The Multimedia Multicasting Problem

Joseph C. Pasquale, George C. Polyzos, Vachaspathi P. Kompella

Computer Systems Laboratory
Department of Computer Science and Engineering
University of California, San Diego
La Jolla, CA 92093-0114, U.S.A.

Technical Report Number CS93-313

September 1993

Abstract

We present the problems associated with multicast communication for multimedia group applications, such as video-conferencing. In particular, when these applications have continuous media I/O components (e.g., audio and video) which must be communicated to a group of receivers, unique problems arise. Perhaps the most fundamental problem is whether feedback should be used in dynamic control; we argue against such reactive control mechanisms and in favor of more proactive mechanisms. We discuss a number of issues, including routing, resource reservation, error control, heterogeneity, and hierarchical coding.

Keywords: broadcast, routing, interactive group communication, continuous media, communication network architectures.

Joseph Pasquale may be contacted at `pasquale@ucsd.edu`, +1-619-534-2673, +1-619-534-7029 (FAX).
George Polyzos may be contacted at `polyzos@cs.ucsd.edu`, 1+619-534-3508, +1-619-534-7029 (FAX).
Vachaspathi Kompella may be contacted at `kompella@cs.ucsd.edu`, +1-619-534-8604, +1-619-534-7029 (FAX).

1. Introduction

One of the ways communication can be characterized is by the number of receivers targeted by a sender. Traditional communication modes have been one-to-one, or unicast, and one-to-all, or broadcast. The generalization of these two extremes is multicasting, the communication of a single message to a select group of recipients which may or may not include the sender.

In multicasting, a sender communicates explicitly with a "group," rather than with individual actors which happen to be members of the group. Consequently the sender does not need to know the group membership, and in fact, the group membership may change over time. This model of communication supports applications where control and/or data are distributed amongst multiple actors. Examples include updates to replicated databases, obtaining a service from one out of a group of servers, and inter-process communication between cooperating processes.

One interesting class of applications which require group communication includes those with multimedia I/O components. Multimedia is the collection of multiple forms of information presentation, such as text and graphics (usually classified as discrete or time independent media), and, audio and video (typically classified as continuous or time dependent media). Sound and moving images are a natural form of communication between humans, making the support of continuous media in computer systems an important issue.

It is only recently that technology has become available for workstations to be equipped with audio and video devices. As these devices proliferate and become standard equipment in personal computers and workstations, and as high-speed fiber optics provide the necessary bandwidth for continuous media networking, we expect multimedia group applications to become standard tools for communication and collaboration. A canonical example of such applications is video-conferencing [4, 49, 51]. More generally, the class of applications called *groupware* [12], and computer-supported cooperative work (CSCW) [21], enhanced with audio and video I/O, require group communication.

Efficient multicasting is a fundamental issue for the ultimate success of these multimedia group applications. While in the past multicasting has been viewed as a service of limited use, often provided as an afterthought, this can no longer be the case. A major market for Broadband ISDN (B-ISDN) is expected to be selective video distribution [55, 56], analogous to CATV channels, but where the relatively small number of active receivers and the large number of channels (sources) make broadcast solutions impractical.

In this paper, we discuss the problem of multicasting multimedia data, particularly multiple time-correlated streams of continuous media (e.g. audio and video). Briefly, the relevant characteristics of this problem are: (1) data must be sent to multiple destinations, (2) the volume of the data is large, and requires high bandwidth, and (3) the value of the data is sensitive to delay; if received beyond a threshold dictated by human perception, they become useless. Since the volume of the data is large, minimizing duplication of transmission is important. And since the data are sensitive to delay, finding short routes which can transport data concurrently to the multiple destinations is important.

The primary goal of this paper is to discuss the relevant issues to solve this problem and put them in perspective, without going into great detail, especially where there are existing papers to which we can refer. With regard to actual solutions, we try to take a point of view or position, or we at least present the various alternatives.

The paper is organized as follows. In Section 2 we discuss the demands made by continuous media on networks. In Section 3 we describe the problems associated with multicasting of continuous media. In Section 4 we present past related work including our proposal for a dissemination-based multicast channel. Finally, conclusions are presented in Section 5.

2. Demands of Continuous Media on Networks

In this section we provide background on the nature of digital audio and video, and the relevant issues regarding network support.

2.1. The Nature of Audio and Video

Due to the very high data volumes of video and audio (when compared with traditional media such as text and graphics) it is important to provide efficient network mechanisms even when high-speed networks are available. For example, CD quality audio requires 1.4 Mb/s and HDTV quality video requires 1.4 Gb/s. Compression, particularly for video, can reduce the required bandwidth by 1-2 orders of magnitude, e.g. compressed HDTV quality video requires bandwidths of 20-40 Mb/s [18], still a significant amount especially considering that this is for only a single video stream. The large data rates make resource considerations important despite the decreasing costs of memory and transmission bandwidth. This is particularly important for multipoint applications which, if not designed with optimization in mind, can easily multiply the cost at least by the number of destinations. Although efficient multicasting has been recognized to be a very desirable service [37, 45], there is still insufficient support for it in most existing network implementations. However, there is on-going work in this area to address this problem, particularly new high-speed switch designs [13, 25, 36, 60, 61]. Also, ST-II, an experimental network layer protocol of the Internet suite [58] incorporates support for multicasting from the outset.

In addition to very high throughput requirements, audio and video are usually associated with real-time interactive applications and thus present rather stringent delay requirements. More precisely, in order for audio and video to be effectively used in interactive communication, i.e., without forcing the communicating subjects to modify their behavior from that of face-to-face communication, the delay between transmitter and receiver is expected to be minimal, with no long-term information storage involved. We use the term continuous media in this sense. It has been determined that a certain amount of delay is imperceptible or tolerable by humans. Various guidelines set this tolerance between 40 and 600 ms [14, 23, 35]. However, it is not just the mean or the maximum delay that needs to be bounded; the jitter, or variance of the delay, might also need to be bounded (e.g. because of buffer[1] limitations at the receiver) [62].

While real-time interactive applications expect interactions with delays that are nearly imperceptible by humans, those real-time applications that are not necessarily interactive (as perceived by humans) are also interesting. For example, typical digital video distribution using a "TV model" could be real-time, but there is no feedback (interaction) expected from the human receiver. On the other hand, if a "VCR model" is adopted for this distribution, the application should probably be characterized as interactive since the user could issue commands to control the flow of information and expect them to take effect immediately (by human reaction time standards). However, there is probably more tolerance to delay and less tolerance to errors, which is uncharacteristic of the bulk of interactive applications.

