WELCOME TO THE GATHERING PLACE

Computer Networks and ISDN Systems 30 (1998) 431–441

COMPUTER NETWORKS and ISDN SYSTEMS

The Interactive Multimedia Jukebox (IMJ): a new paradigm for the on-demand delivery of audio/video

Kevin C. Almeroth [a,*], Mostafa H. Ammar [b,1]

[a] *Department of Computer Science, University of California, Santa Barbara, CA 93016-5110, USA*
[b] *Networking and Telecommunications Group, Georgia Institute of Technology, Atlanta, GA 30332-0280, USA*

Abstract

Straightforward, one-way delivery of video programming through television sets has existed for many decades. In the 1980s, new services like Pay-Per-View and Video-on-Demand were touted as the "killer application" for next-generation Internet and TV services. However, the hype has quickly died away leaving only hard technical problems and costly systems. As an alternative, and what we propose, is a new paradigm offering flexibility in how programs are requested and scheduled for playout, ranging from complete viewer control (true VoD), to complete service provider control (traditional broadcast or cable TV). In this paper, we describe our proposed jukebox paradigm and relate it to other on-demand paradigms. Our new paradigm presents some challenges of its own, including how to best schedule viewer requests, how to provide VCR-style interactive functions, and how to track viewer usage patterns. In addition to addressing these issues we also present our implementation of a jukebox-based service called the *Interactive Multimedia Jukebox* (IMJ). The IMJ provides scheduling via the World Wide Web (WWW) and content delivery via the Multicast Backbone (MBone). We discuss the challenges of building a functioning system and our ongoing efforts to improve the jukebox paradigm.

Keywords: WWW; MBone; Multicast; Video-on-demand

1. Introduction

Straightforward, one-way delivery of video programming through television sets has existed for many decades. In the 1980s, new services like Pay-Per-View and Video-on-Demand were touted as the "killer application" for next-generation Internet and TV services. However, the hype has quickly died away leaving only hard technical problems and potentially very costly systems. Even though VoD has been shown to be technically feasible, service providers have been hesitant to make the investment necessary for wide-scale deployment. Furthermore, almost all of the trials to date suggest VoD is too expensive and there is too little demand. What is needed, and what we propose, is a new paradigm offering flexibility in how programs are requested and scheduled for playout, ranging from complete viewer control (true VoD), to complete service provider control (traditional broadcast or cable TV). Furthermore, our proposed paradigm functions independent of the network topology. Both a cable-TV- and Internet-based jukebox service are possible. And with solu-

* Corresponding author. E-mail: almeroth@cs.ucsb.edu
[1] E-mail: ammar@cc.gatech.edu

0169-7552/98/$19.00
PII S0169-7552(98)00119-6

tions to the key problems facing each network — high quality delivery in the Internet, and bi-direction communication in cable TV systems — the jukebox paradigm could be developed into an attractive commercial service.

The jukebox paradigm we propose is based on the premise of allowing any viewer to watch any other viewer's requested program. Program requests are scheduled on one of a system's channels using a set of scheduling policies. Any viewer who wants to watch a program on a particular channel simply "tunes" to that channel. Content on each channel is delivered from a server to all viewers watching that particular channel.

In this paper, we describe the jukebox paradigm and relate it to other on-demand paradigms. We also describe some of the challenges in providing an on-demand program service. Of particular interest are issues like the best way to handle viewer requests, how to provide VCR-style interactive functions, and how to track viewer usage patterns. We also describe our efforts to prototype a jukebox-based service. The *Interactive Multimedia Jukebox* (IMJ) provides scheduling via the World Wide Web (WWW) and content delivery via the Multicast Backbone (M-Bone). We discuss some of the issues related to building the IMJ and our ongoing efforts in improving the jukebox paradigm.

The jukebox paradigm is an effort to bring together work in several research areas. One area is the scalable delivery of video-on-demand (VoD) service using multicast communication. True VoD, in which all viewers get their own resources, requires very large systems to provide adequate performance. These systems are expensive and provide few revenue opportunities for service providers. One solution is to batch multiple requests for the same program into a group and then service them using one audio/video stream multicast to all group members [1,2]. This solution was mainly proposed for cable-TV based infrastructures which provide many channels, a large numbers of customers, and broadcast-only communication. Within the Internet, the MBone [3] has been the focal point for developing multicast [4] and real-time protocols [5] for the scalable delivery of multimedia streams. Like the IMJ, related efforts are looking at extending the use of the MBone beyond applications like interactive conferencing and program broadcasts [6–9]. Finally, recent work has looked at integrating the services of the MBone and the Real-Time Protocol (RTP) into the the WWW. Several researchers are looking at various ways of using the MBone and multicast protocols to deliver WWW pages [10–13]. Other issues are based on the integration of WWW and MBone-style conferencing [14,15].

