A Dynamic Management Scheme for Real-Time Connections*

Colin Parris, Hui Zhang and Domenico Ferrari
The Tenet Group
Computer Science Division, University of California, Berkeley
and International Computer Science Institute

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 dynamicity of the communicating clients; (b) the dynamicity 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 [4], 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 [6, 13, 18, 1, 2, 11, 7, 3].

These solutions are usually connection-oriented, and require fixed routing and resource reservation on a per-connection basis. In 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 intrinsic dynamicities of client requirements and network state, both of which may change during the lifetime of the connection. For example, in a lossless digital image browsing application, where degradation of image quality is not allowed, variation of the browsing speed corresponds to different requirements for network bandwidth. In addition, the static approach 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 guaranteed performance communication, the adaptation should be graceful (i.e., to be done with the consent of the clients) [9].

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. It attempts to address network and client dynamics in a time scale approximately that of the round trip time of a packet traversing the connection and smaller than the lifetime of the connection. The DCM scheme is based on the framework of the guaranteed performance communication service (i.e., the *Tenet* framework) referred to in [6] . In our

*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.

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.

The paper is organized as follows: in Section 2, we examine the related work and discuss other possible solutions; in Section 3, we briefly review the Tenet scheme, of which the DCM scheme is an extension; in Section 4, we describe the DCM scheme in detail; in Section 5, we present results from simulation experiments; finally we conclude by summarizing the paper in Section 6.

2 Related Work and Alternate Approaches

2.1 Related Work

Besides the Tenet approach, which has been implemented in the Tenet Real-Time Protocol Suite [5], there are a number of other proposals to support quality of service in packet-switching networks. These include the Flow Protocol [18], the Heidelberg Resource and Administration Technique (HieRAT) [15], the Session Reservation Protocol [1], the Asynchronous Time Sharing (ATS) approach [7], the Multipoint Congram-oriented High-performance Internet Protocol [11], and the extended Capacity-Based Session Reservation (CBSRP) protocol [12].

Only two of these schemes permit the modification of the parameters of the communication stream. In the HieRAT scheme [15], which uses ST-II as its reservation protocol [13], connections are established by using the ST-II Control Message Protocol (SCMP). The SCMP specifications include an SCMP **change** message, which can be used to modify parameters, however, the HieRAT implementation does not currently support modification of parameters or routes dynamically.

In the extended version of the CBSRP protocol [12], the user can specify the minimum and maximum values for two parameters: the desired temporal and spatial resolutions of the media to be transmitted. Parameters can be changed between the minimum and the maximum during the lifetime of a connection. The protocol does not take into account delay, delay jitter, and packet loss bounds. In addition, the solution is proposed in a local area network environment, while issues associated with a more general internetworking environment are not addressed.

2.2 Possible Alternative Approaches

Our approach to the problem discussed before 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 tears down 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. This affects the availability of the network. Also, mechanisms like rerouting a channel locally, which is used by the DCM algorithm to reduce 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 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 Background

In this section, we give a brief overview of the current version of the Tenet resource management algorithms [5].

3.1 Tenet Service Model

The Tenet algorithms are based on a communication abstraction called a *real-time channel* [6, 14]. 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 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 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.

3.2 Tenet Algorithms

In order to provide performance guarantees, two levels of control 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 the reservations.

As shown in [17], 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 [16] or RCSP. In this section, we first briefly summarize the properties of RCSP discipline, then give the corresponding admission control tests.

A RCSP server consists of a rate-controller and a static priority scheduler with multiple priority levels. 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, non-preemptively, the packet at the head of the non-empty-queue with the highest priority.

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* <u>*largest*</u> *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:*

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

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 [1].

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*,*

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

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:

$$R_{bu} + \lceil \frac{\overline{d}_{i-1,j}}{Xmin} \rceil \times Smax + \lceil \frac{\overline{d}_{i,j}}{Xmin} \rceil \times Smax \leq B$$

where $\overline{d}_{i-1,j}$ *and* $\overline{d}_{i,j}$ *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.*

4 Dynamic Connection Management (DCM)

DCM is comprised of the *DCM scheme* and the *DCM policies*. The DCM scheme is the collection of algorithms

[1] 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 the property [16]. We assume in the paper that delay-jitter controlling RCSP servers are used.

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 parameters and/or the new route. The DCM policies are usually implemented as management applications, and will not be discussed in this paper.

The DCM scheme will be described from two aspects, the DCM modification contract and the DCM algorithms, which will be presented below.

4.1 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 client's request and the current real-time network load. 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[2]. 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 two contractual obligations that can be made to the client:

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* [3] *of performance violations can occur* during the transition from the primary to the alternate channel (The *Bounded-Violation contract*).

[2] 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.

