This is to certify that the

dissertation entitled

Multicast Videoconferencing over
Internet Style Networks

presented by

Hugh M. Smith

has been accepted towards fulfillment
of the requirements for

Doctoral degree in Computer Science
& Engineering

Matthew W. Mutka
Major professor

Date 12/3/99

Multicast Videoconferencing over Internet Style Networks

By

Hugh M. Smith

A DISSERTATION

Submitted to
Michigan State University
In partial fulfillment of the requirements
For the degree of

DOCTOR OF PHILOSOPHY

Department of Computer Science and Engineering

1999

ABSTRACT

Multicast Videoconferencing Over Internet Style Networks

By

Hugh M. Smith

As the Internet has evolved it has moved from being heavily data oriented, to providing a whole range of services including voice, video telephone and videoconferencing. In this dissertation, we investigate whether the Internet can support interactive multicast videoconferencing. Our initial work involves maximizing the network bandwidth utilization when transmitting Variable Bit Rate video. Specifically this research examines smoothing techniques for MPEG compressed video, which maximized the overall utilization of the network while limiting the frame loss rate. We show that our algorithm, called *Pattern Smoothing*, was able to achieve a 75% network utilization while limiting the frame loss rate to less than one frame every 72 minutes.

Following this work, we examined the transmission of Constant Bit Rate video to heterogeneous groups of receivers. This heterogeneity includes both the network and workstation resources. We present an algorithm to control the transmission rate of a single stream, single channel multicast videoconferencing application. This algorithm, called the *Target Bandwidth Rate Control Algorithm*, maximizes the video displayable at all of the receivers while limiting the worse case loss at the low-end receivers.

There are two problems with the single stream, single channel approach. First, since some of the low-end receivers' links will be saturated, this approach will not interact fairly with TCP. In the worst case, a TCP application will be unable to acquire any of the network resources and will time out. Second, since the algorithm maximizes the displayable video for the aggregate, there is still a tradeoff between meeting the needs of both the high-end and low-end receivers. In order to overcome these limitations, we have developed the *Layered Multicast Control Protocol* (LMCP), which maximizes the displayable bandwidth by stripping the video signal across multiple multicast channels. LCMP extends previous work by including both the sender and receivers in the control protocol. We show through simulation that the LMCP achieves essentially optimal bandwidth utilization at the receivers.

One disadvantage of the basic LMCP approach is that it focuses on a single video session and therefore does not share the network fairly with other LMCP controlled sessions. In order to overcome this limitation we have developed an approach that utilizes information from the routers. This approach takes into account four competing priorities. These are 1) Receiver video stability, 2) Network resource sharing, 3) Maximizing the displayable video at the receivers, and 4) Maximizing network utilization. We show through our simulation results that this approach allows the network to be shared fairly among the video applications and adjusts gracefully to fluctuations in available bandwidth.

Table of Contents

List of Figures

List of Tables

Chapter 1

1 Introduction

1.1 Introduction

As the usage of the Internet grows the demand for non-traditional applications such as videoconferencing will increase. In the future, the transmission of video will be commonplace. The question this thesis investigates is what are the limitations and opportunities for supporting multicast videoconferencing on the Internet. Specifically we will look at techniques that will allow videoconferencing applications to maximize their utilization of network resources, while adjusting their transmission rates to meet available bandwidth constraints.

In developing these techniques there are many issues that must be considered. Since raw video consumes a significant amount of bandwidth, some type of compression is required. The type of compression algorithm significantly impacts how the network and receivers handle the transmission, reception and decoding of the video signal. One

choice to be made in determining the compression type is whether the video will be encoded using a Constant Bit Rate (CBR) or Variable Bit Rate (VBR) encoder. In the CBR approach, the video coder adjusts the quantization parameters to increase or reduce the amount of compression in order to keep its output rate constant. This means that during high periods of motion in the raw video, more compression is required and the image quality will degrade. CBR compression lends itself well to a network transmission using a resource reservation system and therefore, a constant bit rate transmission. By reserving network resources equal to the output of the video coder, we can guarantee the delivery of the entire coded signal while maximizing the utilization of the network.

In a VBR encoder, the amount of compressed signal generated by the video coder varies based on scene content. This approach generates a more consistent image quality but puts heavier demands on the network during transmission. There are two basic approaches that may be utilized in order to transmit VBR video. First, using some type of network resource reservation scheme, the sender may reserve enough bandwidth to transmit the peak rate of the coded video signal. This approach guarantees delivery of the entire video signal but wastes considerable network resources. Second, the video stream may be transmitted using some type of available bit rate (ABR) approach. This transmission approach uses multiplexing of multiple sources, possibly both video and non-video sources, across the network in order to increase the network utilization. This increase in network utilization comes at the expense of some loss of signal in the network. This loss is due to congestion caused by multiple sources competing for the limited network resources.

Because of the high demand put on the network by compressed video, there has been significant research into approaches that maximize both the image quality and the network utilization. In this proposal, we present an approach called Pattern Smoothing [45], which takes into account the advantages of both the CBR and an ABR transmission approaches. In this algorithm, we present a technique that maximizes network utilization, supports the transmission of a VBR encoded signal, and reduces the effects of network loss.

One disadvantage of the Pattern Smoothing algorithm is that it transmits the video signal to all receivers at the same rate. This rate is predetermined by the sender of the video signal and does not dynamically adjust based on the receivers' requirements. In many cases, a video conferencing application is multicasting a video stream to a group of heterogeneous receivers. The receivers may differ greatly in the amount of the video signal they are able to receive and process. As an example, resources may vary from a user with an Intel based 486 PC with a 28 Kb/s modem to a high-end Unix workstation using a 155 Mb/s link. Therefore an approach is needed that will adjust the transmission rate based on the current ability of the receivers to utilize the video signal.

To this end we have developed a control algorithm, called the *Target Bandwidth Rate Control Algorithm* [44], which dynamically controls the output rate of the video coder by receiving and processing bandwidth levels from each of the receivers. The goal of this algorithm is to maximize the aggregate amount of video displayable at the receivers.

This algorithm is unique in that it also allows us to limit the worst case loss experienced by the low-end receivers. In addition to considering the available network bandwidth in the feedback information, this algorithm allows for workstation and user requirements to be considered when determining the sender's output rate.

The problem with this single channel transmission approach is that it requires a trade off between heavy packet loss on the low-end receives and under-utilizing the available bandwidth on the high-end receivers. Therefore, this type of approach serves the aggregate but does not meet the needs of the individual receivers. An alternative approach, which has been presented in the literature, is a multiple channel multicast approach. In this approach, the sender stripes a single video stream across multiple multicast connections. Each receiver then selectively adds and drops channels to meet their individual resource requirements.

The drawback to this approach is that it is strictly receiver controlled. The transmission rates of the multiple channels are predetermined and do not vary. Our research involved the development of a protocol, called the *Layered Multicast Control Protocol* [Smith, 1999 #42], which controls the delivery of the layered signal. In this protocol, receivers transmit feedback information about their resource availability to the sender. The sender utilizes this feedback to adjust the transmission rate for each channel in order to optimize the amount of video signal received. Based on analytical and simulation results, we show that this protocol achieves essentially optimal bandwidth utilization at the receivers.

One weakness of the basic LMCP approach is that it does not share the available bandwidth among multiple video sessions. This is due to the LMCP approach's focus on finding a stable rate at the receivers based on the interaction of receivers in a particular video session and not between sessions. Therefore, receivers who have maintained a stable reception rate over time hold their rate even in the face of congestion. In order to overcome this limitation we developed an approach that utilizes information from the network routers in order to fairly share the available bandwidth. Through our simulation analysis we are able to show that this approach allows the network to be shared fairly while gracefully adjusting to fluctuations in available bandwidth.

1.2 Thesis Organization

This dissertation is organized as follows. The next chapter presents a brief overview of research issues and related work on multicast video conferencing. Our research is presented in the remaining five chapters and the appendix.

Chapter 3 presents the *Pattern Smoothing* algorithm. This algorithm maximizes network utilize when transmitting variable bit rate encoded video. Chapter 4 studies a scheme for controlling the transmission rate of a constant bit rate video stream being multicast to a heterogeneous group of receivers. Chapter 5, 6 and 7 improve on the approach presented in chapter 4. Chapter 5 presents a control protocol that utilizes multiple multicast groups in order to maximize the amount of video displayed at the receivers. Chapter 6 addresses the scalability of the receivers feedback utilized by this approach. Chapter 7 continues this analysis by presenting a modified version of this control protocol that not only

maximizes the displayable video but also fairly shares the network between multiple video sessions. Chapter 8 concludes this dissertation and presents future work.

Chapter 2

2 Issues Involved in Videoconferencing Transmission

2.1 Introduction

In the last ten years, there has been significant research and development in the area of digitized video transmission. Due to the bandwidth requirements of raw video, some type of compression is desirable prior to placing the video onto a network. Figure 1 gives an overview of this process. As shown in this figure, the raw video signal is sent into some type of compression (coder) process. Following this compression, the video signal is segmented into packets in order to be transmitted across the network. On the receiver side, the receiver decompresses (decodes) the video prior to its display. In this chapter, we present the technologies needed to implement this process.

2.2 Video Compression

An overview of the encoding processes is presented in Figure 2. In the first step, the raw video signal is transformed into a domain so that it may be more efficiently encoded.

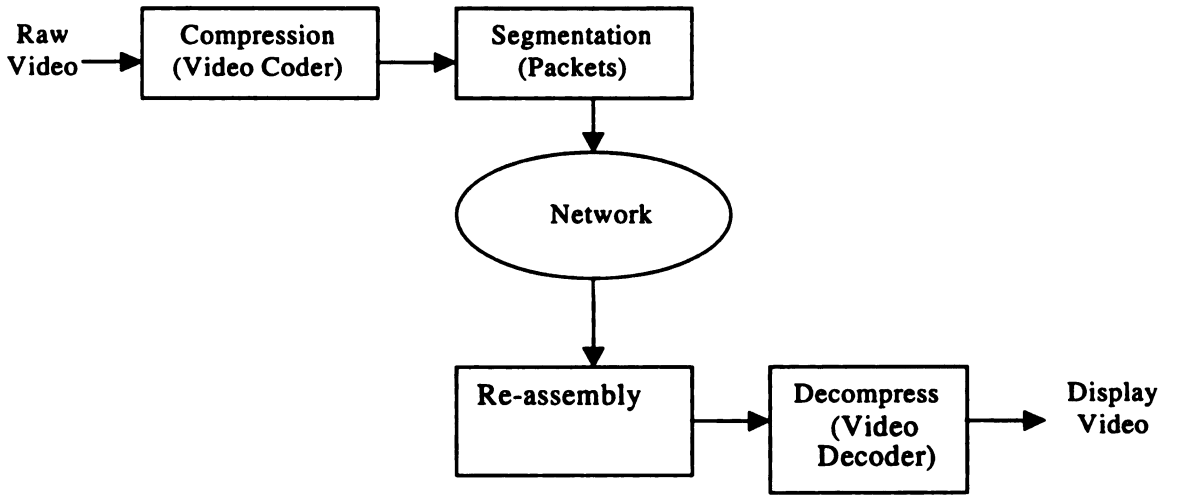


Figure 1: Video Transmission

This step, when done precisely, is considered lossless. By lossless we mean that all information pertaining to the image is retained. The next step is the quantization step. This step is responsible for most of the loss in quality between a compressed image and the original video signal. This step attempts to discard any information that is not needed to obtain the required decoded image quality. Since some precision is lost, the image obtained following this step will be of less quality than the original. The final step in the encoding processes is the entropy coding. In this step, the stream is compressed based on its statistical characteristics. An example of this type of coding is Huffman coding.

There are many characteristics that need to be considered when determining the type of video coder to use for video transmission. These include:

- Constant Bit Rate (CBR) versus Variable Bit Rate (VBR) encoding: The size of compressed video stream is dependent on the precision maintained in the quantization step and the amount of motion in the raw video stream. Therefore, we have a tradeoff. In some encoders, the quantitization step is held constant. Therefore, the output bit rate of the video coder will vary over time. This type of encoder is referred to as a Variable Bit Rate encoder. An alternative approach is to allow the quantization step to fluctuate depending on the amount of motion in the video stream. By monitoring this motion, these types of encoders are able to produce a Constant Bit Rate (CBR).
- Intra-Frame versus Inter-Frame encoding: In the simple encoder discussed previously (Figure 2), each frame of the video signal was encoded separately. This type of encoding is considered intra-frame encoding since only the current frame is required to decode a frame. In many cases, there is little difference between two adjacent frames. This lends itself to an inter-frame encoding scheme. In this approach, a frame is compared to those around it and only the differences are encoded. This is shown in Figure 2 as feedback following the entropy coding step. While this type of encoding can significantly improve the coder's compression rate, it introduces complexity and delays into the transmission process.

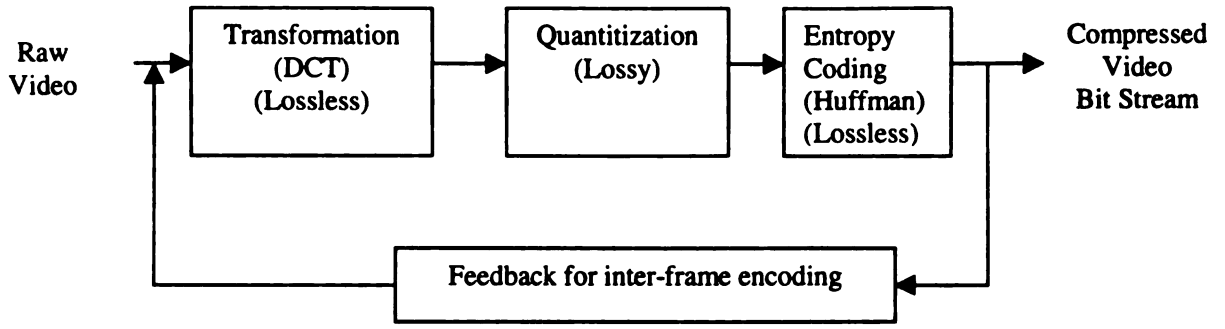


Figure 2: Video Compression (Modified from [43])

- Output Stream Dependencies: In order to transmit the compressed video stream across the network it must be broken up into packets. The sizes of these packets are dependent on the limitations of the network between the video sender and receivers. This causes dependencies between the packets. By dependencies we are referring to the number of packets that must be successfully received in order to decode a frame. These dependencies may be across multiple frames for an intra-frame encoder to less than one frame for some encoding techniques [33].

Keeping these important characteristics in mind, we will now look at some representative approaches to video compression.

2.3 VBR MPEG Compression

The Moving Picture Experts Group (MPEG) has developed multiple standards for video compression [19, 35, 53]. In our work we have been strictly concerned with the MPEG I standard. This compression technique supports both inter- and intra-frame encoding.

And while it is possible to force the encoding to output a CBR stream, MPEG I encoders are generally used to generate a VBR output stream.

An MPEG encoder outputs three types of frames: I (Intra-coded), P (Predicted) and B (Bi-directional). The I frames are intra-framed encoded and therefore do not require any other frame to be present in order to be decoded. P and B frames use both intra-encoding and inter-encoding techniques. P frame encoding uses the previous P or I frame as a reference. B frame encoding information may be obtained from either the proceeding or following P or I frame. Due to the differences in frame encoding techniques, there are large variations in frame sizes. I frames can be an order of magnitude larger than B frames.

MPEG compression groups frames into short sequences called patterns. These patterns begin with an I frame and then contain P and B frames. These patterns may be 1 to 15 frames long and the pattern is repeated throughout the entire video sequence. Patterns are described by two parameters: N represents the distance between I frames, and M the distance between I or P frames [25]. For example, a pattern with N = 6, M = 2 is IBPBPBPBIBP..., and a pattern with N = 1, M= 0 generates IIII....

2.4 VBR Transmission and Smoothing

There are two approaches for transmitting VBR compressed video, such as MPEG I. One is to have the network allocate a Constant Bit Rate (CBR) equal to the peak rate of the video sequence. The other is to use statistical multiplexing and to transmit the video

using an Available Bit Rate (ABR) approach. Due to variations in the compressed video frame sizes, both CBR and ABR transmission techniques do not achieve efficient utilization of the network. The solution to this problem is to smooth the video stream prior to transmission.

There are two approaches for smoothing compressed video. The first approach attempts to smooth the video sequence in order to achieve a constant rate for the entire video sequence and transmit it using CBR transmission. Unfortunately, achieving a constant rate is impractical. The best this approach can do is to minimize the number of transmission rate changes throughout the video. The second approach smoothes the video sequence by decreasing the peak rate and variance of the video stream and transmitting it using ABR transmission. This decrease improves the gain in network utilization achieved by statistical multiplexing multiple VBR sources over a network link using ABR transmission [25, 34].

Smoothing algorithms use a combination of three general techniques [39]. The first is temporal multiplexing, which involves inserting a smoothing buffer somewhere between the sender and receiver. The second smoothing technique is statistical multiplexing, which is accomplished by transmitting video from multiple sources over a single link. The third technique employs work-ahead. In this approach, the data must be prefetched and the receiver must have buffer space available. The data is then sent at a nearly constant rate that does not overflow or starve the receiver's buffer.

There are many attributes that need to be considered when developing a smoothing algorithm. One of these is the support of live video. Many of the smoothing algorithms developed do not support live video; they work only on stored video, where look-ahead processing can be performed. Startup delay and client and receiver buffer size also must be considered. Another important attribute of a smoothing algorithm is whether it is lossy or lossless. Some smoothing techniques allow for data to be lost or dropped. Other techniques are lossless and deliver the entire video stream intact. A final attribute is the level at which smoothing is accomplished. Techniques range from smoothing at the sub-frame level up to smoothing across large chunks of the video sequence. In chapter 3, we present an overview of three different smoothing algorithms and compare their key attributes.

2.5 CBR Hierarchical Compression

One difficulty with many types of video compression is the amount of the video stream that must be received prior to decoding the image. Many types of compression techniques require the entire frame (or more in the case of MPEG encoding) be received. This forces all receivers of the video stream to be able to receive and process the video at a uniform rate. This restriction has led to the development of constant bit rate (CBR) approach that uses a hierarchical encoding scheme.

In a hierarchical encoder, the video image is compressed so that lower quality images of the video are embedded in the lowest bits of the video stream. The receiver decodes the video stream sequentially starting at the lowest quality image. In this way a receiver may

discontinue decoding at any time and produce a copy of the image. The quality of the image will improve as the amount of the video signal decoded increases. One advantage of this approach is that it allows the sender to split the video stream into multiple CBR components and transmits each component on a separate connection. The video receiver then adds channels based on their resource availability.

J. Shapiro [43] first presented a hierarchical encoder in 1993. His approach, called the Zerotree Wavelet Algorithm (EZW), produced a fully embedded signal. Using this algorithm the coder can discontinue coding the image at any time and produce an image that may be decoded at the receivers. In addition, the receiver can stop receiving a video signal at any point and decode the image. By increasing or decreasing the amount of video received, the receiver is able to adjust the image quality to meet its resource requirements. One difficulty with this approach is the complexity of the coding and decoding processes. Vishwanath *et al.*, built on this idea in developing a hierarchical compression algorithm that combined the Discrete Wavelet Transform with a Hierarchical Vector Quantization [51]. This approach is unique in that the encoding and decoding is significantly less complex and many be implemented via table lookups.

McCanne, *et al.*, extended this work in [32, 33]. The benefits of their hierarchical encoding algorithm includes low encoding/decoding complexity, high loss resilience and good compression. Their approach uses a DCT/wavelet based algorithm to implement a layered coding scheme. This approach, called Progressive Video with Hybrid transform, first develops a base layer and then follows with a number of refinement passes. The real

contribution from this work comes from the fact that the authors jointly developed the hierarchical encoding algorithm along with a protocol to allow the receivers to control the amount of video they receive. This combined approach, called Receiver Layered Multicast (RLM) allowed them to show the feasibility of supporting a heterogeneous group of receivers by utilizing a hierarchical encoder. We will present the RLM protocol later in this chapter.

2.6 Internet Multicast Support

IP Multicast is used in order to distribute videoconferencing traffic from one source to many receivers. In a multicast approach, a single stream is sent to many receivers [10]. Unlike a broadcast approach, IP multicast supports the paradigm of group membership. In this paradigm a receiver may join or leave a multicast connection without the knowledge of the sender. This allows the sender to be unaware of the receivers of the multicast transmission. The sender blindly transmits the data to a network address specially designated for use by multicast traffic. The receivers are responsible for joining and leaving the multicast group by making the appropriate requests on their local hosts.

An advantage to this approach is that it scales well to large groups of receivers since the sender is not required to be aware of the receivers. In addition, compared to a broadcast approach where the data is transmitted over all links, in a multicast transmission, data is transmitted only over links leading to an active receiver. When a receiver joins a multicast group, a request to add this connection is propagated from the receiver toward the multicast source. This multicast request message continues until a router currently

taking part in the multicast session is reached. Only then is data routed down a network link toward the receiver. A similar process is used to allow a receiver to leave a multicast group.

The concept of joining and leaving a multicast group was improved in the Internet Group Management Protocol (IGMP), version 2 [6, 15]. This version of the protocol specified a low-leave latency for multicast connections. This improvement is necessary for protocols that require a receiver to quickly add or drop a multicast connection.

Two disadvantages to IP multicast are that it only provides unreliable packet delivery and does not provide for congestion control. There has been significant research into the development of reliable multicast protocols [18, 26, 36, 37, 47, 49]. A common theme in all of these approaches is the receiver requesting the retransmission of a lost data from either the source or other receivers in the multicast group. The MESH protocol developed by Lucas, *et al.* was developed to specifically address the real-time needs of multicast video in wide area networks. In this approach, the network is broken into logical groups based on the locality of the receivers. In each logical group a receiver is designated as the active receiver (AR) and is responsible for handling the processing of lost messages. Due to the real-time nature of videoconferencing traffic, the delay induced by this type of approach significantly limits its usefulness.

2.7 Multicast Video Congestion Control

Many algorithms for controlling the amount of bandwidth used by a multicast videoconferencing application have been proposed over the past few years. These approaches may be classified based on who manages the control algorithm, the sender [1, 3-5, 23, 24] or receiver [7, 31, 33, 42] (see Figure 3). A sender-based algorithm relies on measurement of the current availability of resources from the network or receivers. These algorithms follow the same basic pattern. Feedback is received by the video sender indicating whether there is congestion between the source and the receivers. The sender then calculates an aggregate metric indicating the level of congestion for all receivers. Then using a predetermined threshold the sender increases, decreases or holds constant its transmission rate.