This paper is organized as follows. Section 2 describes our proposed jukebox paradigm and other related paradigms. Section 3 details our implementation of a jukebox prototype called the *Interactive Multimedia Jukebox* (IMJ). Section 4 lists several research issues related to the jukebox paradigm. The paper is concluded in Section 5.

2. Program scheduling paradigms

2.1. Background

Our jukebox paradigm is based on a hybrid of program scheduling paradigms ranging from newer proposals like *true* Video-on-Demand (VoD) and *near* Video-on-Demand to more traditional services like Pay-Per-View (PPV) and broadcast television [1,2,16,17]. Traditional television is based on the premise of delivering as many channels of programming. Viewer choice is the ability to switch between any of the available channels. Affecting what is shown on a particular channel is a slow process of feedback through program "ratings" followed by scheduling and programming changes based on evaluation of the ratings data by the broadcasters. PPV has attempted to give viewers additional choices, but fundamentally, the broadcasters still decide scheduling and timing. For a variety of reasons, PPV has never met the lofty financial goals set by many service providers.

Inherent in a discussion of video service paradigms are comparisons based on two factors. The first factor is the number of viewers who can watch a particular program stream. The second factor is a combination of how much viewer input is considered in program scheduling and whether the program schedule is developed in real-time or pre-arranged. Figure 1 shows the relationship between the paradigms mentioned so far. Existing television services like broad-

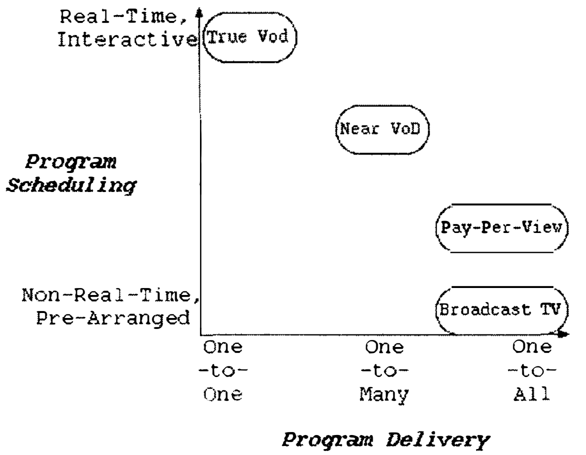


Fig. 1. The relative position of scheduling paradigms.

cast TV and PPV are located in the lower right of the graph. Users must plan their television watching habits around a pre-arranged schedule of programs. While this paradigm is quite limited, it has been the accepted practice since televison's inception.

At the other extreme, true VoD has the highest degree of viewer scheduling control. Program playout is based on specific viewer requests and playout starts as soon as the request can be satisfied. However, there will only be a single viewer per program stream, and so resources necessary to store, load, and transmit the program are allocated to a single viewer. Given that the standard for audio and video compression is likely to be MPEG-2, the storage and bandwidth requirements could easily exceed several Gigabytes for a two hour movie. Trying to make a profit will be very difficult especially since movies can typically be rented for only a few dollars.

With near VoD, the program start time is no longer immediate and some artificial delay is added. The hope is that multiple requests for the same program will be made in a short period of time. These requests are then *batched* and a single program stream is used to service the entire group. The assumption is that the network has an efficient multicast delivery facility and can provide a single program stream to several viewers. Near VoD has proven to be a scalable alternative to true VoD [1,2]. The biggest limitation of near VoD is that it requires large viewer populations to achieve sufficient economies of scale. Furthermore, there is a tradeoff between scalability and the amount of time viewers must wait before a program starts.

2.2. The jukebox paradigm

The key design principal behind the jukebox paradigm is flexible scheduling based on a finite set of channels available to all viewers. The jukebox paradigm is designed to be scalable while offering flexibility in the way viewer requests are handled. The paradigm is based on three properties:

(1) A set of "channels" are multicast to all viewers "tuned" to the respective channel.
(2) Viewers may watch a program playing on any channel or make a request for something of their own choosing. Viewers' requests are scheduled on one of the jukebox's channels using scheduling criteria like shortest wait time, etc.
(3) A schedule of currently playing and scheduled programs, updated in real-time, is available to all viewers. Viewers can watch any program, including those scheduled by others, by tuning to the appropriate channel.

The jukebox paradigm is based on the operation of a music jukebox. Everyone in a room can hear what is being played on a music jukebox and song requests are made by individuals. These requests are queued, and then played in the order they are made. Anyone can make requests but everyone will hear what is played. Our jukebox paradigm offers some additional advantages. First, there can be multiple, distinct channels which means more choices for those who are just surfing. Second, the jukebox paradigm provides a visual interface about what programs are playing and scheduled. This provides an opportunity for viewers to have their decision influenced by what is already scheduled. Put another way, how many movie rental store customers know what they want to rent before entering the store? Most are influenced by the list of available titles or the suggestions of other customers or store employees. Viewers may scan the jukebox schedule and see something interesting which has only just started or will be starting soon. Third, there is opportunity to implement better scheduling policies than simple first-come, first-served.

