Connection-Oriented Service Renegotiation for Scalable Video Delivery[1]

A. Krishnamurthy and T.D.C. Little

Multimedia Communications Laboratory
Department of Electrical, Computer and Systems Engineering
Boston University, Boston, Massachusetts 02215, USA
(617) 353-9877, (617) 353-6440 fax
tdcl@bu.edu

MCL Technical Report 05-01-1994

Abstract–A common approach for supporting continuous media traffic is via resource reservation at connection establishment time. The shortcomings of this approach are the lack of flexibility in renegotiating performance parameters after connection establishment and the inability to react to changing network load conditions. A more desirable goal is the ability to scale the performance of individual connections during a connection's lifespan within the context of a network's capacity. In this paper we propose a protocol for renegotiating continuous media connections in a wide-area environment, considering an overall network perspective on resource availability rather than a per-connection resource management scheme. We also show the scaling gains that can be achieved by adapting the parameters of an encoding algorithm in the user workstations to justify the use of the protocol.

Keywords: Scalable video delivery, connection-oriented services, real-time networks and protocols.

[1]In *Proc. 1st IEEE Intl. Conf. on Multimedia Computing and Systems (ICMCS '94)*, Boston, MA, May 1994, pp. 502-507. This work is supported in part by Motorola Codex through the UPR Program and the National Science Foundation under Grant No. IRI-9211165.

1 Introduction

Rapid advances in workstation and networking technology have enabled the support of multimedia applications over computer networks. These applications differ from others in that they require a wide range of performance guarantees from the network and the user workstations [11]. Even with higher bandwidth availability, a best-effort scheme within the network is not suitable for support of such real-time operations. Quality of Service (QOS) parameters characterize the performance such applications demand from the network. The network resources have to be allocated to the different applications so as to maximize network efficiency as well as to guarantee the delivery of these QOS parameters required by the individual connections.

Present day networks do not necessarily provide guarantees on *end-to-end delay bounds*, *delay jitter* and *cell loss* in the network. A connection set-up protocol is required to allocate bandwidth to individual calls to ensure these performance bounds [7, 8, 9]. Existing protocols and proposals allocate bandwidth for a new connection during connection establishment, and the allocations remain during the period of the connection. However, many real time applications are scalable, i.e., the same perceived quality can be mapped on to a number of QOS requirement vectors. Furthermore, the characteristics of the source and the QOS requirements change over the lifetime of a connection. The service scalability of real-time applications provides a range of parameters the network can work with to support these applications and the flexibility to optimize its resource allocation to support the maximum number of connections.

As computer networks grow to embrace real-time applications, much work is being done to study the establishment of connections which guarantee the delivery of QOS parameters [2, 4, 5, 6]. Protocols suggested for this purpose must be fast and easy to implement at both application and network levels. These protocols run simple tests to determine if the connection can be accepted at each node [10]. The connections are either established or rejected in one round trip between the host and the destination. The decision on connection admission is based on three factors [1]: the QOS requirements of the new connection, the traffic parameters of the new connection, and the current state of the network, i.e., the load that the network is currently supporting. Traffic can be characterized by four parameters and the QOS can be specified by six quantities as shown in Table 1.

We propose a connection set-up protocol for a multi-hop network that enables the support of a wide range of QOS guarantees while trying to maximize the network utilization at the

Table 1: Traffic Characteristics

Par.	**Description**
X_{min}	minimum packet interarrival time
X_{ave}	average packet interarrival time over an averaging interval I
I	the averaging interval
L_b	maximum burst size
D_{max}	maximum delay experienced by a packet
J_d	delay jitter
CL	probability of cell loss
$P_{D_{max}}$	bounding probability that the delay is less than D_{max}
P_{J_d}	bounding probability that the jitter is J_d
P_{CL}	bounding probability on loss due to buffer overflow
W	worth of the packet, based on its priority

same time.

The protocol uses renegotiation of existing calls to provide the resources needed for a new call if they are not already available. The priority of the connection, which is a measure of how much the application or user is willing to pay, is also taken into account.