Once the new transmission rate has been calculated, a mechanism to modify the video

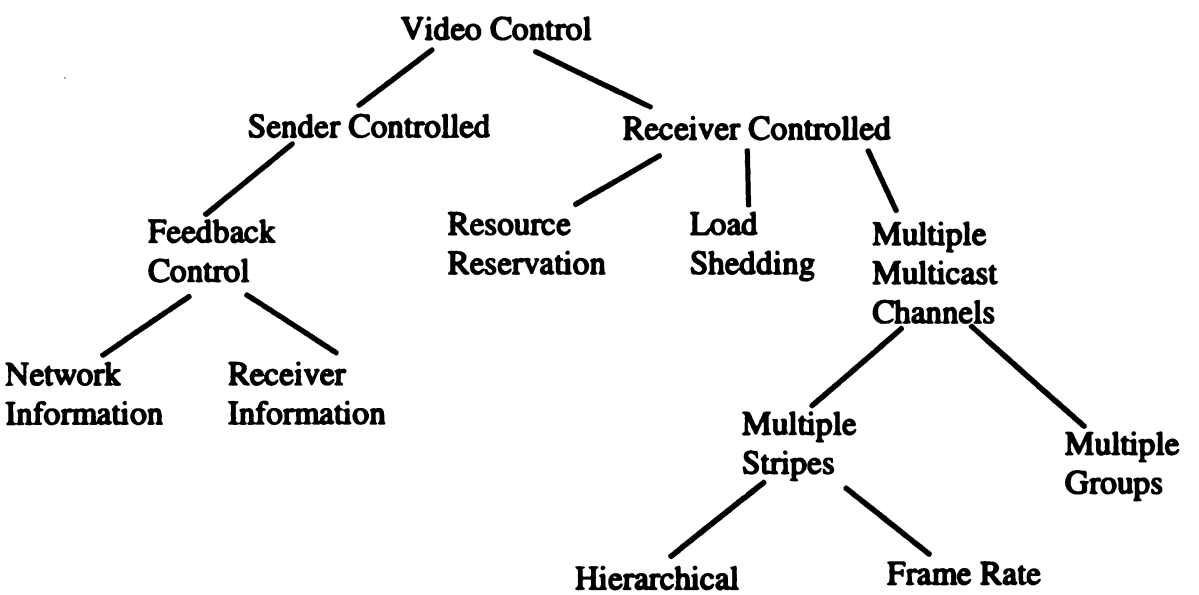


Figure 3: Control Algorithms for Multicast Videoconferencing

stream needs to be considered. There are many attributes which may be adjusted in order to adapt the coder's output rate to meet the new transmission rate. These attributes may be classified as either temporal or spatial [33]. The temporal attributes concern adjusting the rate at which frames are encoded. The spatial attributes relate to the quality of the image transmitted. These quality variables include the number of bits used to represent the color in the image, the number of pixels in the image, the compression algorithm's key frame distribution, and the amount of lossy compression (quantization) applied to the image [20, 21]. Since each of these attributes affect the coder's output rate differently, the sender dynamically determines the best setting based on the calculated transmission rate.

The problem with the sender-controlled approaches is that they do not take into account the severity of loss at the receivers. The feedback only indicates whether or not the receiver is experiencing congestion. We show in chapter 4 that this limitation may translate into unacceptable image quality at the some of the receivers.

2.8 Receiver Based Congestion Control

While there are many methods that a receiver may use in order to manage the amount of video received, in this proposal we are only concerned with a layered multicast approach. In this approach, the video stream is transmitted by striping it across multiple multicast channels (see Figure 4). The receivers are responsible for adding and dropping connections to meet their resource requirements. By striping, we mean splitting a single

video signal into multiple segments and spreading the segments across the multiple channels.

There are two basic methods for striping a video stream: temporal striping and spatial striping. Temporal striping means each segment is one video frame and the sender stripes the frames across the multicast channels as shown in Figure 5-B. Receivers increase their frame rate by adding additional multicast channels. In spatial striping, each video frame is split up into multiple segments (see Figure 5-A). The sender then transmits segments from the same video frame on different multicast channels. A receiver may increase the quality of the video received by adding additional channels. Significant research has been done in each of these areas individually and in combining the two approaches [33] (see Figure 5-C). In this paper, we focus mainly on spatial striping.

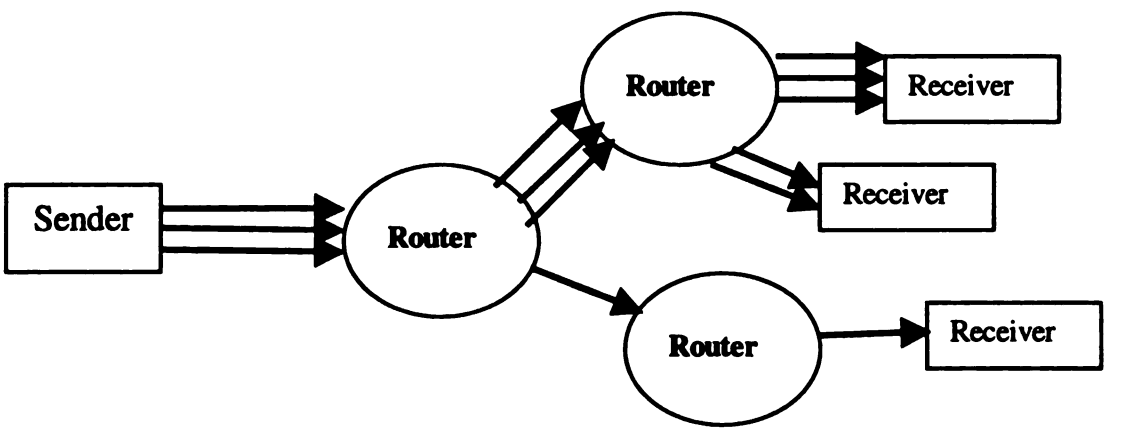
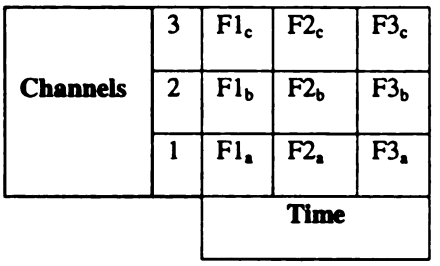
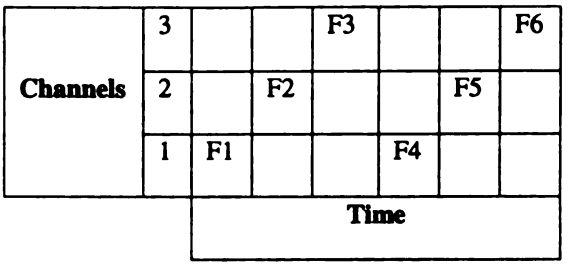


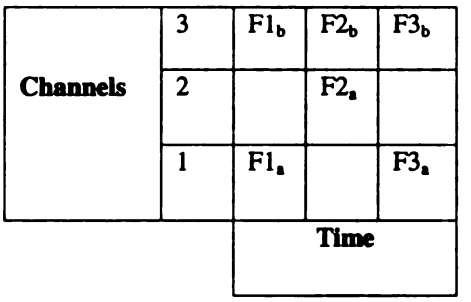
Figure 4: Layered Multicast Transmission (modified from [31])



Spatial Striping (A)



Temporal Striping (B)



Both Spatial and Temporal Striping (C)

Figure 5: Temporal and Spatial Striping

(F1, F2, F3 are 3 video frames and $F\#_a$, $F\#_b$, $F\#_c$ are three layers of frame F#)

In order for a video image to be striped spatially across multiple multicast connections, the video-encoding algorithm must support the ability to segment the video stream into multiple components. In addition, the video decoder must be able to decode the stream without requiring all segments to be present. As presented earlier, one approach to spatial encoding is hierarchical or embedded encoding (initially presented by J. Shapiro [43]).

2.9 Receiver Driven Layered Multicast

One proposed method for managing the layered transmission of video is the Receiver-Driven Layered Multicast (RLM) protocol [31]. This protocol focuses on how the receivers add and drop network connections in order to meet their bandwidth restrictions. This approach assumes the network supports IP-Multicast with fast join-leave capabilities as developed in the Internet Group Management Protocol (IGMP), version 2 [15]. This fast join-leave capability allows the receivers to quickly add or drop a multicast layer in order to meet their changing bandwidth requirements.

The RLM protocol utilizes a layered compression scheme such as those described earlier. The layered encoded video signal is multicasted across a number of network connections. The receiver component of the protocol is responsible for adding and dropping network connections based on packet loss. When network congestion exceeds a threshold, a receiver will drop its highest multicast connection. In order to add additional layers, the RLM protocol uses "join-experiments". The join-experiments test whether a receiver is able to support an additional channel. These experiments are run periodically, using an exponential backoff timer to allow the receiver to achieve a level of stability. During a join-experiment, the receiver joins the next highest multicast group. If the receiver experiences no packet loss, then the experiment is considered a success and the receiver maintains the new connection. If the receiver experiences packet loss then the experiment is said to have failed and the new connection is dropped.

Vicisano, *et al.* [50] built on the RLM protocol and implemented a TCP-like layered multicast congestion control protocol. This protocol is similar to the RLM approach in that control is left entirely to the receivers. This paper provided two important contributions. First, they introduced the idea of Synchronization Points (SP). Instead of receivers independently running "join-experiments," the sender coordinates these experiments. They show that this approach allows the receivers to converge faster to their maximum reception rate. Second, they developed the idea of "sender initiated probes." Just prior to a SP the sender doubles the bandwidth sent to the receivers. These bursts (or probes) have the effect of a join-experiment but with a much shorter duration. These probes allow the receivers to estimate the probability of a successful join-experiment during the next SP. If congestion occurs during a probe the receiver does not perform a join-experiment during the following SP.

Chapter 3

3 Pattern Smoothing for VBR Compressed Video Transmission

3.1 Overview

We introduce in this chapter an algorithm to maximize the transmission of VBR compressed video. This algorithm, called Pattern Smoothing, transmits compressed video via both Constant Bit Rate (CBR) and Available Bit Rate (ABR) channels. To take advantage of the gains achieved through statistical multiplexing of multiple sources over a single link, this algorithm utilizes a CBR channel to reduce the peak rate and variance of the Variable Bit Rate (VBR) of the video stream.

In addition to presenting this new algorithm, we compare it against three smoothing techniques presented in the literature. Key attributes used for comparison include receiver buffer size, live video support, startup delay, losslessness versus lossiness, and smoothing scale. Because network utilization is the most important performance metric for any smoothing algorithm, we provide a performance analysis of the Pattern

Smoothing algorithm via simulation and compare these results to the best of the three presented smoothing algorithms.

3.2 Background

There are two approaches to compressed video transmission. One is to allocate a Constant Bit Rate (CBR) equal to the peak rate of the video sequence. The other is to use statistical multiplexing and to transmit the video at a Variable Bit Rate (VBR) over an ABR channel. Due to video compression techniques, there is a large variation in compressed video frame sizes. Because of this variation in frame sizes, both CBR and VBR transmission techniques do not achieve efficient utilization of the network. The solution to this problem is to smooth the video stream prior to transmission.

There are two approaches for smoothing compressed video. The first approach attempts to achieve a constant transmission rate for the entire video sequence. Unfortunately, achieving a constant rate is impractical. The best this approach can do is to minimize the number of transmission rate changes throughout the video. The second approach smoothes the video sequence by decreasing the peak rate and variance of the video stream. This decrease improves the gain in network utilization achieved by statistical multiplexing multiple VBR sources over one ABR network link [25, 34].

As discussed in chapters 2, smoothing algorithms combine three general techniques. These are temporal multiplexing, statistical multiplexing and work-ahead. While all smoothing algorithms use a combination of these three techniques, their actual

implementation involve tradeoffs in such things as the algorithms ability to support live video or whether or not VCR controls can be provided. Other attributes that must be considered include network utilization, receiver side buffer size and whether the algorithms are lossy or lossless.

In this chapter, we discuss three different smoothing algorithms that have been presented in the literature. In addition, we present a new smoothing algorithm, called Pattern Smoothing that uses both CBR and ABR transmission to achieve high network utilization.

The remainder of this chapter is broken into the following sections. In Section 2, we describe three smoothing techniques that have been presented in the literature. In Section 3, we present a new smoothing technique called Pattern Smoothing. This technique has the advantage of being simple to implement, working with interactive video, requiring minimal receiver buffering, and achieving a high network utilization. Section 4 presents a comparison of the key attributes of each of the algorithms presented. The fifth section covers a performance analysis of the Pattern Smoothing algorithm.

3.3 Smoothing Algorithms

In this section, we describe three smoothing algorithms. These algorithms are referred to as Lossless Smoothing (LLS) [34], Critical Bandwidth Smoothing (CBS) [13, 14], and Optimal Smoothing (OPS) [39]. The LLS algorithm utilizes look-ahead techniques to

reduce the variance in the video transmission stream. This algorithm is based on a given delay bound, the number of frames with known size, and a fixed look-ahead interval. LLS smoothes across one MPEG pattern. The goal of the algorithm is to send an entire pattern at a nearly constant, while meeting the delay bounds and continuous service properties. By sending the patterns at a nearly constant rate, the algorithm tries to minimize the number of rate changes in the video stream. While the LLS algorithm smoothes individual video streams well, no data was provided on the effects of multiplexing multiple streams over a single link. In order to achieve higher network utilization, some type of multiplexing must be used. Therefore, while the smoothing technique is lossless, the actual implementation using statistical multiplexing of many sources may be lossy.

The second smoothing technique is CBS [13, 14]. Intuitively, providing enough buffer space allows an entire video segment to be transmitted at a constant rate. In [13, 14] this idea is evolved into the concept of "Critical Bandwidth Allocation." The authors in [13] define critical bandwidth as "the minimum constant bandwidth necessary to play a video clip through without starvation." Their algorithm determines a critical point, defined as the last frame for which the average frame size for it and all prior frames is maximized. By determining the critical points and recursively breaking up the video at these points, they are able to determine the critical bandwidth for the entire video. This algorithm has the advantage of keeping the bandwidth constant for large periods of time. One disadvantage of this approach is that large buffers are required on the receiving end. As in the LLS technique, this smoothing algorithm is lossless by itself. To achieve higher

network utilization, statistical multiplexing may cause the actual implementation to be lossy.

The third smoothing algorithm is OPS [39]. The OPS algorithm focuses on providing the "greatest possible reduction in rate variability" based on a given receiver buffer size. This algorithm smoothes across the entire video sequence. The key parameters for the OPS technique are the delay bounds and the receiver buffer size. Using these parameters, the OPS algorithm determines the upper and lower bounds for transmitting a video sequence given a fixed size receiver buffer. It then calculates a transmission rate that varies the least number of times and that does not starve the receiver or cause buffer overflow. The OPS algorithm is proven to be optimal in terms of having the minimum peak rate and smallest variance. The algorithm presented in [39] was able to achieve a 75% network utilization with a 1 Mbyte receiver buffer without data loss. The only real downfall of this algorithm is its inability to support live video.

3.4 Pattern Smoothing

In this section we present a new smoothing technique called Pattern Smoothing (PS). This technique smoothes the video across a single pattern only. It uses both work-ahead and statistical multiplexing techniques to smooth out the video stream and to achieve higher network utilization. Because it uses statistical multiplexing, it is possible for frames to be lost. These losses can be kept very small (1 frame every 15-30 minutes or more) and are restricted to P and B frames. The advantages of this technique include

simplicity, ability to handle interactive video, high network utilization and small buffer requirements.

As its name suggests, Pattern Smoothing smoothes a compressed video stream across one MPEG pattern. In this scheme, the video is divided into variable bite rate (VBR) and constant bit rate (CBR) components. By stripping out a CBR component of the video, we can increase overall network utilization by decreasing the variance and peak rate of the VBR component. The VBR component allows us to take advantage of statistical multiplexing. Multiplexing multiple VBR sources over one ABR channel enables us to improve overall network utilization.

The basic approach is to fill the CBR channel with data from the frames in the pattern. Any remaining data is sent over the ABR channel. This is practical in networks, such as ATM, because both CBR and ABR virtual channels can be allocated within a virtual path for a transmission stream. The minimum size of the CBR channel is equal to the maximum I frame size in bits times the number of patterns in a second. The resulting CBR channel size, in bits per second, is sufficient to transmit the largest I frame spread across one pattern time. (Figure 6 shows the operation of the PS algorithm). The I frame is prefetched one pattern in advance and is then slowly transmitted over the CBR channel during the entire patterns transmission time. The I frame is then buffered until it is time for it to be displayed. Consequently, the receiver must have a buffer large enough to hold the largest compressed I frame. Any remaining capacity on the CBR channel is filled with bytes from the P and B frames of the current pattern. By increasing the size of the

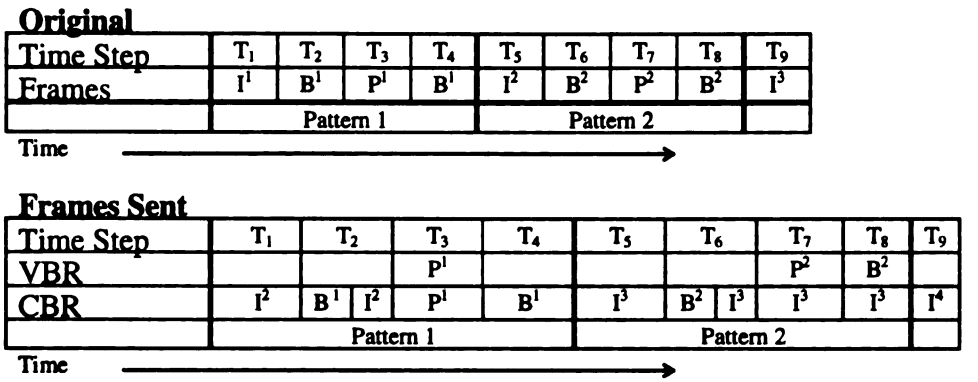


Figure 6: Pattern Smoothing Frame Transmission Schedule

CBR channel we are able to decrease the amount of traffic sent over the ABR channel. Increasing the CBR channel size will improve the performance of the ABR channel by decreasing the peak and variance of the transmitted bursts. However, the increase in the CBR channel size may cause some of the CBR channel to go unused, and consequently, decrease the overall network utilization (see Figure 7).

Increasing the size of the CBR channel allows us to pack bytes from additional frames into this channel. Since P and B frames are not buffered on the receiving end, they must be sent only during their designated time slot. Therefore, if the P or B frame size is larger than the available CBR capacity during the specific time slot, the P or B frame must be split between the CBR and ABR channels. Consequently, there are three ways a frame can be sent: all CBR, all ABR or a combination of ABR and CBR. Since a frame may be split across both channels, the receiving side needs to determine where to distribute the incoming packets. This determination can be done by putting header information into each packet prior to sending the packet onto the network.

Figure 7: VBR and CBR Channel Usage

The choice of which frames from the pattern are packed into the extra CBR capacity may be handled in many different ways. The approaches examined include sequential, P frames first, and random. In the sequential approach, cells are taken from the pattern sequentially and packed into the CBR channel until it is full. In the P frames first technique the extra CBR capacity is first distributed to the P frames and then any remaining capacity is used by the B frames in sequential order. In the random approach, P and B frames from the pattern are picked randomly and placed into the CBR channel. The intention of each of these approaches is to improve the network utilization of the ABR channel by reducing the peak rate and variance of the traffic. After simulating all three of these approaches, no significant performance difference was observed.

3.5 Smoothing Algorithm Comparison

In this section, we present a comparison of the key attributes of the four smoothing algorithms. Table 1 shows an overview of this comparison. The first attribute is the support of live video. Neither the CBS nor the OPS Algorithms support live video. Both

Smoothing Approach	Live Video	Smoothing Scale	Buffer Size	Startup Delay	Multiplexing Required	Lossless	VCR Controls	Bandwidth Utilization
Lossless Smoothing	Yes	1 Pattern	1 Pattern	small	Yes	Yes*	Possible**	NA
Critical Bandwidth Smoothing	No	Long Sequence	2Mb - 32Mb	Variable	Yes	Yes*	More Difficult	NA
Optimal Smoothing	No	Entire video	64KB - 1Mb	Variable	Yes	Yes*	More Difficult	< 85%
Pattern Smoothing	Yes	1 Pattern	64KB	1 pattern	Yes	Yes*	Possible**	< 75%

* The smoothing algorithm itself is lossless, but adding statistical multiplexing to improve network utilization my cause the smoothed transmission to become lossy.

** While VCR controls are possible, different schemes and changes in bandwidth requirements will need to be addressed.

Table 1: Smoothing Algorithm Comparison

algorithms smooth across long sequences of the video. Therefore, these techniques only work on stored video, where some type of look-ahead preprocessing can be performed. The LLS and PS techniques smooth across relatively short sequences of the video. Therefore, if a short delay (166 msecs, about 5 frames) is acceptable then these algorithms can be used with live video.

The second attribute is the smoothing scale. The smoothing scale involves the amount of look-ahead and work-ahead required to smooth the video sequence. Both LLS and PS smooth across one MPEG pattern. Therefore, they only reduce the variance between frames in a pattern. The CBS and OPS techniques, in addition to reducing the frame to frame variability, smooth across scene changes in the video. These scene changes can induce significant frame size fluctuations over time. These algorithms account for this fluctuation by smoothing across large video sequences.

The next attribute is the receiver buffer size. The smoothing scale size determines the size of the receiver buffer. Since LLS smoothes across only one pattern at a time, a buffer large enough to hold one pattern is required. CBS segments the video into large sequences. The receiver buffer size is dependent on the size of the frames and the number of segments into which the video is partitioned. The authors of CBS in [13] showed the effects of buffer sizes between 5 Mb and 40 Mb for the *Star Wars* video. When using the CBS algorithm, the reduction in bandwidth increases as the buffer size grows. While the OPS algorithm provides some benefits without a receiver buffer, a 1 Mb buffer provides excellent smoothing gain and only a minimal additional gain would be achieved by using a larger buffer. In PS, only the I frame needs to be buffered. Therefore, a receiver buffer large enough to hold the largest compressed I frame is required. In this analysis, using a video segment from the *Wizard of OZ*, a buffer size of 64 KB was sufficient.

The startup delay of a smoothing algorithm is determined by the amount of time necessary to fill the receiving buffer so that the buffer is able to continuously feed the receiver's video decoder. The LLS and PS algorithms, due to their small smoothing scale, require short startup times. CBS and OPS algorithms have variable startup delays. If a small startup latency is required, a higher initial bandwidth will be needed. If some type of delay is acceptable, then a smoother startup transmission may be achieved.

Both the LLS and CBS algorithms do not address the transmission of multiple video streams over a single network. Due to the fact that the smoothed streams still have a considerable amount of variability, some type of multiplexing is necessary in order to achieve a satisfactory network utilization. If peak bandwidth allocation was used instead, significant bandwidth would be wasted. The OPS algorithm allows for multiple sources to utilize a single link without data loss. While this scheme achieves a high network utilization, the authors show in [55] that by utilizing statistical multiplexing a 10-60% additional gain may be achieved. By design, the PS algorithm is intended to use both a CBR and a multiplexed (ABR) channel.

All four smoothing algorithms were developed to smooth the video stream without loss of information. Depending on the Call Admission Control algorithm (CAC) and the Quality of Service (QoS) guarantees provided, some data loss in the network may be expected. In the LLS, CBS and OPS approaches, this data loss may come from any frame type, including I frames. Since I frames are required for decoding and playback of an entire pattern, loss of one I frame will be more noticeable than the loss of any other frame type. PS was specifically developed for use with multiplexed links. The data loss for this algorithm is limited to P and B frames.

VCR controls were not specifically addressed by any of the four smoothing algorithms. A simple way to implement controls such as fast forward and fast reverse is to step outside the smoothing algorithm and use different techniques. Problems may arise when normal playback, at some random point in the video, is needed following one of these

operations. Because they smooth over smaller sequences (1 pattern), the LLS and PS techniques allow normal playback to begin with minimal delay. For the longer scale smoothing algorithms, CBS and OPS, additional effort is needed to adequately fill the receiver buffer to guarantee continuous playback.