One of the key features of the jukebox paradigm is that it is scalable while still providing a great deal of viewer choice. In the context of VoD systems, the term *scalable* means a system has the ability to provide service to additional viewers for a diminishing marginal cost. Using this definition, true VoD systems are unscalable because each additional viewer requires roughly the same set of resources required to deliver a program stream as was required by the previous viewer [1,2]. On the other hand, near VoD is scalable because additional viewers can be accommodated by batching them with others who make the same program requests. One major disadvantage of near VoD is that there is still a relationship between the number of channels and the number of viewers. True bandwidth-limited systems may not be able to provide the additional channels needed to meet increased customer demand. Service will begin to degrade and customers will look elsewhere. The jukebox paradigm provides scalability using a fixed number of channels. Additional viewers can watch any channel for only the cost of joining the multicast group for that particular channel. The tradeoff with the jukebox paradigm is that as more viewers make requests the wait time for an individual viewer's request may increase. However, instead of being unable to watch anything, which occurs when a request is blocked in a VoD system, a viewer may be satisfied with something that is already playing or scheduled to start soon. The worst case occurs when there is a long wait time for a viewer's request and there is nothing in the schedule that interests the viewer.

In the jukebox paradigm, the ratio between the number of viewers and the number of channels is an important one. This ratio defines the type of service that viewers can expect. For example, at one extreme, if there is a large number of viewers and only a single channel, there is little chance that a viewer will make a request and then get to watch their program within a short period of time. At the other extreme, there will be as many channels as viewers making requests. In essence, each viewer will have a channel, like in a true VoD system. Furthermore, while there are economies of scale with the jukebox paradigm, they are not as severe as wit near VoD. Jukebox systems can be successful offering only a few channels or many channels. As more subscribers join a service, additional channels can be added.

Another advantage of the jukebox paradigm is its flexibility in request scheduling. A great deal of flexibility can be provided because almost any set of policies can be implemented. The simplest case for a multi-channel system is to schedule a viewer's request on the channel with the shortest wait time. Additional policies might include the following:

- **Content-based scheduling.** Limitations may be imposed based on the content. For example, "R-Rated" programs may be limited to certain channels or can only be played at certain times of day. New releases may be restricted to a certain set of premium channels. Also, specific channels may have a "theme". There could be a classic movies channel, a sports channel, a sit-com channel, etc. Any number of content based scheduling policies are possible.
- **Service provider scheduling.** A service provider may have a desire or obligation to schedule certain programs at certain times. For example, a slot for the evening news may be scheduled each day at the same time. Service providers have an entire spectrum of control. At one extreme, the service provider does all the scheduling and the service becomes traditional TV. At the other extreme, the system is completely demand driven. Somewhere in the middle a service provider might reserve specific times slots on specific channels, or might use a combined system of some on-demand channels, and some broadcast channels.
- **Limited viewer input.** A service provider may want to blunt the ability of individual viewers to control what programs are playing. For example, during peak periods, a service provider may impose limits on the number of programs a single viewer can request. A service provider may also institute a voting procedure. Only when enough votes from enough different viewers are received is a program actually scheduled. Viewers typically dislike having their "rights" restricted, but service providers must walk the fine line between making money and keeping customers happy.

Having described the jukebox paradigm in detail, we now re-examine the graph in Fig. 1. Figure 2 shows the graph with the addition of the jukebox paradigm. Because of its flexibility, the jukebox paradigm extends over a large region. All of the paradigms described in Section 2.1 can theoreti-

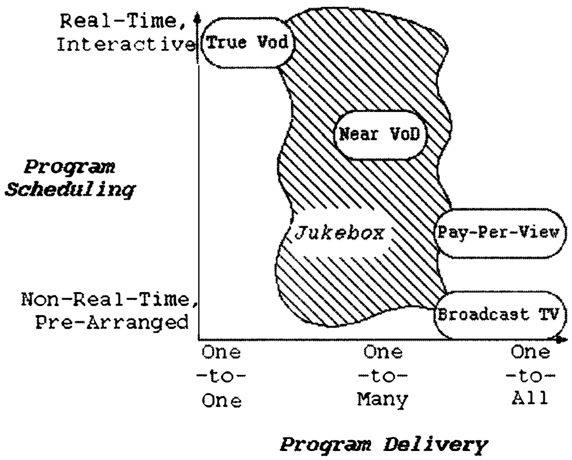


Fig. 2. The relative position of scheduling paradigms including the jukebox paradigm.

cally be implemented using variations of the jukebox paradigm.

2.3. Architecture for a jukebox system

A generic architecture for a jukebox system has four main components. Figure 3 shows the relationship between each of the components, and a description of each follows.

