An implementation of a gateway for hierarchically encoded video across ATM and IP networks

Jean-michel Robinet, Yuhang Au, and Anindo Banerjea
Electrical and Computer Engineering
University of Toronto
CANADA M5S 3G4
Email: {jrobinet, banerjea, yau}@comm.utoronto.ca

Abstract

This paper describes an implementation of an application level gateway for connecting adaptive applications using hierarchical encoded video across ATM and IP networks. The gateway participates in RSVP and ATM signaling. The signaling information, as well as local information about processing load, is used by the receivers to decide the number of layers to join, and by the sources to fine tune the bitrates of the layers to the available capacities. Our approach pushes layering related complexity to the edge of the network, and allows us to use standard ATM UNI and RSVP signaling. The gateway participates in a modified session directory (SDR) protocol, to learn the addressing information necessary to perform signaling translation, and to enable layered sessions to be visible across the IP/ATM boundary. By considering all aspects of the problem, especially session directory issues and dynamic bandwidth selection for the layered hierarchy, we have implemented a system that is much more complete than any of the previous prototypes of layered multicasting. This paper describes the implementation experience and presents some measurements of the performance of the gateway.

Keywords

Layered multicast, Internet Protocol (IP), Asynchronous Transfer Mode (ATM), Gateway, IP/ATM interoperation.

The original version of this chapter was revised: The copyright line was incorrect. This has been corrected. The Erratum to this chapter is available at DOI: 10.1007/978-0-387-35388-3_42

H. R. van As (ed.), *High Performance Networking*

1 INTRODUCTION

In this paper, we consider multicast video applications that use *hierarchical encoding* to handle network heterogeneity, using signaling protocols to probe the network for available capacity. Multicasting is a powerful network abstraction to support point-to-multipoint and multipoint-to-multipoint communication. Every packet sent to a multicast group is delivered by the network to *all* the receivers of the group. Both Internet Protocol (IP) and Asynchronous Transfer Mode (ATM) networks support multicasting, since it is more efficient than multiple point-to-point connections for the same purpose.

Network environments are heterogeneous by nature. This heterogeneity comes from many sources such as link capacity, end stations processing power and display resolution, network protocols and level of Quality of Service (QoS) support. Heterogeneity is especially problematic for multicast applications, since the receivers may not agree on the data rate that they want to receive, or the protocol to use to signal QoS requirements to the network.

Shacham (Shacham. 1992) has proposed multicast transmission of layered video as a solution to data rate heterogeneity. The data is encoded into a low resolution base stream and a series of enhancement streams. This allows different receivers to receive data from the same source at different rates, simply by subscribing to different numbers of multicast streams, identified by multicast address in IP or multicast virtual circuit (VC) in ATM.

The problem of protocol heterogeneity can be solved by mandating a universal protocol, such as IP. However, this requires unnecessary translations, and prevents applications from taking advantage of the native signaling, when the communication is restricted to a single signaling domain. An alternative approach is to perform translation of signaling and QoS semantics at the network boundaries. This approach allows applications in different networks, such as IP and ATM, to communicate with each other, while continuing to take advantage of the native signaling or resource reservation protocols locally. A possible future network scenario involving this approach is a video on demand system, with the sources being video studios directly connected to a high speed ATM backbone, some high end (perhaps HDTV) clients connected directly to the ATM network, and some lower end clients connected over a slower speed IP-based network. There is no need to involve IP software overheads in the data transmission path from the source to the HDTV clients, but at the same time IP provides access to a more heterogeneous and broader set of clients. This is the model we assume for the rest of the paper.

The paper proceeds as follows: Section 2 presents related work and motivation. Section 3 provides some background on application and session layer issues. Section 4 describes the gateway implementation. Section 5 discusses the implementation, the testbed, and the performance of the gateway. We conclude the paper with a summary in Section 6.

2 RELATED WORK AND MOTIVATION

This paper describes an implementation of a gateway to allow adaptive layered video applications to communicate across different protocol domains, specifically IP and ATM networks. Our work is related to prior work on Heterogeneous MultiCast (HMC) (Sudan *et al.* 1997) with a IP/ATM gateway for layered video, but differs in certain key areas that we describe below. In addition, the applications considered in HMC are nonadaptive. Our work is also similar in many respects to Receiver-driven Layered Multicast (RLM) (McCanne *et al.* 1996), which uses receiver adaptation based on packet loss to select the number of layers to receive, but is restricted to an all IP environment. Thus, many of the new problems we face are related to the handling of adaptive applications in a multiprotocol signaling environment.