The remainder of this paper is organized as follows. In Section 2 we define service scaling and demonstrate scaling gains for JPEG compressed video sequences. In Section 3 we propose our session renegotiation protocol. In Section 4 we describe our evaluation of the protocol.

2 Service Scaling

Scaling is defined as subsampling the data stream to present only a fraction of the original contents [3]. Many real-time services are scalable. This means that we can scale them down while maintaining the same *perceived* quality at the receiving end. Furthermore, many applications can tolerate graceful degradation.

Subband-coded video is an example of a scalable video stream. Video information is decomposed into different frequency components, which comprise substreams of different degrees of importance [3]. Individual substreams are mapped onto different connections, each with its own set of QOS parameters. Scaling can be done by adjusting the QOS within a stream (continuous scaling) or by adding and removing sessions (discrete scaling) [3].

Another example of a scalable video stream is one compressed by the JPEG (Joint Photographic Experts Group) standard. One of the parameters specified during compression is the compression quality factor, which can take a range of values from 25 (best quality) to 1000 (worst quality), each of which results in a different average frame size. However, the human viewer can perceive changes in quality only over large changes in the compression factor. To demonstrate this, we conducted an experiment [2] in which 3 video streams were compressed frame by frame with different quality factors and played-out to human viewers. Fig. 1 displays the results of this experiment.

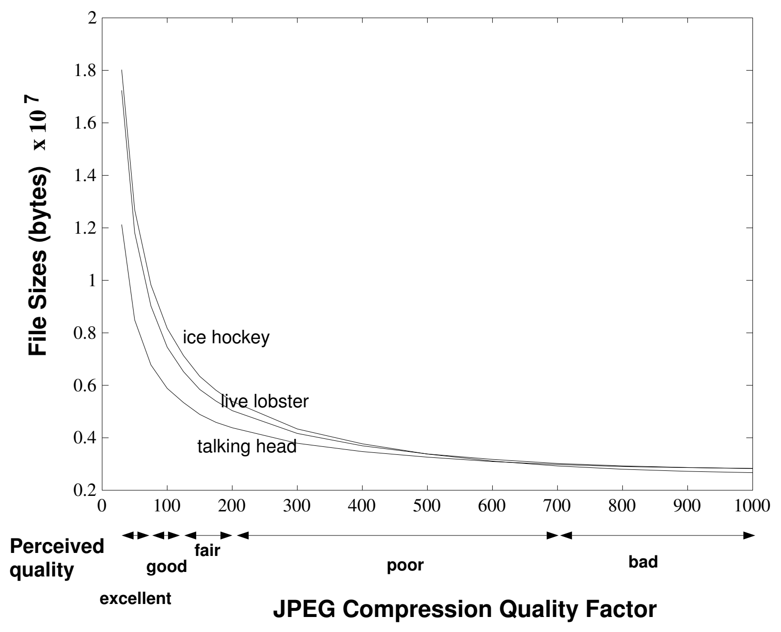

Figure 1: JPEG File Sizes vs. Compression Quality Factor

The figure shows aggregate frame sizes (as files) versus compression quality. The aggregate frame sizes decrease exponentially as the compression factor is increased. The perceived quality is also indicated on the horizontal axis. The results indicate that changes in compression quality are not immediately perceived by the viewer. Thus, for example, a quality of 25 or 35 is still perceived by the user as of excellent quality. This is translated to different frame sizes, or different amounts of data that must be transferred across the network to provide the same perceived quality. If the viewer is willing to tolerate degradation, the session can be

[2]The above experiment was conducted in an ad-hoc manner and should not be interpreted as rigorous. The results only illustrate potential scaling gains.

scaled down further. This provides the network with the flexibility to support a number of network parameters to deliver the same or a tolerable degradation in the quality of service. This experiment demonstrates the existence of QOS regions which can be applied by the network to maximize its resource utilization.