The final attribute in Table 1 is network utilization. The LLS and CBS algorithms did not provide any utilization statistics. The focus of their analysis was to reduce the variance in the video transmission from a single source. The OPS algorithm provided a detailed utilization analysis. Their algorithm performs extremely well and effectively utilizes the available bandwidth. The goal of all three algorithms, LLS, CBS, and OPS, is to reduce the variation over time of the video transmission. Since the OPS algorithm is proven to be optimal in the reduction of peak rate and variance, we may assume that it performs as well, or better than, either of the other two algorithms given a fixed buffer size and delay bound. The next section provides a detailed look at the bandwidth utilization of the PS algorithm and compares the results to the OPS algorithm.

3.6 Pattern Smoothing Simulation and Performance

In this section, we present a performance analysis of the Pattern Smoothing algorithm. First, we provide an overview of the simulation model used to generate the performance statistics. Following this, we present the simulation results and compare these results to the Optimal Smoothing algorithm.

3.6.1 Simulation

The simulation of the Pattern Smoothing Algorithm is divided into CBR and VBR processes. The input to the CBR process, is a file describing a video sequence. For the simulations, we used a 7 minute (12,600 frames) segment of the Wizard of Oz. Each record in the file contains the frame type (I, P, B) and frame size. The CBR channel bandwidth, in bits per second, is also entered into the CBR process. The CBR process then packs frames into the CBR channel. The output of this process is a CBR utilization percentage and a VBR file containing the VBR component of the video. During the simulation runs, we used five different data files, each containing a different MPEG pattern type.

The VBR simulation process is more complex. To simulate the VBR process, we used CSIM, which is an event driven simulator. In order to simulate the network, we used a Burst-Oriented CAC scheme [16]. In this scheme, bursty traffic is handled at the burst level. In this scheme, bandwidth is reserved, using a fast bandwidth reservation scheme, on a burst by burst basis. If inadequate bandwidth is available to handle the burst, then the entire burst is dropped. In the simulations, a burst is one frame. The focus of dropping bursts rather than cells introduces the concept of Burst Blocking Probability (BBP) [16].

BBP is the probability that a burst will be blocked when it tries to enter the network. Therefore, the goal of the Burst-Oriented CAC scheme is to guarantee a maximum BBP. In the simulation, we used a target BBP of $1x10^{-4}$. While frames were

transmitted at 30 frames per second, in the simulations the single source VBR transmission rate ranged from 5.7 to 10.0 frames per second. All other frames were transmitted on the CBR channel. This VBR transmission rate and target BBP gives us a target frame drop rate of 1 frame every 16.7 to 29.2 minutes. The simulation results show smaller BBP values such that the frame drop rate occurred much less often, in the range of one frame every 72 to 366 minutes.

The input into the VBR simulator is the VBR file generated by the CBR simulation, the ABR channel size, the number of sources, and the number of servers. The number of servers determines the maximum number of bursts that can be serviced concurrently. The servers' bandwidth is set equal to 10 times the average frame size. If a server is not available, then the burst (frame) is considered to be blocked and is dropped. In each simulation we ran a total of 20 batches with 60,000 frames per batch. The first batch was dropped to remove any startup anomalies and the remaining batches were averaged together to generate the final results. The output of the VBR simulation is total ABR bandwidth utilization and total BBP.

3.6.2 Performance

In this section we present a performance analysis of the Pattern Smoothing algorithm. This analysis compares the performance results to simulations of a "No Smoothing" and Optimal Smoothing approach. For all three simulations, we used a channel bandwidth of 127.155 Mb/s. In these simulations, frames were transmitted

Video File	MPEG Pattern	BBP	Bandwidth Utilization	Number of Sources
1	N=2,M=1	(no drops)	7%	2
2	N=4,M=2	0.000034	13%	4
3	N=6,M=3	0.000026	13%	8
4	N=10,M=2	0.000015	16%	10
5	N=15,M=3	0.000013	16%	12

Bandwidth = 127.155 Mb/s

Frame drop rate range: 1 frame every 16 to 42 minutes
(excluding file 1 - which had 0 drops)

Table 2: No Smoothing Performance

at 30 frames per second. All three approaches were simulated on five video files. Each video file contained a different MPEG pattern.

The simulations for both "No Smoothing" and Pattern Smoothing use a Burst-Oriented CAC with a BBP target of $1x10^{-4}$. To determine the maximum number of sources supported for each pattern type, the number of sources was increased until the target BBP was exceeded. The 'No Smoothing' performance results represent the base case in which no smoothing is performed. Therefore, no CBR channel or receiver buffer is required. Table 2 shows that the "No Smoothing" implementation achieves a peak bandwidth utilization of 16% and supports a maximum of 12 sources.

Table 3 shows the performance results for the Pattern Smoothing simulations. In these simulations, the CBR channel size for a single source varied from the smallest of 1.14 Mbs for file 5 to the largest of 4.64 Mbs for file 1. Therefore, the CBR channel

Video File	MPEG Pattern	BBP	Total Bandwidth Utilization	Number of Sources
1	M=1,N=2	0.000008	71%	21
2	M=2,N=4	0.000039	73%	36
3	M=3,N=6	0.00003	73%	46
4	M=2,N=10	0.000025	66%	46
5	M=3,N=15	0.000023	59%	49

Bandwidth = 127.155 Mb/s, Receiver Buffer Size = 64 Kbytes
Frame drop rate range: 1 frame every 72 to 366 minutes.

Table 3: Pattern Smoothing Performance

bandwidth for all sources ranged from 56 Mbs to 97 Mbs (44% to 76% of the 127.155 Mbs link). The results show a significant improvement in both bandwidth utilization and number of sources supported. Table 3 shows that this approach achieved a maximum bandwidth utilization of 73% and supports a maximum of 49 sources.

The Optimal Smoothing algorithm's performance results are shown in Table 4. Unlike the other two approaches, the Optimal Smoothing implementation is lossless and; therefore, Table 4 does not require a BBP or cell loss column. For comparison purposes, results from simulating both a 64 KB and 1024 KB receiver buffer are reported.

In comparing the results for the three algorithms, it is clear that the two smoothing algorithms achieve a significant performance improvement over the 'No Smoothing' approach. For a receiver buffer of 64 KB, Pattern Smoothing achieves a higher network utilization than Optimal Smoothing. When the Optimal Smoothing algorithm

		64 Kbytes Buffer*		1024 Kbytes Buffer*	
Video FIle	**MPEG Pattern**	**Bandwidth Utilization**	**Number of Sources**	**Bandwidth Utilization**	**Number of Sources**
1	N=2,M=1	67%	20	81%	24
2	N=4,M=2	68%	34	84%	42
3	N=6,M=3	68%	44	84%	54
4	N=10,M=2	58%	42	79%	56
5	N=15,M=3	56%	48	80%	67

Bandwidth = 127.155 Mb/s

*Receiver side buffer

Table 4: Optimal Smoothing Performance

uses a 1024 KB buffer, it out performs Pattern Smoothing and achieves a maximum bandwidth utilization of 84%.

3.7 Conclusion

In this chapter, we analyzed four smoothing algorithms. As part of this analysis, we compared these algorithms against eight key attributes. In terms of overall network utilization, the Optimal Smoothing Algorithm performs the best given a fixed receiver buffer and delay bound. One downside to this algorithm is its inability to support live video. While both the Lossless and Pattern Smoothing algorithms support live video, the Lossless algorithm was not develop specifically to handle the multiplexing of multiple sources over a single link. In the presence of data loss no guarantee is given to the delivery of I frames. As discussed, an I frame loss may prevent the decoding and displaying of an entire pattern.

The Pattern Smoothing algorithm was written specifically to address the transmission of live video. It utilizes both CBR and ABR channels to transmit multiple video sources over a single link. The concept of BBP was introduced to describe the probability of a frame being lost due to the use of statistical multiplexing. In the simulations the target BBP was set at $1x10^{-4}$, which achieved a frame loss rate of 1 frame every 72 to 366 minutes. The simulation results showed that the Pattern Smoothing algorithm achieves a significant increase in network utilization over a non-smoothed video transmission. In addition, when a 64 KB receiver buffer is used, the Pattern Smoothing algorithm out performs the Optimal Smoothing algorithm. Based on these results, we feel that the Pattern Smoothing algorithm is a viable option for video transmission when working with small receiver side buffers or live video.

Chapter 4

4 A Feedback Based Rate Control Algorithm

4.1 Introduction

There are several limitations to the smoothing approach presented in the previous chapter. First, it requires all receivers to be able to receive and process the same amount of video. In a heterogeneous environment such as the Internet, this restriction is not practical. Second, some type of network resource reservation scheme is required to support the constant bit rate transmission. While the Internet Engineering Task Force (IETF) is currently looking at the implementation of the Resource Reservation Protocol (RSVP) [54], this type of functionality is currently not available in the Internet.

This chapter presents a feedback based rate control algorithm for video conferencing on the Internet. This algorithm adaptively controls the transmission rate of a video stream, which is being multicast to a heterogeneous group of receivers. The fact that the sender is transmitting a single video stream to many receivers introduces a number of issues. These issues include the variable nature of the available network resources and the fact that there is no easy way to determine the availability of these resources. In addition,

even if we were able to track closely the changing network conditions, the best transmission rate for the aggregate means a trade off between high and low end receivers.

The algorithm presented in this chapter, called the *Target Bandwidth Rate Control Algorithm*, dynamically controls the output rate of the video coder by receiving and processing bandwidth levels from each receiver in the videoconference. This algorithm is unique in that it maximizes the aggregate displayable video at the receivers, while bounding the worst-case loss experienced by the low-end receivers. In addition to considering the available network bandwidth in the feedback information, this algorithm allows for workstation and user requirements to be considered when determining the sender's output rate.

In order to show the effectiveness of this approach, we first analyze the algorithm's performance in a simulation environment. In this set of tests, we study the effects of different configurations of receivers in terms of the number of receivers and availability of resources. Additionally, we have implemented this algorithm in the VIC video conferencing system in order to analyze its effectiveness under real network conditions.

4.2 Overview

The problem we are addressing consists of a videoconference between a sender, who captures, encodes and transmits the video and one or more heterogeneous receivers. As shown in Figure 8, this heterogeneity includes differences in workstation resources, local connections to the Internet, types of Quality of Service (QoS) guarantees, and the amount

of bandwidth available on the different network paths between the sender and the receivers.

In this type of multicast environment, we have imperfect knowledge of the current state of the network resources. At best, a receiver can tell how much they are currently receiving and if they are experiencing packet loss. In addition, since multicast transmission involves one sender and many receivers, obtaining timely feedback is difficult without negatively impacting the video sender. The problem is more complex since the network is dynamic and will change. Even if an algorithm existed that could know the current state of the network, there would still be a need to determine a transmission rate that trades-off between under-utilizing the available resources for the high-end users, and high levels of packet loss for the low-end users. Therefore, videoconferencing control algorithms attempt to pick a rate that maximizes the bandwidth usable at the receivers and adapt this rate based on estimates of the current network conditions.

As discussed in chapter 2, there are many algorithms for controlling the amount of bandwidth used by a video application have been proposed over the past few years. These approaches may be classified based on who manages the control algorithm, the sender [1, 3-5, 23, 24] or receiver [7, 31, 33, 42]. In this chapter we are concerned with sender managed feedback control. These types of algorithms rely on measurements from the network or receiver of the current availability of resources. Some of the measurements include packet loss, buffer usage, and packet arrival jitter. In this chapter, we propose a new feedback control algorithm developed specifically to support the

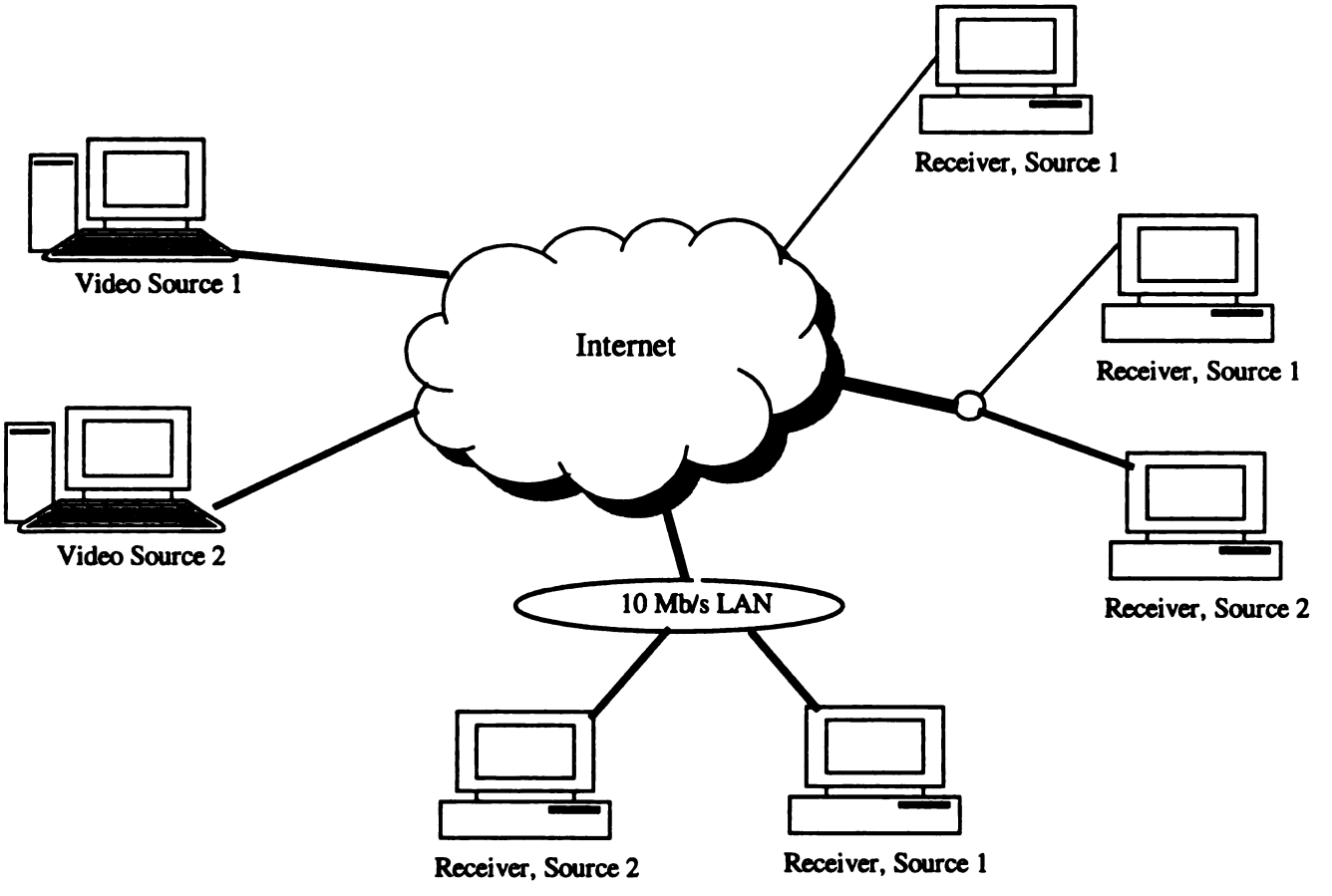


Figure 8: Heterogeneous Networks and Workstations

multicast transmission of video over the Internet. The feedback metric is the received bandwidth as determined by the receiver. This algorithm is different from its predecessors because as it adjusts the sender's transmission rate based on receiver's feedback, it attempts to maximize the usable bandwidth based on the correlation between packets lost in the network and those discarded on the receiver's workstation.

4.3 Motivation

The goal of the video conferencing control algorithm is to pick a transmission rate that maximizes the usable bandwidth on the receiver's workstation. Usable bandwidth is the amount of video that the receiver is actually able to display. This quantity is dependent on the transmission rate of the sender, the network bandwidth available between the sender and the receiver (and therefore the network loss) and the characteristics of the video coder.

The reason the displayable bandwidth is not solely dependent on the transmission rate and the packet loss is due to the restrictions of the encoding/decoding process. These restrictions determine the amount of the video and therefore, the number of packets that must be received successfully before the decoding process may be run. The amount of this dependency is called the application level framing (ALF) [18]. This ALF may range from one packet to an entire frame. If this unit is greater than one packet, there is a dependency between packets. This dependency will cause the loss of usable bandwidth at the receivers to be greater than the loss in the network. As an example, if the ALF is one video frame and there are 20 packets on average per frame, then the loss of one

packet in the network will translate into 19 packets being thrown-away on the workstation.

We implement a maximization function in order for the control algorithm to compensate for the dependency between packets. This function determines a new transmission rate, which maximizes the usable bandwidth, based on the receivers' feedback and an estimate of the workstation loss. Figure 9 gives and example of this type of function. In order to estimate the workstation loss this function uses the calculated network loss times the current average packets per frame. While a more detailed regression analysis is possible to determine the relationship between network loss and workstation loss, the use of average packets per frame gives a good first approximation. We will show that this approximation works well in practice.

Once the new transmission rate has been calculated, a mechanism to modify the video stream needs to be considered. There are many attributes, which may be adjusted in order to adapt the coder's output rate to meet the new transmission rate. These attributes may be classified as either temporal or spatial [33]. The temporal attributes concern adjusting the rate at which frames are encoded. The spatial attributes relate to the quality of the image transmitted. These quality variables include the number of bits used to represent the color in the image, the number of pixels in the image, the compression algorithm's key frame distribution, and the amount of lossy compression (quantization) applied to the image [20, 21]. Since each of these attributes affect the coder's output rate

differently, the feedback control algorithm needs to allow the sender to determine the best setting for each of these attributes based on the calculated transmission rate.

As mentioned previously, the algorithm uses feedback from the receivers in order to adjust the output rate of the video coder. In addition to network constraints, the control algorithm allows for the consideration of the available workstation resources and the user's priority of the video application [8] in determining the receivers' feedback rate. If the receiving workstation is unable to provide the necessary resources to process the incoming video stream, some type of implicit or explicit load shedding will take place [8,

Goal: Find the new Transmission Rate ***T*** that maximizes the displayable bandwidth at **n** receivers:

$$\underset{T}{MAX}\left[\sum_{i=1}^{n}(T - estimated\_workstation\_loss_i)\right]\times\left[\prod_{j=1}^{n} 1_{estimated\_workstation\_loss_i / T < \beta}\right]$$

Where

$$estimated\_workstation\_loss_i = \begin{cases} 0 & if \quad T < feedback\_rate_i \\ (network\_loss_i * packets\_per\_frame) & otherwise \end{cases}$$

$$network\_loss = T - feedback\_rate_i / packet\_size$$

$feedback\_rate_i$ is the current estimate of the available bandwidth between $receiver_i$ and the sender

β is the bound on the worst case loss on the low-end receivers

Figure 9: Maximization Function

12]. While this load shedding will not cause the video display quality to degrade as quickly as network packet loss, it may cause a jerky image display and wastes available workstation resources. The algorithm may also consider the priority of the receiver's application. In a multitasking environment, the user on the receiving end may determine that the video application is of lower importance and may wish to limit the amount of resources committed to receiving and displaying the video stream. While this chapter focuses on the impact of network constraints on the control algorithm, the algorithm allows for workstation resource and user priority requirements to be considered when determining the video coder's output rate.

The remainder of the chapter is organized as follows. In Section 2, we give a brief introduction to the VIC [30] video conferencing system, the MBone [11, 27] and RTP [40, 41]. We then discuss an RTP based feedback control algorithm, which has been presented in the literature. In Section 3, we present the Target Bandwidth (TB) control algorithm and give details of how this algorithm was incorporated into VIC. Section 4 discusses the simulation setup used to test the TB algorithm and presents the performance results of the simulations. Section 5 contains a simulation performance analysis of the TB algorithm. Section 6 presents the VIC performance results. Finally, Section 7 summarizes the chapter and identifies future work.

4.4 Internet Video Conferencing

In this section, we examine the VIC application and some software it uses to support videoconferencing in the Internet. This software includes the MBone and the Real-Time Transport Protocol (RTP). Following this discussion is an introduction to an RTP based Packet Loss Rate Control algorithm. This algorithm was developed to control video conferencing application on the Internet. We will use this algorithm as a base case as we analyze the performance of the TB algorithm.

In order to see how the TB algorithm performs in a live networking environment, We have modified a version of the VIC application to include the TB rate control algorithm. VIC is a video conferencing system developed at UCB/LBL. This system, like the earlier video conferencing tools nv and IVS [48], utilizes IP-multicast and the MBone to deliver video streams over the Internet. VIC provides a flexible user interface and supports multiple compression algorithms. It utilizes the Real-Time Transport Protocol in order to transport the video stream and to deliver control messages between the sender and the receivers. Readers interested in a more detailed description of VIC are referred to [30].

To facilitate the use of IP-multicast over the Internet, VIC utilizes the MBone. The MBone is a virtual multicast network, which operates on the Internet. To support the distribution of multicast data, the MBone utilizes MBone servers running a multicast routing daemon called *mroute*. These routing daemons are geographically distributed and forward multicast packets to the local receivers. From the sender's point of view, the MBone environment looks like a rooted tree with the sender as its root, the *mroute*

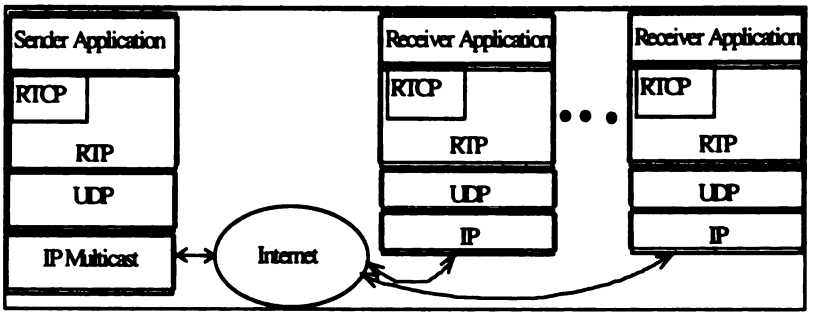


Figure 10: Video Application with RTP

daemons as the interior nodes and the receivers as the leaf nodes. We discuss VIC's use of the MBone later as a way to provide scalability in the TB rate control algorithm.

4.4.1 Real-Time Transport Protocol (RTP)

VIC utilizes RTP for data and control packet delivery. RTP is under development by the Internet Engineering Task Force, Audio-Video Transport Working Group [40, 41]. RTP is being developed for applications requiring real-time services such as video conferencing. It will facilitate the implementation of services such as playout synchronization, active-party identification and media identification. RTP does not provide the underlying end-to-end transport protocol but will work with transport protocols such as UDP.

RTP also provides a control protocol called RTCP. An example of how this would work for a video application is shown in Figure 10. While RTCP packets may be used for many purposes, they are of interest to us because they provide the means to transmit feedback information between the receivers and the sender. This feedback is transmitted in an RTCP Receiver Report packet. A Receiver Report packet is a control packet that is

transmitted by all receivers participating in the videoconference. A part of the receiver report is a QoS feedback field that may be used to convey information such as packet loss and packet inter-arrival jitter. In the Target Bandwidth algorithm, we propose putting the receivers' feedback rate in this field. Readers interested in a more detailed description of RTP are referred to [41].

