modifications, some clients can tolerate the violation of these guarantees during a transition provided that the effect is bounded. This contract is useful, as we believe most clients will expect a slight disruption in service upon modification, and will be ready to accept it as long as it is bounded.

In the case of a Bounded-Violation contract, we remove the parameter constraints and specify an upper bound on the number of packets that will exceed the delay or delay jitter bounds of the alternate channel, or arrive out-of-sequence at the destination during the transition interval. If condition (6) is not satisfied, the number of packets that can arrive out-of-sequence or exceed delay bounds is bounded by $\lceil \frac{\overline{D^p} - \overline{D^a} - Xmin^a}{Xmin^a} \rceil$. The bound will be made known to the client before the channel modification is attempted, and the client can then decide if the modification is worth attempting.

## 5.2 The DCM Algorithms

In this sub-section we will present the high level functionalities of the three DCM algorithms: the channel administration algorithm, the transition algorithm, and the routing algorithm, followed by a detailed discussion of each of them. The key function of these algorithms is to provide the support needed to modify parameters and routes under the constraints of resource sharing and of the modification contracts.

Alternate channel establishment can be examined under two scenarios, i.e. *no resource sharing* and *resource sharing*. Under the *no resource sharing* scenario, an alternate channel is established along a route that is completely disjoint from that of the primary channel or the alternate channel traverses links that are common with those of the primary channel but does not share any of the resources previously reserved by the primary channel. Under the *resource sharing* scenario, an alternate channel is established along a route which traverses some links that are common with those of the primary channel and shares resources along all of the common links[6]. We envision that resource sharing may be desirable[7] as we expect a significant number of instances in which a very large channel (i.e., very resource demanding) or multiple smaller channels are being rerouted or enhanced, and the resources required to accommodate these requests, especially during the transition interval, can only be made available using resource sharing. The Channel Administration algorithm provides the admission control tests and some additional constraints needed to support both of these scenarios. It should be mentioned that the decision to utilize resource sharing is entirely policy dependent and the algorithm merely provides the capability without imposing any judgment as to when it is used.

In determining a suitable route for the alternate channel, the routing algorithm must be able to reflect the inclusion or exclusion of the resource sharing factor. If resource sharing is not accommodated, the resources reserved by the primary channel are not considered by the routing algorithm in determining an alternate route. If resource sharing is accommodated, the routing algorithm compensates for the resources reserved by the primary channel, i.e. by "virtually" removing the resources reserved by the primary channels from its routing database [8] before calculating the alternate route. Thus, the alternate route chosen may have to share previously reserved resources if there are common links. The decision to accommodate resource sharing is a policy decision and only the routing mechanism, which must support either policy, will be considered in this paper.

The modification contracts discussed in Section 5.1 present the performance parameter constraints needed to support the "No-violation" and "Bounded-violation" contracts; however, in the "No-violation" contract additional support is needed to

[5] For a client requesting a connection with bounded delay jitter $\overline{J^a}$, the condition $\overline{J^a} \leq \overline{J^a_{path}}$ must hold.

[6] It may be that the alternate and primary route are the same.

[7] It would increase the utilization of the network.

[8] Routing in our real-time framework is source based and discussed in Section 5.2.3.

| Constraints | Channel Admission Algorithm | Transition Algorithm | Routing Algorithm |
|---|---|---|---|
| NV RS | Test Set (1,2,3,4) | Used | Adjust resources before route calculation |
| NV NRS | Test Set (1,2) | Used | Do not adjust resources before route calculation |
| BV RS | Test Set (1,2,3,4) | Not Used | Adjust resources before route calculation |
| BV NRS | Test Set (1,2) | Not Used | Do not adjust resources before route calculation |

Table 1: Impact of Constraints on DCM Algorithms

prevent packet-ordering or delay bound performance violations. The Transition algorithm provides the additional buffers and the packet reordering mechanism needed to support this contract.

Table 1 summarizes the discussion above. The constraints of resource sharing and the modification contracts are expanded and their effect on the three DCM algorithms are presented. There are two types of modification contracts, a *No-violation contract (NV)*, and a *Bounded-Violation contract (BV)* . The resource sharing constraint is reflected in two states *Resource Sharing (RS)* and *No Resource Sharing (NRS)* . There are four tests associated with the Channel Administration algorithm, the first two tests, Test 1 and 2 presented in Section 4.2, are used for admission control on non shared resources, while Test 3 and 4 (to be presented in Section 5.2.1), are used for admission control on shared resources. Along a path on which resource sharing occurs there may be both shared links and unshared links hence all of the tests (i.e. Tests 1, 2, 3, and 4) may be used for that connection.

### 5.2.1 The DCM Channel Administration Algorithm

The DCM scheme has the same procedural format as the Tenet scheme, as it is an extension of this scheme. In the DCM scheme, the Tenet channel administration algorithm has been supplemented by the DCM channel administration algorithm. In this paper, the DCM algorithms are described only for deterministic real-time [9] services.

The goal of the DCM channel administration algorithm is to establish an alternate route, conforming to the specified traffic and performance parameters, between a source and destination host. This alternate route is established in the presence of a primary route on which the client is currently active. The establishment entails the decision as to the acceptance or rejection of the client's request subject to resource availability; the algorithm must reserve the appropriate resources if they are available, so that an *apriori* guarantee is made. In the establishment of an alternate route we can choose not to utilize or to utilize resource sharing. Both scenarios are examined below.

As discussed previously, in a *no resource sharing* scenario the alternate route is completely resource independent from the primary route and the admissions tests to be applied at each link are those used in the establishment of a primary channel, i.e. Test 1 and Test 2. There is, however, one difference: the transparency procedure (discussed in Section 5.4) is used to ensure that the interface the client sees of the channel after the modification is the same as that seen before the modification.

