Providing A Scalable, Interactive Video-On-Demand Service Using Multicast Communication

Kevin C. Almeroth
Mostafa H. Ammar

Telecommunications/Networking Systems Group
College of Computing
Georgia Institute of Technology
Atlanta, Georgia 30332-0280
kevin, ammar@cc.gatech.edu
(404)853-9911

GIT-CC-94/36

March 4, 1994

Abstract

Video-On-Demand (VOD) systems are typically designed around the provision of a one-to-one connection between a customer and a video server. Such a scheme is not scalable due to bandwidth and video server resource limitations. On the other hand, it is attractive because it can provide an on-demand system and a high degree of interactivity. In this paper we propose the use of multicast communication in a VOD system wherein multiple users may be served simultaneously by a single video stream transmitted by the server. A limited amount of synchronization of customer requests is needed to further enhance the benefits of the multicast operation. Done properly this should not detract from the on-demand nature of the system. We consider how interactive features, giving the illusion of a one-to-one service, can be added to such a system at the expense of added complexity to system operation and set-top box hardware. The performance of our proposed system is investigated through the use of a simulation.

1 Introduction

Interactive TV, "Smart" TV, Video-On-Demand, are some of the latest terms coming out of the battle to provide in-home service to customers through an interface that many families have: the television. With Video-On-Demand, the idea is to offer a very large video library to customers, so that they can select a movie, and watch it any time they want. For the cable TV industry, Video-On-Demand (VOD) offers a lucrative multi-billion dollar industry that is virtually untapped.

The major challenge in providing a VOD service is handling the enormous demands of VHS-quality video in a real-time network, with real-time interaction[3]. Currently, the preferred approach is to offer one channel per customer and service that customer individually[1]. The problem with this type of system is that servicing customers individually is inefficient, expensive, and not scalable. Telephone-based companies and cable-based companies have started working together and experimenting with Video-On-Demand service. Most of these tests are geared only to very small groups; sometimes only about 300 people[1]. In these systems, as the number of customers increases, so must the number of channels. While available bandwidth has increased significantly in recent years, servicing an average-size customer area of 2000 families would require more bandwidth than we have available today[2].

However, one of the best ways to deliver information to more then one customer is through the use of multicast communication. Multicast offers the advantage of being scalable. Earlier work examined some of the issues in using multicast communication as an information delivery mechanism[7, 8]. The interactive nature of Video-On-Demand logically demands individualized service, whereas with the case of multicast communication, true interactivity is difficult to provide.

In this paper, we explore the use of multicast communication to address the issue of providing scalable and interactive Video-On-Demand service. We look specifically at what we can trade-off to improve the usability of multicast communication. We focus on a system that will provide a customer with the ability to pause a movie for arbitrary duration. Some of the techniques we describe can be extended to provide other capabilities such as rewind and fast forward. We will explore these in future research. Consideration of the pause capability, however, yields significant insight into how a system using multicast can be configured to provide an interactive VOD service.

The paper is organized as follows. Section 2 is a discussion of the architecture of a Video-On-Demand system. In Section 3 we discuss the operation of our proposed system. In Section 4, we present a model for the performance evaluation of two systems. Section 5 contains numerical results obtained through a simulation of some sample systems. The paper is concluded in Section 6.

2 System Architecture

The architecture of our Video-On-Demand system is shown in Figure 1. The system components include the Video Server, the Network, and the Set-Top Box. Each is defined by the functions that it must perform, and is described in the following sub-sections.

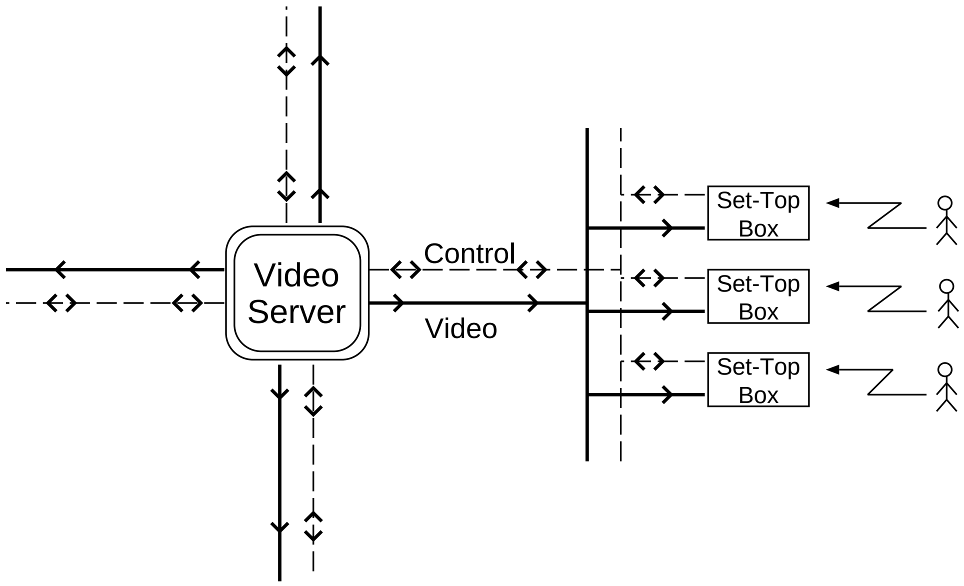

Figure 1: The architecture of our VOD system.