Multimedia applications, and more precisely continuous media components of such applications, may be classified into two categories [7]:

(1) *intolerant and rigid* ones, which require performance guarantees (which typically need to be provided through explicit reservations, even if the guarantees are of statistical nature and allow for occasional misses), and

(2) *adaptive and tolerant* ones, which expect a given quality of service, but can adjust or adapt to temporary degradation of service (in return for much lower cost); these could be satisfied without explicit reservation and micro-management of resources within the network (however, some admission control policy would still be required at the periphery of the network to secure the expected network performance).

2.2. Network Support

Real-time applications provide many challenges for packet-switched networks. Traditionally, these applications have been handled by circuit-switched networks with fixed (channel) data rates, typically in analog form, with no error control, and with variable quality (depending on instantaneous information content and channel quality). The use of variable-bit-rate (VBR) [44] digital transmission for applications such as video and audio makes packet-switching the desired mode of network transport because of the advantages of statistical multiplexing. Statistical multiplexing can realize large economies of scale and can better support constant quality rather than constant bandwidth services. For example, traditional video coders were designed to output a constant-bit-rate (CBR) signal to match the fixed bandwidth communication channels, independently from the actual instantaneous information content of the signal. Coders developed for packet switched networks take advantage of coding techniques that produce constant quality but VBR output. (E.g., when fast action or scene changes occur, a higher bit rate is produced to communicate the increased data volume produced by the differential encoding.)

Many of the techniques used in conventional packet-switched networks are based on assumptions of relatively slow, essentially non-interactive communication. It is important to re-examine those assumptions and reformulate the problems in the context of real-time, high-volume traffic. This is particularly critical in the case of multicasting. For example, one such assumption is the requirement for error-free transmission. The delay constraints of continu-

[1] Jitter can be smoothed out at the receiver by buffering, however, this requires additional buffering space. In addition, this increases the end-to-end delay which has an impact on interactive applications.

ous media do not permit retransmissions for error recovery (at least end-to-end for transcontinental links). Therefore, the (open loop, end-to-end) packet error rate of the transport mechanism becomes an issue. Fortunately, however, real-time audio and video usually exhibit some tolerance to transmission errors, depending on the particulars of their encoding scheme and the level of compression. Thus, audio and video may be characterized as soft real-time traffic, i.e., they require some kind of statistical delay guarantees, such as percentile of packets (or bits) received within the given delay tolerance [14].

A new problem introduced by the use of multiple continuous media is inter-media synchronization. One approach is targeted for traditional, physical or virtual, circuit-based communications, where all media streams are bundled together in a single wide stream (for example, by appending the corresponding audio segment to a video frame) [38]; this greatly simplifies the synchronization problem (assuming a single multimedia source). However, there is a loss in the opportunity for medium specific treatment within the network and the operating system at the destination.

The alternative approach, transmitting the different media separately, leads to a more general and flexible solution, albeit more complex. This solution introduces independence among the media, which has the potential to diminish the effects of communication errors, such as those leading to packet loss due to congestion. This is possible with protocols that support priorities based on the type of information transferred and which use these priorities to control congestion by "dropping" the least important packets, rather than being indiscriminate. Congestion control schemes based on these ideas have been proposed and are now under consideration for B-ISDN using the Asynchronous Transfer Mode (ATM) [43]. For example, graceful video quality degradation in the presence of congestion is usually more acceptable than audio degradation, and can be facilitated through the use of hierarchical coding (discussed in more detail in Section 3.7).

The traditional debate between connections and datagrams resurfaces again in the case of continuous media. In many internetworks, communication is typically stateless, with all information traveling in datagrams. This datagram model is supported by the bursty nature of traffic in conventional networks. Additional semantics, such as connections, are then added on top of the network layer. The real-time requirements of continuous media demand a consistent quality of service. The cost of providing guaranteed quality of service (through resource reservation) is believed to be high enough that it is not worth repeating for each datagram. Furthermore, the long-livedness of the communication sessions further boosts the move towards connections.

Many existing packet-switching networks tend to be naive at dealing with traffic admission. Every new packet is admitted into the network, regardless of the impact it may have on congestion, and the quality of service that existing connections may have been enjoying. This is undesirable when applications bargain for a specific quality of service which they expect to be maintained at more or less the same level. Admission control has to be introduced that determines the impact of a new multimedia connection on the existing network connections, and only admits new traffic when there is no negative impact [7, 43]. Tolerant and adaptive applications can be satisfied without explicit reservations and the necessary per packet switching overhead within the network, whose cost can be significant. Some admission control at the multicast connection level would probably be required, however, to secure the expected overall network performance.

We believe that flexibility in architecture design and implementation is more important than short-term implementation convenience, particularly since detailed requirements for key applications are still not entirely clear and the underlying network and device technology is not going to be homogeneous. Furthermore, approaches supporting distributed designs for multimedia terminals, based on multiple specialized processors for I/O and network access, working in parallel, will be important in the future.

3. Multicasting Continuous Media

We now address the problems that arise when providing multicast transport for distributed multimedia group applications with continuous media I/O components.

3.1. The Feedback Control Problem

A number of dynamic control mechanisms in communications protocols employ feedback from the receiver to the sender. For flow control, the receiver tells the sender to slow down or speed up to adjust to the receiver's current ability to consume packets. For error control, the receiver tells the sender to resend corrupted packets or packets that have not arrived within some time interval, and to acknowledge packets that have been successfully received. However, when there are multiple receivers and a single sender, how to provide feedback efficiently (or whether it should be provided at all) is a non-trivial question.

There are four basic approaches:

(1) ignore the problem,

(2) treat a 1-to-n multicast as n unicasts (even though only a single packet is sent initially),

(3) distribute the feedback control mechanism, and

(4) try to anticipate problems and take preventive action without feedback.

The first solution, ignore the problem, is the simplest. This may be appropriate for connection-less services such as datagrams, where the sender's responsibility is simply to transmit the datagram, and if problems arise along the way (e.g. the packet is corrupted, or the communication subnet or receiver is overloaded and the packet must be dropped), resolution is delegated to a higher level protocol.

In the second solution, the sender transmits a single packet, but then acts as if it just sent n packets to the n receivers, maintaining state information about each of the n "virtual packets" in transit. The sender must then wait until a stable state is reached before sending the next packet, i.e. until it is satisfied that all the receivers are ready for the next packet.[2] For flow control, this means slowing down to the consumption rate of the slowest receiver, forcing all other receivers to be delayed. For error control, this means taking corrective actions based on the most problematic receiver, and forcing all other receivers to be delayed. Unlike flow control, it may not be possible to return to a stable state which applies to all the receivers, as when a maximum number of retransmissions to some receiver have occurred without acknowledgement. In this case, the bad receiver may have to be treated special, perhaps by removing its membership from the multicast group, and allowing any special recovery actions to take place, while the rest of the group can go on receiving packets. Clearly, this is a complex solution, and performance degrades to that of the worst receiver (e.g. slowest, most overloaded, or connected through the slowest communication channel).