4.4.2 Packet Loss (PL) Algorithm

An example RTP based rate control algorithm is based on measuring packet loss at the receivers. This packet loss algorithm was initially discussed in [3, 4] and was later refined by Schulzrinne, *et al.* in [5]. We present an overview of this algorithm in order to compare its performance against the Target Bandwidth algorithm.

The packet loss algorithm uses RTP receiver reports to determine the packet loss being

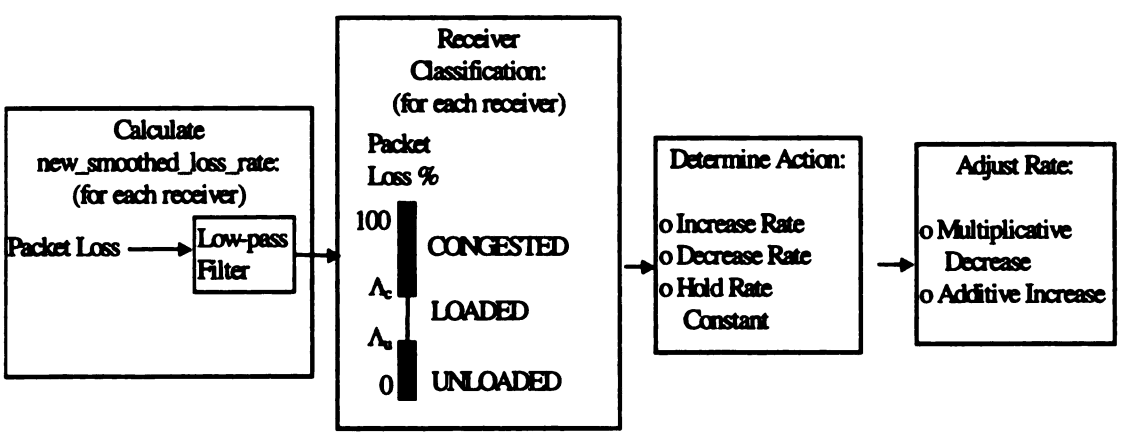


Figure 11:Packet Loss Rate Control Algorithm (modified from [5])

experienced by the receivers. This loss information is used to modify the video coder's output rate. There are four major steps in this control algorithm. Figure 11 shows an overview of the algorithm. This figure is a modified version of the one found in [5].) The steps are:

1. For each receiver determine the packet loss rate.
2. Classify each receiver as UNLOADED, LOADED or CONGESTED.
3. Determine the corrective action necessary based on the percentage of users in each classification. The options are to increase, decrease or hold constant the coder's output rate.

Advantages of this algorithm include its ability to decrease the overall packet loss rate and to follow changes in network bandwidth. In addition, the packet loss information may be calculated without the aid of the video application. Therefore, this approach is fairly independent of the application and only requires application involvement in the final step. On the other hand, this algorithm does have some limitations. One major limitation is that this algorithm does not take into account and adjust to the dependency between packets. Therefore, the algorithm may over or under transmit depending on the current size of the application level framing. In addition, since this algorithm does not look at the effects of its transmission rate on the low-end receivers it may over-transmit and cause the low-end receivers to be able to display little or none of the video signal. Also, these simulations show this algorithm may oscillate for certain configurations. This oscillation, or continuous fluctuation between transmission rates, is due to the dynamic interaction of some of the algorithm's parameters. Finally, since packet loss and packet

size are application and platform dependent it is difficult to make a direct translation between packet loss and the workstations' available resources or the users' priorities of the video application. Therefore, considering this information in the control algorithm would be difficult.

4.5 Target Bandwidth Algorithm

In this section, we introduce the Target Bandwidth (TB) Algorithm. The goal of this algorithm is to maximize the bandwidth utilized at the receivers while attempting to limit the maximum loss experienced by any individual receiver. In addition to considering the availability of the network resources, this algorithm allows for user and workstation requirements to be considered when generating a receiver's feedback.

This algorithm consists of two components. These are the receiver feedback component, which is run on each receiver's machine, and the sender's rate calculation component, which is run on the sending workstation. The receiver component is responsible for generating the feedback that reflects the receiver's current ability to receive and process the video stream. The sender component is responsible for collecting the receiver's feedback and calculating a transmission rate, which maximizes the displayable bandwidth. By displayable bandwidth, we are referring to the amount of the video stream that is usable by the receivers. This bandwidth will vary from the transmitted bandwidth depending on the amount of packet loss in the network and the dependency between packets. This dependency between packets, refers to the number of packets that must be received before the decoder may proceed. In the smallest case, each packet may be

independently decoded. In other cases, the entire frame must be successfully received before any decoding may take place. Therefore, the loss of one packet will cause the entire frame to be lost and translate into a very large loss in usable bandwidth.

4.5.1 Receiver Calculation

The focus of the receiver's calculation is the availability of the network resources. This calculation uses a slow startup algorithm similar to TCP to estimate the current available bandwidth. If the receiver is experiencing packet loss, then the receiver's network feedback rate is set to (1+*increase_rate*) times the currently received bandwidth, where *increase_rate* is set to α, a small value such as 0.02. If the receiver is not experiencing packet loss, then the receiver's feedback rate is calculated by first doubling the increase_rate (e.g., 0.04, 0.08) and then multiplying (1+*increase_rate*) times the currently received bandwidth. In order to prevent very large fluctuations in the feedback rate we limit the *increase_rate* to some predetermined maximum. See Figure 12 for the pseudo code for the network feedback rate calculation.

Unlike many feedback algorithms, the actual calculation of the available bandwidth is performed on the receiver. This allows us to consider requirements in addition to the current network availability. Examples include the user's priority of the application and workstation conditions. By translating these requirements into a common unit such as bits/sec (bandwidth), we can include the requirements in the receiver's feedback calculation. As an example, if the user designates the video application as a lower priority, we can set the feedback rate to some preconfigured minimum rate. While the actual calculation of these two rates is beyond the scope our research, an example of how they may be used is discussed in the VIC implementation section.

```
If (packet_loss) {
    increase_rate = α
    feedback_rate = current_received_bandwidth * increase_rate
}
else {
    increase_rate = increase_rate *2
    if (increase_rate > MAX_INCREASE)
     increase_rate = MAX_INCREASE
    feedback_rate = current_received_bandwidth * increase_rate
}
```

Note: This part of the algorithm is run on each receiver and the feedback_rate is transmitted to the sender for use in the calculation of the sender's transmission rate.

Figure 12:Receiver's pseudo code

4.5.2 Sender Calculation

The sender's calculation is focused on maximizing the amount of video displayable at the receivers. Since the sender wishes to maximize for the aggregate, some receivers will receive less then their maximum rate while others will experience some packet loss. The sender collects the receiver's feedback rates and then determines the new transmission rate, which maximizes the receiver's usable bandwidth. In order to estimate the effects of changes in the transmission rate on the receiver's usable bandwidth, the maximization algorithm takes into account the application level framing and the corresponding dependency between packets.

The sender's algorithm is as follows:

1. Collect the receivers' feedback: $feedback\_rate_i$
2. Using the estimate of workstation loss find a *transmission_rate* that maximizes the displayable bandwidth. (See maximization pseudo code in Figure 13)
3. Smooth and filter-out fluctuations in the transmission rate and pass the new rate to the sender's video coder.

```
max_rate = 0
max_value = 0
value = 0

For (k = 0; k < num_receivers; k++) {
  value = 0
  temp_rate = feedback_rate_k
  for (j = 0; j < num_receivers; j++) {
     network_loss = 0
     if (temp_rate > feedback_rate_j)
       network_loss = temp_rate – feedback_rate_j

     estimated_workstation_loss = network_loss* packets_per_frame

     If (estimated_workstation_loss / temp_rate > β) {
        value = -1
        break-out-of-inner-loop
     }
     value = value + temp_rate – estimated_workstation_loss
    }

    if (value > max_value) {
     max_rate = temp_rate
     max_value = value
    }
}
```

Figure 13: Sender's maximization pseudo code

The sender's algorithm's second step calculates the new transmission rate. This calculation is based on the maximization function presented earlier (Figure 9). As discussed earlier this function calculates the transmission rate that maximizes the receiver's usable bandwidth by estimating the loss of workstation bandwidth. This estimation is based on the application level framing and the corresponding dependency between packets. One shortcoming of this maximization calculation is that some lower-end receivers may experience unacceptable loss. To prevent this from happening the algorithm has been modified to reject a transmission rate where any receiver's estimated displayable bandwidth is less than β% of the transmitted bandwidth.

The final step in the sender's algorithm is to smooth and filter-out large and small short-term (1 to 2 seconds) fluctuations in the transmission rate. The smoothing is done by taking the average of the last η calculated transmission rates. This smoothing prevents large short-term fluctuations from completely dominating the transmission rate. The drawback to this smoothing is it decreases the responsiveness of the algorithm to long-term (tens of seconds or greater) changes in available bandwidth. The second part of this step is to filter out small fluctuations in the transmission rate. This filtering is done by actually only changing the sender's transmission rate if the new calculated rate has increased/decreased a small percentage. Taken together this smoothing and filtering allows the algorithm to remain stable for the majority of the users while short-term bursts (such as a small file transfer) only effect a few receivers. In the simulation and applications runs we have seen that as the network noise levels increase the low-end

receivers begin to drag down the transmission rate. To reduce the effects of these variations the smoothing and filtering values may be increased. It is important to realize that by increasing these values we are reducing the responsiveness of the algorithm to change and therefore increasing the packet loss on the low-end receivers.

There are four key details that allow this algorithm to perform well in a heterogeneous environment such as the Internet. First, the algorithm works with existing network protocols and technologies. Second, the dynamic transmission rate is based on both changes in the video coder and the receiver's resource availability. Third, since the receivers are responsible for generating their own estimates of the available resources, the algorithm allows for the inclusion of other factors such as workstation availability and user priorities. Fourth, the algorithm uses a calculated packet loss rate to *sense* the state of the network. In most transmission paradigms, packet loss is not acceptable. In the case of multicasted video, some loss is to be expected unless the transmission rate is set to the minimum of all receivers. Therefore, the algorithm intentionally sets the receiver's target rate to have a minimum of α% packet loss. It then monitors the actual loss rate to determine if there has been a change in resource availability.

4.6 VIC Implementation

In order to incorporate the TB algorithm into VIC, we made several modifications. On the sending side, VIC was modified to receive and process the receivers' control feedback information via RTP Receiver Reports. This information is then used to calculate the new target bandwidth rate, which is then passed to the encoder. The

encoder then modifies its quality parameters (i.e., frames per second) to meet this new target rate. On the receiver side, the VIC application was modified to calculate the receiver's feedback rate. On a periodic basis, the receiver component of the algorithm is executed and the receiver's feedback is calculated based on the bandwidth received and the current packet loss status. This feedback is then transmitted back to the sender via the RTP Receiver Reports.

In order to allow the TB Algorithm to run more efficiently in a real environment, two modifications were made to the basic algorithm. First, a procedure was needed in order maintain the feedback information from each receiver. This was implemented via a hash table using the receiver's RTP ID as the hash key. In addition, a flag was added to each hash table entry in order to facilitate the removal of any outdated receiver that may have terminated without informing the sender. By maintaining this hash table, we are able to negate the effects of the asynchronous nature of feedback information.

The second improvement to VIC is the inclusion of user priority of the application into the receiver feedback calculations. In VIC, the receiver has the ability to leave the video image in a thumbnail size (icon size) or to increase its size. In the implementation of VIC, if the user does not increase the image size, we set the maximum feedback rate to 750kb/s. In this way, users are able to specify the priority of their video application. While more complex uses of the user priority feedback are possible, this VIC implementation provides insight to the usefulness of this information.

4.6.1 Algorithm Timing

The components of the algorithm on both the sender and receiver are executed periodically. Several factors need to be considered when setting the algorithm's control cycle time. These factors include the overhead on the sender for processing the feedback information and the responsiveness of the algorithm to change. Longer cycle times reduce the overhead of the RTP Receiver Reports on the sending workstation but decrease the responsiveness of the algorithm to changes in network conditions. Since a longer response time only significantly affects a receiver if there is a decrease in the available bandwidth, we are trading off the quality of the video displayed on the low-end receivers versus the overhead of the control algorithm on the sending workstation. We have found that a cycle time of 5 seconds is responsive to changes, while generating less than 10kb/s of traffic for 150 receivers.

4.6.2 Scalability

As the number of receivers in the videoconference grows, the issue of scalability of the control algorithm needs to be addressed. While the overhead of the receivers' control feedback is minimal and may be adjusted by increasing the duration of the control cycle time, at some point this overhead may become detrimental to the functioning of the sending machine or the network. One approach to reduce this overhead is to perform some type of consolidation of the data prior to it reaching the sending machine. From the sender's point of view, the MBone multicast environment looks like a rooted tree with the sender as its root. We propose using this MBone multicast tree as a reduction tree allowing the interior nodes (*mroute* daemons) to receive the feedback from the receivers,

consolidate it and transmit it back to the root (sender). The consolidation at each interior node will consist of quantitizing the incoming feedback into buckets and keeping track of the number of receivers in each bucket. If the number of buckets is kept sufficiently small (< 500) the feedback from an interior node to the root may be done in one packet. This packet would contain the number of receivers in each bucket. In this way, the sender only receives one message from each interior node with a fixed number of buckets regardless of the number of users.

4.6.3 Internet Quality of Service Guarantees

There has been significant research for providing Quality of Service (QoS) guarantees on the Internet. These QoS guarantees involve providing a user's connection with a guaranteed bandwidth given certain delay and peak rate constraints. One such approach is the Internet Engineering Task Force's Resource Reservation Protocol (RSVP) [54]. RSVP is being developed to provide QoS guarantees to multicast connections in the Internet. For the foreseeable future, Internet applications will need to exist in a mixed RSVP/Non-RSVP environment (see Figure 8). One benefit of the TB algorithm is its ability to support such an environment. For the RSVP users, the receiver's feedback rate may be the RSVP guaranteed rate. Non-RSVP receivers would use the standard TB algorithm's calculation for their feedback rate.

4.7 Simulation Performance Analysis

There are a number of simulation conditions that may be varied in order to understand the performance of the TB algorithm. In this section, we analyze how the following variations affect the TB algorithm:

1. Background network noise levels (short-term fluctuations in the network bandwidth)

2. Changes in video coding parameters (dependency between packets)

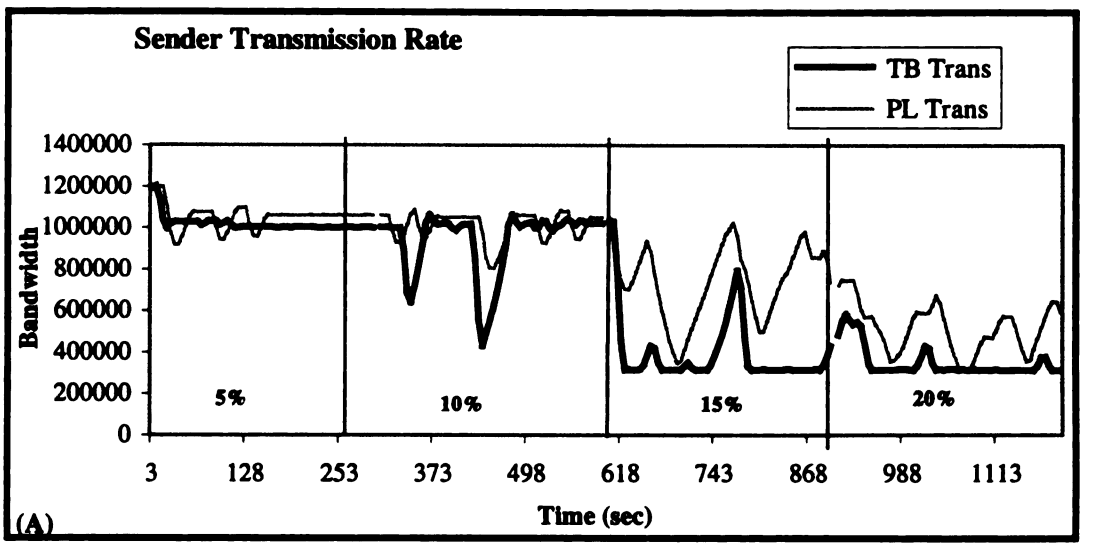
3. Large, long term network fluctuations and changes

4. Receiver group sizes and different distributions of network resources

The first simulation analyzes the effect of short-term network noise. In this simulation, we model short term network noise using a negative exponential distribution. The amount of network noise was modified by varying the noise mean rate from between 1% to 20% of the total network bandwidth. Figure 14-A shows the transmission rate and Figure 14-B shows the average usable bandwidth for simulation runs with noise mean rates of 5%, 10%, 15% and 20% for both the TB and PL algorithms. These graphs show that both algorithms remain fairly stable at noise levels of 5% and 10%, with two exceptions. At simulation time 343 and again at 438 the TB algorithm experiences a period of large adjustment. These are due to a small number of receivers experiencing heavy short-term congestion. This identifies one of the weaknesses of the TB algorithm. Since the algorithm attempts to adjust quickly to fluctuations and restricts the maximum loss rates of the lower end users, the TB algorithm may be strongly affected by the loss rates of a few low-end receivers. The alternative is to ignore the losses experienced on the low-end receivers, but as we show later this leads to those receivers experiencing unacceptable loss rates. Figure 14 also shows the effects of noise mean levels of 15%

and 20%. At these levels of noise, both algorithms tend to become unstable, with the TB algorithm being heavily affected by the low-end receivers.

In the next set of simulations, we examine at the effects of changes in the video coding parameters. Specifically, changes in the dependency between packets. Figure 15-A,B,C,D shows the results of these simulations. In Figure 15-A we see the changes in the transmission rate of the video coder as the dependency between packets moves from an average of 1 packet (e.g., all packets are independent and may be decoded individually) to a dependency of 20 packets. As can be seen in Figure 15-A, the transmission rate for the TB algorithm is higher for small packet dependencies but adjusts to lower rates as the dependency increases. This is due to that fact that as the dependency between packets increases the effects of packet loss on the receiver's increases. Therefore, a higher dependency between frames requires a lower transmission rate in order to accommodate the low-end receivers. Since the PL algorithm does not take packet dependency into account, it does not adjust its transmission rate as the dependency changes.



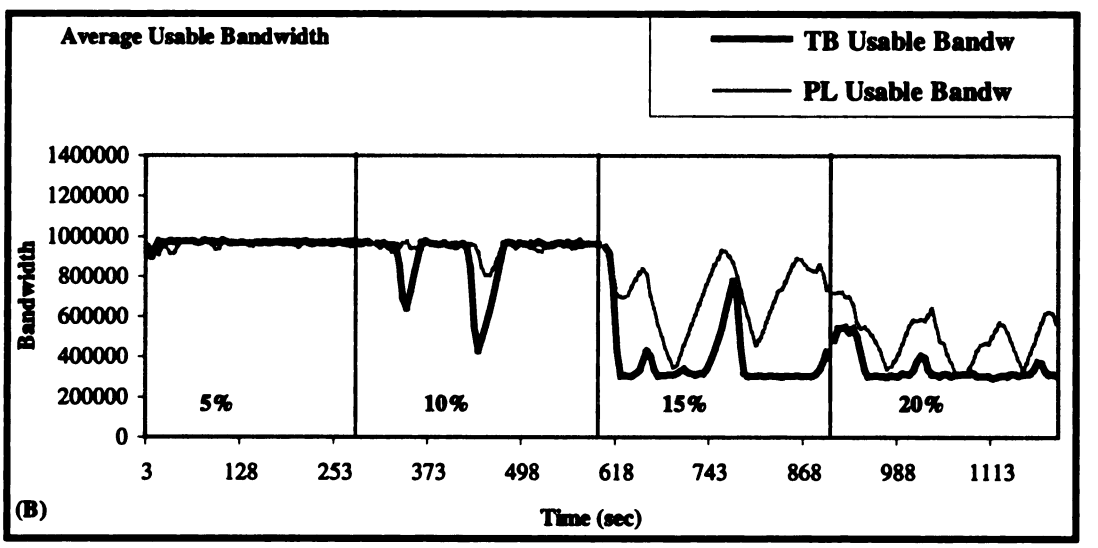


Figure 14 (A,B): Network Noise

Figure 15-B shows how this adjustment in transmission rate affects the average usable bandwidth at the receivers. As this figure shows, in the smaller dependency case the TB algorithm achieves a much higher usable bandwidth. As the dependency between packets increases both algorithms begin to converge on the same average usable bandwidth. If we combine the results Figure 15-A and Figure 15-B during the periods of high dependency (simulation time between 213 and 408 seconds), we see that the PL algorithm is transmitting at a higher rate but the usable bandwidth is the same as the TB algorithm. The effects of this over-transmission can be seen in Figure 15-C and Figure 15-D. These figures show the usable bandwidth for the lowest 7 receivers. As these figures show, during the last 200 seconds of the simulation the low-end users of the PL algorithm experience a significant amount of loss in usable bandwidth. In some cases, 2-3 receivers are receiving none of the video signal for 15 seconds or longer. In contrast, during the majority of this time the TB receivers (Figure 15-C) are receiving a significant portion of the video signal.