1. **The Video Server**: The Video Server is responsible for receiving and processing customer requests; and setting up and maintaining video streams which deliver requested movies. The other main function of the Video Server is to retrieve movie data that is stored on disk, and send it to customers. The Video Server is a control point for processing and forwarding movie frames. The issues involved in retrieving and sending video frames have been the subject of some recent research (see for example[6]).

2. **The Network**: The functions of the network are to transmit video frames from the Video Server to the customer, and to transmit control messages. The topology choice for next generation cable TV plants seems to be a star-bus configuration[4, 5]. The branches of the star network are optical fiber, and extend from the Central Office (CO) to a neighborhood. Inside a neighborhood, the bus network is coaxial cable, and it connects all the customers. A neighborhood can range in size anywhere from 500 to 2500 customers[4].

   Each branch of the star network can be logically separate, because each connection is a different physical cable. Therefore, neighborhoods are physically separate, and can be considered separate when talking about a Video-On-Demand system. However, since every customer is connected via the bus network, a video stream sent to one customer can be received by any other customer in the same neighborhood. Assuming that we are using MPEG-1 to transmit video streams, we expect to have at least 150 channels available per star branch.

3. **The Set-Top Box**: The two classes of functions the Set-Top Box must provide include sending and receiving control messages; and receiving and processing a video stream. This includes receiving and processing input from the customer via a remote control. The customer can either request a movie; or during a movie, request VCR-style functions such as pause, rewind, and fast forward. The Set-Top Box will receive and process these signals, and if needed, send the appropriate requests upstream to the Video Server.

   Part of the responsibility of the Set-Top Box is the playout of a video stream with minimal interruption of service. In fact, any interruption in service is considered to be a significant error. We assume that the Set-Top Box has some video frame buffering capability. This can be used to provide uninterrupted playout. As frames are received, they are buffered until just before playout. Just before a frame is displayed, it is uncompressed with the appropriate hardware. The final component of the Set-Top Box is the processor required to send and receive message; and to process a customer's requests. This processor should be relatively inexpensive considering it only needs to provide a few basic functions.

   An important consideration is the cost of the Set-Top Box. Our proposed VOD system requires that the box have enough memory to buffer a length of movie in the range of 1 to 15 minutes. Based on cost tradeoffs, and our simulations, we consider a buffer capable of storing five minutes of a movie to be sufficient. While this seems large, we hope the cost of memory will drop enough to justify such a requirement. Also, the Set-Top Box will require decompression hardware such as MPEG-1. Again, even though MPEG-1 hardware is relatively expensive now, we hope the cost will drop considerably in the near future.

3 System Operation

3.1 Use Of Multicast

In our system, movies are made available only at the beginning of time slots. The time slot duration is on the order of minutes (in our analysis we use the range 1 to 15 minutes). A customer making a request will thus have to wait for, on average, half a slot duration before the movie can start. For short slot durations (say 5 minutes) this should not affect the "on-demand" nature of the system.

Customers make requests specifying a movie to watch, and a time slot to start watching the movie. The request is sent to the Video Server, which processes all customer requests and produces a schedule for the playout of movies. Based on all the requests for movies, and the number of available channels, the Video Server allocates channels for the playout of requested movies. Customers are informed through confirmation messages whether or not their request was accepted or denied. All scheduling is done prior to the beginning of each time slot.

The synchronization of movie beginning times to a slot boundary allows the server to serve all customers requesting the same movie at the same time using a single multicast channel. Our ability to group multiple movie requests will depend on several factors. The longer the slot duration is, the more likely that several requests for the same movie will arrive during the slot. Of course, long slots detract from the "on-demand" nature of the system. The other factor is the popularity of a particular movie. The system is more likely to be able to group requests for currently popular movies than for a less frequently requested movie. Experience indicates that a small percentage of the movie offerings will experience the largest request volume. It is for these movies that the benefit of the use of multicast becomes apparent.

3.2 Providing Pause Capability

Once the movie starts, the customer has an option of pausing the movie. This pause function provides a level of interaction that represents the most basic and important VCR-like function. To the customer, it will seem like the Video Service is providing a one-to-one video stream. In reality, all Set-Top Boxes that requested the same movie with the same slot are part of a multicast group, and are receiving the same signal. When a customer pauses a movie, the playout of his or her movie will become different from other members in the group. This temporal difference in playout is compensated for by buffering the incoming frames while the movie is paused. This buffering mechanism allows the Set-Top Box to continue to receive the same video stream without affecting the playout of any group members. The buffer acts as a storage and delay mechanism for playing out frames. However, since a customer can pause arbitrarily often, and for a random amount of time; the amount of buffer space needed to guarantee the proper functionality of the pause service has to be very large. This problem can be solved by adopting the following policy: when a customer pauses for longer then a pre-determined amount of time, that customer is removed from the group receiving the video stream, and a new video stream is created for that customer. By limiting the size of the buffer, we specify the upper limit on the amount of time a customer can pause the movie before their Set-Top Box needs to change to a different video stream, i.e. a different multicast group.

An important design aspect of our system is that when a Set-Top Box is moved to a new video stream, a new channel does not always need to be allocated. By defining the buffer size to be equal to the slot length, we can take advantage of the case where other customers might have started watching the same movie, but at the next later time slot. If this is the case, the Set-Top Box that is changing multicast groups can leave its current group, and join a group already in progress.