The third solution is to distribute the control mechanism over the nodes of the multicast tree. Control is now hierarchical (as dictated by the existing multicast tree). A receiver's feedback need not propagate all the way back to the sender. When an intermediate node receives the feedback, it either takes corrective action itself, or it propagates the feedback (which is possibly fused with feedback received from other downstream nodes) upstream to the next higher level node, and the action is recursive. This solution attempts to localize feedback and corrective actions to that part of the tree containing the problems (e.g. misbehaving links or receivers). A potential benefit is that receivers, which have paths to the sender that do not go through nodes currently reacting to such problems, achieve better performance. While this solution may lend itself to elegant algorithms for distribution of complexity, the fact that the complexity of intermediate nodes rises may be too costly, especially given the trend in networks towards simple switching nodes. However, this cost may be completely acceptable for less dynamic control mechanisms which are not overly time critical, e.g., resource reservation for connection setup. The discussion here mainly applies to dynamic control mechanisms.

Finally, the fourth solution is to take preventive action to eliminate or minimize any reliance on feedback. For error control, this means using forward error correction (FEC) [3, 53] to anticipate errors and provide information redundancy, allowing the receivers to reconstruct the information without asking the sender to retransmit. For flow control, this means reserving resources so that receivers and intermediate switching nodes are always able to support the flow rate dictated by the sender. The value of this approach is that there is no complexity due to feedback control mechanisms, and yet is an improvement over simply doing nothing (as in the first solution). The cost is the additional complexity due to these anticipatory/preventive action schemes. In addition, slightly higher bandwidth demands might result from such schemes, e.g., FEC. However, a small percentage increase of the overhead is probably a small price to pay to address the much more significant and difficult problems of rare but intense congestion phenomena in high-speed networks, due to the statistical multiplexing of high burstiness signals (some of which might consume significant percentages of total link bandwidth, and therefore, will not be easily controlled just by relying on the law of large numbers).[3]

We believe the fourth solution is the most appropriate for multicasting of continuous media. Given the non-critical error-tolerance characteristics of most continuous media applications, forward error correction schemes are appropriate (of course, as errors become more critical, the level of redundancy can be increased). As for resource

[2] This simplified description captures the essence of this approach. However, many similar but more sophisticated schemes have been devised; see [5, 41] and the references therein.

[3] Recall that ATM standardization accepted a 10% overhead by selecting a 53 octet cell in order to address other design problems.

reservation, not only is this appropriate for transporting continuous media, but we see an opportunity where hierarchical (layered) coding, with a separate reservation per layer, can be used to great advantage to deal with heterogeneous receivers (as described in more detail below).

Resource reservation schemes will generally employ protocols that use feedback to determine and implement resource allocations. However, since resource reservation takes place once, before the start of actual information flow (rather than multiple times throughout the information flow, as do the dynamic feedback control mechanisms described above), its cost may be acceptable (even if a distributed scheme is used). Therefore, resource reservation is not incompatible with the proactive approach of the fourth solution. To increase the statistical multiplexing gain, however, resource allocation and reservation should be applied to the individual media of multimedia flows (so that their characteristics and requirements can be addressed individually) and to the individual media layers when hierarchical coding is used.

3.2. Multicast Tree Connections

Efficient multicasting uses tree-structured connection topologies. The reason for this is that if a tree determines the route of packets, then the packets flow "in parallel" towards the destinations, and the number of duplicate packets traveling on the network is minimized. There are several problems that present themselves in multicast tree construction. First of all, there is the question of how many multicast trees should be used for group communication (assuming that all parties are expected to both send and receive). If one tree is used between all nodes in a multicast group, then the tree will not be optimal for each possible source. The links in this case are bi-directional and should hold desirable properties in terms of delay in both directions. However, the amount of state information that has to be stored to maintain one tree is smaller.

The alternative is to have a separate tree constructed for each possible source in the multicast group. Then the trees themselves can be made to be quite efficient. However, the total amount of state information needed is larger. Consequently, a change in membership or network topology can cause a large lag before an update of state information is completed. The links in this case will be unidirectional, since each tree will be a directed tree rooted at the source. Furthermore, both the one-tree-per-group and one-tree-per-source approaches can be combined with either one tree for all media or media layers or a separate tree for each medium or medium layer. For example, the quality of service parameters for audio and video are different; it may be best for a multicast route to be designed specifically for one set of quality of service parameters.

The three most important steps in setting up a tree connection for multimedia traffic are:

(1) admission control, or determining whether it is feasible to support the requirements of the new traffic,

(2) routing, or determining what the tree to be used is, and

(3) resource reservation, or allocating the resources needed to maintain the multicast connection.

In order to maintain a tree based multicast connection the three most important issues are:

(1) flow policing, i.e., ensuring that applications do not overload the network with more traffic than they declared,

(2) flow control (also referred to as traffic shaping), i.e., regulating the traffic so that delay-sensitive data meet their deadlines, and

(3) congestion control, i.e. dealing with overload situations gracefully.

Admission control, routing and resource reservation are intimately related, since the determination of whether a new connection can be admitted into the network without overloading is tantamount to discovering which route the new connection must take. Furthermore, the route that is chosen determines the resources that are consumed, and where the resources must be allocated. We focus on the multiple-tree approach (i.e., one rooted tree per source) in this paper.

3.3. Routing

The standard procedure for building a multicast route is to construct a tree along whose edges the information travels. The choice of the tree is typically governed by one of two main objectives, minimizing the cost of performing a multicast or minimizing the delay in completing the multicast. A Steiner tree can be used to minimize the cost of the multicast tree, but the Steiner tree problem is NP-complete [19]. Several good approximation algorithms have been devised for the Steiner tree with a proven constant worst case bound [34, 50]. These have been used to construct low-cost multicast trees [29, 64, 66]. The problem of finding a least delay route can be solved by running Dijkstra's shortest path algorithm [10] and pruning off the branches of the tree that do not contain any of the destination nodes.

In constructing a multicast tree we have to ensure that the quality of service parameters required by the application are met. For multicast traffic two very important factors for the goodness of a route (i.e., tree) are its total cost and the end-to-end delay from the source to each destination. More generally, there are two classes of metrics to be considered here. The first is a measure of the overall cost of forwarding the information to all (interested) receivers; obviously, this is a metric subject to design optimization. The second, is a single receiver oriented metric that characterizes each sender-receiver path, typically from a quality of service perspective. In this case achieving a certain target is the goal (rather than optimization). The above mentioned end-to-end delay is an example of such a metric that typically only needs to be kept below a target rather than being minimized. Another example of a single receiver oriented metric could be the total end-to-end path induced jitter. On the other hand, one could also think of various single receiver oriented costs (for example, if receivers rather than senders are charged individually for the communication costs), in which case optimization is in order.