[3] The number of performance violations is specified as a packet count; however, it will be converted into a unit more easily understood by the human client (i.e., frames, messages, etc.).

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 the transition interval. This contract may be explicitly requested by the client before channel parameter modification or implicitly demanded by the network during channel route modification.

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 re-routed connection on 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

$$D_k < D_{k+1} + s_{k+1} - s_k \tag{2}$$

Notice that the traffic has to satisfy the traffic constraint, or $s_{k+1} - s_k > Xmin$. 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 [4]. Since (2) 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,

$$\overline{D^p} < \overline{D^a} - \overline{J^a_{path}} + Xmin^5 \tag{3}$$

Notice that $\overline{J^a_{path}}$ can be reduced by buffering at the destination. By introducing the transition buffer as described in section 4.2.2, $\overline{J^a_{path}}$ can be reduced to 0. So the necessary condition for a No-Violation contract is

$$\overline{D^p} < \overline{D^a} + Xmin \tag{4}$$

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

[5] We assume here that the traffic characteristics of the primary and alternate channel are the same hence $Xmin = Xmin^a = Xmin^p$.

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.

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 violate their performance guarantees during the transition interval. If condition (4) is not satisfied, the number of packets that can violate their performance guarantees is bounded by $\lceil \frac{\overline{D^p} - \overline{D^a} - Xmin}{Xmin} \rceil$.

4.2 The DCM Algorithms

The key function of the DCM algorithms is to provide the support needed to modify parameters and routes under the constraints of resource sharing and of the modification contracts. These two constraints motivate three specific functions each of which is provided in an algorithm.

The establishment of the alternate channel can be examined under two scenarios: *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 links that are common with those of the primary channel and shares resources along all of the common links. The *Channel Administration Algorithm* provides the admission control tests and some additional constraints needed to support both of these scenarios.

In addition to performance parameter constraints, additional support is needed for the "No-violation" modification contract in order to 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.

The *Routing Algorithm* is used to determine a route from the source host to the destination host taking into consideration the clients requirements and the network's real-time load. For the routing algorithm to support dynamic connection management it must also be able to reflect the inclusion or exclusion of the resource sharing factor. 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.

4.2.1 The DCM Channel Administration Algorithm

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. 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.

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, Test 3 and 4 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 is the delay jitter, can be tuned by properly setting the local delay bounds [16, 17]. Assume that the shared link is an output link of switch i, that the upper bounds and lower bounds on 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, $\overline{D_i^{upper,p}} \leq \overline{D_i^{lower,a}}$ must hold. 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 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 and Test 4 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 (5), (6), and (7)[6].

$$Xmin^c = min(Xmin^a, Xmin^p) \tag{5}$$

$$Smax^c = max(Smax^a, Smax^p) \tag{6}$$

$$\overline{d^c_{\hat{i}}} = \overline{d^a_{\hat{i}}} \leq \overline{d^p_{\hat{i}}} \tag{7}$$

As the resources reserved by Test 3 and Test 4 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:

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

If the condition given below can be met:

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

then the alternate channel can be accepted.

The buffer resource modification test is of the same form as that of the bandwidth and scheduling test above. 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 ith 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:

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

The condition that needs to be satisfied is :

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

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*[7].

In modifying the client's connection the interface to the source and destination hosts must be preserved. The transition from the primary to the alternate channel must be "invisible" to the client as far as this interface is concerned. This interface is usually of the form of a unique identifier at the source and destination host and is preserved by appropriately modifying entries in the virtual circuit routing tables during connection establishment and modification. The manner in which this is accomplished is described in detail in [8].

4.2.2 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 "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[8]:

$$\overline{D^a_{path}} + Xmin - \overline{J^a_{path}} \leq \overline{D^p} < \overline{D^a} + Xmin \tag{8}$$

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

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

[8] There is a distinction between the delay bound specified by the client and the maximum delay of packets on the connection. We denote the latter by $\overline{D^a_{path}}$.

In this case, the buffers needed are

$$\lceil \frac{\overline{D^p_{path}} - \overline{D^a} - Xmin^a + \overline{J^a_{path}}}{Xmin^a} \rceil Smax^a \qquad (9)$$

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 (8) 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 scheduler at the appropriate times. This appropriate time, t_k, for packet k is $t_k = src_k + \overline{D} - \overline{J}$, where src_k is the source time stamp on the packet, and $\overline{D}$ and $\overline{J}$ are the delay and delay jitter bounds on the appropriate channel (i.e. the primary or alternate). 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 (4) be met to ensure that there are no delay jitter performance violations.

4.2.3 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. The administrative requirements are specified by the administrative parameter, which 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 are explained below.