This dynamic scheduling is relatively easy to maintain since the Video Server already has all the information needed. The Video Server knows what movie each Set-Top Box is receiving, and on which channel, and what time each Set-Top Box started receiving the movie. When the video server receives a message from a Set-Top Box that a new channel assignment is needed, the Video Server determines if an existing video stream can be used, or if a new channel, with the appropriate video stream, needs to be created. The process of changing groups continues, if needed, until the movie is over, and the initial request to watch the movie has been satisfied.

It should be noted that there is always the danger that a Set-Top Box's request to change to a different video stream cannot be granted. This will happen when there are no existing video streams to satisfy the request, and no idle channels are available. We consider this type of pause blocking to be unacceptable and would therefore like to design a system where the probability of this blocking is extremely low. Our approach is to trade-off slightly higher initial movie request blocking with lower pause blocking. This is accomplished by reserving a small number of "emergency" channels. These are not available to satisfy initial requests and are used exclusively to serve requests to change video streams that would otherwise be blocked.

3.3 Network Messages

From the network point of view, there are four control messages that can be transmitted. Each message and its fields are shown in Figure 2.

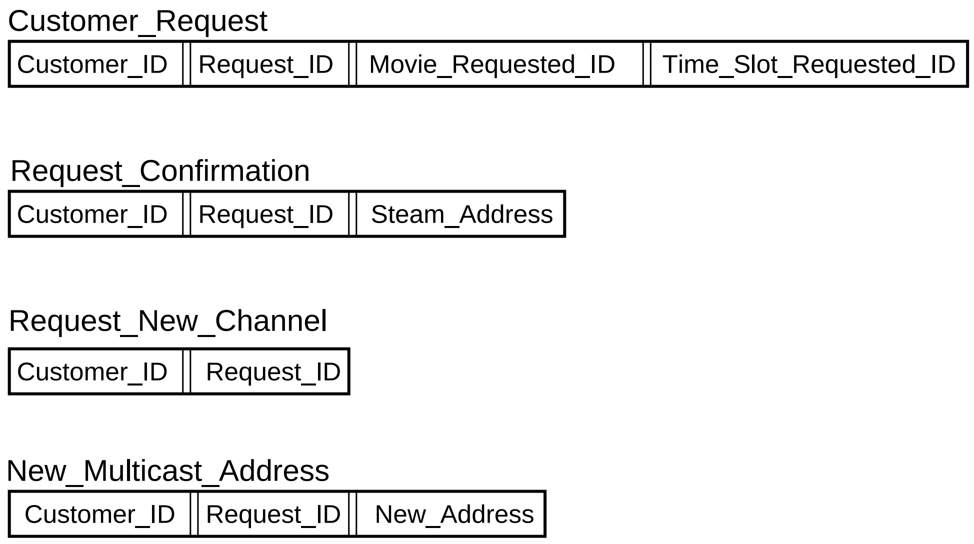

Figure 2: The structure of each message

The fields in the messages have the following meaning:

- **Customer_ID**. This is a unique number assigned to each customer when they first join the VOD service. The number is used to differentiate between customers in a neighborhood, and for billing.
- **Request_ID**. This is a unique number picked pseudo-randomly by the Set-Top Box. This number is used to differentiate between multiple requests by the same customer.
- **Movie_Requested_ID**. This field records the ID number of the movie that the customer is requesting.
- **Time_Slot_Requested_ID**. This field implicates the requested time slot. This can either be the next available time slot (immediate VOD), or a later time slot (reserved VOD).
- **Stream_Address**. This field tells the Set-Top Box on which multicast address it should tune in order to receive the requested video stream. If this field is zero, the request is denied.

A Customer_Request is a message constructed and sent by the Set-Top Box to the Video Server. Using the Customer_ID, the Video Server can access a database to find out what branch of the star network the customer is physically located on, and any other necessary information.

A Request_Confirmation message is sent by the Video Server, and is addressed to a specific Set-Top Box. The response includes the multicast address of a video stream showing the requested movie at the requested time.

A Request_New_Channel message is sent by a Set-Top Box that is already receiving a movie to the Video Server. This message is sent because free buffer space is about to run out. The available buffer space is filled when a customer uses the pause function for a total time greater then what the buffer can handle. This message tells the Video Server that it needs to assign the Set-Top Box to a new video stream.

A New_Multicast_Address message is sent by the Video Server, and is addressed to a specific Set-Top Box. This stream address is for a video stream for the same movie, but at a starting time later in the movie. This later starting time means that the customer will receive frames that have already been stored in the buffer. This change in multicast groups allows the buffer to free up space, because frames that have already been received will be arriving again.

3.4 Video Server Operation

The Video Server has three functions. The first function is to determine whether or not a customer request for a movie can be satisfied. The Video Server uses state information and the following procedure to make the determination.

- If a channel has already been reserved for the requested movie at the requested time; the Video Server simply adds this customer to the already existing group; and sends a confirm message with the group ID.
- If a group does not already exist, and there is a free channel; the Video Server creates a new group; allocates a free channel to the group; and sends a confirm message with the group ID.
- If the group does not exist, and there are not any free channels; the Video Server sends a confirm message with a group ID of zero. This means the request is denied.