3 Proposed Renegotiation Protocol

The objective of the proposed protocol is to provide a framework for renegotiation between the network and hosts to take advantage of service scaling gains. The user specifies a single parameter, Q, which can be quantitative or qualitative, indicating the level of quality that is desired. The connection set-up protocol is responsible for mapping this parameter into a QOS vector. The protocol then uses this vector to set up or reject a connection. The protocol resides in the source and destination, as well as the network elements.

Once the network obtains the parameters corresponding to a call, it has to allocate sufficient resources to the call to ensure that the QOS requirements are met. The total amount of bandwidth available is a fixed quantity that has to be divided among the calls that the network supports. The bandwidth allocated to each call depends on its traffic characteristics and QOS requirements.

Existing protocols allocate the required resources after performing tests to check if the new connection can be supported without disturbing the guarantees given to the old ones. We examine the possibility of adjusting resource allocations during the lifetime of a connection. For a call, Q can be mapped onto an "admissible region" of QOS vectors, each of which satisfy the delivery of Q. The "admissible region" is obtained due to service scaling gains. The network can negotiate on this plane, and choose the vector that allows it to optimize performance, i.e., support as many connections as possible. The larger this region is, the more flexible the adjustments made by the network can be. The network can chose any point on the "admissible region" to maximize its utilization. In the following subsections, we describe the function of each component of the proposed protocol.

3.1 QOS Specification

The application specifies its QOS to the lower layers of the connection set-up protocol in terms of a single parameter Q, which relieves the user of knowing about the lower level QOS details. Q also reflects the willingness of the user to pay for a demanded QOS. The user would have a wide range of Q to chose from, each with a different associated cost, and the choice of a particular Q reflects the users willingness to pay the cost. This introduces prioritization at the highest level. Obviously, the network should give priority to an application which is willing to pay more over one which requires a lower QOS and is willing to pay less.

Once the user specifies a Q, the connection set up protocol tries to establish the call over the network to guarantee the delivery of the specified Q. The user is notified of the success or failure of the connection set up. The user can specify a different Q if the network cannot support the initial quality.

3.2 QOS Mapping

The lower layer maps the application QOS onto an "admissible" region. This region is made up of vectors of the type Φ:

$$\Phi = [X_{min}, X_{ave}, L_b, D_{ave}, J_d, CL, P_{D_{ave}}, P_{J_d}, P_{CL}, W],$$

where the parameters are given in Table 1. Given a high level QOS parameter Q the protocol maps it into a set of vectors Φ such that

$$\begin{aligned} Q &= \alpha X_{min} + \beta X_{ave} + \gamma L_b + \mu D_{ave} + \lambda J_d + \eta CL \\ &\quad + \delta P_{D_{ave}} + \zeta P_{J_d} + \theta P_{CL} + \sigma W \end{aligned}$$

where $\alpha, \beta, \gamma, \mu, \lambda, \eta, \delta, \zeta, \theta$, and σ are all weighting functions/constants. Because Q is expressed as a weighted function of the elements of Φ, it is possible to find many different Φ_i that map to the same Q. All vectors that satisfy $Q \geq Q_{min}$, where Q_{min} is the minimum acceptable QOS are said to map into an "admissible" region of operation. When the application is forced by the network to reduce its QOS, it chooses vectors lying outside the "admissible" space, but makes an intelligent choice such that the QOS degrades gracefully.

3.3 QOS Locus Identification

The protocol chooses two vectors, one corresponding to the ideal parameters for QOS delivery, Φ_{ideal}, and one at the edge of the "admissible" region, corresponding to the worst case parameters which would just satisfy the delivery of the high level parameter Q specified by the user, Φ_{worst} (Fig. 2).

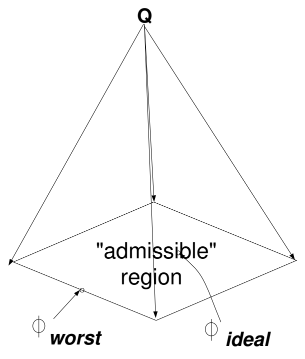

Figure 2: QOS Locus Identification

Instead of sending the whole plane of vectors making up the “admissible” region, only the ideal and worst case vectors are sent to the network at connection set up time. The network will initially attempt to support the ideal vector. If it cannot support this vector it degrades the vector until reaching the limit given by the worst case vector.