The DCM Routing Algorithm provides source routing and is achieved by using a modified, constrained, version of the Bellman-Ford algorithm.

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 3.2 and Section 4.2.1 to determine the minimum queueing delay that this link can offer a connection with these traffic characteristics. The details of the routing algorithm can be found in [10].

The administrative constraints (e.g. resource sharing) 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. With a value of 2 (i.e. resource sharing allowed), the primary route 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, along a minimum spanning tree, the load values of its links to all other nodes. This broadcast is done upon the establishment, modification, or teardown of a channel. If there are no channel establishments, modifications, or teardowns within a specified time interval, periodic link updates are sent by each node to assure other nodes that the link is still active.

5 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 two experiments which exercise the algorithms to determine their correctness and examine applications in which their usefulness would be evident.

The performance indices of interest in the experiments are the throughput[9] of the channels, the maximum delays experienced by the packets as they traverse the route, and the number of out-of-sequence packets received by the destination hosts. The channel establishment times have also been recorded. The DCM algorithms should ensure the fol-

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

lowing (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.

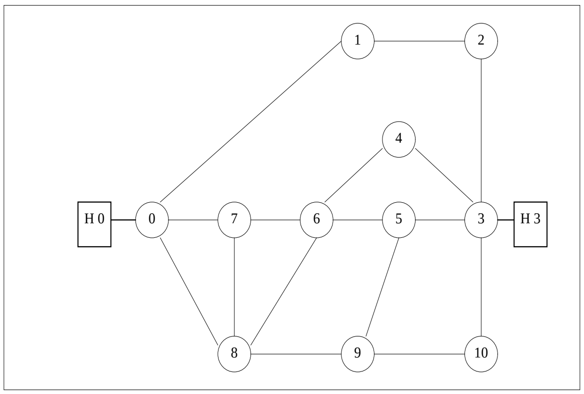

Figure 1: Experimental Network

The topology of the simulated network and its configuration are given in Figure 1. Each link in the network is duplex and has a transmission speed of 10 Mbps. The propagation delays of most of them are 10 ms. There are six links that have propagation delays of 20 ms : link(0,1)[10], link(1,0), link(1,2), link(2,1), link(8,9), link(9,8). The access links connecting hosts[11] to the network have a delay of 2.5 ms. 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/modification time of the connection is the sum of the communication delays (i.e., propagation and queueing), of the processing times of the establishment message (not including the admissions test), of the routing algorithm (at the source node only), and of the admission control tests. Since the communications delays and, to some extent, the message processing times are beyond our control, we shall focus on the processing times of the admissions tests and of the routing algorithms.

The admission control test is executed on a per link basis for each link traversed by the establishment message along the path. 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.

The routing algorithm is executed at the source. The order of growth for this Bellman-Ford based algorithm is $O(NL + ML)$, where *N* is the number of nodes in the network, and *L* and *M* are as described previously.

Under worst-case conditions with the dimensions of the experimental network (with 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 and the routing algorithm took 0.24 ms.

5.1 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 (doubling or tripling is usually sufficient) that permits useful work by the user.

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 20%. The original channel has the following traffic and performance parameters: $Xmin$ = 18.7 ms, $Xave$ = 18.7 ms, I = 1000 ms, $Smax$ = 1024 bytes, and $\overline{D}$ = 80 ms, and the tuned channel's parameters were: $Xmin$ = 15.6 ms, $Xave$ = 15.6 ms, I = 1000 ms, $Smax$ = 1024 bytes, and $\overline{D}$ = 80 ms. The source of the channel was host 0, and the destination host 3; the route used was 0,7,6,5,3. This enhancement was granted by the network at time 8 minutes, and can be seen in the throughput graph of Figure 2. The establishment and the modification times are 94.2 and 98.3 ms (90 ms of each of these times is propagation delay), respectively. These establishment and modification times are round trip times. After the initial tuning, the bandwidth available for normal playback was 0.52 Mbps, and that required for browsing at higher speed was 1.04 Mbps. The playback channel has the following values for the traffic and performance parameters: $Xmin$ = 15.6 ms, $Xave$ = 15.6 ms, I = 1000 ms, $Smax$ = 1024 bytes, and $\overline{D}$ = 80 ms. The channel's parameters needed to accommodate higher speed browsing are: $Xmin$ = 7.8 ms, $Xave$ = 7.8 ms, I = 1000 ms, $Smax$ = 1024 bytes, and $\overline{D}$ = 80 ms. 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

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

[11] All switches have hosts connected to them. In Figure 1, only two hosts are shown for simplicity.

that the bandwidth be reduced to its original value. In the experiment there are two periods during the channel's lifetime in which browsing at a higher speed was required, beginning at 900 and 3000 seconds with each lasting 300 seconds (see Figure 2).