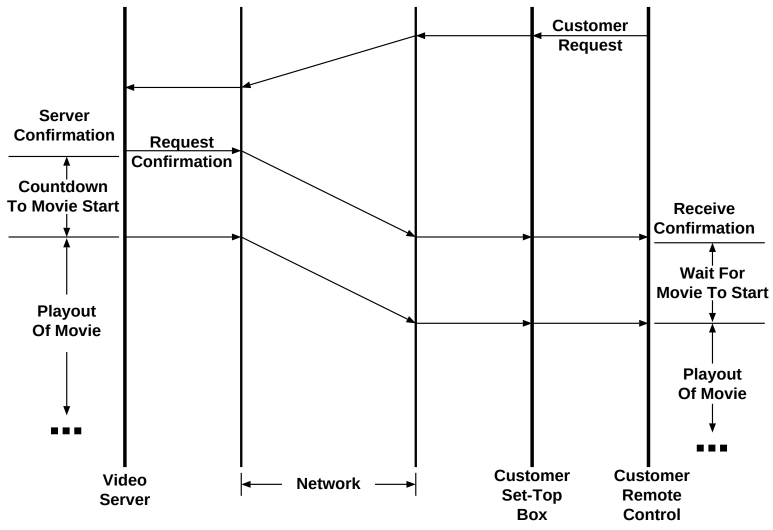

Figure 3: The flow of a customer request and confirmation.

Note that the video server performs straight-forward First Come, First Serve scheduling. That is, no request is denied if a channel is available (free or assigned to the same movie) at the requested time. Other scheduling disciplines are possible, and are relegated to future research.

The second function is to process requests to change multicast groups. A request is received at the Video Server after a customer has paused for a time sufficiently long so as to cause the empty space in the Set-Top Box's buffer to go below a certain threshold. The procedure is as follows:

- The Video Server determines the multicast group of the Set-Top Box making the request.
- The Video Server determines how many Set-Top Boxes currently belong to the multicast group.
- The Video Server determines if a multicast group exists for the same movie, but at the next later time interval.
- Using all this information, one of three scenarios can occur:
    1. The Set-Top Box requesting the multicast group change is the only one in the multicast group, and a multicast group at the next interval does not exist. In this case, the channel is re-allocated

to the next time interval, and the Video Server begins transmitting the movie as if it had started at the later interval.

2. The Set-Top Box requesting the multicast group change is only one of many in the multicast group, and no multicast group exists at the next time interval. In this case, when the Set-Top Box changes groups, a new channel needs to be allocated to transmit the movie at the next interval. If no channels are available, an emergency channel will be used. If no emergency channels are available, the group change is blocked, and the movie continues to play.
3. The Set-Top Box requesting the multicast group change is only one of many in the multicast group, and there already exists a multicast group for the next time interval. In this case, no allocations or de-allocations need to be made. The customer simply leaves one group, and joins the other.
4. The Set-Top Box requesting the multicast group change is the only one in the multicast group, but there exists a multicast group at the next time interval. In this case, the channel that the Set-Top Box is currently using can be de-allocated, and the Set-Top Box joins the later multicast group.

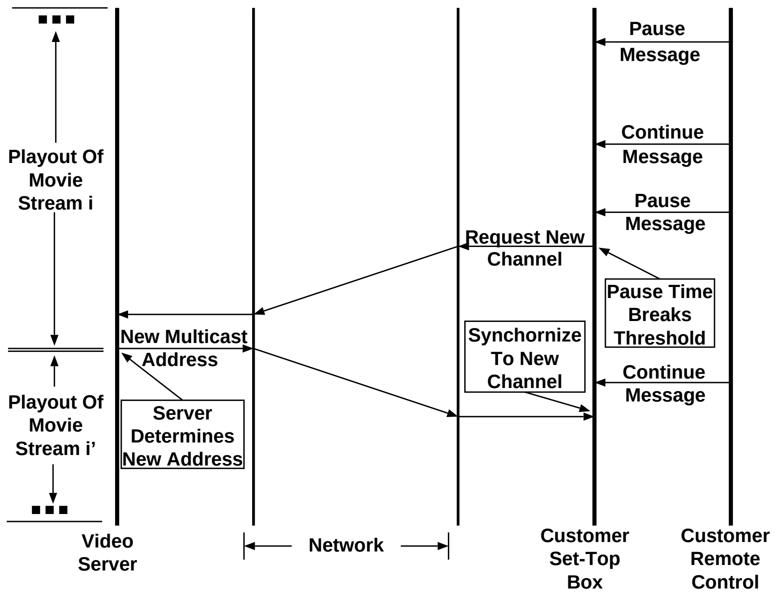

Figure 4: The customer actions that require a multicast group change.

3.5 Set-Top Box Operation

The Set-Top Box is responsible for performing the following functions:

- **Receive and process customer input.** Customer input can be commands for selecting a movie, or can be commands to pause or continue playing a movie.
- **Receive and process movie stream.** The Set-Top Box must be able to listen to a specific multicast address in order to receive the video stream requested by the customer. Also, the Set-Top Box must decode the incoming frames, and send them to the TV to be displayed.

- **Receive and send control messages.** The Set-Top Box uses a separate channel for movie requests, and changes in multicast groups. This control channel is used for any non-movie traffic, and connects all the customers to the Video Server.
- **Buffer management.** The Set-Top Box is responsible for buffering incoming frames, and maintaining information about the state of the buffer. Using this information, the Set-Top Box determines when a multicast group change is necessary. The buffer is the critical element, and it is responsible for compensating for network jitter, and providing uninterrupted service to the customer.

A Set-Top Box that implements these four functions can be built using either current technology, or technology that will soon be available. Based on an estimate of the Set-Top Box cost, building one today may be prohibitive. However, improvements in technology, and mass production should make it affordable in the near future.