In order to use a concrete example we assume that the goal is to minimize the total cost of the multicast tree, while ensuring that the end-to-end delay on each source-destination path is bounded by an application specified delay tolerance. Note that cost and delay are assumed to be known, independent metrics for each link of the network. For example consider the network of Fig. 1, where all edges have unit delay and unit cost. For a delay bound under 3 time units, in Fig. 1a and Fig. 1b, the minimum delay and the minimum cost solutions shown are not optimal, while Fig. 1c shows the ideal solution. We note that while the problem of finding this constrained Steiner tree is NP-complete, for multimedia applications with bounded integer delay constraints in networks with low connectivity, there exist fast routing algorithms that produce trees that, on the average, are nearly optimal [31-33].

The basic algorithm involves constructing a closure graph on the multicast nodes which contains a summary of the best path information between all pairs of nodes in the multicast group. The best path between two nodes is defined as the least cost path between those nodes such that the delay along the path is less than the bound specified by the application. A spanning tree of this closure graph rooted at the source provides a multicast tree. Care must be taken in constructing the closure graph and the spanning tree to ensure that no end-to-end delay bounds are violated. This algorithm has been evaluated on randomly generated graphs of up to 100 nodes, with average connectivity of 15 nodes, for group sizes up to 30 nodes, and various delay tolerances. On the average, the trees constructed using this algorithm are within 5-10% of the optimal cost, and they satisfy the delay requirements of the application. An interesting result of the study is that the shortest delay tree (i.e., the tree of shortest delay paths from the source to the destinations without attempting to optimize on the cost), has a cost that is, on the average, 75% more expensive than optimal. Similar results have been obtained on graphs with the NSFNET topology. Thus, in terms of cost, the shortest delay tree provides a poor solution. Furthermore, using the smallest delay paths is more restrictive than what is needed by most applications, which typically need only a bounded delay.

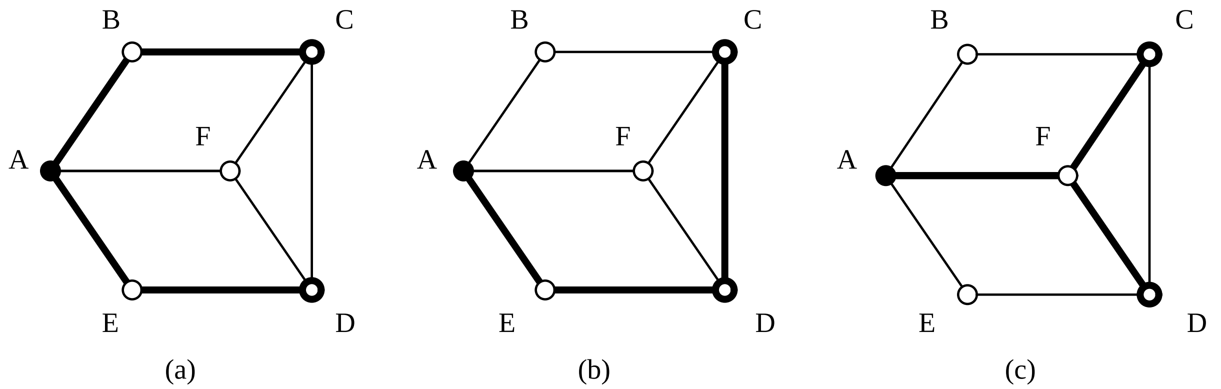

Fig. 1: Multicasting from A to C and D, with unit cost and unit delay on each edge.
a) A minimal end-to-end delay tree.
b) A minimal cost tree.
c) A minimal cost, bounded delay tree, with delay bound < 3.

3.4. Resource Reservation

For guaranteed service, once a route has been determined, the resources that made the route desirable must be reserved, so that the requirements demanded by the application can be guaranteed. For example, in RTIP [15], a real-time extension to IP, it is possible to set up a real-time channel with quality of service parameters that specify the bandwidth required, and the end-to-end delay and the jitter that the application can tolerate. In setting up the channel, a setup message with the appropriate information is sent to the destination. As the message travels along the path from source to destination, the intermediate nodes make a commitment of their resources for the connection. Since they can not know the commitments that nodes downstream will make, each node typically will tend to overcommit resources to this connection. Finally, the destination will examine the resources that have been allocated to the connection by each node, and determine the relaxation on each node's commitments. A return message from the destination to the source then causes each node to allocate the appropriate resources to the connection.

The reservation of resources in a multicast environment is more complex than for unicasts. The difficulty lies in the fact that paths to different destinations may share links. The result is that allocation of resources along some shared links may satisfy the requirements of one destination, but not those of another.

It is possible to extend the resource reservation mechanism to multicasting, with a channel establishment time of at most $1.5 \times T_{\max}$, where $T_{\max}$ is the maximum of all round-trip delays from the source to the destinations. After the multicast tree has been determined, the resources that each node must allocate for the connection have to be reserved. In this phase, a message is multicast along the tree to request resources. In the second phase, a reply from each node propagates towards the source, confirming the allocation. Since it is possible in the first stage that some nodes may overcommit resources, the second phase also permits relaxation of the committed resources. A junction node in a tree will receive several replies, one along each branch. It will need to propagate the most restrictive of the relaxations towards the source. Now, there will be some paths from the junction to the destinations that had less restrictive relaxations. It will be possible to make a further relaxation of resources on these nodes.

3.5. Reconfiguration

One of the problems with connection oriented communication is that it is not very flexible. The question of how to reconfigure a given tree when some parameters change is particularly hard. For example, it may become necessary to change the tree because the network topology has changed. Another reason could be that the quality of service bargained for may be excessively restrictive (and thus expensive) or may be inadequate. Such changes may entail rebuilding the multicast tree.

Group membership dynamics also impact the topology of the tree. When hosts join or leave the multicast group in a datagram oriented network, the group management process has to make note of the changes. However, data forwarding is still relatively simple. In a connection oriented network, however, the trees used for the multicast have to be reconfigured. The optimal solution would involve reconstructing the entire tree. There are low-cost alternatives, like finding the cheapest extension to the existing tree to include a new host, or pruning the branch of the tree that contains a departing host. After several such readjustments, however, the tree may become sub-optimal to the extent that reconstruction may be necessary. References [1, 66] discuss how to deal with group membership dynamics (in the case of multicasting without delay constraints).