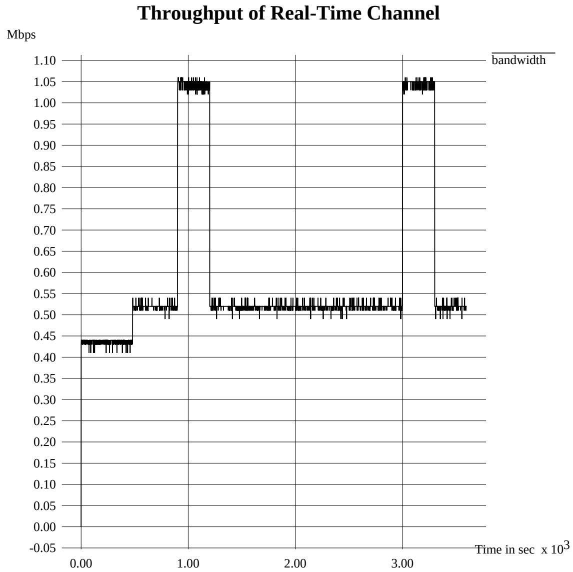

Figure 2: Experiment I - Browser

Table 1 shows the maximum packet delays experienced by packets on both the primary and alternate channels during several periods that span the duration of the experiment. It should be noted that no packet exceeds 80 ms , i.e. the the delays bounds of the primary channels, $\overline{D^a}$ and the alternate channels, $\overline{D^p}$. During the entire simulation experiment there were no out-of-sequence packets recorded at the destination. The route chosen for the first higher speed browsing period was the same, since the original route had sufficient residual resources along its links to support the increase. However, during the second period there were insufficient resources to accommodate the channel along the original route and the alternate channel traversed the route 0,8,9,10,3. It should be noted that the alternate channel establishment time along this route was 125.5 ms (110ms of this time is the propagation delay), which 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).

5.2 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 the original channel between hosts 0 and 3 were: $Xmin$ = 15.6 ms, $Xave$ = 15.6 ms, I = 1000 ms,

Time Period	**Max. Delay**
Delay Bounds (0 -3600 s), $\overline{D}^p and \overline{D}^a$	80 *ms*
Initial tuning period (0 - 900 s), I	77 *ms*
1st browsing period (901 - 1200 s), $B1$	53 *ms*
1st Interim period (1201 - 3000 s), $R1$	54 *ms*
2nd browsing period (3001 - 3300 s), $B2$	59 *ms*
2nd Interim period (3301 - 3600 s), $R2$	68 *ms*

Table 1: Maximum Packet Delays in Primary and Alternate Channels

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

Table 2: Result of Experiment II

$Smax$ = 1024 bytes and $\overline{D}$ = 160 ms, while that of the modified channel were: $Xmin$ = 15.6 ms, $Xave$= 15.6 ms, I = 1000 ms, $Smax$ = 1024 bytes, with the first delay bound $\overline{D}$ being 90 ms and the second 60 ms. The route used by the original channel was 0,7,6,5,3, and route 0,8,9,10,3 was used after both modifications. The route change ensured that there were out-of-sequence packets. As the necessary condition for the Non-Violation bound is not met, a Bounded-Violation modification contract is used for the modification. Table 2 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. In Table 2, $\overline{D^a}$ 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 2, the DCM performance-violation guarantees were met.

6 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 to a variety applications. These examples were divided into three categories: the dynamicity of client requirements, the dynamicity of the network state, and the tradeoffs between QoS and 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 disruption 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 (or *primary*) channel to the new (or *alternate*) channel and to ensure that the performance guarantees of the new 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. Many simulation experiments were conducted to determine the validity of the scheme. They proved that the scheme is indeed viable and useful in dealing with dynamics in the seconds time interval.

References

[1] D. P. Anderson, R. G. Herrtwich, and C. 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.

[2] I. Cidon, I. Gopal, and R. Guerin. Bandwidth management and congestion control in PlaNET. *IEEE Communications Magazine*, pages 54–64, October 1991.

[3] 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.

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

[5] D. Ferrari, A. Banerjea, and H. Zhang. Network support for multimedia: a discussion of the Tenet approach. To appear in *Computer Networks and ISDN Systems*.

[6] D. Ferrari and D. 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.

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

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

[9] C. Parris, G. Ventre, and H. Zhang. Graceful adaptation of guaranteed performance service connections. In *Proceedings of IEEE GLOBECOM'93*, Houston, TX, November 1993.

[10] Colin Parris, Hui Zhang, and Domenico Ferrari. Dynamic management of guaranteed performance multimedia connections. *ACM Multimedia Journal*, (to appear), 1994.

[11] G. 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.

[12] 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.

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

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

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

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

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

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