In the *resource sharing* scenario, the alternate route is resource dependent on the primary route, and shares resources along one or more of the links that comprise the primary route. In this scenario the admissions tests applied to the common links, Test 3 and 4, reserve resources for the larger (in terms of resource reservations) of the two channels. Test 1 and 2 are still applied to the links that are not common to both routes. Test 3 and 4 are modifications of Tests 1 and 2, respectively, and take into consideration resources that are already reserved for the primary channel at that link, to ensure that there is no duplication of resources. If the primary channel has acquired resources that are greater than those of the alternate channel, these tests need not be applied. If Test 3 and 4 are successful, we have guaranteed that enough resources are available for the higher performance channel but not for both of the channels, therefore, we need to avoid the situation in which packets from both channels arrive at the link simultaneously, as resources are not reserved for both channels. This is achieved by using the delay jitter control mechanism and by properly setting the local delay bounds parameters along the paths of both the primary and the alternate channels. Notice that in a delay-jitter controlled network, the delay of a packet from a source to a switch does not have only an upper bound, but also a lower bound, and the difference between the two bounds, which

[9] These deterministic services have a deterministic delay bound and a zero buffer overflow probability.

is the delay jitter, can be tuned by properly setting the local delay bounds [ZF93, ZK91]. Assume that the shared link is an output link of switch $i$, that the upper bounds and lower bounds of delay from source to the $i$ th switch for packets of the primary and alternate channels are denoted by $\overline{D_i^{upper,p}}$, $\overline{D_i^{lower,p}}$, $\overline{D_i^{upper,a}}$ and $\overline{D_i^{lower,a}}$ respectively. To ensure that the packets arriving at the shared link obey the traffic specification, the following condition must be satisfied:

$$\overline{D_i^{upper,p}} \leq \overline{D_i^{lower,a}} \tag{7}$$

If both of these actions, passing the establishment tests and fulfilling the above delay bounds condition, can be done successfully the channel can be accepted, otherwise, it is rejected.

Resource Sharing Admissions Control Tests.

These modified tests are only applied to the common links if any of the performance requirements of the alternate channel are *greater* than those of the primary channel. When at least one of the conditions (provided below) holds, the performance needs of the alternate channel are *greater* than those of the primary channel. These conditions correspond to the throughput, delay, and delay jitter performance of the channels, respectively, and are: $\frac{Smax^a}{Xmin^a} \geq \frac{Smax^p}{Xmin^p}$; $\overline{D}^a < \overline{D}^p$; and, $\overline{J}^a < \overline{J}^p$.

To ensure that the transition from the primary to the alternate channels is as smooth as possible it is necessary to retain ample resources so that packets on either the primary or alternate channels can meet their requirements. This can be achieved by a judicious choice of parameters upon which resource reservation at this common link will be based. In Test 3 below, an initial adjustment is made to virtually remove the resources currently reserved for the primary channel, and then resources are reserved for the composite channel defined by equations (8), (9), and (10)[10].

$$Xmin^c = min(Xmin^a, Xmin^p) \tag{8}$$

$$Smax^c = max(Smax^a, Smax^p) \tag{9}$$

$$\overline{d_{\hat{i}}^c} = \overline{d_{\hat{i}}^a} \leq \overline{d_{\hat{i}}^p} \tag{10}$$

If all of the performance indices of the alternate channel are less than those of the primary channel, no admission test need be applied, as sufficient resources for the alternate channel have already been reserved. As the resources reserved by Test 3 below ensure that the primary channel packets as well as the alternate channel packets meet their obligations, they may be in excess of those needed for the alternate channel. These excess resources are only present during the transitional period, and will be recovered by the network upon the tear down of the primary channel.

*Test 3: For an alternate channel request with the traffic specification ($Xmin^a, Xave^a, I^a, Smax^a$), at any switch, and a local delay bound requirement of $\overline{d^{a,l}}$, and a primary channel with specification ( $Xmin^p, Xave^p, I^p, Smax^p$ ) and a delay bound $\overline{d^{p,m}}$, first the resources are adjusted to "virtually" remove the primary channel and then Test 1 is applied.*

*Adjustment:*

$$R_{ba,m'} = \sum_{q=1}^{m'} \sum_{j \in C_q} \lceil \frac{\overline{d^{m'}}}{Xmin_j^q} \rceil Smax_j^q + \overline{Smax} - \lceil \frac{\overline{d^{p,m'}}}{Xmin^p} \rceil Smax^p \quad m' = m, \cdots, n. \tag{11}$$

*If the condition given below can be met:*

$$R_{ba,m'} + \lceil \frac{\overline{d^{c,m'}}}{Xmin^c} \rceil \times Smax^c \leq \overline{d^{c,m'}} l \quad m' = l, \cdots, n \tag{12}$$

*then the alternate channel can be accepted.*

The buffer resource test modification is of the same form as that of the bandwidth and scheduling test above, but an adjustment must be made to ensure that there is no duplication of previously reserved resources. Again, the reserved resources are adjusted, and the test condition applied to the adjusted resources.

*Test 4: For an alternate channel request with the traffic specification ($Xmin^a, Xave^a, I^a, Smax^a$), at the $i$ th switch, and a delay bound requirement $\overline{d^a}$, and a primary channel with a traffic specification ($Xmin^p, Xave^p, I^p, Smax^p$) and a delay requirement $\overline{d^p}$, the adjustment is:*

[10] The subscript $\hat{i}$ indicates the local delay bound at the common link $i$.

$$R_{bu_{adj}} = R_{bu} - \lceil \frac{\overline{d^p_{i-1}}}{Xmin^p} \rceil Smax^p - \lceil \frac{\overline{d^p_i}}{Xmin^p} \rceil Smax^p \quad (13)$$

*The condition that needs to be satisfied is :*

$$R_{bu_{adj}} + \lceil \frac{max(\overline{d^p_{i-1}}, \overline{d^a_{i-1}})}{Xmin^c} \rceil Smax^c + \lceil \frac{\overline{d^p_i}}{Xmin^c} \rceil Smax^c \leq B \quad (14)$$

*where $R_{bu}$ is the current buffer space in use (in bits),* B *is the maximum buffer size (in bits) allocated to that output link, and $\overline{d^p_{i-1}}$ is the delay bound in the $i-1$ th switch along the route*[11].

Note also that this modified test is only performed if the performance indices of the alternate channel are greater than those of the primary channel. In the case where the performance requirements are less restrictive, no resources are released during the establishment phase; rather, the excess resources are reclaimed during the tear down of the primary channel.

### 5.2.2 The DCM Transition Algorithm

The DCM transition algorithm ensures that the transition from the primary to the alternate channel does not violate the DCM modification contract. It is invoked for a connection with a No-violation modification contract when there is a possibility that packets on the alternate route may arrive at the destination before packets on the primary route. In this case, transition buffers need to be reserved at the destination and the re-sequencing of packets needs to performed.

Packets along the alternate route may arrive at the destination before packets along the primary route only when the following condition holds:

$$\overline{D^a} + Xmin^a - \overline{J^a_{path}} \leq \overline{D^p} < \overline{D^a} + Xmin^a \quad (15)$$

In this case, the buffers needed are

$$\lceil \frac{\overline{D^p} - \overline{D^a} - Xmin^a + \overline{J^a_{path}}}{Xmin^a} \rceil Smax^a \quad (16)$$

During the transition, packets arriving earlier from the alternate route will be held in these buffers until all packets from the primary route have arrived and been passed to the receiver.