Similar techniques can be applied in the delay constrained case. For example, the problem of adding a new member to the tree can be simply (but obviously sub-optimally) solved by finding the best path from the multicast group to the new node that satisfies the delay bound. More complicated strategies could look at all the nodes in the tree. One obvious strategy is to maintain the closure graph, and update it every time a new node joins the group. The recomputation to produce a spanning tree would take little time, since the multicast group size is small. Thus, the approach to reconfiguration could be to prune the tree when nodes leave, and to do a partial reconstruction when nodes join.

3.6. Error Control

In point-to-point channels, error-free communication is usually achieved by error detection and retransmission. Error control is particularly difficult with multipoint connections because of the feedback control problem. A naive multicast error control scheme could be based on positive acknowledgements. In that case, if communication is hampered by transmission errors, retransmissions will reach all members of the group, including the ones that received the packets correctly. In order to minimize the overhead, alternative schemes use negative acknowledgments, taking advantage of the high reliability of links. These schemes are based on the observation that if errors occur only sporadically, then recipients need respond to the sender only on errors, asking for retransmissions. This requires synchronization packets to be sent periodically to update all receivers on which packets were transmitted successfully [42]. Various schemes improving on traditional Automatic Repeat reQuest (ARQ) techniques have been devised [5, 41].

However, the real-time requirements of multimedia applications do not typically permit ARQ solutions (at least in general settings, e.g., when transcontinental links are included in the path). Various combinations of forward error correction schemes and error recovery and concealment techniques will probably be necessary. Interestingly, the choice of open-loop error control suggested by the real-time character of continuous media also solves the multipoint feedback control problem by eliminating the need for feedback at the lowest layers of the protocol hierarchy.

An important characteristic of real-time interactive applications such as video-conferencing is that the applications can tolerate some number of errors without significant degradation in quality. The consequence is that even without fully reliable transmissions, applications can be made to recover gracefully from loss of segments (at least for some media components). There is, thus, a trade-off involved here with real-time continuous media between smooth progress and complete reliability. Typical video and audio information will be compressed for efficient transportation. This decreases the redundancy in the data, and thus limits the error resiliency of the applications in question. Hierarchical coding for continuous media can improve the error resiliency of multimedia applications.

3.7. The Value of Hierarchical Coding

Hierarchical coding techniques (also referred to as layered, component, or sub-band coding) split continuous media signals into components of varying importance [20, 30]. The aggregation of these components reconstructs the original data, but subsets of them can also provide various degrees of approximation to the original signal. For example, a simple form of hierarchical coding could decompose a video frame into two sub-samples: (1) a low resolution component containing one quarter of the pixels, and (2) a high-resolution component containing the remaining three quarters of the pixels. For a receiver which uses a presentation window of size one quarter the size of the frame generated by the transmitter, receiving the second part of the signal would not be useful, and can actually be counter-productive because it can be a severe strain on local resources.

Therefore, the use of hierarchical coding can be very important as it gives the system software at the receiver the capability of allocating resources based on local (i.e., the receiver's) specifications and priorities. This might entail deciding to gracefully and dynamically degrade the quality of the received and played-back signal when resources are limited. Also, it might facilitate local processing and presentation of the signal in ways not intended by the sender [39]. Furthermore, there is an interesting interaction between hierarchical coding and multicasting. Hierarchical coding enables destinations to adjust the quality of the signal they receive each, independently and without the source actually being aware of this adjustment (of course, the adjustment is only possible towards lower quality). This is a very important property considering the feedback control problems of multipoint communication, and can also be used to effectively address many compatibility problems.

Hierarchical coding is also very important for the efficient use of high-speed networks. The main issue for such network architectures (e.g., ATM) is the significant congestion control problems which can arise due to the statistical multiplexing of very high burstiness signals [11, 59]. A key property of any solution (that is not based on resource reservation at the peak rate of the sources) is the ability to shed load quickly without causing an avalanche of retransmissions of dropped traffic [40]. With layered coding of continuous media, when network congestion arises it is possible to drop the less important signal components without causing service interruption, and without the need for retransmissions. This should only lead to a reduction in service quality (which may be unnoticeable in many cases). Since continuous media will constitute the bulk of the network traffic, this technique can be very effective as a last resort for congestion control. Many of the proposed congestion control techniques rely on this feature [11, 40].[4] Furthermore, routing algorithms may take into account the different needs and capabilities of the destinations by, for example, forwarding only usable components to select destinations [54].

There are, of course, trade-offs between the benefits derived from the independence of the components and the total volume of data generated by compressing components individually (in order to achieve their independence). First of all, coding (compressing) the various layers independently would in general result in less efficient compression (this however, does not seem to be a concern). More at issue is the cost of splitting the signal into

[4] Similar strategies can be used even at the destination. When a presentation deadline is approaching and there is risk of the processor missing the deadline, the system software could decide to focus on processing and presenting only the most important components.

components, and (maybe more importantly) signal reconstruction at the receivers. Given that the use of layered coding has been limited at best until now, system support (from the operating system, window system, framebuffer hardware) for signal reconstruction is non-existent, or even worse, there is a significant penalty for not conforming to the traditional, supported presentation formats. Hopefully, system support for hierarchical coding will improve in the future.

Interestingly, some compression standards, notably JPEG [65], support various forms of hierarchical coding Another related observation is that with hierarchical coding, the basic (e.g., low resolution) layers which are important for signal continuity are highly compressible. A reasonable strategy then is to transmit those through connections with explicit guarantees, and send the other layers on connections without explicit guarantees or require less stringent guarantees.

3.8. Dealing with Heterogeneity

It is expected that there will be many types of multimedia terminals and that possibly many formats for representation of each medium type will exist [28]. This has been the case for traditional text-based computer communications (where the compatibility problems are considerably simpler than for multimedia). The presentation layer in the OSI architecture addresses the issues of data representation compatibility problems. When incompatible transmitters and receivers communicate, translation is necessary. This service can be provided in one of three places:

(1) at the transmitter,

(2) at the communications channel (or within the network), or

(3) at the receiver.

If the third solution is possible, there is no real compatibility problem. The second approach is the typical solution (e.g., protocol converters). The first solution is similar to the third in the point-to-point case, but is rather inefficient in the multipoint case. It requires the sender to translate its data format once for every incompatible receiver. This translation consumes sender resources that are usually highly contended. Furthermore, it requires the network to transport a higher volume of information because sharing of links is not possible. In essence, severe compatibility problems limit the effectiveness of multicast solutions and the communication model reverts to a series of unicasts.