The first three functions are relatively straightforward to implement given the appropriate hardware. The most interesting, and least intuitive of the four is management of the buffer. Figure 5 shows a buffer in a typical state of operation.

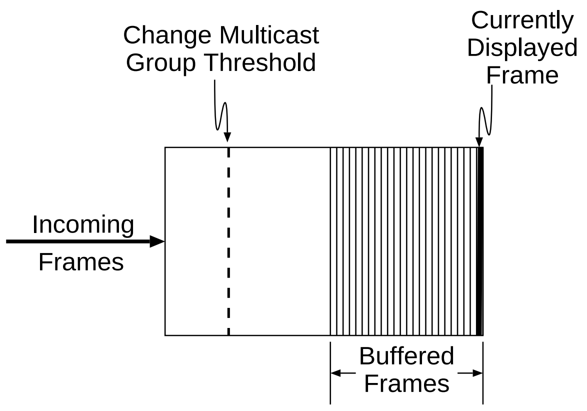

Figure 5: Typical state of the buffer in the Set-Top Box.

The typical state of the buffer includes some number of frames that have not yet been played out. These frames exist because of two reasons. First, it is necessary to provide a small cushion between frames being played out, and frames being received. This cushion provides effective network jitter control. Second, when a customer pauses the movie the number of buffered frames will increase.

Our buffer is logically equivalent to a queue. Frames arrive, and are placed at the end of the queue. Playout of frames is from the beginning of the queue. The physical implementation does not have to be a queue, but can be something like a circular buffer. Normal operation of the buffer occurs when the customer is watching the movie, and not using the pause function. During normal operation, frames arrive at roughly the same rate as they are played out.

When the customer presses the pause key, the playout of frames is stopped, while new frames continue to arrive. The net effect on the buffer is the number of frames stored increases. Figure 6 shows an example of how the state of the buffer changes during a pause in the movie.

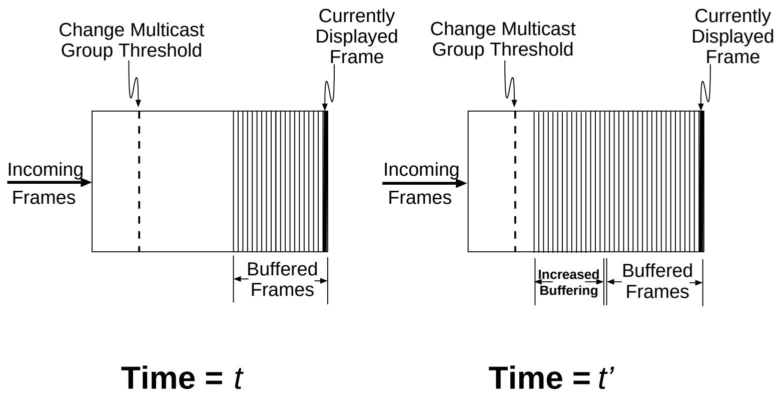
Change Multicast Group Threshold
Currently Displayed Frame
Incoming Frames
Buffered Frames
Time = t
Change Multicast Group Threshold
Currently Displayed Frame
Incoming Frames
Increased Buffering
Buffered Frames
Time = t'

Figure 6: Effect of a pause action on the state of the buffer in the Set-Top Box.

In some cases, the buffer will be large enough to accommodate the cumulative time of all of a customer's pause requests. If however, a customer uses the pause function for more time then the size of the buffer, a change in multicast groups is required. When the number of frames in the buffer exceeds a pre-defined threshold, the Set-Top Box sends a message to the Video Server requesting a new multicast group. This threshold is based on the maximum time required to receive a response from the Video Server. More specifically, this time is equal to the propagation delay of sending the request; plus the time to process the request; plus the propagation delay of sending the response. In addition, a small safety period should be added to accommodate any jitter or unforeseen small delays. If the Set-Top Box does not receive a response before the buffer fills up completely, continued use of the pause function is blocked, and the movie starts playing. A pause function block represents a significant problem, and should occur very infrequently. If the Set-Top Box receives a message from the Video Server with a new multicast address, it will immediately tune to the new multicast address. This new address is for a video stream playing the same movie, but with a later start time. Therefore, by tuning to the new multicast address, the customer will start receiving frames that are already stored in the buffer. The buffer can now be cleared up to the point where old frames will be duplicated by frames from the new video stream. The effect of changing multicast groups is that by switching to a later video stream of the same movie, the buffer can be emptied since the customer will again receive those exact same frames. Figure 7 shows what happens when the buffer breaks the threshold, and a multicast group change occurs.

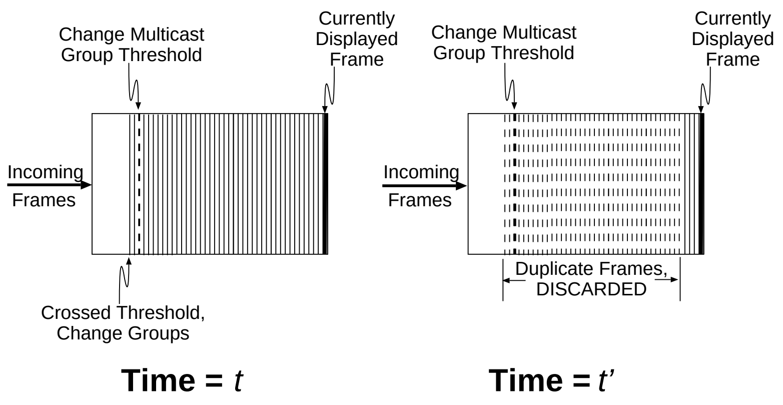
Change Multicast Group Threshold
Currently Displayed Frame
Incoming Frames
Crossed Threshold, Change Groups
Time = t
Change Multicast Group Threshold
Currently Displayed Frame
Incoming Frames
Duplicate Frames, DISCARDED
Time = t'

Figure 7: Effect of crossing the thresholds on the state of the buffer in the Set-Top Box.