Both of the above systems only allow the receiver to select from a static set of layers, based on network bandwidth availability. If the set of bandwidths being transmitted is not well tuned to the set of bandwidths available, these systems perform poorly. We believe that it is unreasonable to require the user to know *a priori* the set of network and receiver capacities required, and configure the sources with the correct set of layer bitrates. We implement feedback mechanisms across the network, so that the bitrates transmitted on the layers from the sources are adapted to the set of network bandwidths and receiver capacities dynamically. This functionality is orthogonal to the functionality provided by SCUBA (Amir *et al.* 1997), where the information from different sources is dynamically mapped on to a static set of layers in response to receivers expressing interest in particular sources. Our adaptation model also allows us to push all layering related complexity to the application layer, using standard User to Network Interface (UNI) (UNI 3.0. 1993) signaling within the ATM network and ReSerVation Protocol (RSVP) (Zhang *et al.* 1993, Braden *et al.* 1996) signaling within the IP network, instead of modifying the protocols to handle layering as suggested in HMC.

Previous work in layered multicast has neglected the problem of session advertisement. The session directory information provided by SDR (Handley *et al.* 1995) allows a receiver to learn of the existence of a session, and provides sufficient information (such as multicast addresses and port numbers in IP) so that the receiver can join the session. In an IP/ATM layered environment, the problems of how layered sessions are specified in session advertisement messages, how receivers in one domain learn about sources in the other domain, or how to reconcile the multipoint-to-multipoint nature of IP multicast with the point-to-multipoint nature of ATM virtual circuits (VCs), have not been previously dealt with. For example, in Sudan's work the gateway must be manually configured with a *static* mapping between layers, IP multicast addresses, ATM addresses, participant identifiers (independent of domain specific addresses), and traffic descriptors for the layers (in the ATM to IP case).

Our work addresses these problems in a more general way, by extending

the Session Directory (SDR) protocol to handle layered sessions, ATM addresses, and the point-to-multipoint nature of ATM multicast. The gateway participates in the session directory protocol in both domains and performs session advertisement translation. These issues are particularly important for our system, since the applications are adaptive and use the session directory protocol to advertise *changes* in the structure of layers transmitted by the sources, in response to feedback about network and receiver heterogeneity.

The approach of RLM is targeted to an all IP environment, and does not depend on the existence of signaling protocols (e.g., RSVP). If RSVP signaling is available, receiver adaptation can be much more stable than in RLM, while at the same time responding very quickly to available capacity. In RLM, stability and response time must be traded off against each other, depending on complex interactions between the duration of congestion and the time the network takes to prune a connection after a receiver leaves. Since prune times of currently deployed multicast routing and group membership protocols in the Internet are long (Gupta *et al.* 1997), a method like RLM must necessarily have poor response times to be stable(McCanne *et al.* 1997). In addition, using RSVP and ATM UNI signaling allows us to provide QoS guarantees.

Finally, Sudan *et al.* do not consider the case of connecting an ATM network to an IP network without RSVP support. We handle the case where RSVP support is not available, by using a loss based mechanism adaptation similar to RLM. We help to handle some of the stability and latency problems raised by this approach, by performing priority service based on layer number at the gateway, to concentrate loss due to congestion on the highest layers. This provides a clearer signal to the receiver to adapt, while reducing the impact of the loss on the received video quality.

Our applications are based on Zakhor's software codec (Taubman *et al.* 1994), which is capable of encoding digital video into a very large number of fixed size sublayers. Previous work (Banerjea *et al.* 1997) showed how these can be combined dynamically to a smaller number of transport layers, allowing the bitrate to transport layer mapping to be adapted by the source in response to network feedback. We have added signaling support (RSVP over IP and UNI over ATM) and adaptation mechanisms to the above, to create adaptive layered network conferencing and video server applications.

3 APPLICATION AND SESSION LAYER PROCEDURES

In this section, we briefly summarize the feedback algorithms used by the receiver and the source to adapt to the network capacity and receiver load, and the signaling and session directory functionality required to handle layered applications on IP and ATM. Details can be found in (Yau *et al.* 1997).

Our layered application uses three different adaptation mechanisms, which work over different time scales and distances. The first is adaptation to receiver load. Since our application performs decoding of the layered video in software,

the number of layers it can receive depends on the CPU load. The receiver monitors the time to process a frame of video to determine the number of layers it is able to process. This feedback is entirely local to the receiver's machine and involves the shortest time interval.

The application uses network signaling mechanisms (RSVP over IP and UNI over ATM) to determine network bandwidth availability. When the receiver load allows, the receiver probes the network with a reservation attempt to add the next layer. There is no danger of unstable behavior, since the layer is only added if its addition cannot cause congestion. This feedback involves the network and occurs over slightly longer intervals than CPU Monitoring.

In the absence of signaling mechanisms (for example, IP networks without RSVP) the application uses a loss based feedback mechanism similar to RLM for network adaptation. However, this leaves us with the problems of stability and responsiveness mentioned before. We return to this issue in Section 4.