Facilitating translation at the receiver through synthesis of the signal from separate components can be achieved through appropriate hierarchical coding. Thus, this technique can help diminish compatibility problems for continuous media and also address other problems such as real-time delivery, error control, and network congestion control.

3.9. Considerations at Lower Layers

The overall philosophy of the network layer can have a serious impact on its ability to support multicasting. It is not surprising that the main available support for multicast at the network layer is in the form of IP multicast [8, 63], and that there is little support at the transport layer. Santoso and Edida [52] address the semantics of reliable transport layer multicast. Serious problems arise in using multicast techniques when the multicast group extends over multiple interconnected networks with different architectures, or when support for multicasting is inconsistent. This is particularly difficult for multimedia multicasting which requires adequate support, instead of the least common denominator approach traditionally taken in these situations (which guarantees connectivity, but not much beyond that).

For example, because of the lack of unanimous support for multicasting in the networking community, IP multicasting is implemented in only a few machines. Thus, less than efficient ways to route multicast packets over an internetwork are being used. When IP multicast datagrams are sent over the Internet, they are encapsulated in IP packets with standard IP headers, and sent to sites that understand IP multicasting. There, the packets are unwrapped, and the real datagrams are appropriately dealt with. This technique, known as tunneling [63], is suboptimal.

Support for multicasting may exist at the data-link and physical layers. For example, Ethernet has been designed to support multicast addressing. However, processors are often forced to perform extra processing when receiving a multicast packet since many available Ethernet controllers only support partial address filtering [26]. Another example is the ability to recognize that a packet is being multicast, and that multiple transmissions over different links, while using the same buffer, are required. New ATM switch designs [60, 67] increasingly support (in hardware) parallel transmission of multicast packets over multiple outgoing links. Such support will be necessary in order to achieve the high performance expected by multimedia applications.

4. Related Work

A number of research issues have evolved from the study of multicasting. A good survey on multicasting can be found in [17]. Chanson *et al.* provide a bibliography of multicast and group communications [6]. A compilation of recent salient work in multicasting can be found in [2]. The basic graph theoretic problems have been how to get the multicast communication to the destinations in the fastest possible time, and how to get the multicast communication done for the least cost. This latter problem is also known as the Steiner tree problem [19, 22]. Jiang [27] presents two heuristic Steiner tree algorithms for broadband stream multicast routing taking into account link capacities. Tode *et al.* [57] propose two multicast routing algorithms which distribute the copy operation of packets over all nodes along the multicast path.

A very similar problem to that of multicast routing exists in routing in VLSI circuits. The objective here is to make the cheapest connections (in terms of the length of the wire used). It is also important to keep the number of hops in the path from the source of the signal to the destination, since each hop adds delay. The difference in the two problems is that in the VLSI circuit, additional points (called Steiner points) can be introduced in the network that act as junctions, whereas in computer networks, junctions have to be found among the existing nodes in the network.

The pioneering work on IP multicasting by Deering and Cheriton [9] is fueling some of the current protocol work, which is based on a group communication protocol, the Internet Group Management Protocol (IGMP) [8]. This protocol specifies how multicast groups can be formed and managed with little change to the basic infrastructure of IP. The two major changes to the implementation of IP to accommodate multicasting are the provision of IP multicast addresses and the IGMP protocol. Multicast groups are created by individual hosts explicitly joining the groups, i.e., using IGMP to include them in the address list corresponding to a multicast address. The actual multicast routing is done by routers that learn the shortest path routes to the destinations. These routers then forward multicast packets along those shortest paths.

Recent experiments using IP multicasting include the telecasting of the Internet Engineering Task Force (IETF) meetings. Live audio and slow scan black-and-white video from the meetings are distributed to hosts included in the multicast group. Tunneling [63] is used to send data over portions of the Internet that do not support multicast routing.

An alternative network layer protocol in the Internet suite, specially designed for continuous media transmission, is ST-II [58]. This experimental protocol supports multicasting and resource negotiation appropriate for continuous media, but the protocol itself does not include specific implementation mechanisms, e.g., for multicast route set-up or network resource reservation (instead, such mechanisms must be provided outside of the protocol). Partridge and Pink report on their experiences implementing ST-II in [46].

A communication system designed to support heterogeneity and which provides support for multicasting is described in [24]. Hierarchical coding and layered error control techniques supporting multicasting to heterogeneous receivers, or receivers reachable through paths with different characteristics, are described in [54]. Hierarchical coding has been studied extensively in the area of signal processing. It has recently been recommended as a potentially effective mechanism for congestion control for high-speed networks carrying digital continuous media [20, 30]. Provision of operating system or network system software level support for hierarchical coding, particularly in relation to multicasting, is mentioned in [16, 47].

We are addressing the problems described above in the design of a multicast communication channel for multimedia called the Multimedia Multicast Channel (MMC) [48]. The MMC provides a communication abstraction similar to a "television broadcast channel" for a group of distributed processes, where a source disseminates multimedia information such as video and audio to a varying number of destinations which are "tuned in" to the MMC. The MMC is based on a communication paradigm which is a major departure from more traditional ones: the source and the destinations are very loosely coupled in their control and data exchanges. In general, the source's main concern is to push various media streams into the MMC; in particular, it is not concerned with where the streams end up or how they are used. A destination's main concern is what to extract from the MMC, which is viewed as offering multiple media streams, some or all of which are of interest.

5. Conclusions

We have presented the major problems associated with multicasting continuous media. These problems present themselves over those of conventional multicasting (e.g., formation of groups, addressing) because of the high bandwidth and bounded delay requirements of video and audio. These characteristics affect the connection setup problem in how routes are determined and how resources are reserved. We suggest minimizing reliance on feedback control mechanisms, in favor of mechanisms which are anticipatory of problems and take preventive action. Given an environment of heterogeneous receivers, the way in which continuous media is encoded has a major effect on efficiency and flexibility. We believe hierarchical coding and open loop control are fundamental to effective multimedia multicasting.