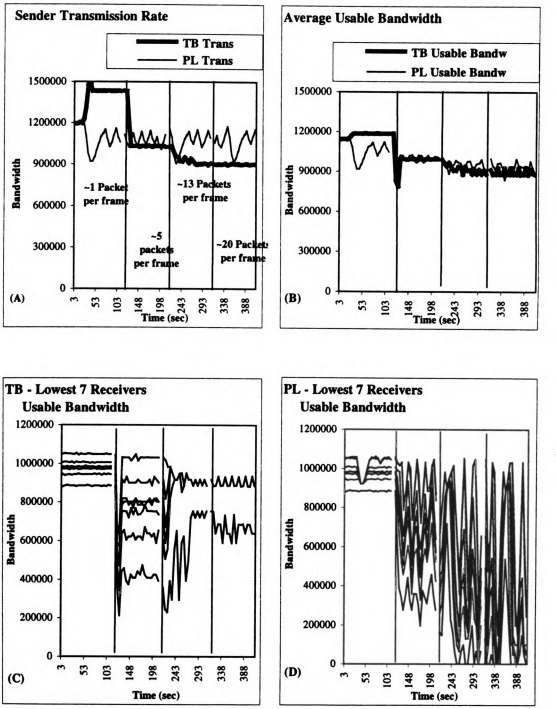
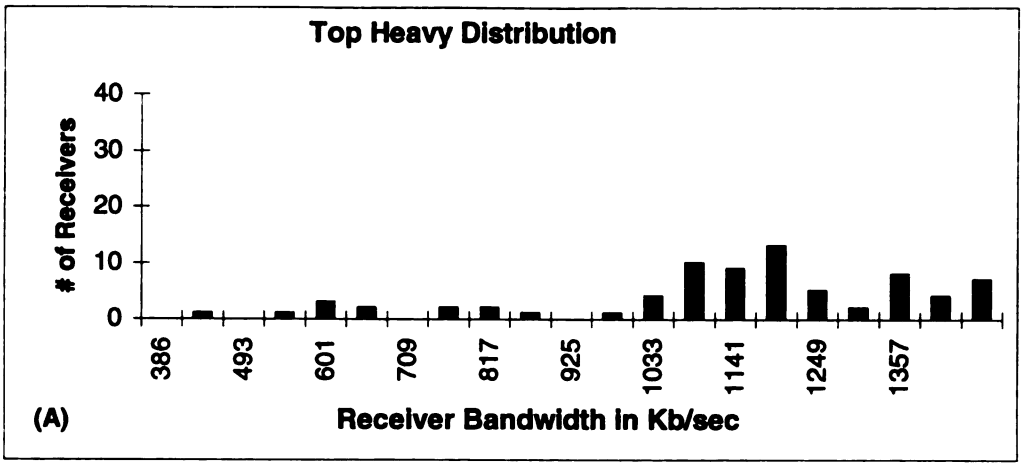
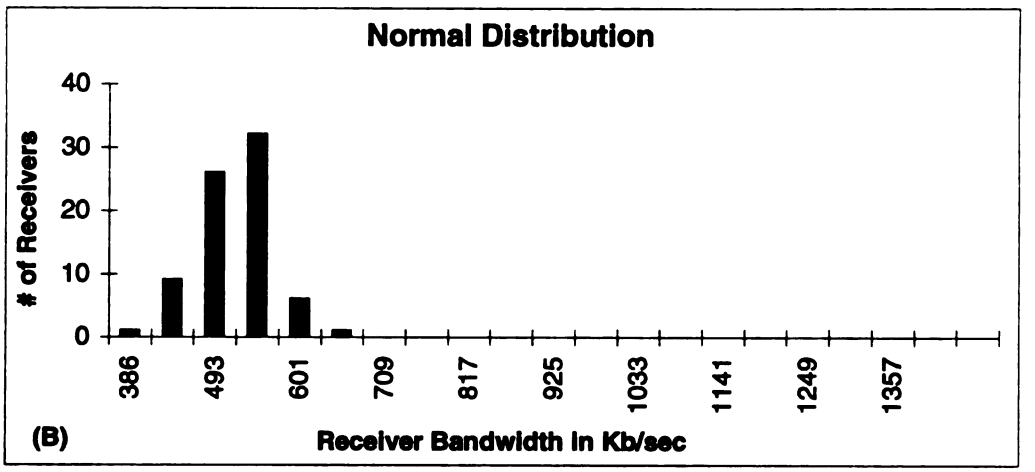


Figure 15 (A,B,C,D): Dependency Between Packets

Based on many simulation runs over varying distributions we have found that for cases with small packet dependencies the TB algorithm tends to have a higher transmission rate and higher average usable bandwidth without the negatively affecting the low-end receivers. As the dependency grows, the TB algorithm will tend to transmit at a lower rate than the PL algorithm. While the effects of the higher transmission rate of the PL algorithm on the average usable bandwidth are mixed, (We show later there are receiver distributions where the PL algorithms average usable bandwidth is much higher than the TB Algorithms) this higher transmission rate has significant negative effects on the low-end receivers.





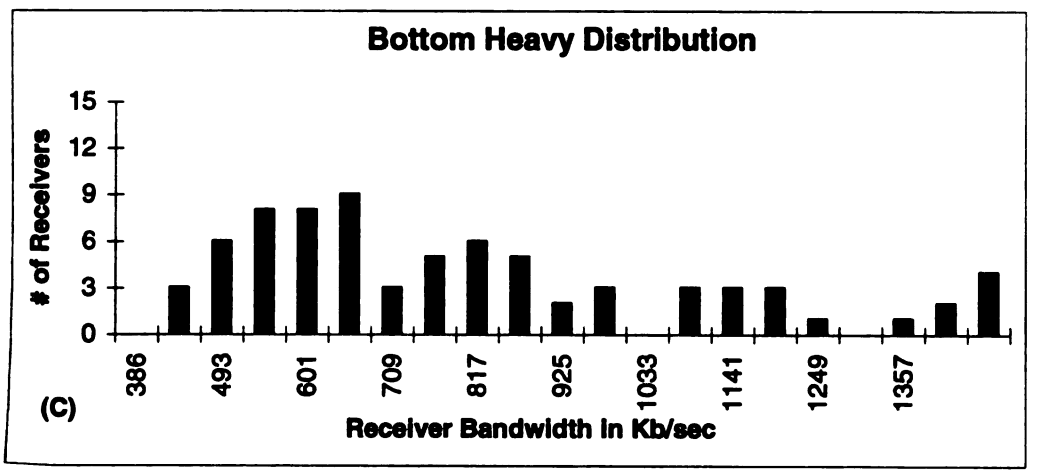


Figure 16: Receiver Distributions

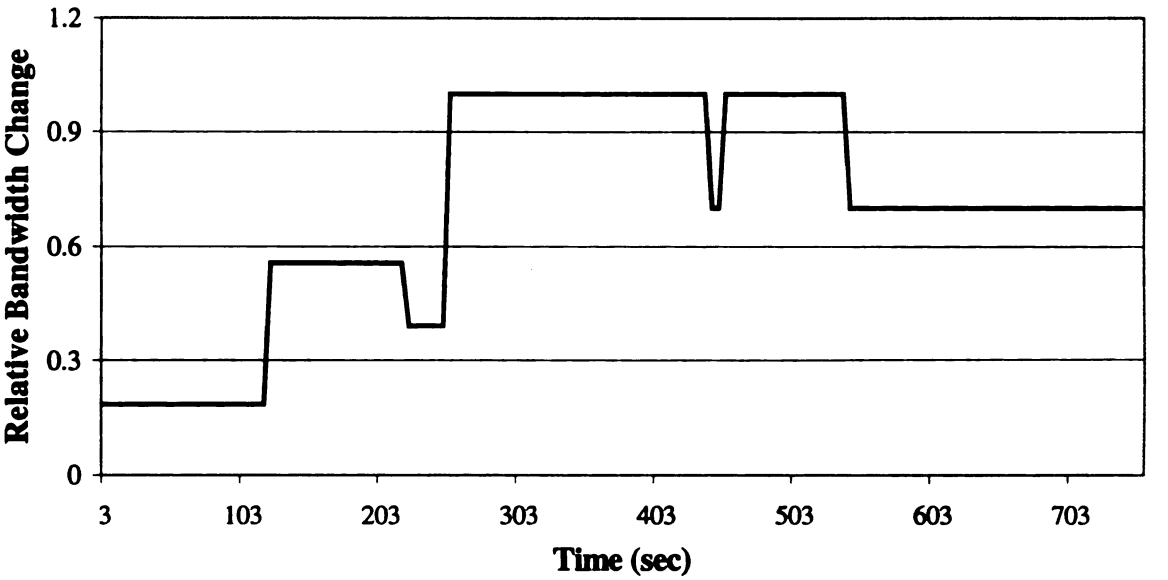


Figure 17: Timing and Magnitude of Bandwidth Changes

The final set of simulations study the effects that different distributions of receivers' network resources and the number of receivers has on the sender's transmission rate, the receiver's average usable bandwidth and the low-end receiver's usable bandwidth. In these simulations, we analyzed runs with receiver sizes of 10, 30, 75 and 150 users. In order to simplify the discussion we have included only data for runs with 75 users. In addition, we examine three different distributions of receiver's resources. These receiver bandwidth distributions are show in Figure 16 for receiver size of 75. These figures show the approximate mean rate for the receivers at the start of the simulation. While the mean rate of the receivers' bandwidths varies during the simulation, the overall shape of the distribution remains fairly constant.

In order to understand the effects of the different distributions and varying number of receivers we ran the simulation over dynamic network bandwidth conditions. These simulations are meant to look at longer term (tens of seconds to minutes) and larger fluctuations in available network bandwidth. Figure 17 tracks how the average receiver bandwidth varied over time for the final simulation runs. The actual bandwidth amounts, in Figure 17, were intentionally left off since they will vary based on the distribution of the receivers. Figure 18-(A,B,C,D) , Figure 19-(A,B,C,D), Figure 20-(A,B,C,D) show the results of the simulation runs for 75 receivers using the conditions in Figure 17 for the three distributions in Figure 16.

Figure 18-A, Figure 19-A, and Figure 20-A show the transmission rates for all three distributions for 75 receivers. These figures show that the PL algorithm is more aggressive in its transmission rate, especially at the higher transmission rates. This is because the frame size is greater and therefore the dependency between packets is greater. The packet dependency forces the TB transmission rate lower. Figure 20-A, which displays results for the top-heavy distribution, shows that the PL algorithm is significantly more aggressive. The TB algorithm is constrained in the top-heavy distribution by the loss rates of the lower end receivers. As you can see in all three figures, the PL algorithm tends to oscillate. As we will show later, this oscillation negatively affects the usable bandwidth, especially on the low-end receivers.

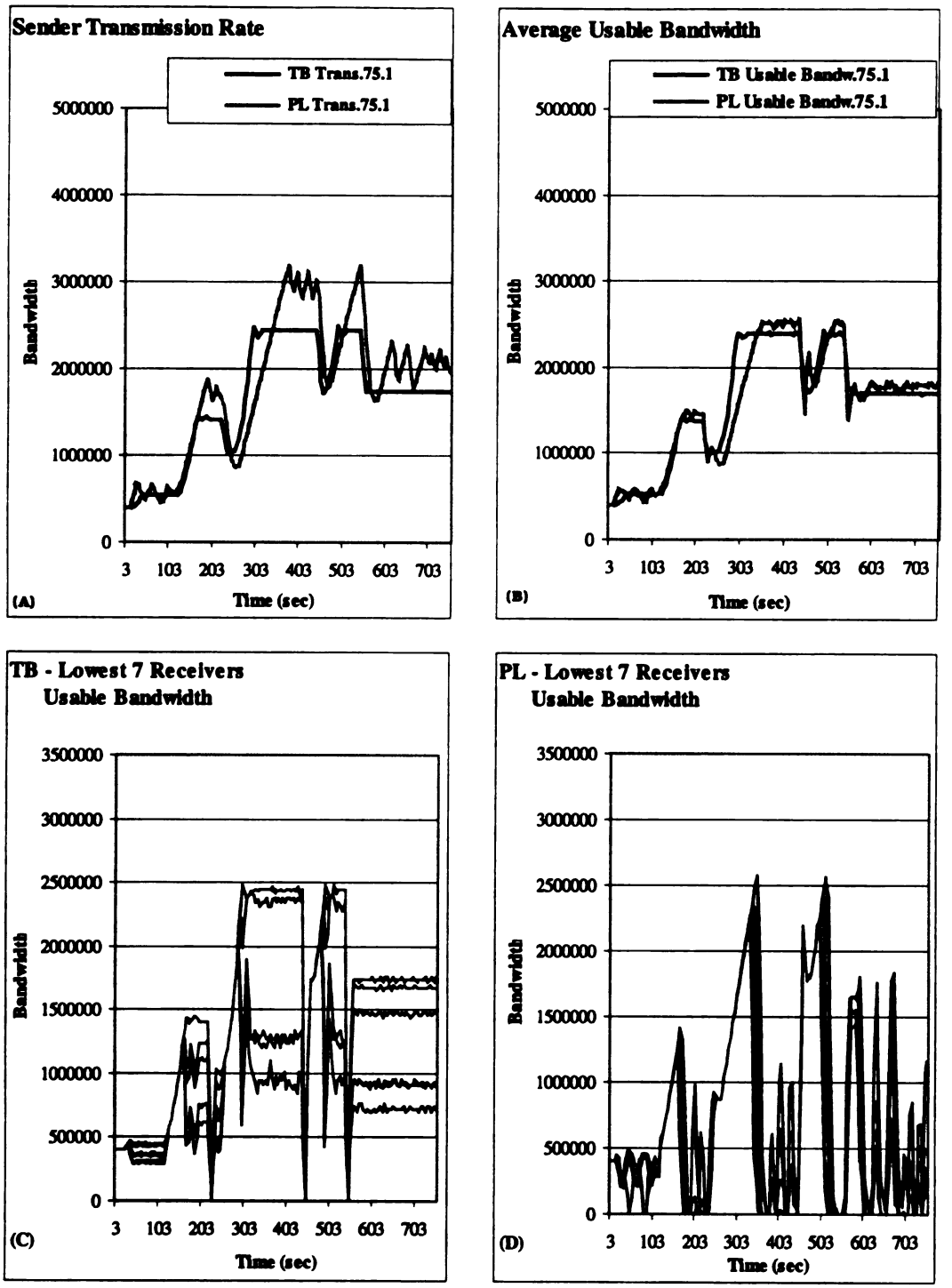


Figure 18 (A,B,C,D):Bottom Heavy Distribution, 75 Receivers

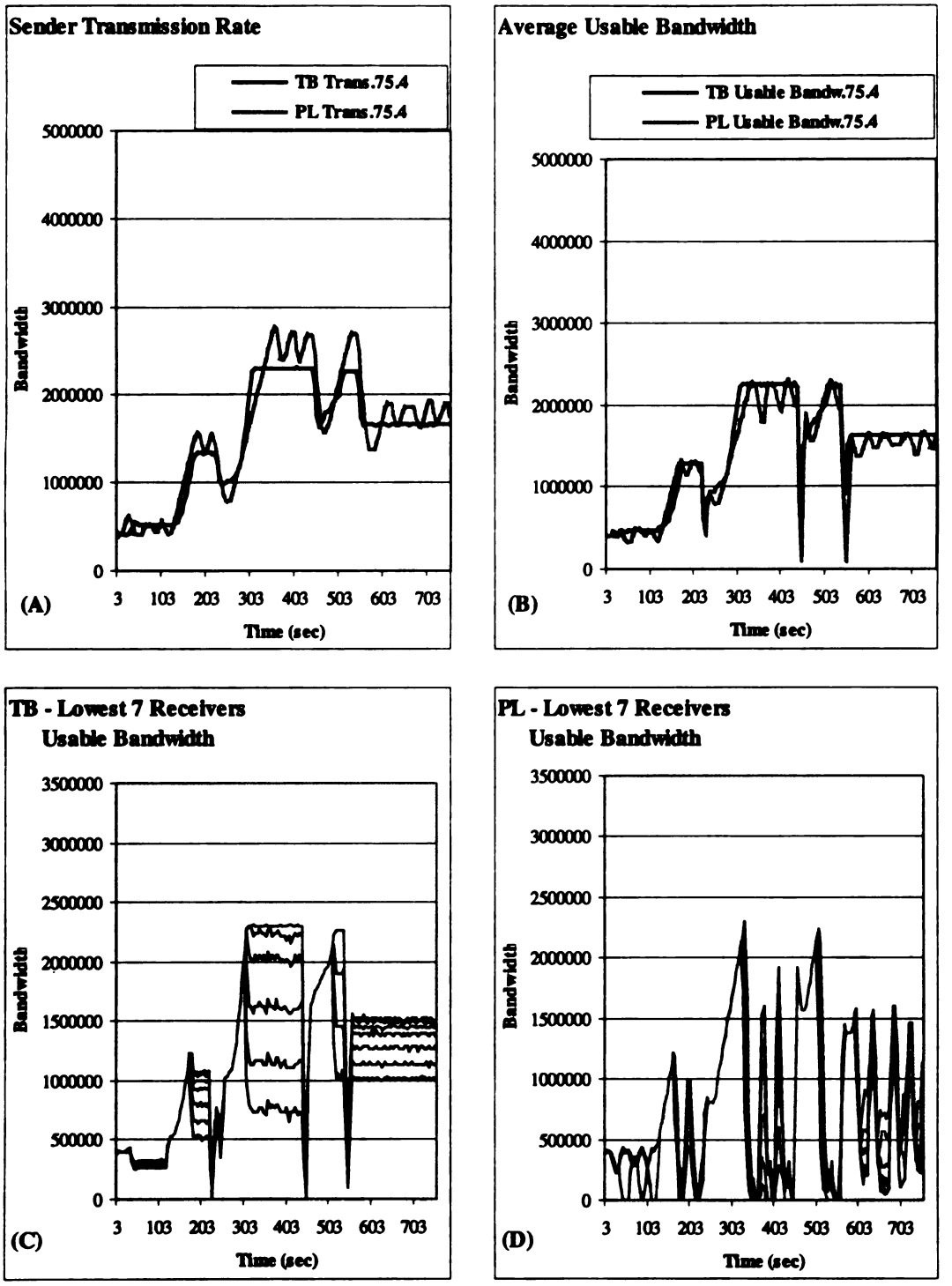


Figure 19 (A,B,C,D):Normal Distribution, 75 Receivers

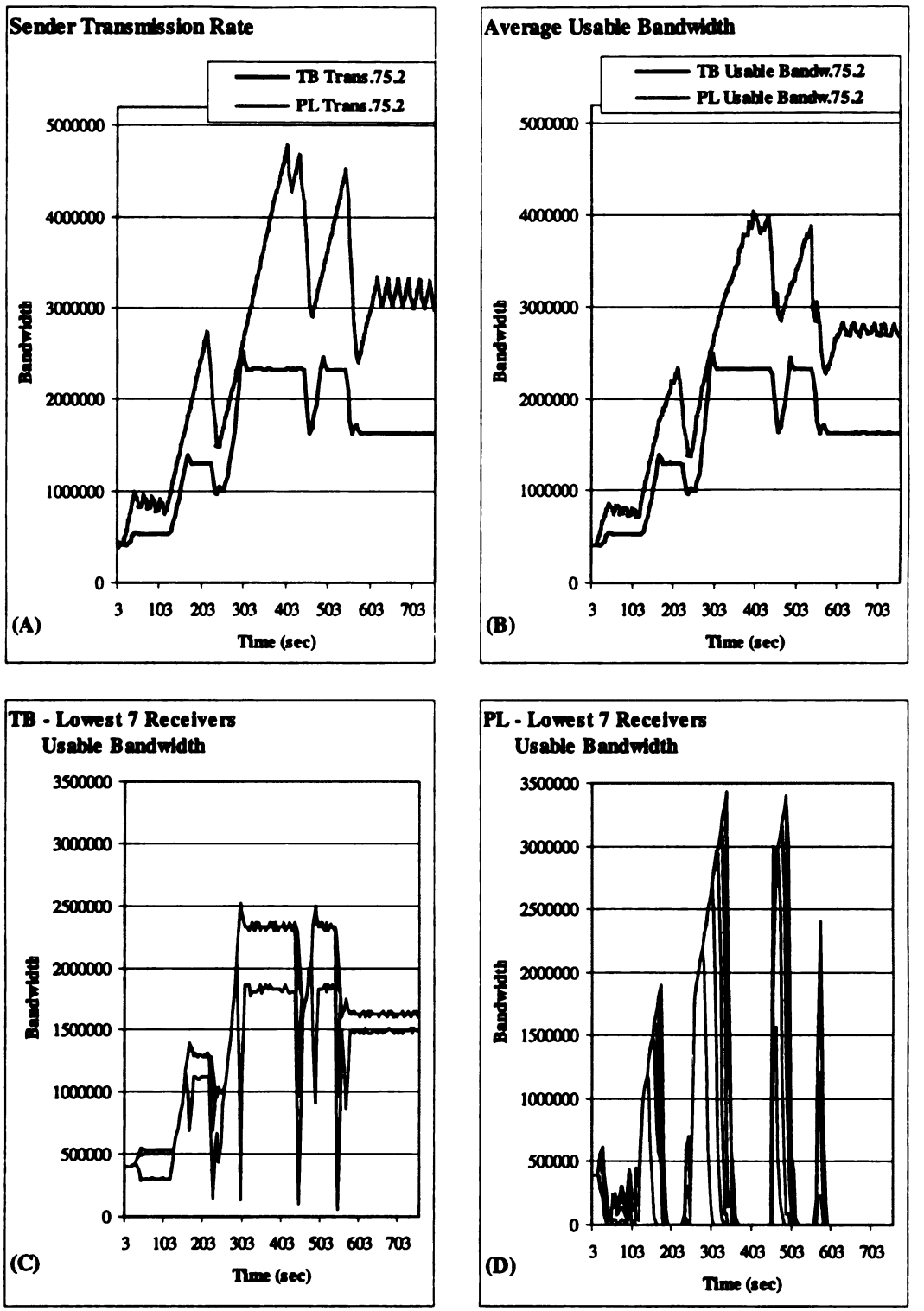


Figure 20 (A,B,C,D):Top-Heavy Distribution, 75 Receivers

Figure 18-B, Figure 19-B, and Figure 20-B show the average usable bandwidth on the receiver's. These graphs allow us to see the effects of the transmission rates across the entire 75 receivers. The average usable bandwidth for the PL algorithm in the top heavy and bottom heavy cases (Figure 18-B, Figure 20-B) is higher than the TB algorithm. This is due to the constraints of the lower-end users on the TB algorithm. While the average usable bandwidth is higher, we will show the negative effects on the lower-end receivers are significant. Figure 19-B shows that the average usable bandwidth in the normal distribution is comparable between the two algorithms. This is due to the tightness of the distribution of receivers. In the normal distribution, the low-end receivers do not dominate the transmission rate for the TB algorithm.

While the average usable bandwidth graphs show how the algorithm's choice of transmission rate impacts the aggregate, it does not tell the entire story. Figure 18-C,D, Figure 19-C,D, and Figure 20-C,D show the usable bandwidth for the lowest seven receivers. Key points from these figures are:

1. While both algorithms experience large dips in usable bandwidth during large drops in available bandwidth (simulation times 228, 448, 548), on the whole the lower end receivers fair much better under the TB algorithm. This is due to the TB algorithm restriction on the maximum loss in its transmission rate calculation.
2. For the PL algorithm, the lowest receiver's experience considerable oscillation and extended periods of time when no usable bandwidth is being received. This is due to the more aggressive transmission rate that causes the lowest receivers to experience

high packet loss rates. For the top-heavy distribution, all seven low-end receivers experience periods of over 45 seconds when they are able to display none of the video signal. This can be seen starting at simulation times: 368, 518 and 593 in Figure 20-D.

3. While the PL algorithm transmission rate for the normal distribution (Figure 19-D) is only slightly more aggressive than the TB algorithms, the oscillation in the transmission rate accounts for most of the heavy losses on the low-end receivers.

In summary, we have found that the TB algorithm adjusts well to fluctuations in network availability and changes in the video coder. In addition, we have shown that the algorithm is successful at maximizing the displayable bandwidth at the receivers while limiting the maximum loss at the low-end receivers. One item that must be considered when utilizing this algorithm is the fact that a few low-end receivers may dominate the transmission rate. This is due to their impact on the maximization function.

4.8 VIC Results

In order to analyze the performance of the TB algorithm's implementation in the VIC system we need to understand how changes in resource availability affect the low-end receivers. Therefore, a small set of users will be sufficient to test the performance of this implementation in varying network conditions. The environment used was MSU's ATM test bed. The configuration for this is shown in Figure 21. This configuration consists of four Sun Ultras connected to two Fore ATM ASX-200 Switches via 155mb/s links. In addition, a HP analyzer was connected to the two switches to

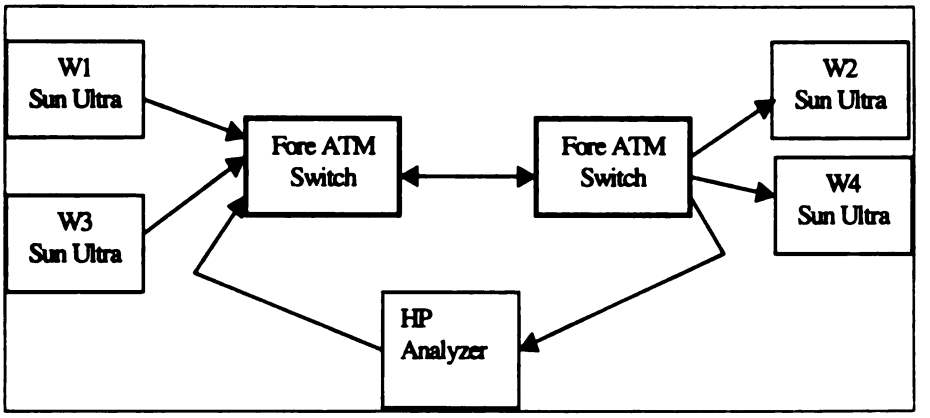


Figure 21:ATM Test Bed

provide background traffic. This configuration was chosen because it gives us tight control over the available network capacity.