3.4 Request Vector Propagation

The request goes through all the nodes that are on the path from host to destination, and is evaluated at each node. Each node performs tests to see if sufficient processing and link bandwidth are available to allocate to the new channel to provide the requested QOS, without violating the QOS guarantees of existing connections. The node determines the minimum delay, jitter, probability bounds, and cell loss probability for the new connection based on the data characteristics from the ideal and worst case vectors after reserving bandwidth for the connection without affecting the existing connections, and passes this information on to the next node (Fig. 3). [3]

The request reaches the destination, along with the parameters added by each node on the path. The destination host receives information about the state of the network (provided by each node along the path), as well as the QOS requirements and data characteristics of

[3]These parameters depend not only on the characteristics of the traffic and the state of the network, but also the scheduling policies at the node. The policy used is transparent to the protocol, which only requires the calculation of these parameters at the nodes.

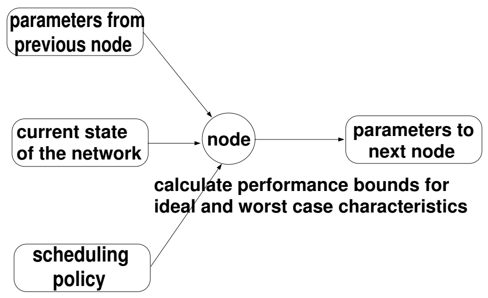
parameters from previous node
current state of the network
node
parameters to next node
calculate performance bounds for ideal and worst case characteristics
scheduling policy

Figure 3: Request Vector Propagation

the new connection. Based on this information, the node can make a decision on whether or not the connection can be supported.

3.5 Connection Establishment

The destination node evaluates the end-to-end QOS that the network can provide for the two cases (ideal and worst case data characteristics). The QOS parameters computed using the data characteristics from the ideal vector (QOS_{NI}) is then compared with the QOS requirements in the ideal vector (QOS_{HI}). This requires comparison of each QOS parameter in the vector. If the QOS provided by the network is better than what is required, the allocations made at each node are relaxed (computed at the destination node), and a connection establishment message is sent back along the same route to the host. When a node receives this message, it allocates bandwidth as requested by the destination (less than or equal to what was reserved). The message reaches the host and the connection is established.

If the QOS provided by the network is worse than that required by the ideal vector, the QOS is compared with the worst case vector. If the QOS is better than these parameters ($QOS_{NI} > QOS_{HW}$), then the connection is established. The destination sends back a message confirming the allocation of the reserved bandwidth at each node and finally to the host with the QOS parameters that are adopted for the connection. The host changes its protocol to map the application onto the vector corresponding to the established QOS. Note that the data characteristics change correspondingly to reflect the new mapping. The reservations made in the channel are for the data characteristics of the ideal vector, and so the resources allocated are actually more than what is required to support the QOS for this connection (Fig. 4).

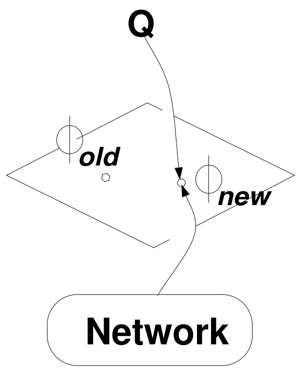

Figure 4: Vector Renegotiation

If the QOS provided by the network corresponding to the ideal data characteristics is worse than the worst case QOS required by the connection, the QOS provided with worst case characteristics (QOS_{NW}) is compared with the required QOS (QOS_{HW}). If this QOS provision is better than what is needed for the worst case, then the connection is established for the worst case. Again, the delay bounds at each node are relaxed and a message is sent back to the host with the established vector. The host changes its protocol to map the application onto this worst case vector.