In the case of a connection requesting a delay-jitter bound, the channel administration algorithm makes the delay bound at the last switch equal to the delay jitter bound. If $\overline{J}^a \neq \overline{J}^p$, contract violations can be avoided by maintaining a delay jitter equal to min($\overline{J}^p$,$\overline{J}^a$). If (15) holds, the transition buffers, specified above, will be used to ensure that delay-jitter performance guarantees are not violated during the transition. All out-of-sequence packets along the alternate route will be buffered at the destination and passed up to the client at the appropriate time. In the previously discussed delay bound case, upon arrival of all packets on the primary route the buffered out-of-sequence packets are all passed up to the client immediately. However, to preserve the exact traffic pattern in the delay jitter bound case, the out-of-sequence packets will be passed up to the client at the appropriate times. This appropriate time, $t_k$, for packet $k$ is $t_k = src_k + \overline{D^a} - \overline{J^a}$, where $src_k$ is the source time stamp on the packet, and $\overline{D^a}$ and $\overline{J^a}$ are the delay bounds and delay jitter bounds on the alternate channel. As mentioned previously, if the No-violation contract is desired, the tight coupling of delay bounds and delay-jitter bounds in our scheme necessitates that the conditions specified by equation (5) be met to ensure that there are no delay jitter performance violations.

### 5.2.3 The DCM Routing Algorithm.

The DCM routing algorithm is designed to find an shortest path route based on the constraints imposed by the traffic characteristics, the performance and administrative requirements, and the source/destination host pair. A shortest path route is one that minimizes total cost which is defined below. This routing algorithm taken by itself would provide a significant contribution to the Tenet scheme as the scheme currently uses internet routing which considers neither the real-time network

[11] It should be noted that $\overline{d^p_{i-1}}$ actually refers to the delay in the switch on the primary route preceding the $i$th switch along the alternate route. This preceding switch along primary route may not be $i-1$th switch on that route but for the same of notational brevity we allow this exception.

load nor the traffic and performance parameters of the channel. The DCM channel administration algorithm obtains from the DCM routing algorithm a route for the specified source/destination host pair that obeys the specified routing constraints. In requesting this route, the values of various traffic, performance, and administrative parameters are required by the routing algorithm. The traffic and performance parameters pertaining to the alternate and primary channel have been previously described. The administrative parameter is used to facilitate the presence or absence of resource sharing. This parameter can take three values: if the parameter value is 0, the routing algorithm assumes that a primary route is needed and obviously no resource sharing occurs; if the parameter value is 1, the routing algorithm determines an alternate route that does not share resources with the primary route; if the parameter value is 2, an alternate route that can share resources with the primary route is determined. The manner in which these administrative requirements are satisfied is explained below.

The DCM Routing Algorithm provides source routing and is achieved by using a modified, constrained, version of the Bellman-Ford algorithm[12]. The goals of the routing algorithm were to maximize throughput, to obtain routes in a timely manner, and to maximize the probability that the route provided will be successfully established (i.e. the route will be established with the traffic and performance specifications given by the client). The routing algorithm calculates a minimal-cost route subject to a delay constraint. The cost of the route is the number of links comprising the route while the delay constraint ensures that the sum of the delay values of the links is less that the delay bound, $\overline{D}$, required by the client. The delay value attributed to a link is the sum of the minimum queuing delay offered by the node to a real-time channel with these traffic characteristics and the propagation delay along the output link. While the propagation delay is fixed, the queuing delay experienced in the RCSP scheduler is variable, and is dependent on the current channel resource reservations on the corresponding output link and the traffic characteristics of this new channel. This queueing delay is calculated by using the admissions tests provided in Section 4.2 and Section 5.2.1 to determine the minimum queueing delay that this link can offer a connection with these traffic characteristics.

The algorithm proceeds with the following steps:

1. A directed graph is created in which the nodes correspond to switches and hosts in the network and the edges to the links connecting these switches and hosts. The weights attributed to the edges represent the link delay values. These delay values are computed just prior to applying the algorithm, thereby using the most recent link information obtained from routing update messages.

2. A constrained modified Bellman-Ford algorithm is then applied to this graph to determine a possible route.

   (a) For the delay bound case the algorithm proceeds as follows:

      i. Consecutive searches are performed on all 1, 2, .., N-2-hops paths from the source to the destination node, where N is the number of nodes in the network, until the delay condition $\sum_{l(s,r)} w_l \leq \overline{D}$ is satisfied, where $\overline{D}$ is the delay bound of the channel, $w_l$ is the weight of link *l*, and *(s,r)* is the links connecting the source *s* to the destination *r*. A *constraint* is placed on the number of possible searches by stopping at the *hop level* [13] at which the delay bound condition is satisfied.

      ii. At this *hop level* or cost, the path with the minimum delay (i.e. $\min\sum_{l(s,r)} w_l$ ) that meets the delay condition is chosen.

   (b) For the delay jitter bound $\overline{J}$ case, the following steps are performed:

      i. Assuming a path with n hops, the minimum queuing delay offered by the link of the destination node, $d_n$, is first examined. If $d_n \leq \overline{J}$ , then the algorithm proceeds, else the channel cannot be accepted, as the delay jitter bound condition cannot be satisfied.

      ii. Consecutive searches are performed on all 1, 2, .., N-2-hops paths from the source to the destination node until $\sum_{l=1}^{n-1} w_l \leq \overline{D} - d_n - \pi_n$, where $\overline{D}$ is the delay bound of the channel, $\pi_n$ is the propagation delay associated with the last link *n*.

      iii. At this hop level, the path with the minimum delay that meets the delay condition is chosen.

This algorithm limits its search space thereby reducing its computation time. Currently, as we consider bandwidth to be our premium resource, the algorithm seeks to reduce consumption of this resource by selecting a path with the minimum number of hops so as to maximize network throughput. It also increases the probability that channel establishment will be

[12] In a network with N nodes, the fundamental Bellman-Ford algorithm [Bel58] searches for the shortest paths between a specified source and destination node starting from all possible one-hop paths and continuing until the N-2-hop paths are examined.

[13] This *hop level* is the number of hops from the source to the destination node and is also the cost of the route.

successful as it determines the queuing delays based on the traffic characteristics of the channel and the most recent real-time network load information.

The administrative constraints are achieved by modifying the weights associated with the edges (i.e. links) of the graph before applying the algorithm. With an administrative parameter value of 0 and 1, no adjustment is made to the edges of the graphs. Thus in the presence of an primary channel, i.e. when the parameter is 1, resources used by the primary channel are not considered. With a value of 2, the primary channel's resources are virtually removed from the edges corresponding to the links comprising the primary route before calculating the weight of the edge.