The receivers provide feedback to the session originator about the link capacities and receiver processing powers of the active receiver and network environment. The originator adjusts the bitrates being transmitted on each layer of the encoding hierarchy accordingly, and advertises the changed layer hierarchy information to the sources using the session directory protocol. The sources modify their transmitted hierarchies accordingly. This feedback involves both sources and receivers. For scalability reasons, it occurs over the longest time interval. Note, however, that this does not imply the system is not responsive, since the receiver adaptation (compared to RLM) is fast. The system adapts slowly to changes in the receiver set, and in comparison, all previous approaches did not adapt in this respect at all.

In the IP network, a receiver changes its video quality by joining or leaving multicast groups using the Internet Group Management Protocol (IGMP). An IP multicast group is a many-to-many communication abstraction, so the receiver receives data from all sources transmitting to it. The receiver must also use RSVP to specify its QoS requirements, and make reservations for the layer. By waiting till the RSVP reservation request is successful before adding the next layer, the receiver can ensure that the network is not overloaded.

In the ATM domain, a receiver changes its fidelity by joining or leaving a multicast Virtual Circuit (VC). In ATM 3.x, the join request can only be initiated by the source; hence, the receiver must send a request to the source to added to a specific layer. An ATM multicast VC is strictly one-to-many, so the ATM receiver must know the Service Access Point (SAP) address of each source in order to send the requests to join. The Session Directory (SDR) protocol must be modified to carry this source specific information. We accomplish this by adding a ATM_SRC message, which is transmitted by each source on the ATM network, and carries the address information.

We also extend the SDR message to carry layering information. This message is periodically retransmitted by the session originator, and conveys information about the number of layers, and the multicast address, and bitrate

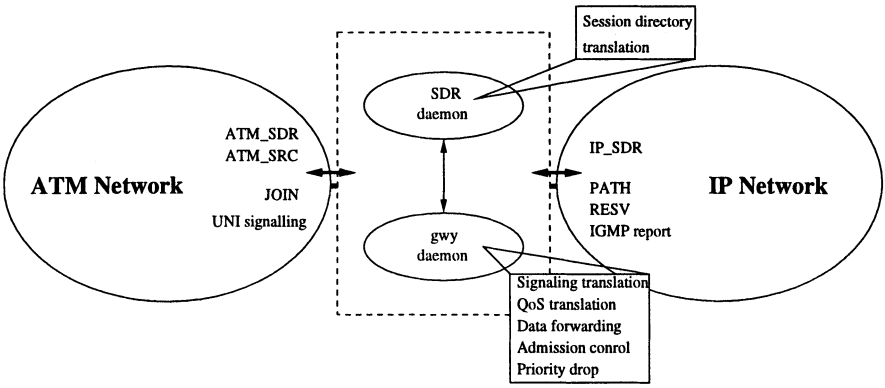

Figure 1 Gateway design

associated with each layer. Based on feedback received from the receivers, the originator transmits a changed SDR message. On receiving the new SDR message, sources adapt their transmitted streams; receivers can detect and adapt to the change in the data stream.

4 GATEWAY PROCEDURES

To maintain transparent interdomain connectivity, the gateway performs the following tasks:

- Translation of the connection setup messages.
- Translation of the traffic parameters.
- Translation of the session directory messages.
- Admission control.
- Data forwarding.
- Priority service.

As shown in Figure 1, two daemons, the SDR daemon and the gateway daemon, are responsible for the above tasks. The SDR daemon is only responsible for translating session advertisement messages from one domain to the other one. The gateway daemon is responsible for the other tasks.

4.1 Signaling translation

The gateway has to translate the signaling messages and map the traffic parameters between the two domains. In the following subsection, we consider the case of an IP source and an ATM receiver, and after that, the case of an ATM source and an IP receiver.

IP to ATM: The gateway learns of the existence of a new session from the session directory (SDR) protocol. It joins the multicast groups in order to

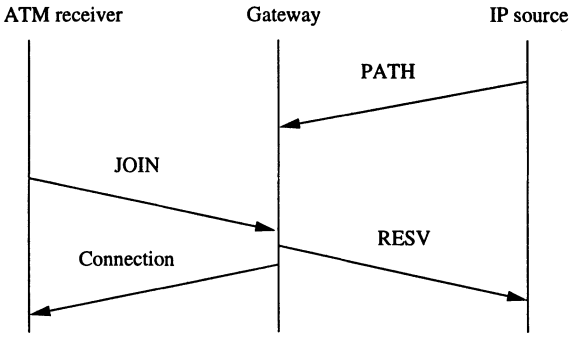

Figure 2 End-to-end signaling: IP to ATM case

receive PATH messages for each group and hence learn the existence of an IP source. The gateway also uses SDR to announce the session on the ATM network. Figure 2 shows how the gateway translates the messages to have a coherent end-to-end signaling.

The gateway learns of a new source when it receives a new PATH message on the base layer multicast address. After performing local admission control tests, the gateway behaves like a new source on the ATM network, and advertises its address in an ATM_SRC message. It also adds the layer by creating a multicast VC (with currently no receivers) for this layer with the QoS translated from the PATH message, and updating the database.