References

1. L. Aguilar, J. J. Garcia-Luna-Aceves, D. Moran, E. J. Craighill, and R. Brungardt, ‘‘Architecture for a Multimedia Teleconferencing System,’’ *ACM SIGCOMM ’86 Symposium*, pp. 126-136 (August 1986).
2. M. Ahamad, Ed., *Protocols for High-Speed Networks*, IEEE Computer Society Press, Technology Series (1990).
3. E. W. Biersack, ‘‘Performance Evaluation of Forward Error Correction in ATM Networks,’’ Proc. *SIGCOMM ’92*, pp. 248-257 (August 17-20, 1992).
4. S. Casner, K. Seo, W. Edmond, and C. Topolcic, ‘‘N-Way Conferencing with Packet Video,’’ *Third International Workshop on Packet Video*, (March 22-23 1990).
5. S. R. Chandran and S. Lin, ‘‘Selective-Repeat-ARQ Schemes for Broadcast Links,’’ *IEEE Transactions on Communications* **40**(1), pp. 12-19 (January 1992).
6. S.T. Chanson, G.W. Neufeld, and L. Liang, ‘‘A Bibliography on Multicast and Group Communications,’’ *Operating Systems Review* **23**(4), pp. 20-25 (Oct. 1989).
7. D. D. Clark, S. S. Shenker, and L. Zhang, ‘‘Supporting Real-Time Applications in an Integrated Services Packet Network: Architecture and Mechanism,’’ Proc. *ACM SIGCOMM’92*, pp. 14-26 (August 1992).
8. S. Deering, ‘‘Host Extensions for IP Multicasting,’’ RFC 1112, Network Information Center, SRI International, Menlo Park, CA (August 1989).
9. S. E. Deering and D. R. Cheriton, ‘‘Multicast Routing in Internetworks and Extended LANs,’’ *ACM Transactions on Computer Systems* **8**(2), pp. 85-110 (May 1990).
10. E. W. Dijkstra, ‘‘A note on two problems in connexion with graphs,’’ *Numerische Mathematik* **1** pp. 269-271 (1959).
11. A. E. Eckberg, ‘‘B-ISDN/ATM Traffic and Congestion Control,’’ *IEEE Network* **6**(5), pp. 28-37 (September 1992).
12. C. A. Ellis, S. Gibbs, and G. L. Rein, ‘‘Groupware — Some Issues and Experiences,’’ *Communications of the ACM* **34**(1), pp. 38-58 (January 1991).
13. K. Y. Eng, M. G. Hluchyj, and Y. S. Yeh, ‘‘Multicast and Broadcast Services in a Knockout Packet Switch,’’ Proc. *IEEE INFOCOM ’88*, pp. 29-34 (March 1988).
14. D. Ferrari, ‘‘Client Requirements for Real-Time Communication Services,’’ *IEEE Communications Magazine* **28**(11), pp. 65-72 (November 1990).
15. D. Ferrari and D. C. Verma, ‘‘A Scheme for Real-Time Channel Establishment in Wide-Area Networks,’’ *IEEE Journal on Selected Areas in Communications* **8**(3), pp. 368-379 (April 1990).
16. D. Ferrari, J. C. Pasquale, and G. C. Polyzos, ‘‘Network Issues for Sequoia 2000,’’ Proc. *IEEE Compcon Spring ’92*, (February 24-28, 1992).
17. A. J. Frank, L. D. Wittie, and A. J. Bernstein, ‘‘Multicast Communication on Network Computers,’’ *IEEE Software* **2**(3), pp. 49-61 (1985).
18. D. Le Gall, ‘‘MPEG: A Video Compression Standard for Multimedia Applications,’’ *Communications of the ACM* **34**(4), pp. 46-58 (April 1991).
19. M. R. Garey, R. L. Graham, and D. S. Johnson, ‘‘The Complexity of Computing Steiner Minimal Trees,’’ *SIAM Journal on Applied Mathematics*, (34), pp. 477-495 (1978).

20. M. Ghanbari, ‘‘Two-Layer Coding of Video Signals for VBR Networks,’’ *IEEE Journal on Selected Areas in Communications* **7**(5), pp. 771-781 (June 1989).

21. I. Grief, Ed., *Computer Supported Cooperative Work,* Morgan Kaufmann Publishers (1992).

22. S. L. Hakimi, ‘‘Steiner’s Problem in Graphs and its Implications,’’ *Networks* **1** pp. 113-133 (1971).

23. D. Hehmann, H. Salmony, and H. Stuettgen, ‘‘Transport services for multimedia applications on broadband networks,’’ *Computer Communications* **13**(4), pp. 197-203 (1990).

24. D. Hehmann, R. G. Herrtwich, W. Schulz, T. Schutt, and R. Steinmetz, ‘‘Implementing HeiTS: architecture and implementation strategy of the Heidelberg high-speed transport system,’’ Proc. *2nd International Workshop on Network and Operating System Support for Digital Audio and Video,* (November 1991).

25. A. Huang and S. Knauer, ‘‘STARLITE: A Wideband Digital Switch,’’ Proc. *IEEE GLOBECOM ’84,* pp. 121-125 (December 1984).

26. L. Hughes, ‘‘Survey of Multicast Address Handling Techniques for Ethernet Communication Controllers,’’ *Microprocessors and Microsystems* **13**(9), pp. 563-568 (Nov. 1989).

27. X. Jiang, ‘‘Routing Broadband Multicast Streams,’’ *Computer Communications* **15**(1), pp. 45-51 (Jan.-Feb. 1992).

28. R. K. Jurgen, Ed., ‘‘An Abundance of Video Formats,’’ *IEEE Spectrum,* pp. 26-28 (March 1992).

29. B. K. Kadaba and J. M. Jaffe, ‘‘Routing to multiple destinations in computer networks,’’ *IEEE Transactions on Communications* **31**(3), pp. 343-351 (March 1983).

30. G. Karlsson and M. Vetterli, ‘‘Packet Video and Its Integration into the Network Architecture,’’ *IEEE Journal on Selected Areas in Communications* **7**(5), pp. 739-751 (June 1989).

31. V. P. Kompella, J. C. Pasquale, and G. C. Polyzos, ‘‘Multicasting for Multimedia Applications,’’ Proc. *IEEE INFOCOM ’92,* pp. 2078-2085 (May 1992).

32. V. P. Kompella, J. C. Pasquale, and G. C. Polyzos, ‘‘Multicast Routing for Multimedia Communication,’’ *IEEE/ACM Transactions on Networking* **1**(3),(June 1993).

33. V. P. Kompella, J. C. Pasquale, and G. C. Polyzos, ‘‘Two Distributed Algorithms for Multicasting Multimedia Information,’’ Proc. *Second International Conference on Computer Communications and Networks,* (June 1993).

34. L. Kou, G. Markowsky, and L. Berman, ‘‘A Fast Algorithm for Steiner Trees,’’ *Acta Informatica* **15** pp. 141-145 (1981).

35. AT&T Bell Laboratories, *Engineering and Operations in the Bell System*. 1984.

36. T.T. Lee, ‘‘Nonblocking Copy Networks for Multicast Packet Switching,’’ *IEEE Journal Selected Areas in Communications* **6**(9), pp. 1455-1467 (December 1988). (Special Issue: Broadband Packet Communications)

37. B. Leiner, Ed., ‘‘Critical Issues in High-Speed Networking,’’ RFC 1077, Network Information Center, SRI International, Menlo Park, CA (November 1988).