Routing updates are currently done on a per-channel-establishment basis. Updates are accomplished by having every node broadcast the load values of its links to all other nodes. This broadcast is done upon the establishment of a new channel, after the node has sent the reverse channel establishment message to the previous node on the new channel's route ,and follows every channel tear down. Upon receiving an update packet, the receiving node updates its local link-state table. If there are no new channel establishments within a specified time interval, link updates are sent by each node to assure other nodes that the link is still active. All route update broadcasts are done along a minimum spanning tree.

### 5.3 Fast Establishment and Granularity of Control in DCM

In this section we describe the method used to reduce the channel establishment and modification time, as well as, the granularity of control possible under DCM.

The response time of the DCM scheme is the time interval between a client's establishment or modification request and the response of the network. This interval is dependent on the client's request, the current network load, and the establishment procedure, and is commonly referred to as the establishment time[14]. With establishment we have the following two cases: if the client requests a modification that does not increase the level of any of the performance parameters, then the response is **acceptance**, and the client can immediately modify its traffic characteristics to reflect this response; and if the client requests a modification that increases the level of at least one of the performance parameters, then the response can be an **acceptance** or a **rejection** of the request. If the response is an acceptance, then at the instant the response is passed to the client, the excess resources are available and the client can now begin to use these resources.

The establishment time should be as short as possible for client satisfaction. This time is dependent on the accuracy of the routing algorithm (i.e. the ability of the algorithm to provide a route that will permit the establishment of the channel requested by this client) and the number of retires that are permitted under the current management policy. In DCM we attempt to reduce this establishment time by (1) having the routing algorithm utilize the most recent real-time network state information as well as the traffic and performance requirements of the client[15] to determine an available route before establishing the channel and, (2) exploiting the *time value* of the network state information so as to navigate unavailable links during channel establishment.

In order to guarantee that the route selected by the routing algorithm will provide the greatest probability of success, the routing algorithm uses the most recent network state information in its database and the traffic and performance requirements of the channel in computing a route. The accuracy of this network state information is based entirely on the routing update mechanism. In our model, route update information is disseminated mainly upon the establishment and tear down of a channel. Currently, these route updates are sent via a minimum spanning tree connecting all nodes in the network [16].

The value of network state information decreases in time if it is not updated. At its inception, an update packet contains the exact state of the node[17] from which it was generated; as it is sent to nodes further away from this source node, the network state changes and the information provided in the packet is a less accurate description of the state of the source node. Beyond a certain amount of time the information contained in the routing update packet may be entirely inaccurate. As a channel establishment message moves from the source node towards the destination node, it encounters nodes that have received more recent network state information from the destination node. Thus, the time value of the information pertaining to the route along which the establishment message is traveling is increasing. This indicates that the nodes have more precise information on the state of the network along that route as the establishment message moves towards the destination. If the establishment message encounters a link which has insufficient resources to accommodate the request, it returns to the node preceding this unavailable link and requests a new route from this node to the destination. This

[14] In this paper, the interval is referred to as the *primary establishment time*, in the case of the response time of an initial request, and as the *alternate establishment time* in the case of the response time of the alternate channel.

[15] The current Tenet scheme utilizes internet routing which does not directly utilize any network or client real-time information.

[16] While a sufficiently large network will suffer from inconsistent views of the network due to the large delays experienced in the broadcast, for small networks this mechanism is adequate. Other mechanisms suitable for large networks are being examined.

[17] The nodes contain information on the state of the links they support.

node, which has more recent information than the source of the establishment package, computes the route from itself to the destination. This computation takes into consideration the resources that were reserved before the unavailable link, and the traffic and performance parameters contained in the establishment message. Each node that attempts to route the establishment packet because of an unavailable link knows the previously reserved portion of the route and removes these links (with the reserved resources) from the graph, before applying the routing algorithm, thereby preventing looping in the path. This addition to the establishment procedure should reduce the establishment time of a channel, and can be useful also in the case of link failures, where the establishment message is immediately rerouted to avoid the failed link.

The DCM algorithms can be applied to modify the performance parameters or the route of an entire channel. It can also modify a segment of the channel. The smallest segment of a channel that can be modified is a single link. Control can be applied at the link (or *local*) level or at the route (or *global* ) level. We make no distinction between a modification affecting a single link and a modification affecting any subset of a channel's links (excluding the entire route); both of these are called *local* modifications. The modification of an entire channel is referred to as a *global* modification. The DCM modification contracts are equally applicable to channel segments.

There are advantages and disadvantages to local and global control. Local control has the advantages that the routes obtained by the routing algorithm usually have a higher success rate than those of global routes, as the routing information gathered within a small radius is normally more accurate, and the channel establishment time is small due to the short distances traveled. However, local routing algorithms do not possess the global knowledge that the source has and may not be aware of routing constraints (these constraints may be cost or security restrictions) and can cause localized saturation as only a small subset of links are considered.

### 5.4 Transparency Procedures in DCM

The transparency procedure must ensure that the interface to the client is preserved during the transition from a primary route to an alternate route. This transition must be "invisible" to the client as far as this interface is concerned. The interface is usually in the form of a unique identifier that a client uses when transmitting on the channel. This unique identifier must be preserved at the source and a similar identifier must be preserved at the destination. After the transition, the unique identifier will refer to the alternate channel, and the client's packets will be sent along this alternate route, while at the destination the receiver will continue to receive its packets from its usual receiver abstraction.

While preserving this unique identifier, the client's real-time packets must be a switched from the primary to the alternate path at the appropriate time. The transfer to the alternate path, while maintaining the same unique source and destination channel identifiers, is done during the alternate channel's establishment. On the forward pass, each intermediate node creates an instance of a channel structure which stores the state of the channel in that node, and an entry into a virtual circuit routing table that indicates the outgoing virtual circuit identifier and the outgoing link. Along the alternate route, the node preceding the destination configures its virtual circuit routing table so that it points to the same entry as the node preceding the destination in the primary path. In this manner both the primary and alternate channels point to the same destination virtual circuit entry. Note that this action is taken regardless of the conditions of the routes (i.e., whether the alternate and primary routes are completely disjoint, partially disjoint , or identical), as new channel structures and new table entries are required in all conditions. On the reverse pass the switching between the primary and alternate channel takes place when the virtual circuit table entry corresponding to the source is changed to point to the alternate route. This change is accomplished by modifying the outgoing virtual circuit identifier and the outgoing link identifier at the source node. The previous outgoing virtual circuit and outgoing link identifiers are maintained and used to tear down the primary channel. A more thorough description of these implementation details can be found in [PZF92].