3.6 Connection Renegotiation

If the QOS provided for the worst case is still worse than that required by the worst case, the destination identifies nodes on the path and calculates the delay, jitter, and cell loss bounds these nodes must provide to enable the establishment of the connection. The nodes selected are those with a low worth and a high delay, jitter, or cell loss bound. These nodes then adjust the bandwidth allocated to the *existing* connections so that they still remain within the worst case QOS and try to meet the requirements requested by the destination. Each node N tries to maximize its worth N_w, which is the sum of the worth of the connections it supports, subject to their QOS constraints.

The bounds for delay, jitter, and cell loss are calculated based on the worst case vectors that were used to set up the connections through nodes $N, N_1, .., N_c$. The destination goes ahead with connection establishment if source hosts of all these nodes are able to provide the required performance. These nodes then send messages to the source hosts of the connections whose allocations have been changed to remap the QOS vector.

If the node cannot satisfy the requested QOS parameters even after reducing the bandwidth allocation of its existing connections to the edge of their QOS requirements, then it identifies the lower priority connections (lower than those that of the connection to be set up) that go through it and schedules these connections for QOS degradation outside of their "admissible" region. Thus, some of the constraints in the optimization problem are replaced by weaker constraints. The schedule is organized so that the degradation is graceful. The node can schedule lower priorities for degradation to ensure that enough bandwidth is provided for satisfying the request for the new connection. If sufficient bandwidth is still not available, then it sends back a message to the destination host of the new connection. On receiving a refusal from this node, the destination host refuses the connection (or queues-up the connection set-up request).

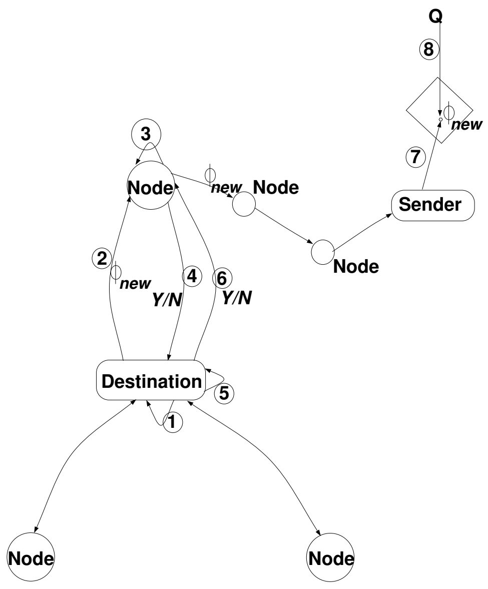

Figure 5: Connection Renegotiation

These procedures are illustrated in Fig. 5. The destination host identifies the nodes and computes the new vectors for them (1) and sends these vectors to the corresponding nodes (2). The nodes run algorithms to reduce the bandwidth of other connections through them to support the new vector, without violating the QOS guarantees of those connections. If necessary, the node identifies connections for graceful degradation from the guaranteed QOS so as to maximize the node's worth (3). The node sends a message to the destination telling it of its ability to support the new vector (4). The destination host gets these messages from all the queried nodes and decides if the connection can be supported with the new vector. It sends a message back to the node with this decision (5). If the destination gives the go ahead, the node modifies the QOS vector for the existing vector and updates it at all the nodes supporting the affected connection (6). The hosts of the affected connections adjust the vector in the "admissible" region to reflect the new vector (7). The translation protocol makes the adjustment to map the high level Q onto the new vector in the "admissible" region (8).

When a connection is released, it frees up the resources it had reserved in the network. The nodes which obtain these additional resources identify and scale up the connections they support.

4 Discussion

The renegotiation protocol we propose takes advantage of QOS mapping rescalability to improve overall network utilization while delivering the QOS guarantees of individual connections. It provides for communication between the elements of the network and the host and destination processors, and proposes a set of parameters and framework for this communication.

When the host specifies its QOS requirements, it has no knowledge of the state of the network. The renegotiation protocol provides a way for the the network to provide feedback to the source, and enables the source to decide on a suitable mapping scheme.