When an ATM receiver wants to receive a particular source and layer, it sends a JOIN request to the gateway. On receiving such a JOIN, the gateway looks up the corresponding IP source from the database, and makes a reservation, using a RESV message to the IP source. Finally, the gateway adds the receiver to its ATM multicast VC, and updates the forwarding table.

In case RSVP signaling is not present, the gateway will not be able to depend on PATH messages and NULL PATH messages to learn of the joining and leaving of IP sources. In this case, it joins all multicast groups, and uses the presence of data to learn about sources. On learning of new sources, it advertises them using ATM_SRC messages. It uses timers to learn about sources leaving. In order to assist the receiver in performing network adaptation, the gateway performs priority service on the layers of the same session. This is further described in subsection 4.3.

ATM to IP: When a session is created by an originator on the ATM network, the gateway learns about it from an SDR message. The gateway SDR daemon acts as an originator on the IP side, by choosing IP multicast addresses for the session and sending SDR messages. Figure 3 shows how the gateway translates the signaling messages, when the source is in the ATM network and the receiver is in the IP network.

The gateway learns the existence of a new ATM source by receiving an ATM_SRC message. The gateway sends as many JOIN messages as layers to the ATM source. It translates the QoS learned from the ATM signaling, performs the local admission control tests, and sends as many PATH messages

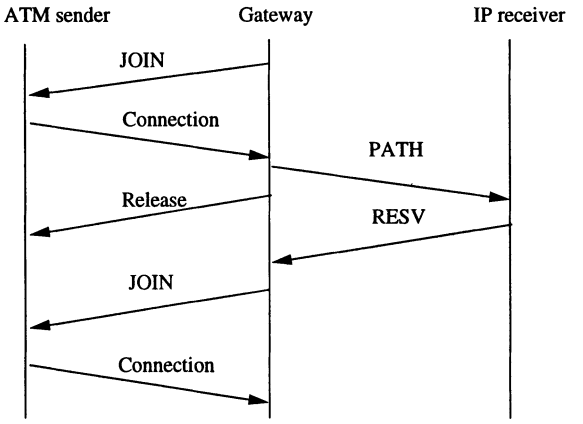

Figure 3 End-to-end signaling: ATM to IP case

as existing layers to the corresponding multicast groups. The gateway then tears down the multicast VCs, since no receivers exists yet. For each ATM source, the gateway is seen as a different IP source, distinguished by source UDP port number, by the receivers in the IP domain.

The gateway learns about the first IP receiver for a layer by receiving a RESV message on a multicast group. After performing local admission control, the gateway sends a JOIN message to the ATM source requesting the particular layer. For subsequent IP receivers, no new state needs to be setup at the gateway, and the RESV message is not even forwarded to the gateway process by RSVP unless the reservations change.

If the gateway does not support RSVP, the signaling is very simple. The gateway learns the existence of a new ATM source by receiving an ATM_SRC message. The gateway waits for an IP receiver to show interest by sending an IGMP report message. Once a receiver exists, the gateway sends a JOIN message to the ATM source, updates the database and forwarding table, and forwards the data to the IP side. The gateway also learns about receivers leaving using from the IGMP protocol, and deletes the corresponding ATM connections when all receivers for a given multicast address leave.

4.2 Data forwarding

To forward the data streams from the senders to the correct set of receivers, the gateway must identify: (1) the sender, (2) the layer, and (3) the receivers.

When a packet is received on the IP side, packet header contains the IP multicast address, the UDP destination port, the IP source address, and the UDP source port. Based on this four fields, the gateway looks up a multicast VC handle from the forwarding table and transmits the data. When a packet is received on the ATM side, the gateway uses the multicast VC handle to look up the addresses to put on the outgoing packet header in the forwarding table.

Note that the forwarding state of the gateway is extremely simple. Packets are actually queued in the receive socket buffers (in the kernel). The gateway daemon processes packets one at a time, reading from one socket and immediately transmitting to another. In order to implement priority service, it just associates a priority with each socket, and serves them in priority order. In order to implement controlled discarding (such as RED In Out (RIO)), it uses IOCTL calls to read the socket buffer length. This results in a very simple and robust gateway design.

4.3 Participation in adaptation procedures

Where the gateway acts as a receiver into a network, either ATM or IP, it ensures that it adds and drops layers in order of the layer number. This functionality is redundant, since the receivers exhibit this property individually. However, it helps to stabilize the system against the effect of accidental use of the multicast address space of a layered session by an unrelated receiver, or against incorrect behavior on the part of layered receivers.

The gateway is responsible for forwarding feedback messages from the receivers to the originator of the session across the ATM/IP boundary. This allows the originator to get the full picture about the set of receiver and network capacities, so it can make its decision about changing the transmitted hierarchy.