## 6 Simulation Experiments

In this section we provide our experimental design and then present a small subset of the simulation experiments that were performed using the DCM mechanism. We present three experiments which exercise the algorithms to determine their correctness and performance . These experiments also depicted useful applications that can be supported by DCM. These experiments sought to exercise the major components of DCM scheme. In the first experiment, parameter and route modifications are conducted under a "no-violation" modification contract, with the application being that of a browser. In the second experiment, parameter and route modifications are conducted under a "bounded-violation" modification contract. This experiment illustrated a client's application tuning its parameters to improve its perceived quality. The third experiment is a route modification experiment with a focus on exercising local control under a "no-violation" modification contract. This experiment illustrates the ability of DCM to provide guaranteed performance support in a mobile environment.

The performance indices of interest in the experiments are the throughput[18] of the channels, the maximum delays experienced by the packets as they traverse the route, the numbers of out-of-sequence packets received by the destination hosts, and the alternate channel establishment times. The DCM algorithms should ensure the following (subject to the modification contract): that the delay bounds of all packets on the primary and alternate channels are met; that the throughput of the alternate channel correctly reflects the modification (if a bandwidth modification has been made); and that there are no out-of-sequence packets, or in the case of a bounded-violation contract, the number of out-of-sequence packets should not exceed the violation bound.

In the simulation experiments, the delays of all packets on the primary and alternate channel were recorded. Throughputs were verified by examining the number of *packets per second* received at the destination host over the channel's lifetime. All simulation data was taken with a millisecond granularity, and the simulations were run for $7.2 \times 10^6$ milliseconds (i.e., two hours of simulated time ). In the simulator, packet arrivals were recorded and used to calculate the average throughput rate at each second of simulated time. This value is displayed in the throughput graphs given below. In this paper, the most important measurement to be derived from the delay histograms is the maximum delay experienced by the packets during transit from the source to the destination host. These maximum values were recorded and displayed in tabular form. Out-of-sequence packets were recorded by an out-of-sequence counter associated with each channel.

The topology of the simulated network and its configuration are given in Fig. 2. In this simple network there are 11 nodes (i.e.,switches or gateways) connected by 16 links. Attached to each of the nodes is at least one host which is connected by a local link with a constant delay of 2.5 ms. In Fig. 2, only three of the twelve hosts are shown, host 0, host 3, and the mobile host, host m. All of the links have the same maximum speed of 10 Mbps.

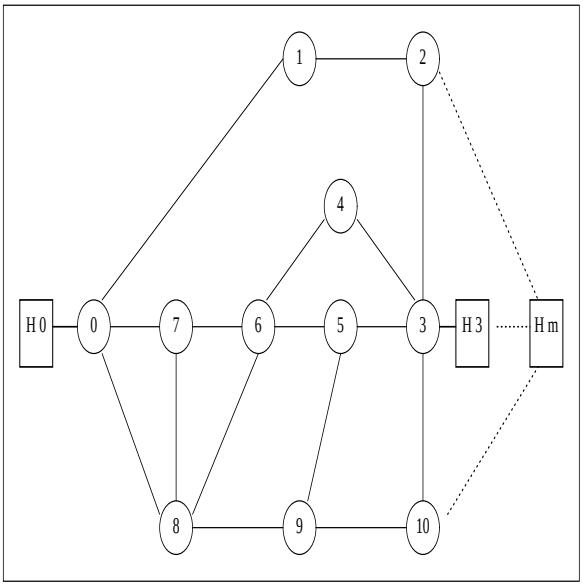



Figure 2: Experimental Network

In Fig. 2 the edges connecting the nodes represent pairs of simplex links with opposite directions. The propagation delays of most of them are 10 ms. There are six links that have propagation delays of 20 ms : link(0,1)[19], link(1,0), link(1,2), link(2,1), link(8,9), link(9,8). In all of the experiments there was at least one real-time channel originating from each host, all links had real-time traffic present, and the average bandwidth reserved in the network was 53%, with some links having 80% of their bandwidth reserved. During the course of the modifications new channels were being established and existing channels were terminated.

The establishment times of the connections (primary and alternate) are important in assessing the viability of DCM. The establishment time is the sum of the communication delays (i.e., propagation and queueing), and the processing times of the establishment message (not including the admissions test), the routing algorithm (at the source node only), and the admission control tests. Since the communications delays and, to some extent, the message processing times are beyond

[18] For simplicity, in the simulations we assumed that all of the bits sent in a packet were useful.

[19] A link is characterized by the pair *(s, d)* where *s* refers to the source gateway and *d* refers to the destination gateway.

our control, we shall focus on the processing times of the admissions tests and of the routing algorithm.

The admission control test is executed on a per link basis for each link traversed by the establishment message along the path. As can be determined from the admissions control tests (Sections 4.2 and 5.2.1), for the worst case, the order of growth for this algorithm is $O(LM)$, where *L* is the number of links in the network and *M* is the number of priority levels in the RCSP queue.

Under worst-case conditions, the routing algorithm must be executable at each node along the path. The algorithm first forms the directed graph, and then runs the constrained-modified Bellman Ford algorithm (described in Section 5.2.3). Thus, in the worst case, the order of growth for this algorithm is $O(N^2L + MNL)$, where *N* is the number of nodes in the network, and *L* and *M* are as described previously.

Using the dimensions of the experimental network (there were ten priority levels in the RCSP queues) and running the algorithms on a DECstation 5000/240, the admissions control tests took 3.3 microseconds at each node. In the worst-case scenario the routing algorithm took 2.4 ms on the same machine. It should be noted that the run time order of the routing algorithm assumes worst-case conditions (i.e., in which the network is very heavily loaded, and at each node encountered by the establishment message the initially chosen link was unavailable).

### 6.1 Experiment I

In the first experiment, Experiment I, the application considered is that of a *still image lossless browser*, which is used to browse a sequence of large still images. This type of application is used, for example, by environmental scientists who examine large satellite maps to determine minute changes over a period of time. Lossy compression techniques cannot be used on these images as they may remove these minute changes and low frames speeds are used in normal playback operation. As these images cannot tolerate lossy compressions browsing is best achieved by increasing the bandwidth of the channel by any factor that permits useful work by the user.