- **Scheduling control and schedule display**: The scheduler receivers viewer requests, performs scheduling, controls the video server, and provides a schedule of programs to all viewers.
- **Video server**: The video server transmits audio/video streams into the network. Programs are stored in compressed format, ready for transmission. The video server does not require significant CPU resources, but a system providing a large number of high quality channels may require fast disk access techniques [18].
- **Network**: The network must provide an efficient multicast facility and have sufficient bandwidth to meeting viewer quality expectations. The network must also have bi-directional capability in order to receive viewer requests and feedback. While this second requirement may seem obvious, it is worth mentioning in the face of today's asymmetric networks.
- **Receivers**: Receivers must be able to receive, decode, and display an audio/video stream. In addition, receivers must also be able to make program requests, and browse a program schedule. One of the keys to most streaming audio/video applications, especially when designed for entertainment, is that they not require high end workstations.

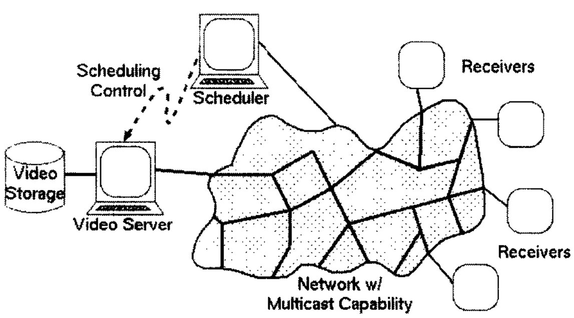


Fig. 3. Generic architecture for a jukebox system.

At this point, it is worth mentioning how system capacity is measured. A *channel* is defined to be the set of resources in the servers and network necessary to provide continuous delivery of a program to all viewers. For certain topologies, like cable TV, a channel is an easily defined. Systems are able to provide some number of simultaneous logical channels.

3. Jukebox system prototype

3.1. Prototype details

The jukebox paradigm has been implemented, and the prototype is called the *Interactive Multimedia Jukebox (IMJ)*. The IMJ uses the WWW for scheduling and program information and the MBone for multicast delivery of programs. By going to the IMJ home page (located at http://imj.gatech.edu), a viewer can see how many channels are available on the jukebox; what programs are currently scheduled for playout including start and end times; and what programs are in the jukebox library. Content being played on each channel is transmitted to all group members. Figure 4 shows the top of the IMJ home page including a snapshot of a sample real-time schedule.

Figure 4 shows that the set of IMJ scheduling policies is very simple. The time-to-live (TTL) value in IP packets is used to limit the scope of transmission. The TTL for channels 1 and 2 is set to

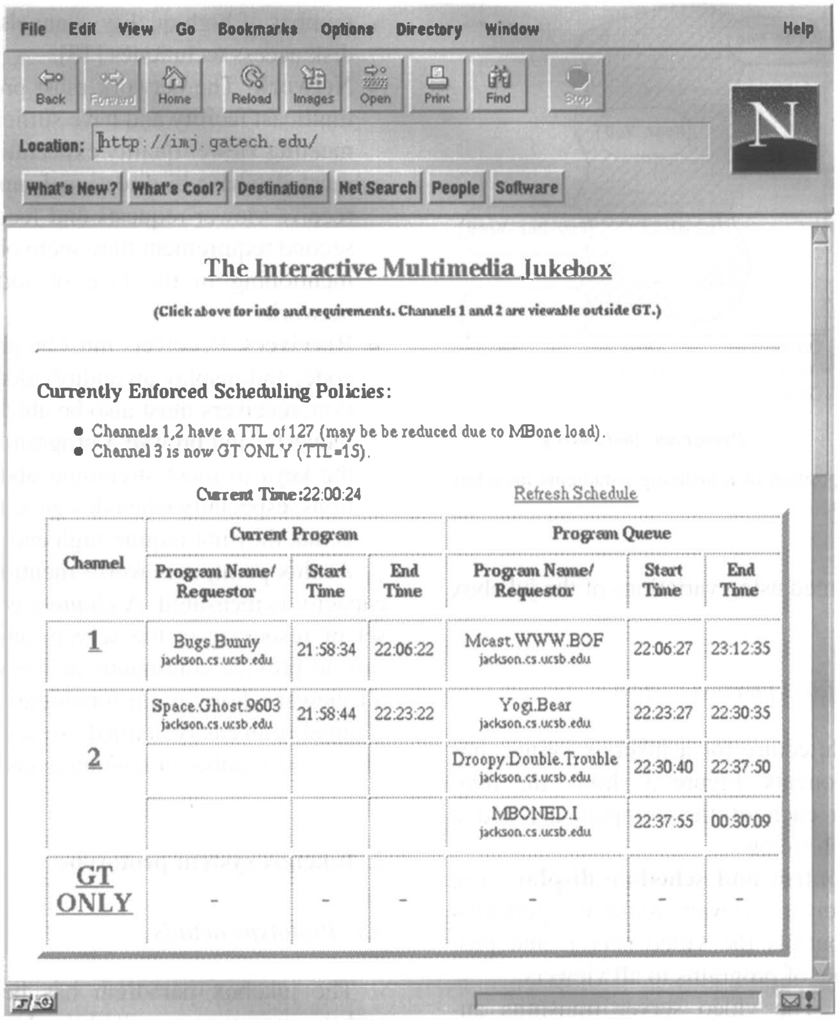


Fig. 4. Snapshot of the IMJ scheduling page.