When the originator decides to change the layer hierarchy, it transmits a new SDR message. These messages are translated and forwarded from one network to the other by the gateway. This allows sources on both sides of the network to learn about the new hierarchy. The sources then start to transmit data according to the new hierarchy. On the IP network, they start sending the data according to the new scheme right away, and also transmit PATH messages to notify the network and the receivers about the changed resources needed. If resources are available, the reservations are modified by RESV messages from the receivers without a need to tear down the current distribution tree. On the ATM network, the sources tear down the current distribution tree and wait for new requests to join from the receivers. The new multicast VCs are set up according to the modified bitrate requirements.

The gateway participates in this adjustment of the distribution trees. The signaling actions taken by the gateway as part of the adaptation process are all the result of messages from the network sent by the receivers (e.g., RSVP RESV messages or ATM JOIN request), or the senders (e.g., RSVP PATH messages or ATM_SRC messages). The gateway never initiates any adaptive action, therefore its state is simple. For example, no timers need to be kept.

However, in the absence of RSVP, the gateway must detect the coming and going of IP sources by using timers, since no explicit notification is provided by the network. The gateway also assists the adaptation process at the receiver by performing priority service of packets, based on layer number. The receiver performs loss based load shedding similar to RLM in the absence of RSVP.

Priority service concentrates the loss onto the highest layers in the case of congestion. This has two advantages. Firstly, the receiver can compute the loss rate for each layer separately, and this will be much higher than if the same loss was spread across all the layers. This gives a clearer signal to the receiver to drop the highest layer. For instance, the drop trigger thresholds can be set higher, so a few accidental packet drops do not cause the receiver to back off. This makes the adaptation more stable while still remaining responsive. The second advantage is that if the loss is concentrated on the highest layer, the visual quality of the video is less effected than if the same loss is spread across all the layers.

5 IMPLEMENTATION STATUS AND PERFORMANCE

The current implementation of the gateway at the user level has several limitations. The number of sessions that can be simultaneously handled is severely limited by the number of sockets available to a single UNIX process. The gateway has also not been optimized for high throughput. Finally, the support for local admission control of the gateway resources and controlled dropping of low-priority packets under overload is not complete. The version of the gateway that we used for the measurements implements simple priority service, with the priority based on layer number and QoS support.

The gateway runs on top of the socket interface, and binds virtual circuits on the ATM network and multicast groups on the IP network to sockets. Thus, for example, for a single session with seven layers, one ATM sender, one IP sender, and arbitrary number of ATM and IP receivers, the gateway daemon uses twenty three sockets. Since a UNIX process has access to a maximum of 64 sockets, this limits rather severely the size and number of the sessions we can create. One possible solution to this problem lies in moving the gateway implementation into the kernel. This would reduce the memory copy overhead as well, leading to an improvement in the maximum throughput capacity. This may be appropriate for a stand alone machine, with the sole function of connecting layered applications across ATM and IP.

The following experiments were conducted on a network testbed consisting of an ATM LAN and a 10 Base-T Ethernet LAN. The ATM LAN consists of a Fore ASX-200WG switch, with multiple UltraSparc workstations connected using Fore SBA200 ATM interface cards, and running ForeThought 4.1.0 driver software that implements an application programming interface to the ATM UNI 3.0. The machines on the Ethernet LAN run the ISI implementation of RSVP on Solaris 2.5.1. The ATM and Ethernet LANs are connected by a Sparc20 workstation running Solaris 2.5.1 with ATM and Ethernet network interface cards. This workstation runs the gateway software.

Our implementation is not optimized for fast forwarding performance. For example, it would be possible to move the forwarding function into the kernel to lower memory copy overheads and improve throughput. Hence, the first

experiment we performed is to test the maximum data forwarding capacity of the gateway. We found that even with multiple sessions running simultaneously, the gateway is capable of forwarding data up to the full capacity of the Ethernet without overload. The signaling performance is also not degraded at this high load, indicating that the gateway has sufficient CPU cycles to spare.

The next set of experiments presented explore the time to add a layer. Our receivers use signaling to probe the network for available capacity. In the worst case, this requires one round trip for each layer; the latency of this process is of concern.

Figure 4 shows the round trip latency to add a layer with a single source on the ATM network, and a single receiver on the IP network. The round trip is measured from when the RESV message is transmitted by the receiver, to when the first data packet arrives at the receiver. It is important to note that the case of a single receiver is the worst, since it requires a full round trip for each layer. A second receiver on the Ethernet would observe much less latency; it would start receiving the data as soon as it bound a socket to the multicast address, as the data is already being transmitted on the LAN.