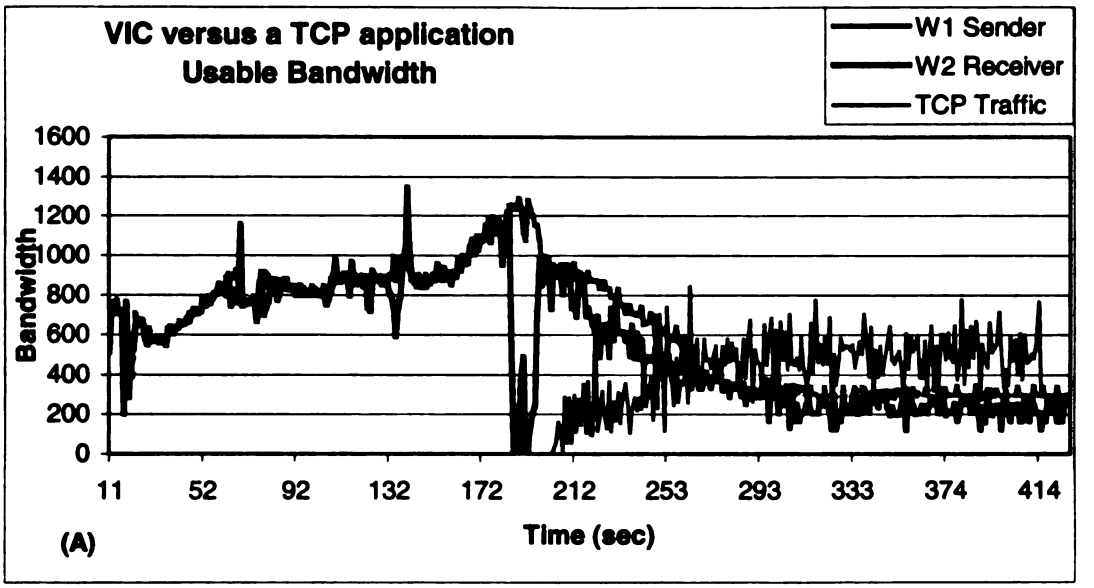
In order to monitor the performance of the algorithm in these test environments, we utilized the PGRT BRISK visualization system [2, 38, 52]. This system allowed us to monitor the performance of the VIC application on each of the workstations in real-time. In addition to providing an on-line performance tool, the BRISK system enabled us to extract the performance data for off-line analysis.

The focus of these tests was similar to the tests run via simulation. These tests focused on how variations in network resources impact the TB algorithm's ability to operate effectively. The first test examines the effects of TCP traffic on the TB algorithm. In this test, the HP analyzer was set up to consume 154 MB/s of the 155 MB/s link between the two ATM switches. This left a little over 1 MB/s for VIC and the TCP application.

In this test, we wish to determine how these two sources will interact. Using the same parameters as in the simulation model, the TCP source begins to dominate, see Figure 23. After one minute, the TCP source has begun to consume more bandwidth than the video application. In a separate simulation, not shown, we increased the smoothing parameter to 3 cycle times and the filtering parameter to ±6%. These increases had two effects. In some of the runs these increased parameters tripled the time it took for the TCP source to dominate the link to over 3 minutes. In a subset of runs the TCP source was unable to acquire any of the link and timed out.

Based on these findings we have come up with three conclusions.

1. In the first test (2 cycle smoothing and filtering of ±2%), long term TCP application will begin to dominate the link, while TCP applications of a few tens of seconds will interact well with the application. We tested this last point by FTPing small to medium size files (up to 2MB) over the congested link while VIC was running. While there was some fluctuations in the video transmission rate and loss in usable bandwidth, the two sources (FTP and VIC) were able to share the link.
2. In the second test (3 cycle smoothing and filtering of ±6%) the video application is fairly greedy. The tradeoff here is who should dominate the link, the long-term TCP application or the video source. This highlights the problem of multicasting video. Either the video application must transmit at the rate of the lowest receiver or there will be periods of congestion and someone will have to be greedy. This leads us to next point.



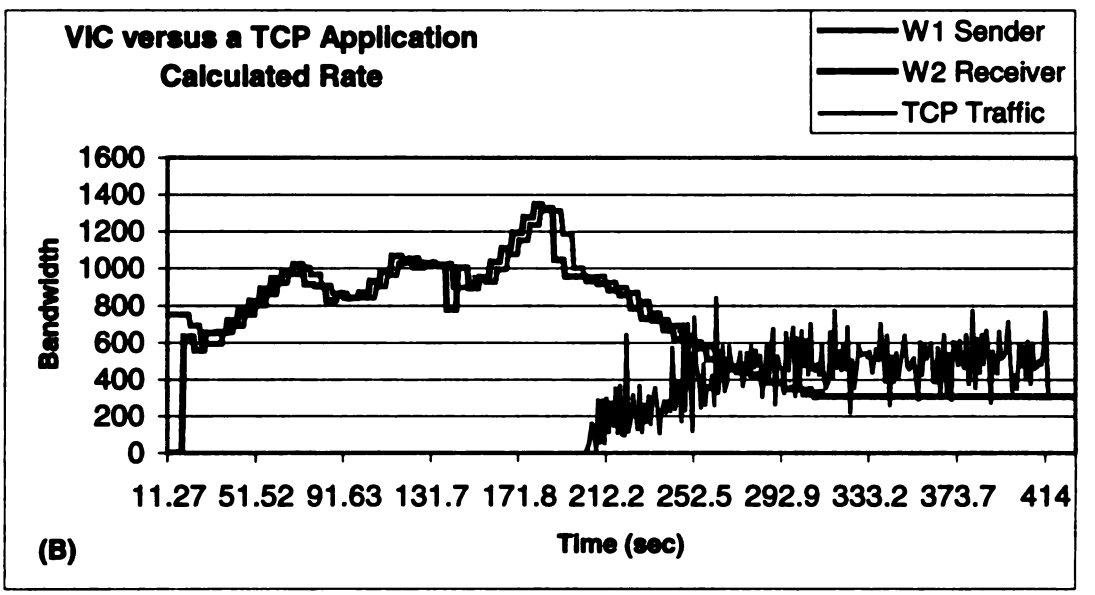


Figure 23: VIC with TCP Traffic

3. In the current Internet, the routers use a drop-tail queue management mechanism. This type technique allows for some applications to dominate and other applications to starve. Current research into other queue management algorithms, such as RED [17], will help alleviate this problem. In this type of algorithm, no source is able to completely dominate the link.

The second test involves the use of two video sources competing over the same link. The VIC application was first started between the workstation pair W1 and W2 and 130 seconds later on the pair W3 and W4. Figure 24 shows the results of this simulation. On this graph the lines for W1 and W3 represent the sender's target bandwidth rate. Lines W2 and W4 represent the receiver's feedback rate. As can be seen in this figure, as the second source is initiated the target bandwidth as determined by sender W1 drops to accommodate this new network demand. Once a state of equilibrium is reached, the two sources maintain a fairly stable target bandwidth rate as seen by lines W1 and W3.

4.9 Summary

This chapter presented a feedback based control algorithm for video conferencing on the Internet. This algorithm, called Target Bandwidth (TB), utilizes RTP control packets to transmit feedback information regarding the availability of the receiver's resources. The algorithm was designed to maximize the usable bandwidth at the receivers while limiting the maximum loss rate at the low-end receivers. In order to do this the algorithm estimates the loss at the workstation based on the receivers' feedback, the calculated network loss and the average packets per frame. From an implementation standpoint this algorithm uses exiting technologies and is of low complexity.

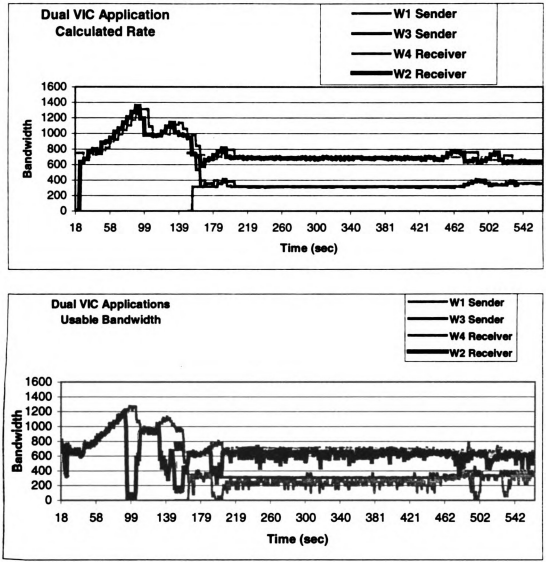


Figure 24: Multiple VIC applications

In order to show the effectiveness of the TB algorithm, we analyzed it using a simulation model and in the VIC video conferencing application. The simulations allowed us to analyze the algorithms performance under dynamic networking conditions. We studied the effects of background network traffic, long-term fluctuations in network resources, changes in receiver group sizes and differences in the distribution of network resources.

The results showed that the TB algorithm is stable under various network conditions and adjusts well to network rate fluctuations.

As part of the VIC application tests, we analyzed the performance of the TB algorithm while it interacts with different types of network traffic. Specifically we studied the effects of the algorithm competing with a TCP based application and the interaction between two VIC applications. The TCP test showed that with long-term interaction with a TCP application, someone will have to be greedy. This is an inherent problem with multicasting video in the Internet. Current research into queue management mechanisms will help to alleviate this problem. In the second set of test we were able to show that two video applications sharing the same link reach a state of equilibrium and remain fairly stable.

Chapter 5

5 Bandwidth Allocation for Layered Multicast Videoconferencing

5.1 Introduction

In the previous chapter, we examined an algorithm to control a videoconferencing application using a single multicast group. While we showed that this approach is able to control the transmission rate of the source in order to maximize the aggregate displayable bandwidth, this approach has two limitations. First, since the low-end receivers network links have the potential to be saturated, they may not interact fairly with TCP. A TCP connection assumes that when congestion occurs all sources will throttle back their transmission rates. A TCP connection interacting with a low-end TB controlled connection will experience at best poor performance and will most likely time out. This is because the TB algorithm does not respond to congestion on any individual link. The

second limitation of this approach is that it will leave many receivers unsatisfied if the resource requirements of all receivers vary significantly. Even when an optimal transmission rate is chosen, some of the receivers will be experiencing packet loss while others will not have their resources fully utilized.

In this chapter, we examine an alternative approach. This approach utilizes multiple multicast groups in order to meet the varying bandwidth requirements of the receivers. In this approach, called the Layered Multicast Control Protocol (LMCP), the sender stripes a single video signal across multiple multicast groups. The receivers selectively add and drop channels to meet their individual needs. In addition, the receivers send feedback to the sender, and the sender uses this feedback to adjust the transmission rate for each channel. In this chapter we present and analyze three different algorithms the sender might use for computing the transmission rates for each channel. Our analytical and simulation results show that this new protocol significantly improves upon the performance of both previous protocols. In particular, this protocol achieves essentially optimal bandwidth utilization at the receivers.

5.1.1 Overview

One of the difficulties in multicasting a video stream over the Internet is determining the rate of transmission that best meets the receivers' requirements. Previous rate-control can be divided into two types. The first is a sender based rate-adaptation approach [5, 44] where the sender multicasts a single video signal and adjusts its transmission rate based

on feedback from the receivers. This approach works well if all receivers have similar network resources, but it performs poorly if the receivers have large differences in their available network resources. The TB algorithm presented in chapter 4 is an example of this type of approach.

The second approach is a receiver based rate-control protocol [7, 22, 28, 31, 50]. In this solution, the single video stream is split into multiple segments in order to transmit the stream across multiple multicast channels. The receivers are then responsible for adding and dropping the multicast channels to best meet their available resources. The sender is not dynamic and transmits the video stream at pre-determined (fixed) rates [31, 50]. While this approach may better meet the diverse requirements of the receivers, due to its lack of sender rate-adaptation, it will not maximize the utilization of the receivers' bandwidth. An example of this approach, the RLM protocol, was presented in chapter 2.

In this chapter, we present a new approach that combines the strengths of the previous two approaches. More specifically, we utilize both multiple channel transmission concepts from the receiver-based approach and the transmission rate adaptation concept from the sender based approach. In order to implement sender based rate-adaptation, we have developed a mechanism for the receivers to transmit estimates of network congestion to the sender. These estimates are generated by taking advantage of certain features of the receiver based rate-control algorithm. In addition, we have implemented four different transmission rate calculation algorithms that are used by the sender to analyze this feedback in order to select a transmission rate for each multicast channel.

The remainder of this chapter is organized as follows. First we present our protocol, the Layered Multicast Control Protocol (LMCP). We then provide a statistical analysis of this protocol. Finally, we present the results of a simulation implementation of LMCP.

5.2 Layered Multicast Control Protocol (LMCP)

We introduce a new control protocol that combines the strengths of both the sender based rate-adaptation protocol and the RLM protocol [28, 31]. In this approach both the sender and all receivers are actively involved. The receivers not only perform the basic RLM protocol, they also dynamically approximate their available bandwidth and provide this as feedback to the sender. The sender processes the receivers' feedback to determine the optimal rate of transmission for each of the multicast layers (see Figure 25)

5.2.1 The LMCP Receiver Feedback Calculation

Each receiver dynamically estimates the *bottleneck rate* between the sender and that receiver (the receiver's available bandwidth). By bottleneck rate, we mean the maximum transmission rate available on the path between the sender and the receiver. Each receiver calculates this bottleneck rate as follows. When a receiver adds a multicast channel through a join-experiment, the receiver's LMCP process monitors the bandwidth received across all of its multicast layers. More specifically, it maintains a byte counter which counts the total number of bytes received across all multicast layers since the first packet arrived on the most recently added channel, and it maintains a timer which

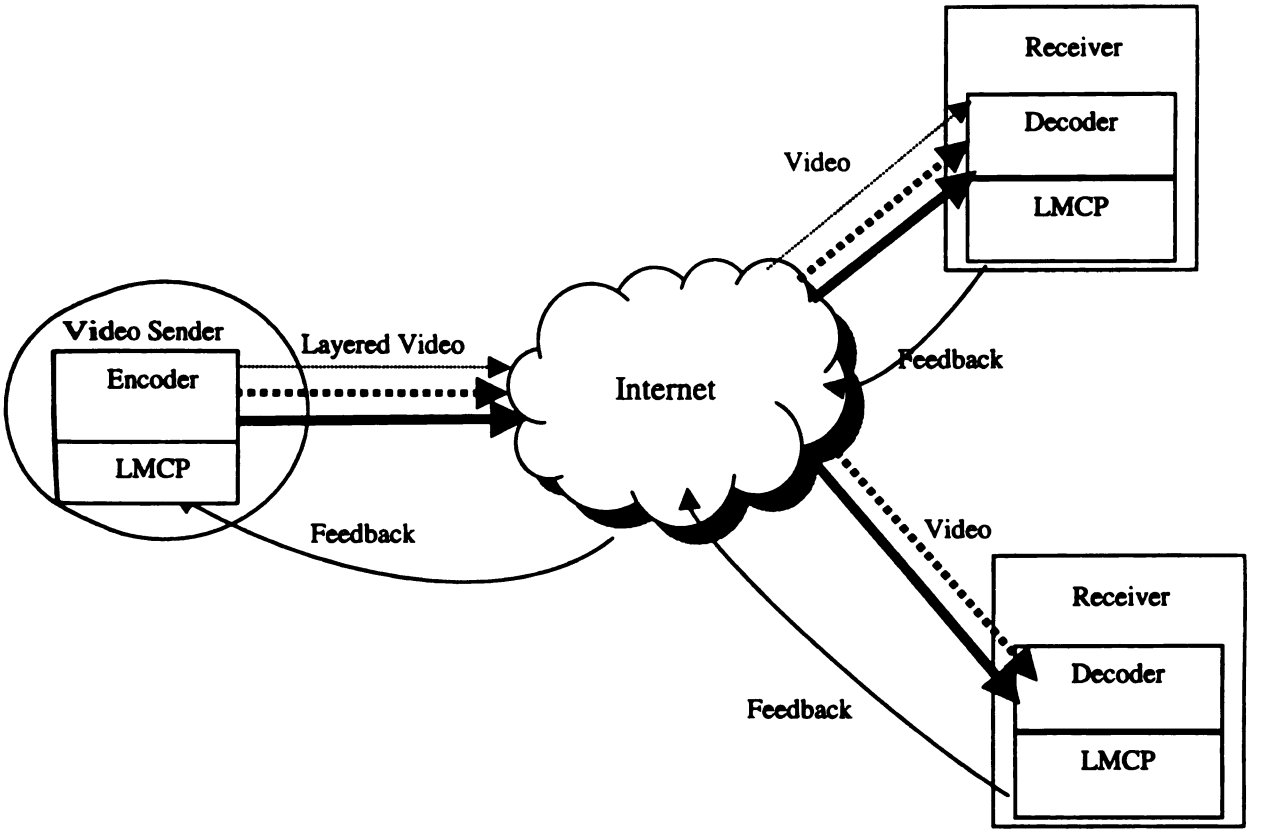


Figure 25: Layered Multicast Control Protocol (LMCP)

measures the amount of time that has passed since the new channel was added. If the newest channel is dropped due to congestion, the receiver estimates the bottleneck rate by dividing the current byte counter value by the current timer value. This estimate is then transmitted to the sender. On the other hand, if a join-experiment succeeds (and therefore there is no congestion), then the LMCP feedback process resets the timer and byte counter and waits for another join-experiment.

5.2.2 LMCP Sender Transmission Rate Calculation

The sender side of the protocol is responsible for collecting the receivers' feedback and calculating a new transmission rate for each multicast channel which maximizes the amount of the video signal received by the receivers. This problem can be formally stated as follows: Given n receivers where receiver i can receive r_i units of transmission, find a set of k cumulative transmission rates t_j such that $\sum_{i=1}^{n}(r_i-(r_i-t_j))$ is maximized where t_j is the maximum cumulative transmission rate such that $t_j \leq r_i$. Typically, $k << n$. This objective function can be simplified to minimizing $\sum_{i=1}^{n}(r_i-t_j)$. Figure 26 gives a more detailed description of this problem using set $\mathbf{R} = \{r_i\}$ and set $\mathbf{T} = \{t_j\}$. The objective can be restated as finding an optimal mapping from R $\Rightarrow$ T. Note, if set T is an optimal set of transmission rates (so transmitting using set T maximizes the bandwidth received by the receivers), then T$\subset$ R.[1]

---

[1] Since channels are dropped by the receivers when congestion occurs, picking a transmission rate $t_i \varepsilon T$ such that $r_a < t_i < r_{a+1}$ for some a, where $r_a, r_{a+1} \varepsilon$ R (and assuming R is sorted) and $r_a \neq r_{a+1}$, means r_a cannot receive channel i and setting $t_i = r_{a+1}$ will provide optimal results. Since $t_i \varepsilon$ R then T$\subset$ R.

Goal:

Given a fixed number of multicast connections, k, maximize the usable bandwidth at the n receivers

Inputs:

Let $\mathbf{R} = \{r_i \mid i \leq n\}$

- o R is the set of all the Receivers' bottleneck rates
- o r_i is receivers i's bottleneck rate

Let $\mathbf{T} = \{t_j \mid j \leq k\}$

- o T is a set of cumulative transmission rates
- o t_j is the cumulative transmission rate over connections 1 to j in bits/second

Task:

Find a mapping from R $\Rightarrow$ T that maximizes the usable bandwidth on the receivers.

So our Maximization Function is: $Max\left[\sum_{i=1}^{n}(r_i - (r_i - t_j))\right]$ or: $Min\left[\sum_{i=1}^{n}(r_i - t_j)\right]$

where t_j is the maximum channel such that $t_j < r_i$

Figure 26: Sender Transmission Rate Calculation

5.2.3 LMCP Sender Transmission Rate Algorithms

In this section, we present four different algorithms for computing transmission rates (mapping the set R $\Rightarrow$ T). The first is a dynamic programming solution that finds the optimal set of transmission rates T. By optimal, we mean a set of rates T that provides

the maximum usable bandwidth at the n receivers for a given number k of channels. This algorithm has a run time complexity of $O(kn^3)$ and a space usage of $O(n^2)$ which makes it impractical for large values of n. See appendix A for an overview of this algorithm. While we independently developed this dynamic programming algorithm, it was originally presented by Shachan in 1992 [42].

The second approach is a non-optimal divide and conquer algorithm, which produces transmission rates that are very close to those produced by the optimal dynamic programming algorithm in many cases. While the complexity of this approach is $O(\binom{2k}{k} * kn)$ which is exponential in the number of channels, for reasonable k ($k < 6$) and $n >> k$, this approach has a significant runtime performance advantage over the dynamic programming algorithm. See appendix B for an overview of this approach.

Finally, we consider two less adaptive approaches which serve as a baseline which we can use to judge how well the dynamic programming and divide and conquer algorithms perform. In the first approach the sender's transmission rates are fixed. One possibility for these channel rates are powers-of 2 (e.g.; 32 Kb/s, 64 Kb/s, 128 Kb/s). The second approach picks bandwidths based on a fixed percentile which is dependent on the number of channels. In this approach $t_j = R_{offset*(j)}$, where offset = $\left\lfloor \frac{n}{k} \right\rfloor$. Since these approaches are not as adaptive or dynamic as the previous two algorithms, they do not perform well

under certain distributions of receivers' bottleneck rates. However, when the number of channels, *k*, is fairly large, the fixed percentile approach performs well.

5.3 Statistical Analysis

In this section, we compare the performance of three of the transmission rate calculation algorithms presented in the previous section. Unlike the simulations presented in section 4, this comparison is strictly analytical. Our purpose is to compare how the three algorithms perform under varying distributions of receiver bottleneck rates. The key metric for comparison is the percentage of the total bandwidth used. This *percentage-used* metric is calculated as the sum of the bandwidth-received for all *n* receivers divided by the sum of the total possible received. The higher the *percentage-used* the better, with a *percentage-used* of 1 meaning all receivers are receiving at their maximum rate. The calculation of this metric is:

$$Percentage\text{-}used = \frac{Total\_received}{Total\_possible} = \frac{\sum_{i=1}^{n}(r_i - (r_i - t_j)}{\sum_{i=1}^{n} r_i},$$

where t_j is the maximum transmission rate such that $t_j \leq r_i$

In this section, we will not be concerned with how the algorithms interact with RLM but only how well they perform given a set R of receivers' bottleneck rates. Therefore, we will assume that $r_i \varepsilon R$ is known for each receiver.