127 which means anyone in the world can receive these sessions. The TTL for the "GT Only" session is set to 15 which limits transmissions to the Georgia Tech campus. Only two global channels are provided so as not to put an undue bandwidth load on the world-wide MBone.

With the WWW providing the interface, the MBone provides the multicast delivery service over the Internet and the MBone audio and video tools provide delivery, decoding and display functions. Before being made available on the IMJ, content is encoded as an RTP packet stream using the rtpdump utility [19]. Quality levels for both audio and video are set at typical MBone session levels. Audio is encoded at roughly 39 Kbps using the Intel DVI audio format. Video is encoded at a constant bit rate of 128 Kbps using the H.261 coding standard. The IMJ library currently has almost 70 hours of programming. Plans to increase the library size are in the works and depend on the availability of new sources.

The actual architecture of the IMJ is very similar to what is shown in Fig. 3. There is an HTTP server on one machine which serves requests for the IMJ home page and accepts program requests. Program requests are processed using a Perl script which passes the program name and information about the request to the scheduling daemon via a standard UNIX pipe. The scheduling daemon, written in C, processes and schedules the requests, and then updates the IMJ home page and its schedule in real-time by modifying the WWW page. Receivers are expected to periodically update their page by reloading the page from the IMJ WWW server. Embedded HTML flags automatically reload the page every 5 minutes. When it is time to start a particular program, the scheduler uses a remote shell command to the video server to start the audio and video streams via the rtpplay utility [19]. Stream synchronization is provided via RTP [5]. Because there are only two channels and per-stream bandwidth is relatively low, there is no need for specialized server hardware. Programs are stored as standard UNIX files and accessed from a disk local to the server via NFS.

3.2. Content for the IMJ

One of the biggest challenges to making the IMJ a success was our ability to make interesting content available. The two most important partners to-date are Turner Broadcasting Systems, Inc. (TBS) and the Internet Engineering Task Force (IETF). Our goal is to archive all future IETF MBone broadcasts on the IMJ. This will enable us to provide a useful service to interested members of the Internet community. The content contributed by TBS has also been very important. Their cartoons are ideal for the jukebox because they are of ideal length, and their simple video images encode very well. Our current agreement allows us to transmit content at a quality-limited rate of 128 Kbps. The main reason, which is mostly precautionary, is that lower quality content is less likely to be digitally recorded and does not detract from TBS's "real" TV channels. Readers should bear in mind that licensing content for Internet transmission is a difficult problem and content providers are justifiably hesitant.

4. Jukebox research issues

The jukebox paradigm and the IMJ implementation have created a number of challenges in a number of areas. While some of these issues have been explored for related paradigms, the jukebox paradigms requires a re-examination of some issues. In this section we concentrate on describing issues that are either already provided or that we expect to provide in the IMJ system.

4.1. Advanced jukebox service

- **Providing interactivity.** Because the jukebox is inherently a one-to-many service, it does not provide a specific mechanism for individual viewer interactivity. However, if sufficient network and server resources exist, VCR-like functions can be provided. One of two types of interactivity can be provided:
  - **Limited interactivity**: Instead of allowing a viewer continuous playout control, VCR actions are only allowed in increments. This allows interactivity to be provided using only a small number of additional groups at specific offset intervals. This type of interactivity is comparable to a similar concept proposed for near VoD systems [1]. While the advantage is that fewer resources are required, the disadvantage is that playout control is more coarse-grained. Also, this type of interactivity is only useful for large systems were interactivity is expected to occur frequently.
  - **Full interactivity**: Full interactivity can be provided using one of two methods. The first method requires the server to split the viewer from the main IMJ channel and provide the viewer with a single stream. When the first VCR action is initiated, resources are allocated to the viewer for the remainder of the program. At the end of the program, the viewer returns to the real-time jukebox schedule. The second method, and the method we are implementing in the IMJ, is to put the processing burden on the viewer. The viewer is responsible for buffering the server's transmission. Once buffered, the viewer can initiate VCR-style functions. The server transmits at a constant rate, and viewers use buffering to capture the program stream but then may be watching different parts in the program. The advantage of this method is that the processing burden is removed from the server. Furthermore, the amount of buffer viewers are willing to dedicate to a stream dictates the freedom with which they can move around within a stream.