If the customer continues to pause, and the buffer fills up again, the same process will be repeated. The customer will leave the group it just changed to, and join a new group with an even later start time. The justification for making the buffer capable of holding a time slot's worth of frames is because ideally, the Video Server should not have to allocate a new channel and create a new video stream when a Set-Top box requests a group change. If the buffer is smaller then the time slot, there is no chance of finding an already existing video stream that matches a Set-Top Box's request, and a new stream will always have to be created. Making the buffer larger then necessary is not cost-effective.

4 Performance Evaluation

In order to understand the performance of our proposed Video-On-Demand system, we developed a simulation. The various parameters and performance measure used in the simulation are described next.

4.1 System Parameters

The simulation is parameterized by the following:

1. **Number Of Customers Making Requests**. This is the number of customers who will make requests on one branch of the star network. Each branch should be able to serve more customers since not all customers will request to watch a movie during the peak period on every single night.
2. **Customer Request Pattern**. We consider a duration of 6 hours divided into a number of fixed length slots, ranging from 1 to 15 minutes. A customer will request exactly one movie during this period. A request is equally likely to be for any slot.
3. **Total Number Of Channels**. This is the total number of channels available for transmitting video streams.

4. **Number of Emergency Channels**. This is the number of channels reserved for preventing the blocking of pause requests. These channels cannot be used for satisfying initial customer requests to watch a movie.

5. **Length Of The Movie**. The length of the movie determines how long a channel needs to be allocated.

6. **Time Slot Length**. This defines the time intervals on which movies can start. It also defines the maximum amount of time a customer has to wait before a successfully scheduled movie starts playing.

7. **Movie Selection Statistics**. The movie selection process is used to model how customers will go about choosing movies. We have assumed that customers will request the more popular movies more often. We assume the availability of 100 movies. The probability that a customer requests movie i is given by q_i. We assume the following request pattern:

$$q_i = \begin{cases} 0.085 & \text{for } i = 1, \cdots, 10 \\ 0.005 & \text{for } i = 11, \cdots, 30 \\ 0.000714 & \text{for } i = 31, \cdots, 100 \end{cases}$$

8. **Pause Use Statistics**. Once the movie starts, a customer can only affect the playout of the movie by using the pause button. Figure 8 represents the viewing pattern for one customer watching one movie. After a request has been made, and the customer starts watching the movie, the customer's use of the pause function can be modeled in the following manner. The customer watches for a period of time, and then presses the pause button for some amount of time. After releasing the pause button, the customer continues to watch the movie until the next pause time. When and for how long the customer uses the pause button is extremely important because it affects when and how often customers change multicast groups. User behavior can be modeled in several different ways. For our simulator we assumed that when the customer pauses the movie, it is equally likely to be a short period of time, or a long period of time. A short pause duration is uniformly distributed from 30 to 60 seconds, and a long pause is uniformly distributed from 2 to 4 minutes. The time between the conclusion of one pause request and the start of the next is exponentially distributed with a mean time of 30 minutes.

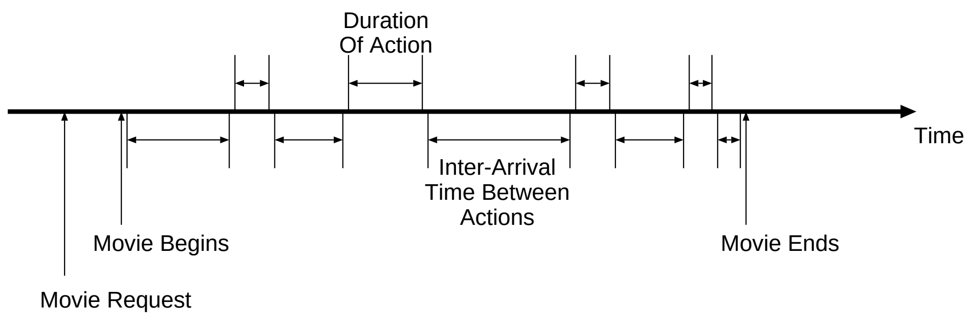

Figure 8: Small scale time-line showing movie playout.

4.2 Performance Measures

We used the following two performance measures:

1. **Initial Request Blocking**. This is the percentage of initial requests for movies that were blocked. A request is blocked when there are no free channels to allocate for a new movie request.
2. **Pause Function Blocking**. This is the percentage of customer pause attempts that were blocked because there were no free channels or emergency channels.

In our numerical examples to follow, we use these two measures to compare the performance of our system with one that allocates one channel per movie request. In such a unicast system, initial request blocking may occur. However, pause blocking will never occur.

5 Numerical Results

In the examples to follow we consider the effect of five parameters on the performance of the system. Table 1 shows the range of values we studied. The nominal values are the ones used when each particular parameter was not varied. The movie selection and pause use statistics were not varied and the values described in Section 4.1 were used throughout.