In order to study the effects of the varying distributions of receivers' network resources, we will consider three different distributions of receiver bottleneck rates which we label as clustered, top-heavy, and uniform. Figure 27-(A, B, C) illustrates these three basic distributions. While we analyzed the results for numbers of receivers ranging from 10 to 950, the underlying rate distributions were the same for all cases.

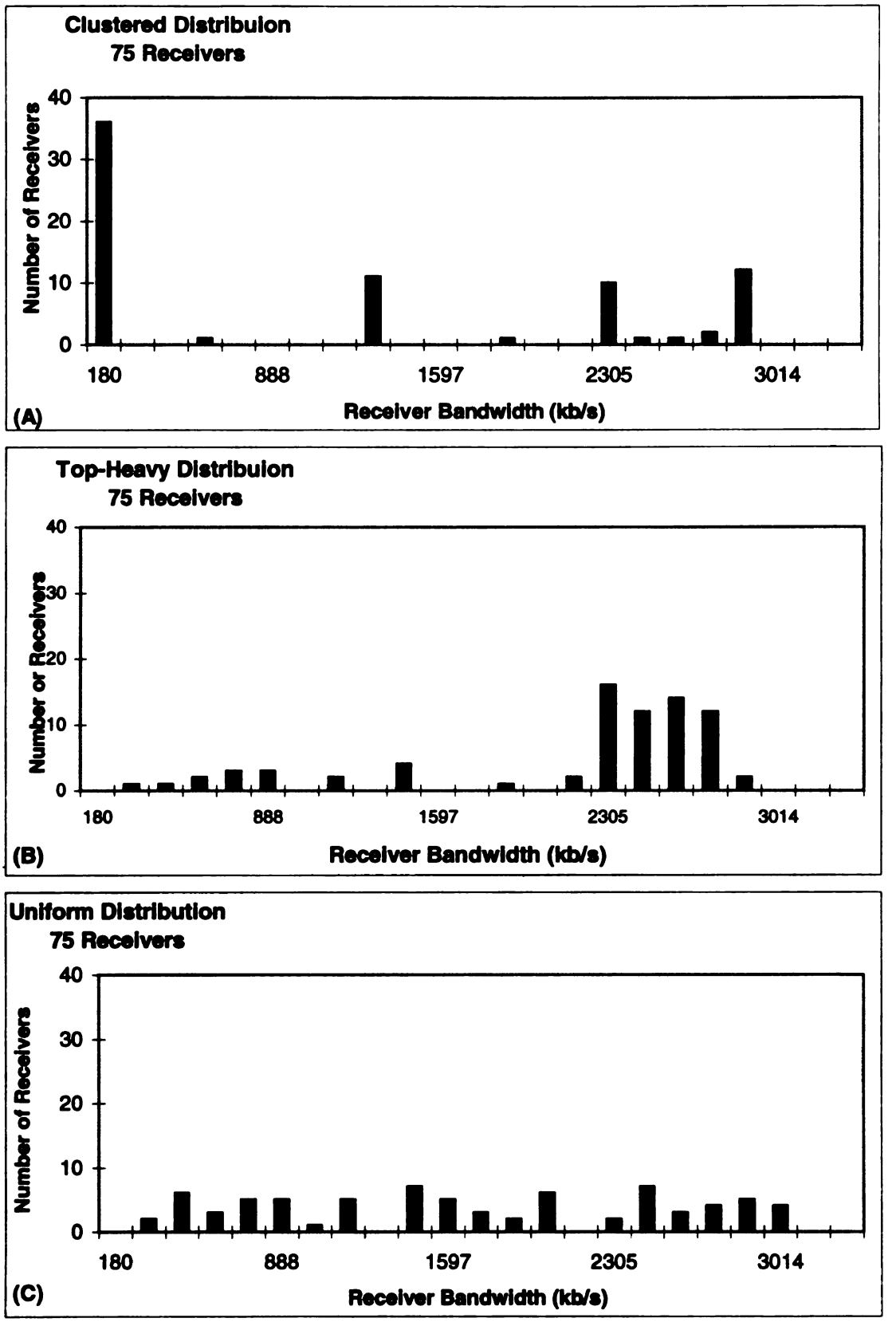


Figure 27: Receiver Distributions (A,B,C)

As Figure 28-(A,B,C) shows, the two adaptive algorithms achieve a very high *percentage-used.* In particular, for both the clustered and top-heavy distributions, the algorithms achieve a *percentage-used* of 94% or better using 5 or more channels. The performance of the fixed percentile approach is inferior to the performance of the two dynamic algorithms for both the clustered and top-heavy distributions, but it has essentially the same performance for the uniform distribution. Also, as can be seen in Figure 28-A, there are instances where adding a channel in the fixed percentile approach causes the *percentage-used* to go down. This is due to the transmission rate being determined solely on the number of channels without taking into account the distribution of the receivers' bottleneck rates.

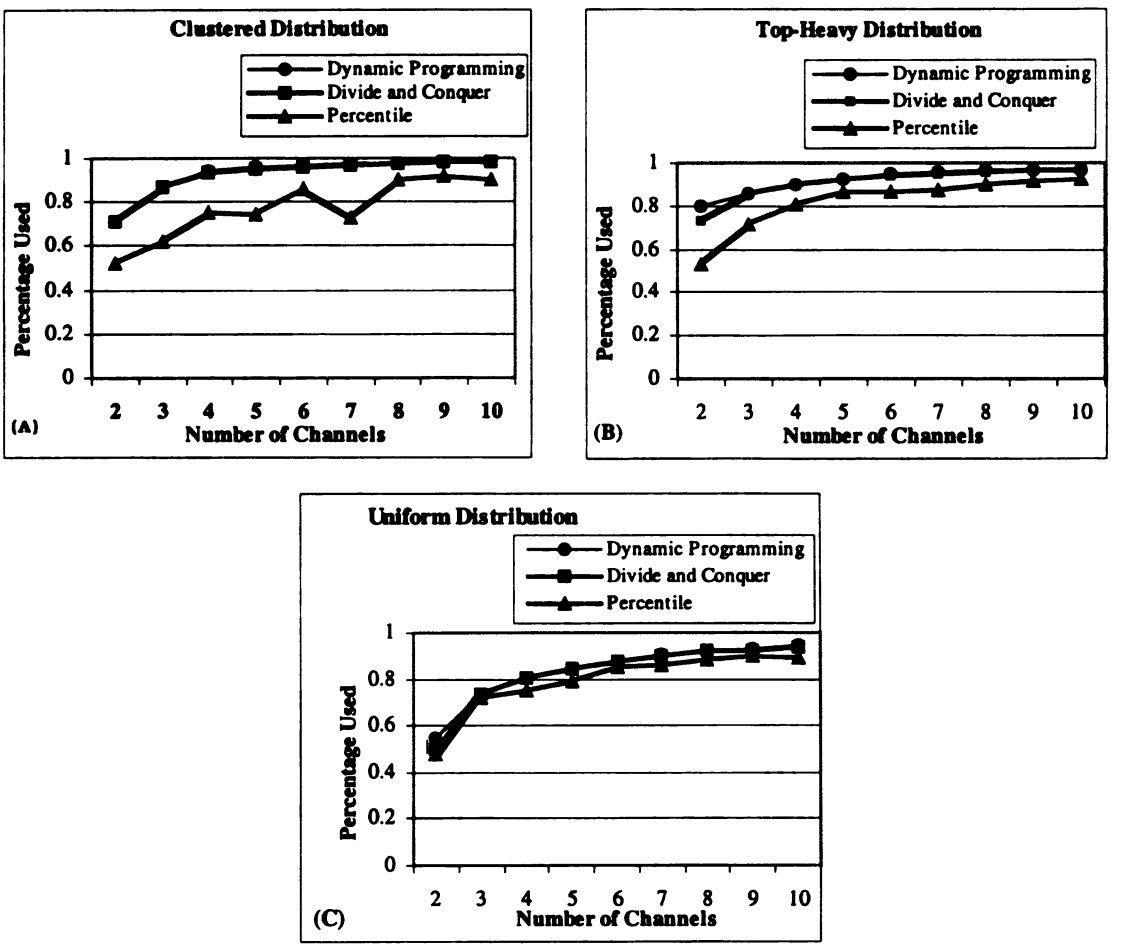


Figure 28: *Percentage-used*, Analytical Performance Results (75 receivers)

5.4 LMCP Simulation

In order to analyze the LMCP in more detail we have implemented it using the *ns* [29] simulation package. This simulation analysis allows us to look at how the protocol performs under more realistic conditions. Specifically, how the protocol works with a set of receivers' feedback that is calculated dynamically rather than known in advance. In addition it requires us to specify the mechanisms that must be in place in order to support our approach. In this section we give an overview of some of the key details of the *ns* implementation, discuss the changes required to support our protocol, and then present the performance results. In an effort to simplify our discussion, we have limited our simulations to the optimal dynamic programming algorithm.

5.4.1 NS Implementation

Our ns simulation was built upon S. McCanne's [29, 31] RLM *ns* implementation. Changes were necessary in both the receiver and sender's RLM code. The receivers' code was modified to generate an estimate of the available bandwidth as determined by measurements during the RLM join-experiments. This feedback, along with a receiver ID, is periodically transmitted to the sender via a control channel.

The implementation of the sender's side of the LMCP algorithm requires some additional features in order to allow it to work in a simulation environment. First, the sender was modified to accept the receivers' feedback and store this information in a hash table.

Second, an aging process was added to maintain this hash table and remove outdated information after three iterations of the control algorithm. This aging process allows the sending application to be pseudo-stateless. While the sender keeps track of the feedback by receiver ID, no notification of receiver startup or termination is required. The aging process also improves the scalability of the algorithm by not requiring feedback from each receiver every time period.

Two additional changes were required to support the sender's rate control algorithms. First, in order to provide service to all receivers, the lowest multicast channel was locked into transmitting at *min_rate*[1]. This is necessary in order to prevent the control algorithm from picking a transmission rate that would prevent some low end users from receiving any of the video channels. While this channel was included in the optimal calculation of transmission rates, it was not allowed to vary.

The second change is the addition of a floating channel. This channel is the k+1st layer of the video stream. It is used to help the high-end receivers estimate their maximum rate. This prevents the lower end receivers from completely dominating the sender's calculation of the transmission rates. While this channel carries video like all the other transmission channels, both the sender and the receivers utilize it slightly differently in the LMCP algorithm. The sender does not consider this channel in the calculation of the

---

[1] For our purposes we set *min_rate* to 32Kb/s. In any specific video application, any setting of this rate needs to take into account the video coder's limitations and the type of receivers the sender wishes to support.

transmission rates. Instead this channel's transmission rate is set to a multiple of the rate of the T_k channel. The receiver utilizes the video on this channel in the same way as all other channels. The only difference is how the receiver uses this channel in its bottleneck rate calculation. If a receiver has a successful join-experiment on this floating channel, they calculate the bits per second being received across all of the multicast channels and transmit it as their feedback rate even though no packet loss was experienced. While this feedback is not a true bottleneck rate, it does allow the algorithm to take into account the high-end receivers.

5.4.2 Simulation Results

In this section we present the results of our *ns* simulation runs. Figure 29 shows the network used in these runs. In this set of simulations we used a network with 10 receivers. Results from simulations using 4 and 5 transmission channels are presented. Since the float channel is not considered in the sender's algorithm, these simulations were run with k=3 and k=4. Table 5 shows the key parameters used for these simulation runs.

Figure 30-(A, B) shows the calculated channel settings (set T) for simulation runs with 4 and 5 transmission channels. These figures show the accumulated transmission rates for each of the channels. The initial setting of the channels were power of 2's starting from 32kb/s (32, 64, 128, 256, 512). Two hundred seconds into the simulations the sender's LMCP algorithm is run on the receivers' feedback and the channel transmission rates are adjusted accordingly. By utilizing these initial rates and delaying the running of the

sender's algorithm we provide time for the RLM algorithm to perform join-experiments on the higher end channels.

Figure 31-(A,B) shows the bandwidth received by the 10 receivers for the two simulation

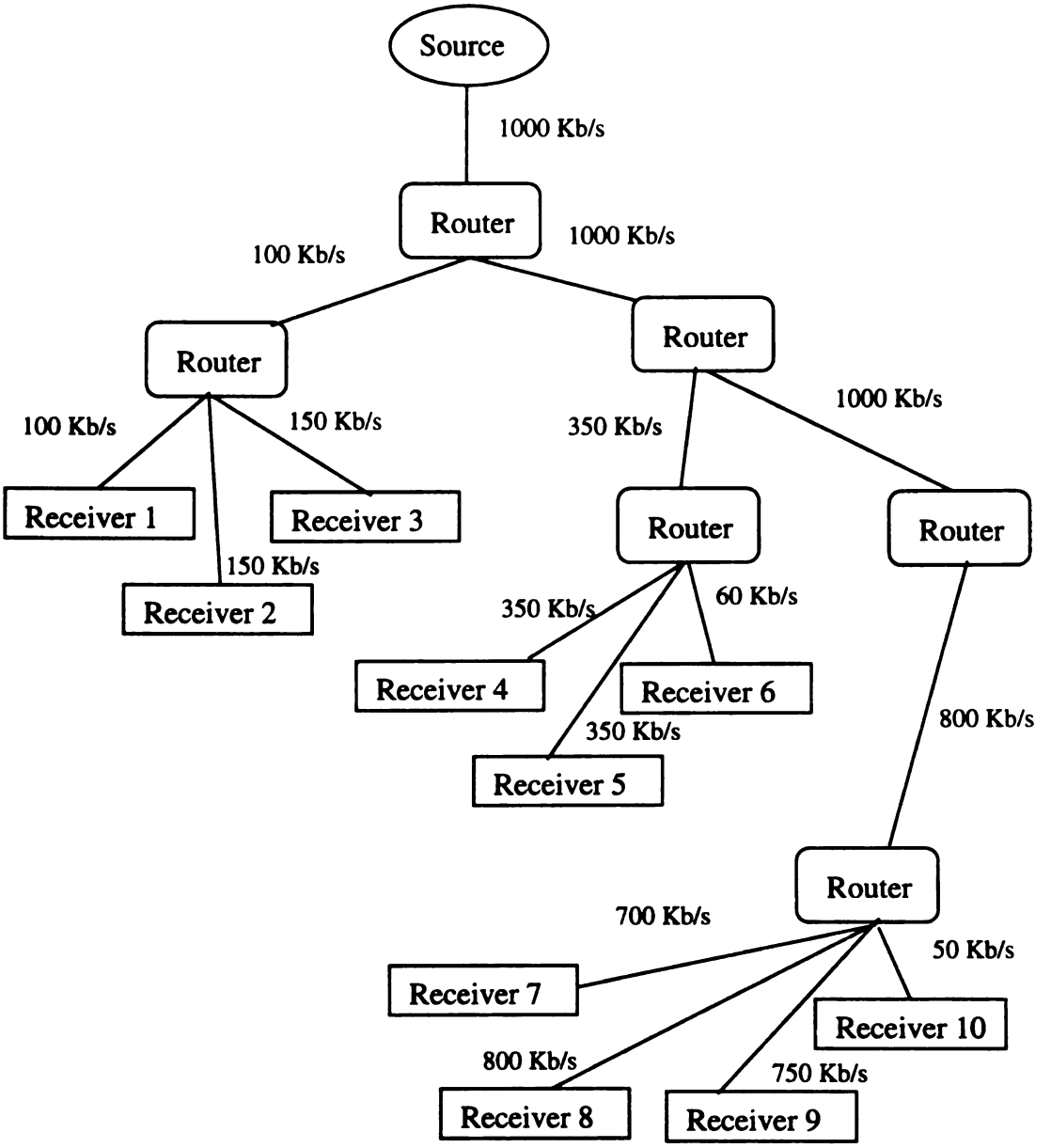
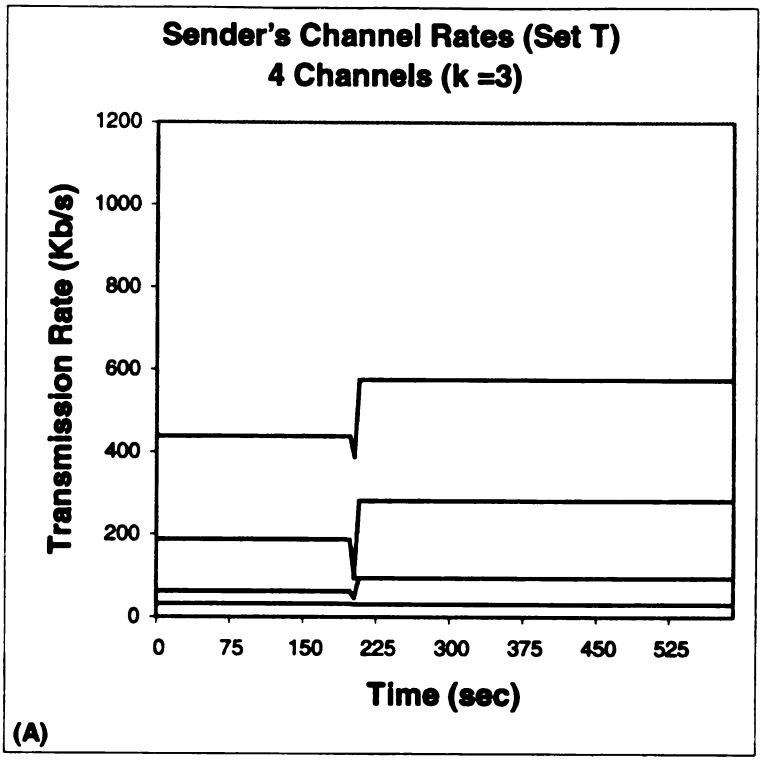


Figure 29: Simulation Network

runs. These figures show that the LMCP protocol quickly stabilizes on the new transmission rates. This can be seen by the overall stability of the receivers' rates and the sender's smooth transmission rate. These figures also show that the amount of bandwidth received by the receivers experiences short-term fluctuations during the course of the simulations. These fluctuations are due to the interaction of the receivers' join experiments. This type of fluctuation is typical for a RLM implementation. Figure 30-(A,B) also shows that the channel transmission rates remain fairly stable after the optimal algorithm has been run. Therefore, any fluctuations in the amount of bandwidth received by the receivers are due to the RLM protocol.

Parameters	Setting
Simulation Duration	600 Seconds
Sender Initial Rates	32, 64, 128, 256, 512 Kb/s
Period of Sender Algorithm	Every 20 Seconds
Sender Initial Algorithm Run	200 Seconds into Simulation
Float Channel	$2 * T_k$ (max = 1Mb/s, min = 300Kb/s)
Fixed, lowest channel *min_rate*	32 Kb/s
Receiver Feedback Period	Every 5 Seconds with a random offset
Router	Drop Tail Queuing, Prune time of 1 second
Packet Size	1000 Bytes

Table 5:Simulation Parameters



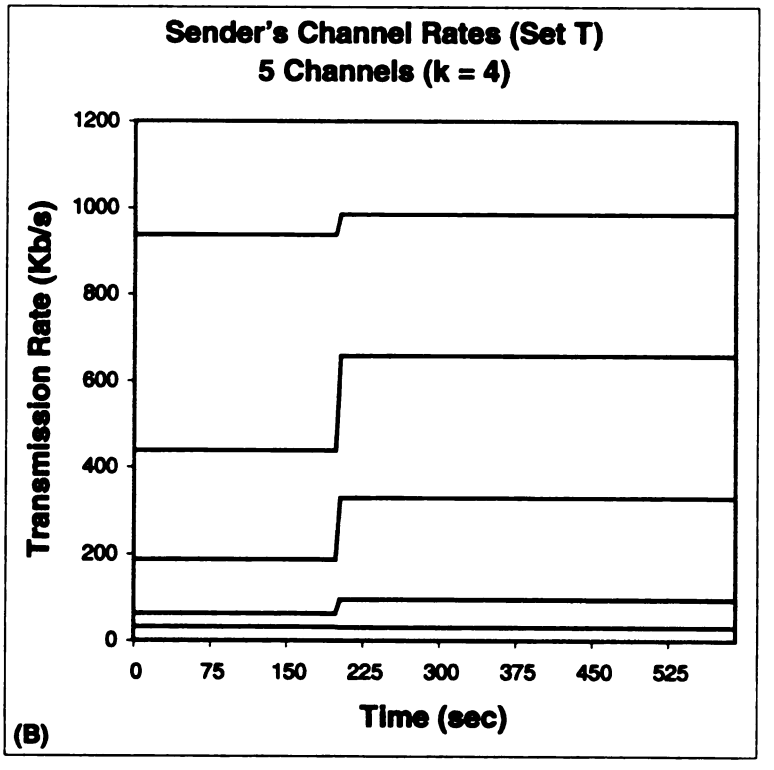
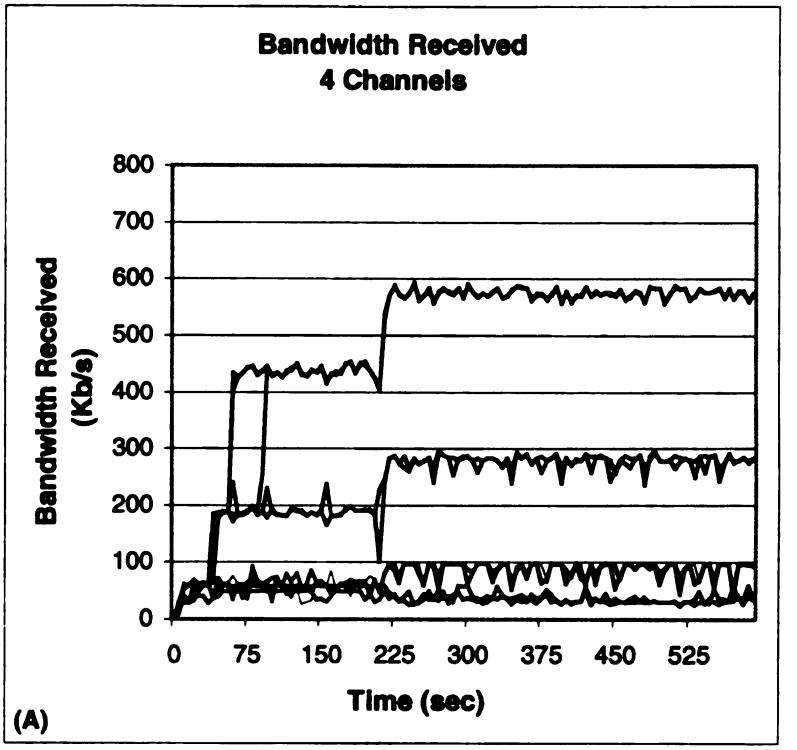


Figure 30: Sender's Transmission Rates (A,B)

Figure 32-A shows the total bandwidth used for all the receivers for both the 4 and 5 channel simulations. For comparison, this figure also shows the results of a fixed-transmission rate simulation using 5 transmission channels. In this fixed channel implementation, the sender continues to transmit at the initial rates (32k, 64k, 128k, 256k, 512k). This figure shows that the dynamic approach used in LMCP significantly outperforms this static approach. Also shown in this figure is a line at 3360 Kb/s that represents the maximum possible bandwidth received. This line was generated by adding all of the receivers' bottleneck rates. Figure 32-B shows the *percentage-used* for the two LMCP simulations and the fixed channel. This *percentage-used* is calculated by dividing the sum of bandwidth received by the maximum possible (3360 Kb/s). Once the LMCP algorithm is run on the sender (after 200 seconds), the average *percentage-used* for the 5-channel simulation is 87%. In comparison the fixed channel results in a *percentage-used* average of 58%.