The X-axis shows the layer number N, while the Y-axis shows the time to add the Nth layer averaged over eighty repetitions of the experiment, and the 95% confidence intervals. We note that the average time to join a layer increases with the number of layers being added. Figure 5 shows the IP to ATM case. Table 5 shows the breakup of the round trip time into its major components, shown as a (MAX, AVERAGE, MIN) triple in units of milliseconds. These components are:

- RSVP processing: the delay in the receiver host from when the decoder decides to add a layer to when the RESV message is sent. This processing is performed on the receiver, by the RSVP library and daemon code. The processing time of layer 1 is greatest, because an RSVP thread is created.
- RESV processing: the delay in the gateway machine from when the gateway receives a RESV message to the time the multicast VC is setup. This involves setting up the internal tables, sending a JOIN message to the ATM source, and then performing ATM UNI signaling.
- Data sent: the delay from the time the VC is setup to the time the first packet arrives at the gateway. A major component of this delay is the time spent waiting for the next round of transmission. In addition, for layer 1 the source has to wake up a thread before data is transmitted.
- Forwarding: the delay in the gateway from when the data is read to when the data is transmitted. This is not shown in the table since it is almost fixed at 1 ms.
- Data read: the delay at the receiver, from when the data arrives in the kernel buffer to when the application reads it from the kernel buffer.

We note that the major components of the delay, as well as of the variability, are the time to send the data from the source and the time to receive the data

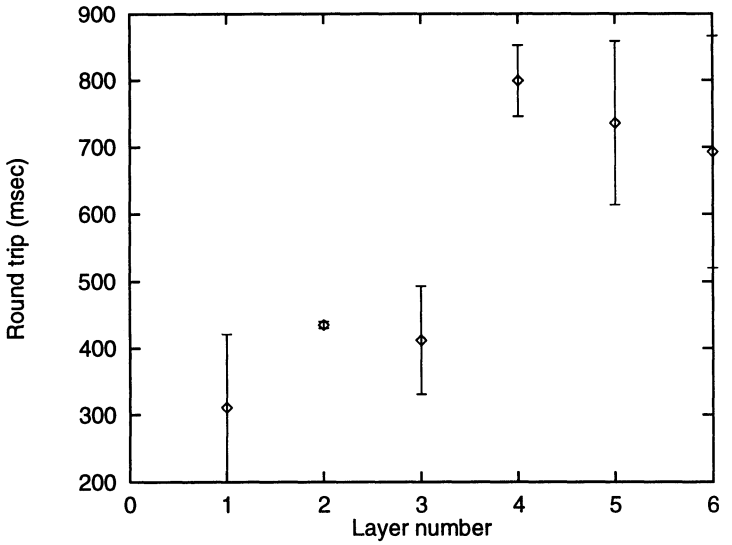

Figure 4 Round trip latency for adding a layer: ATM to IP

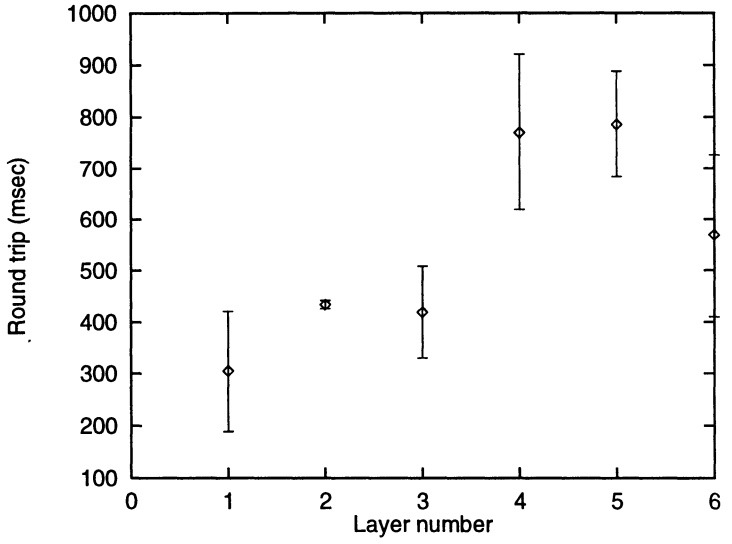

Figure 5 Round trip latency for adding a layer: IP to ATM

at the receiver. At the source, this time is the time from when the ATM connection has been successfully set up to when the next frame of data is due to be transmitted. Since the encoder places 4 frames of data into a single packet and the frame rate for this experiment was ten frames per second, the inter-transmission time is 0.4 seconds. We see that the send time is spread between 0 and 0.4 seconds as expected. At the receiver end, the time to receive the packet is also similarly affected by the time to process a frame,

Table 1 Break up of time to add a layer; ATM to IP case

Layer	*RSVP proc.*	*RESV proc.*	*data sent*	*data read*
0	(18/23/81)	(37/41/64)	(25/232/409)	(0.3/0.4/0.7)
1	(3/6/10)	(36/43/140)	(170/373/382)	(0.2/0.4/0.7)
2	(3/5/11)	(36/43/173)	(101/135/484)	(210/234/290)
3	(3/5/7)	(124/149/217)	(308/373/402)	(230/271/332)
4	(3/5/7)	(55/115/196)	(286/375/418)	(2/260/312)
5	(3/5/15)	(35/72/123)	(238/358/385)	(2/317/331)