Factor	Range Of Values Tested	Nominal Values
Number of Requests	480 to 1760 requests	1200
Total Number Of Channels	100 to 220 channels	150
Emergency Channels	0 to 8 channels	2
Length Of The Movie	90 to 130 minutes	120
Time Between Movie Slots	1 to 15 minutes	5

Table 1: The range of values studied, and the nominal value for each factor.

Our results are presented in five graphs. Each graph has three lines: one line for the Request Blocking Probability in the multicast system; one line for the Pause Function Blocking probability in the multicast system; and one line for the Request Blocking Probability in the unicast system. The Pause Function Blocking probability for the unicast system is not included since it is always zero.

5.1 Number of Emergency Channels

Figure 9 shows the impact of increasing the number of emergency channels on the blocking probabilities. Since the unicast system does not use emergency channels, there is no increase in the blocking probability. For the multicast system, as we increase the number of emergency channels, the Pause Blocking probability decreases, and the Request Blocking Probability increases. Since it is important to minimize blocked pause

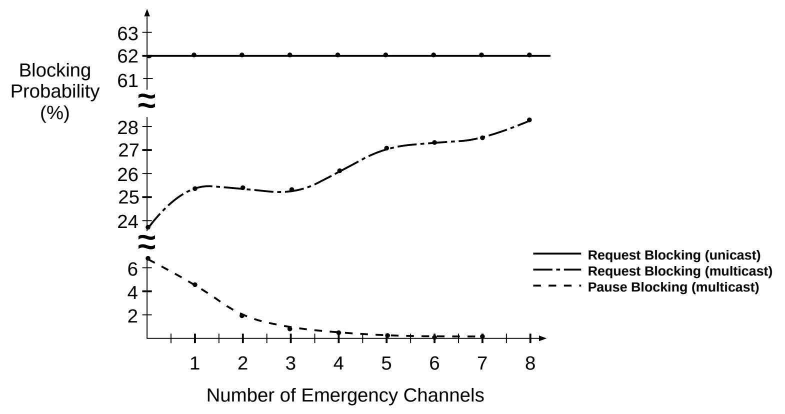

Figure 9: Blocking probability as a function of the number of emergency channels.

attempts without dramatically increasing requests blocked, we decided a good tradeoff would be to allocate 2 emergency channels.

5.2 Number of Customers Making Requests

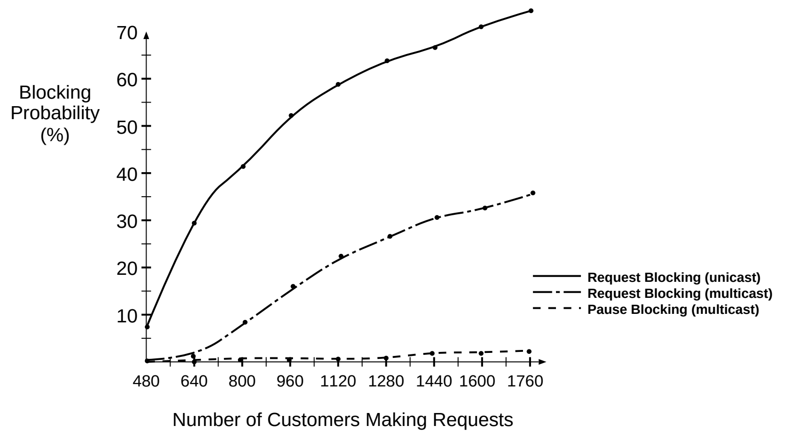

Figure 10: Blocking probability as a function of the number of customers making requests.

Figure 10 shows the impact of increasing the number of customers who make requests. Both request blocking probabilities start out low, but as the number of requests increases, the unicast system initial request blocking probability increases more quickly. The multicast system is much better at handling a large number of customers. The pause blocking rate is relatively stable because the system can only service a maximum number of requests at any one time. Once this maximum is reached, the number of customers who are being serviced will stabilize, and so will the number of pause attempts, and the number of blocked pause attempts. The pause blocking rate is kept even lower since 2 emergency channels are reserved for avoiding pause blocks.

5.3 Number of Channels

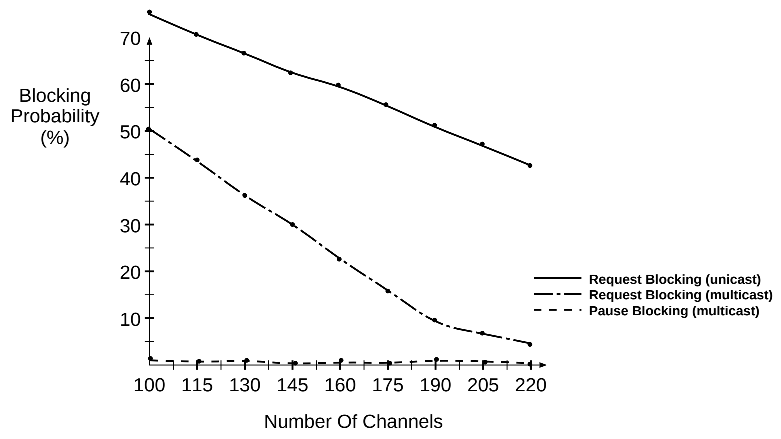

Figure 11: Blocking probability as a function of the number of channels.