The traffic and performance parameters of the different states of the connection are provided in Table 2(a) below. In this experiment the source of the connection is host 0, the destination is host 3, and the initial route chosen traverses the switches 0, 7, 6, 5, and 3. Initially, insufficient bandwidth was requested for the channel, and the traffic parameters had to be tuned to obtain adequate bandwidth. To this end, an initial request was made for a throughput performance increase of 16%. After tuning the parameters of the channel, channel modifications were done to reflect the applications needs for fast browsing and playback. The parameters associated with each of these states, fast browsing and playback, are also provided in Table 2(a). The bandwidth of the channel is doubled for the duration of time that the client needs the higher speed (this is usually short compared to the channel's lifetime), after which the application requests that the bandwidth be reduced to its original value. In this experiment the start of the fast browsing period and its subsequent duration were statistically derived. The start of the browsing periods were randomly chosen from an exponential distribution with mean 10 minutes, and the durations randomly chosen from a uniform distribution with an upper bound of 15 minutes and a lower bound of 5 minutes. The networks real-time background load was dynamic in that real-time channels were being created and terminated during the entire experiment thus encouraging route changes to accommodate alternate channels. These route changes exercised the routing algorithms under the resource sharing constraints. The results of this experiment are displayed in Table 2(b) and Fig. 3. In the experiment there were nine channel modifications (i.e. one channel tuning and four browsing periods) with three route changes. The average (round trip) and maximum modification times, 101.3 and 122.6 ms respectively, reflect the changes in routes due to modifications. The average and maximum packet delays indicate these changes as well and also verify that no packet exceeded its delay bound of 80 ms. No out-of-sequence packets were observed. This experiment verified the schemes in that modifications were accomplished within the constraint of the modification contracts.

It should be noted that the maximum alternate channel modification time along this route 122.6 ms, of which 110 ms was propagation delay, is less than the values associated with electro-mechanical switches such as those found on VCRs (the time between activating the switch and the response is usually in the range 0.5-0.8 sec). The maximum alternate modification time is dominated by the propagation delay which accounts for 89.7% of the time.

This experiment also illustrates the ability provided by DCM to balance the network load or adapt to the error thresholds requirements of connections by modifying the routes of the connections under a "no-violation" contract. These modification would then be done in a manner that would be transparent to the client.

### 6.2 Experiment II

The delay experienced by a real-time channel between hosts 0 and 3 was excessive, and a channel modification request was made to the network to reduce the delay from 160 ms to 90 ms to 60 ms. The traffic and performance characteristics of

| Parameters | Initial | Playback | Fast Browse |
|---|---|---|---|
| Xmin *(ms)* | 18.7 | 15.6 | 7.8 |
| Xave *(ms)* | 18.7 | 15.6 | 7.8 |
| I *(ms)* | 1000 | 1000 | 1000 |
| Smax *(bytes)* | 1024 | 1024 | 1024 |
| $\overline{D}$ *(ms)* | 80 | 80 | 80 |
| Bandwidth *(Mbps)* | 0.42 | 0.51 | 1.0 |

(a)
Traffic and Performance Parameters

| Metrics | Values |
|---|---|
| Number of Modifications | 9 |
| Number of Route Changes | 3 |
| Maximum Setup Time *(ms)* | 122.6 |
| Average Setup Time *(ms)* | 101.3 |
| Maximum Packet Delay *(ms)* | 76.0 |
| Average Packet Delay *(ms)* | 71.5 |

(b)
Results

Table 2: Experiment I - Parameters and Results.

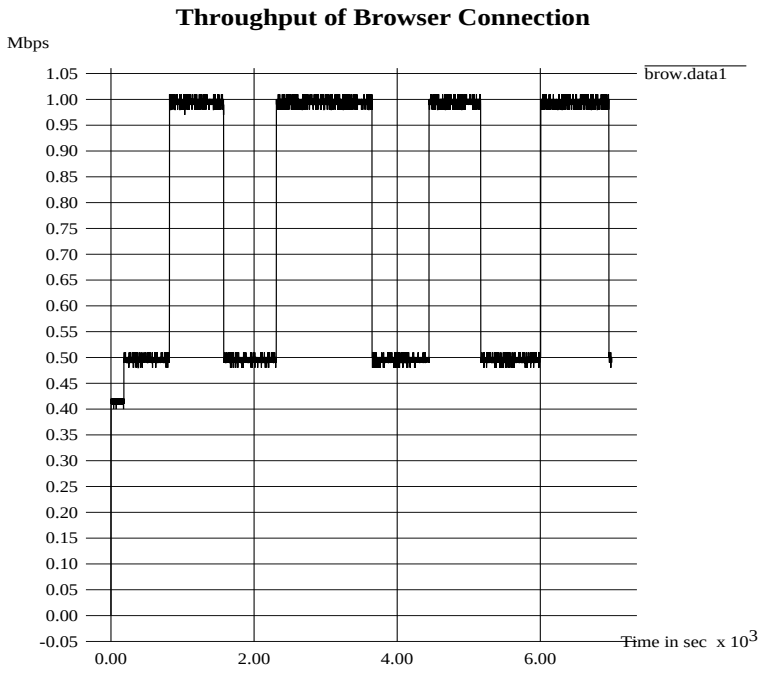



Figure 3: Experiment I - Browser

the original and adjusted channels are given in Table 3(a).

The route change in this experiment ensured that there were out-of-sequence packets. As the delay bound conditions given in eqn (5) cannot be met, a "bounded-violation" modification contract is used for the modification. Table 3(b) provides the packet violation bounds and the actual number of packets that violated their performance bounds. The establishment and modification times were 97.5 ms, 121.4 ms and 121.1 ms, respectively. These delays are short given that the filtering capability of human vision is incapable of distinguishing any pauses between frames 33 to 70 ms apart. Hence the visual response seems reasonable especially when the propagation delay is on average is 103.3 ms. In LAN and MAN environments the delays should be well within the distinguishing range mentioned above.

In Table 3(b), $\overline{D^a}$ is the modified delay bound requested by the client, $\overline{D^p}$ is the delay bound along the primary route, *Bound* ($\lceil \frac{\overline{D^p} - \overline{D^a} - Xmin^a}{Xmin^a} \rceil$) is the offered performance violation bound, and *Packet Count* is the number of packets whose performance contracts were actually violated. As can be seen in Table 3(b), the DCM performance-violation guarantees were met.

| **Parameters** | **Initial** | **Adjustment 1** | **Adjustment 2** |
|---|---|---|---|
| Xmin *(ms)* | 15.6 | 15.6 | 15.6 |
| Xave *(ms)* | 15.6 | 15.6 | 15.6 |
| I *(ms)* | 1000 | 1000 | 1000 |
| Smax *(bytes)* | 1024 | 1024 | 1024 |
| $\overline{D}$ *(ms)* | 160 | 90 | 60 |
| Route | 0,7,6,5,3 | 0,8,9,10,3 | 0,8,9,10,3 |

(a)
Traffic and Performance Parameters

| $\overline{D_p}$ | $\overline{D_a}$ | Bound | Packet Count |
|---|---|---|---|
| 160 | 90 | 4 | 3 |
| 90 | 60 | 1 | 1 |

(b)
Results

Table 3: Experiment II - Parameters and Results.