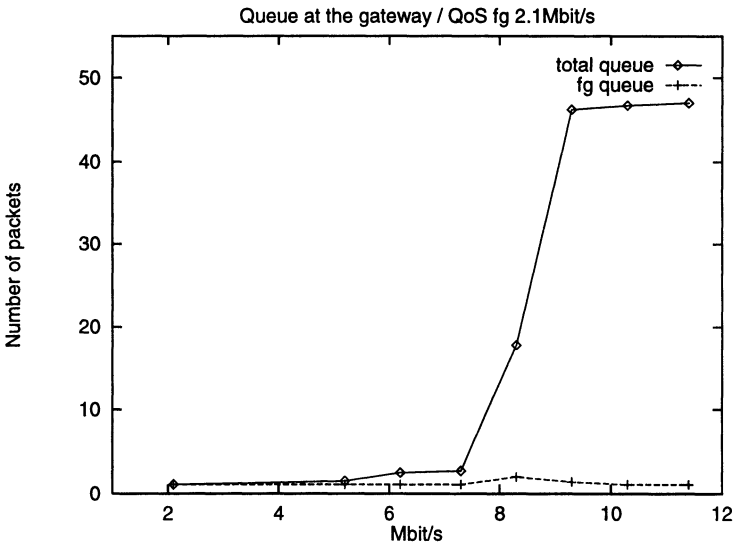

Figure 6 Gateway queue size

since while the decoder is processing the previous frame it does not check if new data has arrived. When the decoding threads become idle, the thread which is blocked on the receive gets a chance to run and retrieve the data from the network. This time increases with layer number and saturates near 0.4, since the receiver does not take on more layers than it can handle. Thus, the major element affecting the time to add a layer is the unit of transmission, which is this case happens to be four video frames.

In order to maintain stability, the receiver has to perform some measurements of the last layer added before it can safely add the next layer. The intervals of time for this measurement increase as the receiver becomes loaded, to avoid oscillatory behavior. These intervals are of the order of seconds, which is quite large compared to the round trip time to add a layer. Thus, the signaling latencies are acceptable for the applications under study, and optimizing the signaling overhead by using 'bundled VCs' as proposed in HMC is not necessary.

Figure 6 shows the queue lengths for two simultaneous streams. The foreground stream is a 2.1 Mbps stream, carried with QoS support, using ATM and RSVP signaling. The background stream is a best effort stream with bitrate increasing from zero to nine megabits per second. The X-axis shows the total bitrate of the combined traffic, while the Y-axis shows the queue length for foreground queue (diamond symbols) and total queue length (plus symbols). We note that the foreground queue remains small, even when the sum of bitrates exceed the capacity of the Ethernet (10 Mbps). This happens because of two reasons. Firstly, since the foreground traffic is shaped at the entrance to the ATM network, a burst of traffic is never injected in this stream. Secondly, the QoS stream is protected from the bursty behavior of the best effort background stream by the per stream queueing and priority service in the gateway (as well as in the switches, etc). This experiment shows the effectiveness of the priority service and QoS mechanisms; even under overload conditions, all the queueing delay and loss are concentrated on the low priority stream.

6 CONCLUSION

We have presented an implementation of a gateway for connecting layered applications across ATM and IP networks. This implementation improves on previous research by extending the feedback algorithms for adaptation all the back to the source. This allows the source to select the correct number of layers, and the bitrates for each layer, to accommodate the current network and receiver capacities. Our adaptation model has three different control loops, one limited to the receiver, a longer one involving the receiver and the network, and a third (longest) control loop involving the source, network and receiver. The combination of the three gives us allows the feedback to be scalable, stable, and still responsive. The gateway participates in the source adaptation by translating the feedback messages, and updating the layered hierarchy advertisements (in the session directory protocol) when they change.

We also extend previous research by considering the addressing and naming translation issue. The extension of the session directory protocol to the ATM environment allows us to compensate for the lack of a multipoint-to-multipoint abstraction on the ATM network, since the receivers can find out about source information from the session directory. The gateway participates in the session directory protocol, to become aware of new sessions and sources, to advertise them on the other side, and to translate the addresses, port numbers and other information that the receivers need to join a session. The gateway acts as a proxy for each IP source on the ATM network, and acts on behalf of IP receivers on the ATM side. Instead of using preconfigured tables for the address translation, the gateway exchanges the necessary information through the session directory protocol.

In another departure from previous work, our applications take care of the complexity of layering, such as ensuring that resources are not wasted for

higher layers when lower layers are not available, at the edge of the network. This simplifies the network from the point of view of network scalability. It also allows us to perform our experiments with a standard ATM and RSVP installation. Our experiments show that the signaling is not a major factor in the latency of the receiver based adaptive control.