Figure 11 shows that as the number of channels increases, the request blocking probabilities decrease. The requests blocked in the multicast system decrease more rapidly and approach a reasonable level much more quickly. A unicast VOD system would require significantly more channels to approach the same level of service. The pause attempts that are blocked remains low because of the 2 reserved emergency channels.

5.4 Length of Movie

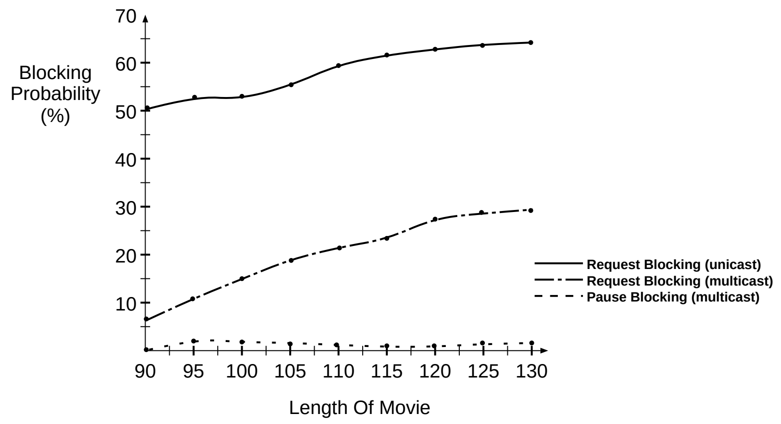

Figure 12: Blocking probability as a function of the length of the movie.

Figure 12 shows that as the length of a movie increases, the number of requests that are blocked also increases. This result is somewhat intuitive because a longer movie requires more channel resources for a longer period of time. The increases in blocking probability for both systems are about the same, and the

Pause Blocking Probability remains consistently low.

5.5 Slot Length

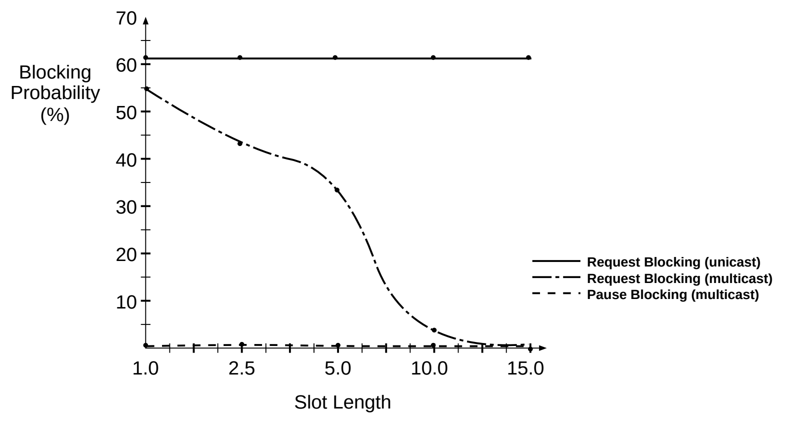

Figure 13: Blocking probability as a function of the slot length.

Figure 13 shows the impact of increasing the time between movie slot times. It is significant because it shows the tradeoff between quick response time and the Request Blocking Probability. The unicast system is not affected by varying inter-movie times because there are no advantages to delaying the video stream. However, in the multicast system, the fewer time slots offered, the more movies requests will be bunched at time intervals. Since a 10 to 15 minute wait to see an on-demand movie is probably unrealistic, a maximum wait time of 5 minutes, and an average wait time of 2.5 minutes seems more reasonable.

6 Concluding Remarks

Some efforts to develop Video-On-Demand systems have been hampered by the high cost of providing individualized service. The two problems that we must solve are the inefficient use of resources, and the inability of the Video-On-Demand system to service a large number of customers. Some solutions suggest increasing the number of resources dedicated to satisfying customer requests, but this makes the system too expensive and therefore, undesirable.

In this paper, we proposed a new type of system that utilizes the advantages of multicast communication. A performance analysis of our system shows that we can service more customers with fewer resources, and provide a reasonable level of interactivity. This allows us to create a system that can service even a large neighborhood of customers for reasonable cost. This is achieved at the expense of somewhat increased system complexity. We believe that our system can be extended to include other features like rewind and fast forward, using techniques similar to ones described in this paper. Our future research will focus on such an extension.

References

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

[2] A. Cook and J. Stern. Optical fiber access–perspectives toward the 21st century. *IEEE Communications*, pages 78–86, Feb 1994.

[3] A. Gelman, H. Kobrinski, L. Smoot, and S. Weinstein. A store-and-forward architecture for video-on-demand service. In *ICC '91*, 1991.

[4] R. Karpinski. Fiber: Not just for telcos anymore. *Telephony: Transmission Special Supplement*, pages 6–14, Dec 2 1991.

[5] A. Karshmer and J. Thomas. Computer networking on cable tv plants. *IEEE Network*, pages 32–40, Nov 1992.

[6] P. Venkat Rangan, Harrick M. Vin, and Srinivas Ramanathan. Designing an on-demand multimedia service. *IEEE Communications Magazine*, pages 56–64, Jul 1992.

[7] J. Wong and M. Ammar. Analysis of broadcast delivery in a videotex system. *IEEE Transactions on Computers*, pages 863–866, Sep 1985.

[8] J. Wong and M. Ammar. Response time performance of videotex systems. *IEEE Journal on Selected Areas in Communications*, pages 1174–1180, Oct 1986.