- **Advanced reservations.** Instead of scheduling programs as soon as possible, a service provider may allow viewers flexibility in when a program is scheduled to start. Users may be allowed to specify that a program start at some time beyond the current schedule. Advanced reservations are a nice feature but they create two problems. The first problem is an issue of fairness. Allowing scheduling arbitrarily far in advance creates the potential for viewers to abuse the system and schedule programs without really knowing whether they will be around to watch them. The second and more interesting technical problem is how to deal with "dead periods". Dead periods occur as the playout time of a program approaches. Consider for example, a program scheduled in advance to start at 8:00pm. As the on-demand schedule fills and approaches 8:00pm, there will come a time when a requested program will be too long to fit in the period between the last on-demand program and the 8:00pm reservation. The schedule is left with a gap in which no program will fit. The solution to this problem involves using three techniques:
  - **Schedule programs with an awareness for the mean and median program length**: In some cases, a program may be scheduled on an alternate channel if it creates a high probability of a significant, unfillable dead period.
  - **Automatically schedule a program that fits into the available time slot**: If there is a program in the library which would nicely fill an existing dead period, go ahead and schedule it even though no viewer has requested it. Any program is better than no program.
  - **Adjust the start time of the advanced reservation**: If is a dead period is only a couple of minutes or seconds shorter than a requested program, the reservation could be pushed forward slightly. Or, if the dead period is too long, the reservations could be moved up. However, violating promised start times may cause dissatisfaction among some viewers, but slight scheduling changes may not be too bad.
- **Service pricing.** The cost of using the jukebox system is an important consideration for a service provider. From a technical perspective, pricing can be used as an equalizer. It can be used to control viewer behavior, discouraging actions which require precious system resources, and encouraging behavior which allow more viewers to be satisfied. A typical pricing scheme might include two types of charges:
  - **Monthly access charge**: Simple access to all the jukebox channels is worth something. The more channels and the better the content the more a service provider can charge. A service provider offering a number of broadcast and on-demand channels could easily charge current cable TV rates. Monthly access charges will likely account for a majority of a system's revenue.
  - **Per-request charge**: There can be a per-request charge each time a viewer makes a request. The dilemma here is making this charge small enough to encourage viewers to make requests but large enough to discourage viewers from randomly requesting programs that they have no real intention of watching. Without market testing we cannot attempt to guess what viewers would be willing to pay.

  There are also numerous opportunities for service providers to charge additional fees for premium services. Additional services include advanced reservations, server-based interactivity, access to premium content like new release movies, premium quality including an HDTV version of a program, etc.

4.2. Tracking usage in the IMJ

Understanding how the IMJ is used is critical to understanding many aspects of the system. From a research point-of-view, we can find and correct problems and learn about behavior. From a pay-for-service point-of-view, tracking usage enables a service provider to decide when new programs should be added and when old programs are no longer worth offering. Furthermore, if a jukebox system is offered as an alternative to TV, service providers would like as much information about viewing habits as possible. Consider the importance broadcasters put on ratings. Tracking usage is based on our ability to collect information from three sources:

- **Program requests**: Data about who, when, and what programs are requested. This data is collected by the scheduler and archived in a log file.

- **Jukebox schedule**: Data about viewers who may not necessarily request a program but who check to see what has been scheduled. The IMJ WWW server access logs can be used for this information. Furthermore, because the IMJ page re-loads itself every 5 minutes we can monitor those viewers who point their browsers at the IMJ page for extended periods of time.
- **Program viewers**: Because the IMJ uses the MBone tools for audio/video delivery, information can be collected about how many people are watching each session and for how long. This information is collected using a tool designed and developed by the authors called *Mlisten* [20]. In the past, it has been used to monitor multicast group behavior in traditional MBone sessions.

The IMJ effort has been immensely successful as a prototype but the data to-date does not suggest that the IMJ is in danger of competing with international broadcasting companies. However, at its peak, the IMJ has had almost 50 viewers spread across its two channels. Thousands of different viewers have visited the WWW site over the past year. And requests continue to be made at an average rate of a several per hour.

4.3. Dealing with heterogeneity

The issue of providing the best quality possible to each and every viewer is very difficult. Different viewers have different bandwidth capabilities and these capabilities can vary from minute to minute based on transient network conditions like congestion. Some viewers might have Internet connectivity and MBone capability at one or even several Megabits per second while other viewers might be accessing the MBone via a 64 Kbps ISDN link. An even more exciting possibility is that viewers are connected to a jukebox system via a cable TV infrastructure. Requests can still be made via a dial-up connection but programs are delivered via a pure digital signal or a digital signal converted to a standard television signal. In fact, this option is very close to being deployed in campus housing at Georgia Tech. Given that viewers will want different quality levels, there are a number of solutions worth evaluating.