We deal with the case when RSVP is not present, by using a loss based mechanism similar to RLM. In this case, the receiver responds to congestion by detecting increased packet loss and dropping layers. Since the gateway is itself likely to be a bottleneck (going from a high speed ATM to a low speed Ethernet network), we concentrate loss at the gateway on to the highest layers by performing priority service. This gives the clearest feedback to the receivers, since the percentage of lost packets on the highest layer is maximized. It also minimizes the effect of the loss on the visual quality of the video. Finally, this action, being taken by an application level entity, does not cause any increase in the network complexity or a violation of layering.

In conclusion, we consider all aspects of the layered multicast problem at the ATM/IP gateway. We contend that this makes our system more usable, more complete, more flexible, and more stable than previous prototypes of layered multicasting.

REFERENCES

Amir, E. McCanne, S. and Katz, R. (1997) Receiver-driven Bandwidth Adaptation for Light-weight Sessions, *ACM Multimedia,* pp. 415-426, Seattle, WA.

Banerjea, A. Tan, W-T. and Zakhor, A. (1997) Evaluation of a Layered 3-D Subband Compression Scheme for Multicasting Cine-Angiograms across Heterogeneous Networks, *Proc. SPIE Medical Imaging'97,* pp. 265-276, Newport Beach, CA.

Braden, B. Zhang, L. Berson, S. Herzog, S. and Jamin S. (1996) Resource Reservation Protocol (RSVP) – Version 1 Functional Specification, *Internet Draft,* Work In Progress.

Gupta, A. and Speer, M. (1997) Measuring the performance of IP multicasting, *Sun Microsystems Laboratories Document, SMLI-97-0048,* Palo Alto, CA.

Handley, M. and Jacobson, V. (1995) SDP: Session Description Protocol, *Internet Draft,* Work In Progress.

McCanne, S. Jacobson, V. and Vetterli, M. (1996) Receiver-driven Layered Multicast, *Proceedings of SIGCOMM'96,* pp. 117-130, Stanford, CA.

McCanne, S. Vetterli, M. and Jacobson, V. (1997) Low-complexity Video Coding for Receiver-driven Layered Multicast, *IEEE Journal on Selected Areas in Communications 16,* 6, pp. 983-1001.

Shacham, N. (1992) Multipoint Communication by Hierarchically Encoded Data, *Proceedings of IEEE INFOCOM'92,* pp. 2107-2114, Firenze, Italy.

Shenker, S. and Breslau, L. (1995) Two issues in Reservation Establishment, *Proceedings of SIGCOMM'95*, pp. 14-26, Cambridge, MA.

Shenker, S. Partridge, S. and Guerin, R. (1997) Specification of Guaranteed Quality of Service, *Internet Draft*, Work In Progress.

Shenker, S. and Wroclawski, J. (1997) General Characterization Parameters for Integrated Service Network Elements, *Internet Draft*, Work In Progress.

Sudan, M. and Shacham, N. (1997) Gateway Based Approach For Managing Multimedia Sessions over Heterogeneous Domains, *Proceedings of IEEE INFOCOM'97,* Kobe, Japan.

Taubman, D. and Zakhor, A. (1994) Multirate 3-D Subband Coding of Video, *IEEE Transactions on Image Processing 3,* 5, pp. 572-588.

ATM User Network Interface Specification, Version 3.0. (1993) *Prentice Hall.*

Wroclawski, J. (1996) Specification of the Controlled-Load Network Element Service, *Internet Draft*, Work In Progress.

Yau, Y. Robinet, J-M. and Banerjea, A. Session and application layer issues for layered multicasting, *document in preparation.*

Zhang, L. and Deering, S. Estrin D. Shenker, S. and Zappala D. (1993) RSVP: A New Resource ReSerVation Protocol, *IEEE Communications Magazine 31*, 9, pp. 8-18.

7 BIOGRAPHY

Jean-michel Robinet received a B.S. in Computer Engineering from the Florida Institute of Technology in August 1996. He is currently a M.A.Sc. candidate at the department of Electrical and Computer Engineering at the University of Toronto. His graduate research is on multimedia networking in heterogeneous environments.

Yuhang Au received his B.S. in Computer Engineering from the University of Kansas in August, 1996. He is currently a M.A.Sc. candidate at the Department of Electrical and Computer Engineering at the University of Toronto. His graduate research is on video conferencing in heterogeneous networked environments.

Anindo Banerjea received a B. Tech degree in computer science and engineering from the Indian Institute of Technology, Delhi in 1989, and a Ph. D. in Computer Science from the University of California at Berkeley in 1994. His dissertation topic concerned the problem of providing fault recovery in networks with guaranteed performance (real-time) services. He was also involved in the design and implementation of the Tenet realtime protocol suite, which provides real-time services on wide area inter-networks. As Assistant Professor at the Department of Electrical and Computer Engineering at the University of Toronto, Anindo is continuing his research in multimedia networking. He is interested in heterogeneous networking environments, integration of Asynchronous Transfer Mode (ATM) technology with the Internet, QoS sensitive multicast routing and real-time networked multimedia applications.