### 6.3 Experiment III

Experiment III depicts a mobile application where the nodes at the periphery of the network are considered as base stations. The Tenet and DCM frameworks are capable of supporting heterogeneous network topologies comprised of both wired and wireless links [Fer92]. In this experiment a channel exists between host 0 and host m, where host m is the mobile host. Host m follows a path in a downward direction, and therefore connects to node 2, 3, and 10. A handoff takes place between these nodes to permit the real-time route to exists at all times between host 0 and host m. In this scenario we assume that the mobile host has intelligence and is the actual endpoint of a real-time channel. Thus, a handoff becomes a rerouting of the channel so that, in the new route, the node preceding host m is the new basestation which receives the handoff. For the purpose of this experiment, these reroutings will be under local control. It should be noted that reroutings could also be global, but to reduce establishment time a local rerouting is preferred if it can be accommodated by the channel's performance requirements and the available network resources. A combination of local and global reroutings can also be used to increase the availability of the network for these reroutings. With this method, a local rerouting is first accomplished to ensure a quick handoff; then, after the handoff has been achieved, the channel can again be rerouted to find a more efficient route through the network. The first rerouting is not likely to be optimal, as only local information is used. The channel's traffic and performance parameters are given Table 4(a).

The set of routes that are needed to satisfy the mobility requirements of the channel are 0,1,2, 0,1,2,3 and 0,1,2,3,10. In this experiment the duration of time that the mobile host was connected to a base-station (i.e. nodes 2, 3, or 10) was randomly chosen from a uniform distribution with a lower bound of 5 seconds and an upper bound of 20 minutes. The results of this experiment is shown in the Table 4(b). The modification times reflect local control times as they indicate the times needed to add links (2,3) and (3,10). It should be noted that in both cases 25.0 ms of these delays represent propagation delays hence the modification times are acceptable for support even in small cells (i.e. with a radius that can be spanned in 5 seconds).

Fig. 4(a) illustrates the route taken by the connection in handing off to each of the base stations 2, 3, and 10. Fig. 4(b) shows that the throughput of the channel is unaffected by the rerouting. As shown in Table 4(b) all packets met their delay bounds. There were no out-of-sequence packets recorded at host m.

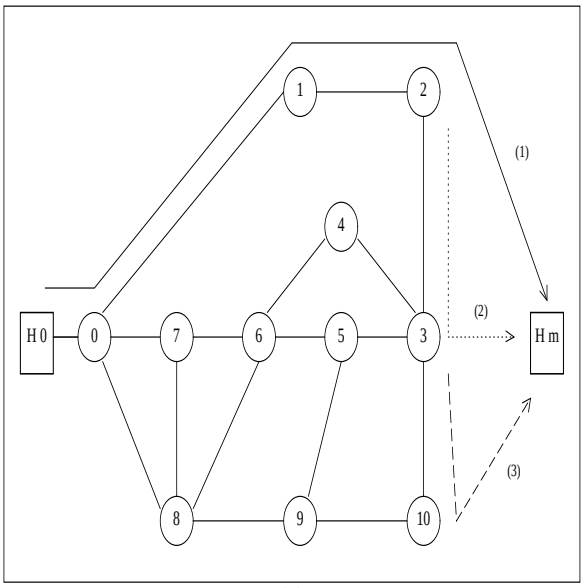
| Parameters | Values |
|---|---|
| Xmin *(ms)* | 15.6 |
| Xave *(ms)* | 15.6 |
| I *(ms)* | 1000 |
| Smax *(bytes)* | 1024 |
| $\overline{D}$ *(ms)* | 80 |

(a)
Traffic and Performance Parameters

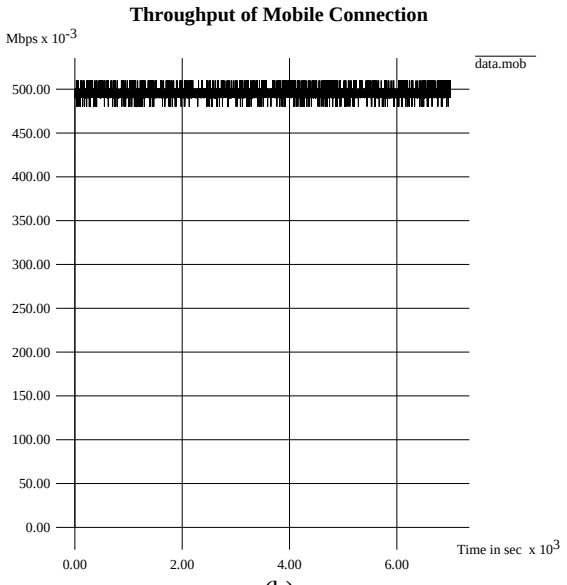
| Metrics | Values |
|---|---|
| Number of Modifications | 14 |
| Number of Route Changes | 14 |
| Establishment Time *(ms)* | 98.1 |
| Average Modification Time *(ms)* | 30.5 |
| Maximum Modification Time *(ms)* | 32.0 |
| Maximum Packet Delay *(ms)* | 70.0 |
| Average Packet Delay *(ms)* | 57.9 |

(b)
Results

Table 4: Experiment III - Parameters and Results.




(a)




(b)

Figure 4: Experiment III - Mobile Hosts.

## 7 Conclusion

In this paper we presented the Dynamic Connection Management (DCM) scheme, which permits the modification of traffic and performance parameters, and routes of real-time connections. We motivated this work by explaining the benefits that DCM would provide in a variety of examples. These examples were divided into three categories: the dynamics of client requirements, the dynamics of the network state, and the tradeoffs between QoS and network availability. The DCM scheme is based on three algorithms: the DCM administrative algorithm, the DCM transition algorithm, and the DCM routing algorithm; and is subject to the DCM modification contract. This contract specifies the degree of disruption that a client may experience during a modification. This degree can range from no performance violations to a bounded number of performance violations. The administrative algorithm reserves the network resources, in the presence or absence of resource sharing, needed to support the transition from the original (i.e. *primary*) channel to the new (i.e. *alternate*) channel and to ensure that the performance guarantees of the alternate channel are satisfied. The DCM transition algorithm ensures that the performance violations specified in the DCM modification contract are adhered to during the transition. The DCM routing algorithm determines a route from the source to the destination host according to the traffic and performance requirements and the resource sharing factor. The DCM scheme also supports mechanisms that enable modifications to a connection to be made to a segment of the connection (*local control*) or to the entire connection (*global control*). Faster establishment and modification is also possible as the DCM scheme utilizes real-time network and client state to compute the path before establishment and utilizes the time value of the network state information to navigate unavailable links during establishment. Simulations experiments were conducted to determine the validity and usefulness of the scheme. Our experiments showed that the scheme was indeed valid in that all traffic, performance and route modifications were realized within the constraints of the modification contracts and, under our workload and topological conditions, establishment and modifications times were short and dominated by the propagation delay of the routes. At present, the scheme is being expanded for use in an internetworking environment, and is being implemented on the Sequoia 2000 network testbed.