(1) A server transmits multiple, independent streams of different bandwidths, and receivers move between these streams based on the measured bandwidth of the stream, the receiver's capacity, and the measured loss [21–23]. In an extended version, the server may change the bandwidth or quality of service of a particular stream in response to more course grain feedback from receivers. The limitation of the basic scheme is that it actually increases bandwidth transmitted from the server because of the duplication of data encoded in each stream.

(2) A server transmits several streams created by dividing a single stream into layers [24]. The streams are dependent on each other and each higher layer of quality requires the receiver to join an additional multicast group [25,26]. A viewer would typically join as many groups as possible without causing congestion on some link along the path.

Both solutions are in the process of being implemented within the MBone. When they are available on a wide scale we will work to deploy them in the IMJ and evaluate their effectiveness.

5. Conclusions

This paper discusses our proposed jukebox paradigm, an exciting alternative to other video-on-demand paradigms. The jukebox paradigm we propose is based on the premise of allowing any viewer to watch any other viewer's requested program. Program requests are scheduled on one of a system's set of channels using a set of scheduling policies. Any viewer who wants to watch a program on a particular channel simply "tunes" to that channel. Content on each channel is delivered from a server to all viewers watching that particular channel. This paper also describes our efforts to prototype a jukebox system called the *Interactive Multimedia Jukebox* (IMJ). The IMJ provides scheduling via the World Wide Web (WWW) and content delivery via the Multicast Backbone (MBone).

References

[1] K. Almeroth and M. Ammar, The use of multicast delivery to provide a scalable and interactive video-on-demand service, *IEEE Journal on Selected Areas in Communications*, 14: 1110–1122, August 1996.

[2] A. Dan, D. Sitaram, and P. Shahabuddin, Scheduling policies for an on-demand video server with batching, in: *ACM Multimedia '94*, San Francisco, CA, October 1994.

[3] S. Casner, Frequently Asked Questions (FAQ) on the Multicast Backbone (MBone), USC/ISI, December 1994, available from ftp://ftp.isi.edu/mbone/faq.txt

[4] S. Deering and D. Cheriton, Multicast routing in datagram Internetworks and extended LANs, *ACM Transactions on Computer Systems*, May 1990, pp. 85–111.

[5] H. Schulzrinne, S. Casner, R. Frederick, and V. Jacobson, RTP: A transport protocol for real-time applications, Tech. Rep. RFC 1889, Internet Engineering Task Force, January 1996.

[6] W. Holfelder, Interactive remote recording and playback of multicast videoconferences, in: *4th International Workshop on Interactive Distributed Multimedia Systems and Telecommunication Services (IDMS '97)*, Darmstadt, September 1997.

[7] A. Klemets, The design and implementation of a media on demand system for WWW, in: *Proc. 1st World-Wide Web Conference*, Geneva, May 1994.

[8] R. Vetter and C. Jonalagada, Multimedia system for asynchronous collaboration using the multicast backbone and the World Wide Web, in: *Proceedings of the Annual Conference on Emerging Technologies and Applications in Communications*, Portland, OR, May 1996, pp. 60–63.

[9] P. Parnes, M. Mattsson, K. Synnes, and D. Schefstrom, mMOD: the Multicast Media-on-Demand system, Centre for Distance-Spanning Technology, May 1997 (submitted).

[10] K. Almeroth and M. Ammar, Scalable delivery of Web pages using cyclic best-effort (UDP) multicast, in: *IEEE Infocom*, San Francisco, CA, USA, June 1998 (to appear).

[11] T. Liao, WebCanal: a multicast Web application, in: *Proc. 6th International World Wide Web Conference*, Santa Clara, CA, USA, April 1997.

[12] P. Parnes, M. Mattsson, K. Synnes, and D. Schefstrom, The mWeb presentation framework, in: *Proc. 6th International World Wide Web Conference*, Santa Clara, CA, USA, April 1997.

[13] R. Clark and M. Ammar, Providing scalable Web service using multicast delivery, in: *IEEE Workshop on Services in Distributed and Networked Environments*, Whistler, Canada, June 1995; also *Computer Networks and ISDN Systems*, 29(7), 841–858, 1997.

[14] R. El-Marakby and D. Hutchinson, Integrating RTP into the World Wide Web, in: *World Wide Web Consortium Workshop on Real Time Multimedia and the Web*, Sophia Antipolis, France, October 1996.

[15] M. Handley, Applying real-time multimedia conferencing techniques to the Web, in: *World Wide Web Consortium Workshop on Real Time Multimedia and the Web*, Sophia Antipolis, France, October 1996.

[16] T. Little and D. Venkatesh, Prospects for interactive video-on-demand, *IEEE Multimedia*, 14–23, Fall 1994.

[17] J. Allen, B. Heltai, A. Koenig, D. Snow, and J. Watson, VCTV: A video-on-demand market test, *AT&T Technical Journal*, 7–14, January/February 1992.