38. W. H. Leung, T. J. Baumgartner, Y. H. Hwang, M. J. Morgan, and S.-C. Tu, ‘‘A Software Architecture for Workstations Supporting Multimedia Conferencing in Packet Switching Networks,’’ *IEEE Journal on Selected Areas in Communications* **8**(3), pp. 380-390 (April 1990).

39. A. Lippman, ‘‘Feature Sets for Interactive Images,’’ *Communications of the ACM* **34**(4), pp. 92-102 (April 1991).

40. D. M. Lucantoni and S. P. Parekh, ‘‘Selective Cell Discard Mechanisms for a B-ISDN Congestion Control Architecture,’’ *7th International Teletraffic Congress Symposium*, (1990).

41. K. Mase, T. Takenada, H. Yamamoto, and M. Shinihara, ‘‘Go-Back-*N* ARQ Schemes for Point-to-Multipoint Satellite Communications,’’ *IEEE Transactions on Communications* **31**(4), pp. 583-589 (April 1983).

42. P. V. Mockapetris, ‘‘Analysis of Reliable Multicast Algorithms for Local Networks,’’ Proc. *of the 8th Data Comm. Symp.,* pp. 150-157 (Oct 1983).

43. Special Issue: Congestion Control in ATM Networks, *IEEE Network* **6**(5),(September 1992).

44. M. Nomura, T. Fujii, and N. Ohta, ‘‘Basic Characteristics of Variable Rate Video Coding in ATM Environment,’’ *IEEE Journal on Selected Areas in Communications* **7**(5), pp. 752-760 (June 1989).

45. C. Partridge, ‘‘Workshop Report, Internet Research Steering Group Workshop on Very-High-Speed Networks,’’ RFC 1152, Network Information Center, SRI International, Menlo Park, CA (April 1990).

46. C. Partridge and S. Pink, ‘‘An Implementation of the Revised Internet Stream Protocol (ST-2),’’ Proc. *2nd International Workshop on Network and Operating System Support for Digital Audio and Video*, (Nov. 1991).

47. J. C. Pasquale, G. C. Polyzos, E. W. Anderson, K. R. Fall, J. S. Kay, V. P. Kompella, S. R. McMullan, and D. Ranganathan, ‘‘Network and Operating System Support for Multimedia Applications,’’ Technical Report CS 91-186, University of California, San Diego, La Jolla, CA (March 1991).

48. J. C. Pasquale, G. C. Polyzos, E. W. Anderson, and V. P. Kompella, ‘‘The Multimedia Muticast Channel,’’ Proc. *Third International Workshop on Network and Operating System Support for Digital Audio and Video*, (November 1992).

49. P. V. Rangan, H. M. Vin, and S. Ramanathan, ‘‘Designing an On-Demand Multimedia Service,’’ *IEEE Communications Magazine* **30**(7), pp. 56-64 (July 1992).

50. V. J. Rayward-Smith, ‘‘The Computation of Nearly Minimal Steiner Trees in Graphs,’’ *Intl. J. Math. Educ. Sci. Tech.* **14**(1), pp. 15-23 (1983).

51. S. Sabri and B. Prasada, ‘‘Video Conferencing Systems,’’ *Proceedings of the IEEE* **73**(4), pp. 671-688 (April 1985).

52. H. Santoso and S. Edida, ‘‘Transport Layer Multicast: An Enhancement for XTP Bucket Error Control,’’ *IFIP Transactions C (Communication Systems),* **C-14** pp. 333-349 (1993). (High Performance Networking IV. IFIP TC6/WG6.4 Fourth International Conference, Liege, Belgium, 14-18 Dec. 1992).

53. N. Shacham and P. McKenney, ‘‘Packet Recovery in High-Speed Networks Using Coding and Buffer Management,’’ Proc. *IEEE INFOCOM ’90*, pp. 124-131 (June 1990).

54. N. Shacham, ‘‘Multipoint Communication by Hierarchically Encoded Data,’’ Proc. *IEEE INFOCOM ’92*, pp. 2107-2114 (May 1992).

55. W. D. Sincoskie, ‘‘Video On Demand,’’ *IEEE GLOBECOMM ’90*, (1990).

56. J. Sutherland and L. Litteral, ‘‘Residential Video Services,’’ *IEEE Communications Magazine* **30**(7), pp. 36-41 (July 1992).

57. H. Tode, Y. Sakai, M. Yamamoto, H. Okada, and Y. Tezuka, ‘‘Multicast Routing Algorithm for Nodal Load Balancing,’’ Proc. *IEEE INFOCOM ’92*, pp. 2086-2095 (May 1992).

58. C. Topolcic, Ed., ‘‘Experimental Internet Stream Protocol, Version 2 (ST-II),’’ RFC 1190, Network Information Center, SRI International, Menlo Park, CA (October 1990).

59. L. Trajkovic and S. J. Golestani, ‘‘Congestion Control for Multimedia Services,’’ *IEEE Network* **6**(5), pp. 20-26 (September 1992).

60. J. S. Turner, ‘‘Design of a Broadcast Packet Switching Network,’’ *IEEE Transactions on Communications* **36**(6), pp. 734-743 (June 1988).

61. J. S. Turner, ‘‘Managing Bandwidth in ATM Networks with Bursty Traffic,’’ *IEEE Network* **6**(5), pp. 50-58 (September 1992).

62. D. C. Verma, H. Zhang, and D. Ferrari, ‘‘Delay Jitter Control for Real-Time Communication in a Packet Switching Network,’’ Proc. *TriComm ’91*, pp. 35-43 (April 1991).

63. D. Waitzman, C. Partridge, and S. Deering, ‘‘Distance Vector Multicast Routing Protocol,’’ RFC 1075, Network Information Center, SRI International, Menlo Park, CA (November 1988).

64. D. W. Wall, ‘‘Mechanisms for Broadcast and Selective Broadcast,’’ STAN-CS-82-919, Computer Science Department, Stanford University (1982).

65. G. K. Wallace, ‘‘The JPEG Still Picture Compression Standard,’’ *Communications of the ACM* **34**(4), pp. 30-44 (April 1991).

66. B. M. Waxman, ‘‘Routing of multipoint connections,’’ *IEEE J. on Selected Areas in Comm.* **6**(9), pp. 1617-1622 (Dec 1988).

67. W. De Zhong, Y. Onozato, and J. Kaniyil, ‘‘A Copy Network with Shared Buffers for Large-scale Multicast ATM Switching,’’ *IEEE/ACM Transactions on Networking* **1**(2), pp. 157-165 (April 1993).