## References


[AHS90] David P. Anderson, Ralf Guido Herrtwich, and Carl Schaefer. SRP: A resource reservation protocol for guaranteed performance communication in internet. Technical Report TR-90-006, International Computer Science Institute, Berkeley, California, February 1990.

[Bel58] R. Bellman. On a routing problem. *Quarterly of Applied Mathematics*, pages 87–90, 1958.

[CESZ91] R. Cochi, D. Estrin, S. Shenker, and L. Zhang. A study of priority pricing in multiple service class networks. In *Proceeding of the SIGCOMM'91*, September 1991.

[CGG91] Israel Cidon, Inder Gopal, and Roch Guerin. Bandwidth management and congestion control in PlaNET. *IEEE Communications Magazine*, pages 54–64, October 1991.

[CSZ92] D. Clark, S. Shenker, and L. Zhang. Supporting real-time applications in an integrated services packet network: Architecture and mechanism. In *Proceeding of the SIGCOMM'92*, pages 14–26, August 1992.

[DB89] D.C. Darragh and R.L. Baker. Fixed distortion subband coding of images for packet-switched networks. *IEEE Journal on Selected Areas in Communications*, 7(5):826–832, June 1989.

[DHH+93] L. Delgrossi, C. Halstrick, D. Hehmann, R.G. Herrtwich, O. Krone, J. Sandvoss, and C. Vogt. Media scaling with heits. In *Proceeding of ACM Multimedia '93*, August 1993.

[FBZ92] Domenico Ferrari, Anindo Banerjea, and Hui Zhang. Network support for multimedia: a discussion of the Tenet approach. Technical Report TR-92-072, International Computer Science Institute, Berkeley, California, October 1992. Also to appear in *Computer Networks and ISDN Systems*.

[Fer90] Domenico Ferrari. Client requirements for real-time communication services. *IEEE Communications Magazine*, 28(11):65–72, November 1990.

[Fer92] Domenico Ferrari. Real-time communication in an internetwork. *Journal of High Speed Networks*, 1(1):79–103, 1992.

[FV90] Domenico Ferrari and Dinesh Verma. A scheme for real-time channel establishment in wide-area networks. *IEEE Journal on Selected Areas in Communications*, 8(3):368–379, April 1990.

[GG91] Michael Gilge and Riccardo Gusella. Motion video coding for packet switching networks – an integrated approach. In *SPIE Visual Communications and Image Processing '91*, November 1991.

[Gha89] M. Ghanbari. Two-layer coding of video signals for VBR networks. *IEEE Journal on Selected Areas in Communications*, 7(5):771–781, June 1989.

[KV88] G. Karlsson and M. Vetterli. Subband coding of video for packet networks. *Optical Engineering*, 27(7):574–586, July 1988.

[LP91] A. Lazar and G. Pacifici. Control of resources in broadband networks with quality of service guarantees. *IEEE Communications Magazine*, 1991.

[PKF92] C. Parris, S. Keshav, and D. Ferrari. A framework for the study of pricing in integrated networks. Technical Report TR-92-016, International Computer Science Institute, Berkeley, California, March 1992.

[PT89] Guru M. Parulkar and J. S. Turner. Towards a framework for high speed communications in a heterogeneous networking environment. In *Proceedings of INFOCOM'89*, pages 655–668, Ottawa, Canada, April 1989.

[PVZ93] Colin Parris, Giorgio Ventre, and Hui Zhang. Graceful adaptation of guaranteed performance service connections. In *Proceedings of IEEE GLOBECOM'93*, Houston, TX, November 1993.

[PZF92] Colin Paris, Hui Zhang, and Domenico Ferrari. A mechanism for dynamic re-route of real-time channels. Technical Report TR-92-053, International Computer Science Institute, Berkeley, California, April 1992.

[SCB92] Samuel Sheng, Anantha Chandrakasan, and Robert W. Brodersen. A portable multimedia terminal. *IEEE Communications Magazine*, pages 64–75, December 1992.

[Sto92] Michael Stonebraker. An overview of the Sequoia 2000 project. In *Proceedings of COMPCOM 92*, San Francisco, CA, February 1992.

[Top90] Claudio Topolcic. Experimental internet stream protocol, version 2 (ST-II), October 1990. RFC 1190.

[TTCM92] Y. Tobe, H. Tokuda, S.T.C. Chou, and J.M.F. Moura. QoS control in ARTS/FDDI continuous media communications. In *Proceeding of the SIGCOMM '92*, pages 88–98, August 1992.

[TYT91] F. Teraoka, Y. Yokote, and M. Tokoro. A network architecture providing host migration transparency. In *Proceedings of the SIGCOMM'92*, pages 209–220, 1991.

[VHN92] C. Vogt, R. Herrtwich, and R. Nagarajan. HeiRAT- The Heidelberg Resource and Admistration Technique: Design Philosophy and Goals. Technical Report No. 43.9213, IBM Technical Report, IBM ENC, 1992.

[VZF91] Dinesh Verma, Hui Zhang, and Domenico Ferrari. Guaranteeing delay jitter bounds in packet switching networks. In *Proceedings of Tricomm'91*, pages 35–46, Chapel Hill, North Carolina, April 1991.

[YH91] Nanying Yin and Michael G. Hluchyi. A dynamic rate control mechanism for integrated networks. In *Proceedings of INFOCOM'91*, 1991.

[ZF93] Hui Zhang and Domenico Ferrari. Rate-controlled static priority queueing. In *Proceedings of IEEE INFOCOM'93*, pages 227–236, San Francisco, California, April 1993.

[Zha89] Lixia Zhang. *A New Architecture for Packet Switched Network Protocols*. PhD dissertation, Massachusetts Institute of Technology, July 1989.

[Zha93] Hui Zhang. *Service Disciplines for Integrated Services Packet-Switching Networks*. PhD dissertation, University of California at Berkeley, November 1993.

[ZK91] Hui Zhang and Srinivasan Keshav. Comparison of rate-based service disciplines. In *Proceedings of ACM SIGCOMM'91*, pages 113–122, Zurich, Switzerland, September 1991.