[18] A. Chervenak, D. Patterson, and R. Katz, Storage systems for movies-on-demand video servers, in: *IEEE Symposium on Mass Storage Systems*, Monterey, CA, USA, September 1994, pp. 246–256.

[19] H. Schulzrinne, README for RTPtools, Columbia University, August 1997, available from ftp://ftp.cs.columbia.edu/pub/schulzrinne/rtptools/

[20] K. Almeroth and M. Ammar, Multicast group behavior in the Internet's multicast backbone (MBone), *IEEE Communications*, 35: 224–229, June 1997.

[21] X. Li and M. Ammar, Bandwidth control for replicated-stream multicast video distribution, in: *High Performance Distributed Computing (HPDC)*, Syracuse, NY, USA, August 1996.

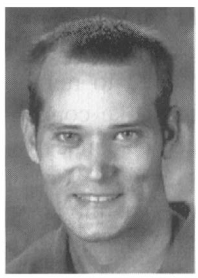
[22] S. Cheung, M. Ammar, and X. Li, On the use of destination set grouping to improve fairness in multicast video distribution, in: *IEEE Infocom*, San Francisco, CA, USA, March 1996.

[23] J.-C. Bolot, T. Turletti, and I. Wakeman, Scalable feedback control for multicast video distribution in the Internet, in: *ACM Sigcomm*, September 1994, pp. 58–67.

[24] X. Li, S. Paul, P. Pancha, and M. Ammar, Layered video multicast with retransmission (LVMR): Evaluation of error recovery schemes, in: *Proc. of the Int. Workshop on Network and Operating System Support for Digital Audio and Video '97*, St. Louis, MO, USA, May 1997.

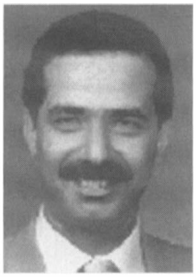
[25] I. Wakeman, Packetized video — options for interaction between the user, the network and the codec, *The Computer Journal*, 55–67, Jan. 1993.

[26] S. McCanne, V. Jacobson, and M. Vetterli, Receiver-driven layered multicast, in: *ACM Sigcomm*, Stanford, CA, USA, August 1996, pp. 117–130.



Kevin C. Almeroth received his B.S. (1992), M.S. (1994), and Ph.D. (1997) degrees in computer science from the Georgia Institute of Technology in Atlanta, Georgia. He is currently an assistant professor in the Department of Computer Science at the University of California, Santa Barbara. At UCSB he is managing the Networking and Multimedia Systems Lab (NMSL). His research interests include computer networks and protocols; Internetworking; multimedia systems; performance evaluation; and distributed systems.

Dr. Almeroth has he has held several research positions while a graduate student including positions at IBM's T.J. Watson Research Labs, Hitachi Telecommunications, and Georgia Tech's Office of Information Technology. More recently he has been involved in the Internet Engineering Task Force (IETF), IP Multicast Initiative (IPMI), the Internet-2 Multicast Working Group, and the Corporation for Education Network Initiatives (CENIC) Academic Applications Council. He has been a member of both the ACM and IEEE since 1993. Home page: http://www.cs.ucsb.edu/~almeroth/



Mostafa H. Ammar received his Ph.D. Degree in Electrical Engineering from the University of Waterloo in Ontario, Canada, 1985. His S.M. (1980), S.B. (1978) degrees were acquired in Electrical Engineering and Computer Science at the Massachusetts Institute of Technology, Cambridge, MA. Dr. Ammar is currently an Associate Professor in the College of Computing at Georgia Tech. He has been with Georgia Tech since 1985. For the years 1980-82 he worked at Bell-Northern Research (BNR), in Ottawa, Ontario, Canada, first as a Member of Technical Staff and then as Manager of Data Network Planning.

Dr. Ammar's research interests are in the areas of computer network architectures and protocols, multipoint communication, distributed computing systems, and performance evaluation. He is the co-author of the textbook "Fundamentals of Telecommunication Networks," published by John Wiley and Sons.

Dr. Ammar is the holder of a 1990-1991 Lilly Teaching Fellowship and received the 1993 Outstanding Faculty Research Award from the College of Computing. He is a member of the editorial board of IEEE/ACM Transactions on Networking and Computer Networks and ISDN Systems Journal. He was the co-guest editor of a recent issue (April 1997) of IEEE Journal on Selected Areas in Communication on "Network Support for Multipoint Communication". He was the Technical Program Co-Chair for the 1997 IEEE International Conference on Network Protocols.

Dr. Ammar is a Senior Member of the IEEE and a member of the ACM and a member of the Association of Professional Engineers of the Province of Ontario, Canada. Home page: http://www.cc.gatech.edu/fac/Mostafa.Ammar/index.html