As in existing protocols, it is the destination which runs the final tests to determine whether or not the call can be admitted. However, we provide a way for the destination to change the state of the network as far as the limits of scalability of the already existing connections allow to accommodate the request. The protocol attempts to improve network efficiency by making use of the scalability of not only the new connection, but also of those which already exist. Such a scheme is viable when requests for connection establishment are not too frequent.

We believe that capitalizing on scalability will lead to substantial gains in network utilization when the network is heavily loaded. The protocol requires each node to run an optimization algorithm which should lead to a higher network efficiency. Finally, we observe that renegotiation between the elements of the network leads to an increase in call set up time over simpler existing protocols, but we envisage the protocol to work for real time applications such as video sessions which are established for relatively long periods of time. A small increase in connection set up time is justified for such applications. Renegotiation is required only if the state of the network cannot handle the initial request, and involves simple algorithms to be run at the the elements of the network and the source and destination processors.

5 Conclusion

Many real-time services are scalable. To take advantage of scalability gains, a mechanism is needed for the network and hosts to communicate with each other. In this paper we have proposed a protocol for renegotiating continuous media connections with the intent of providing gains achieved through service scaling. The proposed approach overcomes the lack of flexibility in renegotiating performance parameters after connection establishment and the inability to react to changing network load conditions. Further evaluation of the protocol will be done by means of simulation and implementation on a LAN testbed to demonstrate the effectiveness of our protocol in translating service scaling gains to an improvement in network efficiency.

References

[1] X. Chen, "Modeling Connection Admission Control," *Proc. Infocom 93*, San Francisco, CA, March, 1993, pp. 274-281.

[2] S. Crosby, "In-call Renegotiation of Traffic Parameters," *Proc. Infocom 93*, San Francisco, CA, March, 1993, pp. 638-646.

[3] L. Delgrossi, C. Halstrick, D. Hehmann, R. Herrtwich, O. Krone, J. Sandvoss, and C. Vogt, "Media Scaling for Audio Visual Communication with the Heidelberg Transport System," *Proc. ACM Multimedia 93*, Anaheim, CA, August, 1993, pp. 99-104.

[4] D. Ferrari and D.C. Verma, "A Scheme for Real-Time Channel Establishment in Wide-Area Networks," *IEEE Journal on Selected Areas in Communication*, Vol. 8, No. 3, April, 1990, pp. 368-379.

[5] J-I. Jung and D. Seret, "Translation of QoS Parameters into ATM Performance Parameters in B-ISDN," *Proc. Infocom 93*, San Francisco, CA, March, 1993, pp. 748-755.

[6] A.A. Lazar, A. Temple, and R. Gidron, “An Architecture for Integrated Networks that Guarantees Quality of Service,” *Intl. Journal of Digital and Analog Cabled Systems*, Vol. 3, No. 2, 1990.

[7] F.Y.S. Lin and J.R. Lee, “A Real-Time Distributed Routing and Admission Control Algorithm for ATM Networks,” *Proc. Infocom 93*, San Francisco, CA, March, 1993, pp. 792-801.

[8] R.T. Olson and L.H. Landweber, “Design and Implementation of a Fast Virtual Channel Establishment Method for ATM Networks,” *Proc. Infocom 93*, San Francisco, CA, March, 1993, pp. 617-627.

[9] T.E. Tedijanto and L. Gun, “Effectiveness of Dynamic Bandwidth Management Mechanisms in ATM Networks,” *Proc. Infocom 93*, San Francisco, CA, March, 1993, pp. 358-367.

[10] O. Yaron and M. Sidi, “Calculating Performance Bounds in Communication Networks,” *Proc. Infocom 93*, San Francisco, CA, March, 1993, pp. 539-546.

[11] Q. Zheng, K.G. Shin and E. Abram-Profeta, “Transmission of Compressed Motion Video over Computer Networks,” *Proc. Compcon 93*, San Francisco, CA, February, 1993, pp. 37-46.