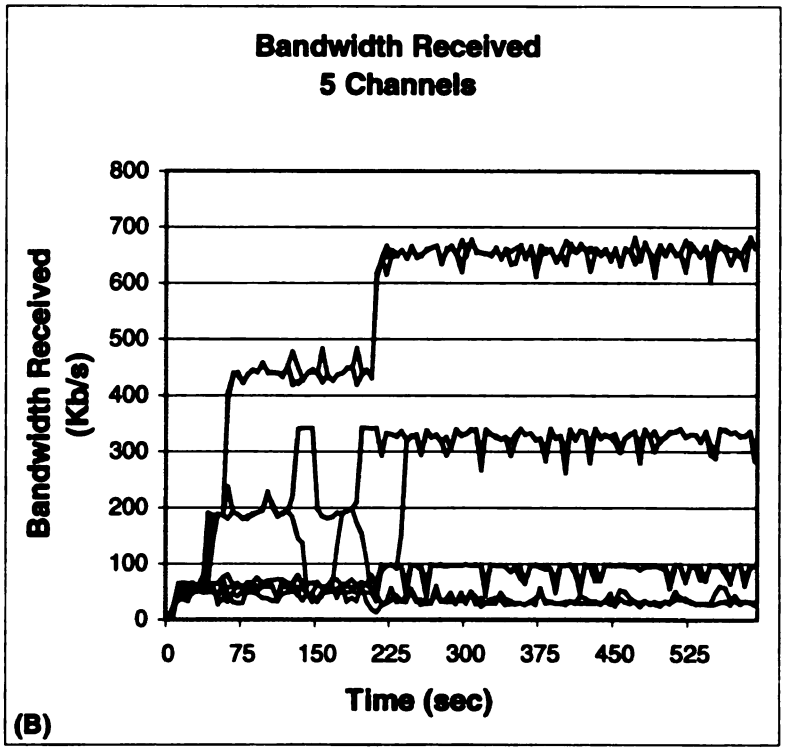
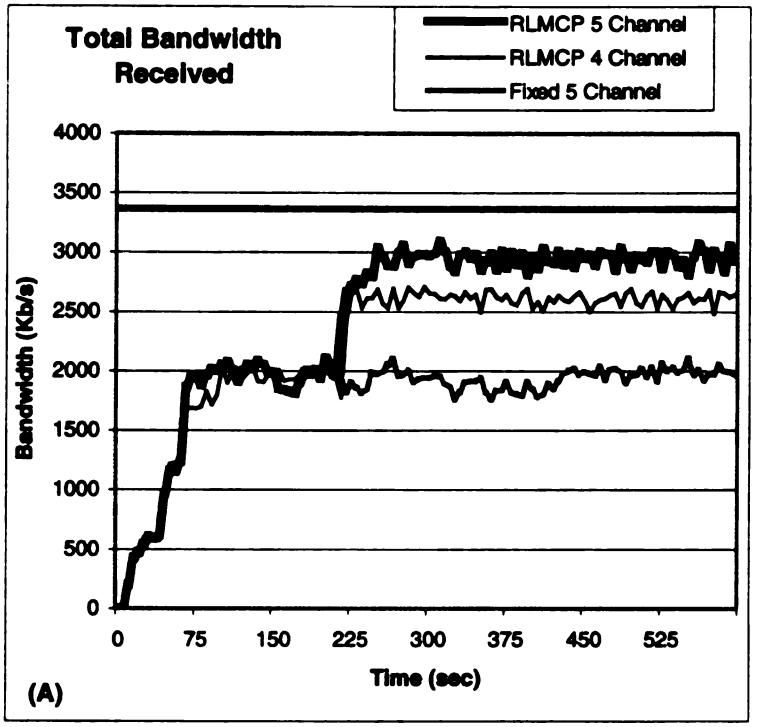


Figure 31: Receivers' Bandwidth Received (A,B)



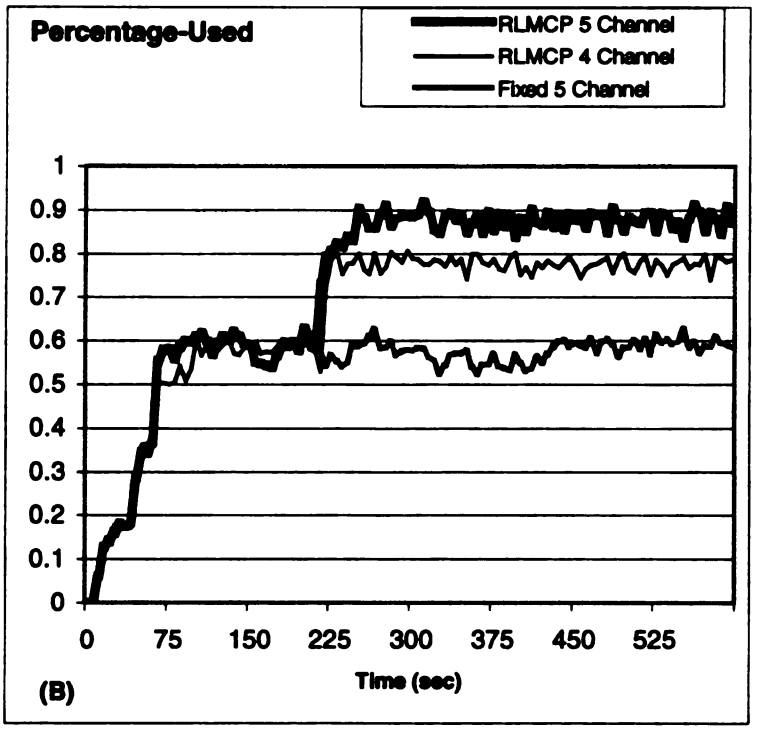


Figure 32: Total Received (A) and Percentage-used (B)

5.5 Summary

In this chapter we presented the LMCP protocol. In this approach, the receivers continue to add and drop layers utilizing RLM as their available resources dictate, and they also send their bottleneck rates to the sender as feedback. Meanwhile, the sender uses this feedback to adjust the transmission rates on each channel. Note that we presented a method for calculating and transmitting this feedback, which has very little over-head on the receivers and the sender. Furthermore, on the sender side, we presented four algorithms for calculating transmission rates for each of the multicast channels. One algorithm produces optimal transmission rates while a second algorithm produces near optimal transmission rates

In order to understand the performance of these two algorithms, we presented a statistical analysis using three different distributions of receivers' bottleneck rates. This analysis showed that both of these algorithms deliver a high *percentage-used* for all three distributions. In addition we analyzed the run-time performance of the two algorithms as the number of receivers increases. This analysis showed that for large numbers of receivers, the run time performance of the optimal dynamic programming algorithm becomes prohibitive. While the divide and conquer algorithm, which is less sensitive to the number of receivers, had a significantly better run-time performance with no perceptible loss in bandwidth received by the receivers.

Using the *ns* simulation package, we examined how the LMCP algorithm performs in a more realistic environment with 4 and 5 transmission channels. These simulations showed that the protocol continued to achieve a high *percentage-used*, even when the bottleneck rates are estimated dynamically rather than known in advance. In the simulations with 5 transmission channels, the protocol achieved an average *percentage-used* of 87%. In comparison, a fixed rate approach achieved an average *percentage-used* of 58% for the same set of receivers

Chapter 6

6 Feedback Scalability

6.1 Introduction

In the last chapter we introduced the Layered Multicast Control Protocol, which controls the transmission rate of multicast video to a heterogeneous group of receivers. This approach utilizes feedback from the receives in order to determine transmission rates which maximize the video displayed at the receivers. In the last chapter we addressed the issue of scalability of the sender's control algorithm by introducing a divide and conquer algorithm (see appendix B). This chapter addresses the scalability of the receivers' feedback.

As the number or receivers increase into the hundreds, the amount of feedback arriving at the sender will increase beyond the sender's ability to receive and process the information. There are two approaches for dealing with this problem. First, the control period, which is the amount of time between successive runs of the sender's algorithm, may be lengthened. This will allow the receivers to increase the time they wait before

transmitting their feedback packet. This will decrease the overhead of the feedback on the sender. The downside to this approach is that it reduces the responsiveness of the control protocol to changes in available bandwidth. The second approach is to reduce the number or receivers providing feedback in a control period. In this approach, a random subset of the receivers generate and transmit their feedback during a control period. The sender uses this subset of rates in order to calculate its transmission rates. In this chapter we develop an algorithm for implementing this type of approach through statistical sampling and discuss the accuracy of this approach.

The remainder of this chapter is organizes as follows. In the next section we discuss the performance metric used in order to calculate n, our sample size. In section 3 we show the derivation of n. Section 4 utilizes the equation for n in an iterative algorithm to determine the sender's transmission rates. Section 5 presents a performance analysis of this algorithm, specifically looking at how this sampling impacts our performance metric *percentage-used*. We conclude this chapter by summarizing our findings.

6.2 Statistical Sampling

In the last chapter we introduced the metric *percentage-used*, which is calculated as:

$$Percentage\text{-}used = \frac{Total\_received}{Total\_possible} = \frac{\sum_{i=1}^{N}(r_i - (r_i - t_j)}{\sum_{i=1}^{N} r_i} = \frac{\sum_{i=1}^{N} k_i}{\sum_{i=1}^{N} r_i}$$

Where N is the total number of receivers, $r_i \in R$ is receiver i's bottle neck rate and $t_j \in T$ is the maximum transmission rate such that $t_j \leq r_i$ and $k_i = r_i - (r_i - t_j)$.

In the LMCP, the sender's control algorithm picks transmission rates, which maximize this *percentage-used.* In this section we are interested in determining the number of receivers which must be sampled in order to allow us to calculate a *percentage-used* to within an error of ε and a confidence of $1-\alpha$. Stating this formally we are looking for the smallest *n* such that $P(|P - \hat{P}| > \varepsilon) \leq \alpha$, where *P* is the true *percentage-used* and $\hat{P}$ is the *percentage-used* achieved using only a subset of the receivers' bottleneck rates.

One difficulty in calculating this *n* is the independence of the two variables, k_i and r_i, in the calculation of *percentage-used.* The fourth column in Table 6 shows the correlation coefficient, ρ, between the two variables, k_i and r_i, used in this calculation. This table is

Distribution Number	Percentage-used	*n* ε=.05, α=.05	Correlation k_i and r_i	Correlation w_i and r_i
1	.92	97	.93	.60
2	.90	54	.98	.54
3	.87	63	.98	.52
4	.86	49	.97	.22
5	.87	41	.98	.08
6	.86	40	.98	.13

Table 6: Variable Correlation for the 6 receiver distributions show in Figure 33

based on 500 receivers using receiver bottleneck rate distributions randomly picked between slightly clustered to uniformly distributed. The histograms for these distributions are shown in Figure 33. The calculation of ρ is based on Pearson's product moment correlation coefficient as given in [9]. The value of ρ may range between –1 and 1. As can be seen in this table, there is a very high correlation between these two variables regardless of the distribution.

As an alternative to maximizing *percentage-used*, the sender's control algorithm may minimize the *percentage-wasted*; where *percentage-wasted* = 1 – *percentage-used* and is calculated as:

$$Percentage\text{-}wasted = \frac{Total\_wasted}{Total\_possible} = \frac{\sum_{i=1}^{N}(r_i - t_j)}{\sum_{i=1}^{N} r_i} = \frac{\sum_{i=1}^{N} w_i}{\sum_{i=1}^{N} r_i}$$

Where $r_i \in R$ is receiver i's bottle neck rate and $t_j \in T$ is the maximum transmission rate such that $t_j \leq r_i$ and $w_i = r_i - t_j$.

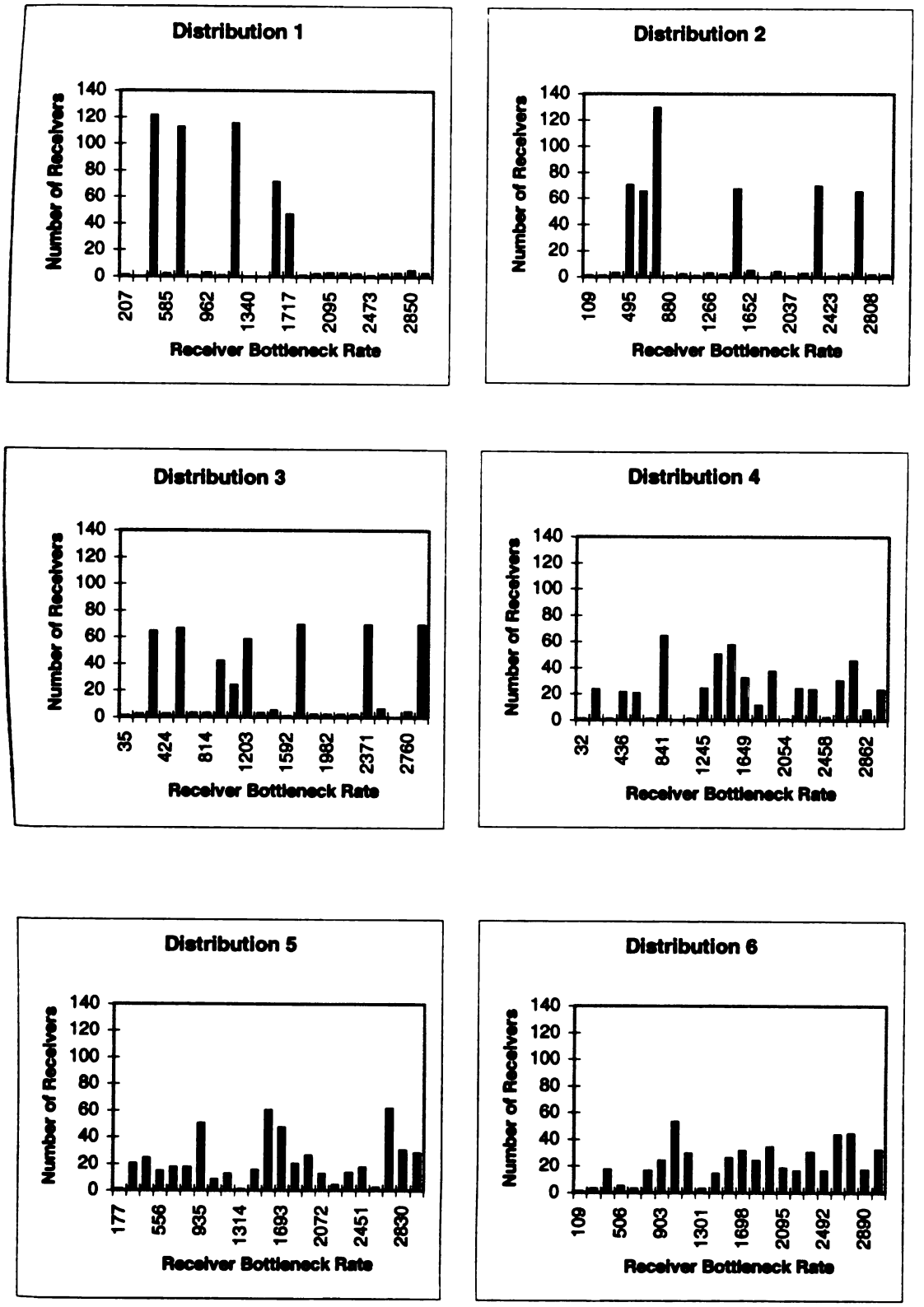


Figure 33: Histograms of Receivers' Rate Distributions

We may then restate the problem as looking for the smallest n such that $P(|W - \hat{W}| > \varepsilon) \leq \alpha$, where W is the true *percentage-wasted* and $\hat{W}$ is achieved using only a subset of the receivers' bottleneck rates. As the fifth column in Table 6 shows, the correlation coefficient, ρ, for the two variables w_i and r_i is significantly less. Therefore, we may assume that the distribution of *percentage-wasted* is asymptotically normal for sufficient size n.

6.3 Derivation of *n*

In this section we derive the calculation for n using W, the *percentage-wasted* metric.

$$\text{Let } W = \frac{\sum_{i=1}^{N} w_i}{\sum_{i=1}^{N} r_i} = \frac{\frac{1}{N}\sum_{i=1}^{N} w_i}{\frac{1}{N}\sum_{i=1}^{N} r_i} = \frac{\mu_w}{\mu_r}$$

With variance σ_w and σ_r

And

$$\text{Let } \hat{W} = \frac{\sum_{i=1}^{n} w'_i}{\sum_{i=1}^{n} r'_i} = \frac{\frac{1}{n}\sum_{i=1}^{n} w'_i}{\frac{1}{n}\sum_{i=1}^{n} r'_i} = \frac{\hat{\mu}_w}{\hat{\mu}_r}$$

With $t'_j \in T'$, where T' is calculated using $r'_i \in R'$ with R' is a random subset of R of size n, and t'_j is the maximum transmission rate such that $t'_j \le r'_i$ and $w'_i = r'_i - t'_j$

Then based on the central limit theorem

$$\hat{W} = \frac{\mu_w + \frac{\sigma_w}{\sqrt{n}} Z_w}{\mu_r + \frac{\sigma_r}{\sqrt{n}} Z_r}$$

So

$$\hat{W} = \frac{\mu_w}{\mu_r} \left(\frac{1 + \frac{\sigma_w}{\sqrt{n}\mu_w} Z_w}{1 + \frac{\sigma_r}{\sqrt{n}\mu_r} Z_r} \right)$$

Using $\frac{1}{1+a} = 1 - a + a^2 - a^3 + \ldots$ we get

$$\hat{W} = W \left(1 + \frac{\sigma_w}{\sqrt{n}\mu_w} Z_w \right) \left(1 - \frac{\sigma_r}{\sqrt{n}\mu_r} Z_r + O\left(\frac{1}{n}\right) \right)$$

So

$$\hat{W} = W + \frac{W}{\sqrt{n}} \left(\frac{\sigma_w}{\mu_w} Z_w - \frac{\sigma_r}{\sqrt{n}\mu_r} Z_r + O\left(\frac{1}{n}\right) \right)$$

$$\text{Let } Z_p = \left(\frac{\sigma_w}{\mu_w} Z_w - \frac{\sigma_r}{\mu_r} Z_r \right)$$

Then

$$\mathrm{var}(Z_p) = \frac{\sigma_w^2}{\mu_w^2} + \frac{\sigma_r^2}{\mu_r^2} - \frac{2\sigma_w\sigma_r}{\mu_w\mu_r}\rho$$

Then

$$P(|W - W| > \varepsilon) = P\left(\frac{\sqrt{n}(|W - W|)}{W\left(\sqrt{\frac{\sigma_w^2}{\mu_w^2} + \frac{\sigma_r^2}{\mu_r^2} - \frac{2\sigma_w\sigma_r\rho}{\mu_w\mu_r}}\right)} > \frac{\sqrt{n}\varepsilon}{W\left(\sqrt{\frac{\sigma_w^2}{\mu_w^2} + \frac{\sigma_r^2}{\mu_r^2} - \frac{2\sigma_w\sigma_r\rho}{\mu_w\mu_r}}\right)} \right) < \alpha$$

Then

$$\Phi\left(\frac{\sqrt{n}\varepsilon}{W\left(\sqrt{\frac{\sigma_w^2}{\mu_w^2} + \frac{\sigma_r^2}{\mu_r^2} - \frac{2\sigma_w\sigma_r\rho}{\mu_w\mu_r}}\right)} \right) \geq 1 - \frac{\alpha}{2}$$

Therefore,

$$n \geq \left(\frac{Z_{1-\alpha/2}}{\varepsilon} \left(W\left(\sqrt{\frac{\sigma_w^2}{\mu_w^2} + \frac{\sigma_r^2}{\mu_r^2} - \frac{2\sigma_w\sigma_r\rho}{\mu_w\mu_r}} \right) \right) \right)^2$$

6.4 Sender's Algorithm

In the last section we developed an equation to determine n, the sample size needed to archive a $\hat{W}$ with an error of ε and a confidence level of $1-\alpha$. One difficulty with this equation is that it requires knowledge about the set of receivers. Specifically, we need to

know the entire set R in order to derive our n. In order to over come this limitation we have developed an iterative algorithm. This algorithm is shown in Figure 34. The algorithm is iterative in the sense that we continually estimate n based on a subset of the receivers feedback rates and adjust our sample size according to this newly calculated value. At the same time the sender uses the receivers feedback to calculate its transmission rates.

In order to obtain a subset of the receivers' bottleneck rates our algorithm utilizes a technique developed in [4]. Their approach utilizes probabilistic probing in order to estimate the number of receivers in the videoconference. In addition, they provide an algorithm to obtain feedback from a randomly determined subset of receivers. We utilize these techniques in steps 1 and 2 in order to obtain the feedback necessary to calculate our new transmission rates.

In step 3 of our algorithm we calculate $\hat{n}$ utilizing the equation developed in the last section and the feedback obtained in step 2. For this calculation we are using $\varepsilon = .05$ and $\alpha = .05$. It should be noted that this is only an estimate of n since we are utilizing only a subset of the receivers in our calculation. While an averaging of the $\hat{n}$'s might more accurately represent the true n, in order to be conservative in step 4 we only adjust our sample size δ to larger values of $\hat{n}$. In Step 5 of our algorithm we utilize the feedback to determine the sender's transmission rates.

1) Estimate N, the total number of receivers in the videoconference.

2) Initiate the process to receive feedback from δ receivers.

3) Estimate $\hat{n}$ based on the receivers' feedback obtained in step 2.

4) If $\hat{n} > \delta$ then set $\delta = \hat{n}$.

5) Utilize feedback to determine the sender's transmission rates.

6) Goto step 2.

Figure 34: Sender's Iterative Algorithm to Determine Transmission Rates

Number of Sender's Multicast Groups	5
Number of Receivers	500
Iterations of Sender's Algorithm	60
α	.05
ε	.05
δ	120

Table 7: Parameters Used in Figure 35 and Figure 36

6.5 Performance Analysis

In order to determine the effectiveness of the algorithm given in Figure 34, we have analyzed its performance with hundreds of distributions. The receiver bottleneck rate distributions varied from a small number of clusters, to very clustered, to uniformly distributed. To simplify our discussion we will limit our analysis to the distributions shown in Figure 33 and summarized in Table 6. The parameters used in this analysis are shown in Table 7.

There are really two steps key steps in our algorithm. First is the calculation of $\hat{n}$ in step 3. Due to the correlation between w_i and r_i we have found that a fairly large δ, our minimum sample size, is necessary in order to achieve a realistic $\hat{n}$. For our analysis we

have initially set $\delta = 120$[1]. Figure 35 shows histograms for 60 calculations of $\hat{n}$ using a sample size of 120 for the 6 distributions in Figure 33. The number shown (e.g. n=?) on each histogram is the true value of n calculated using all 500 receivers. As we can see by these histograms, the estimated value of n tends to be normally distributed around the true n. The second key step in our algorithm is the calculation of the sender's transmission rates in step 5. Figure 36 shows the metric *percentage-used* for 60 iterations of our algorithm for the distributions in Figure 33. The histograms in Figure 36 show that for any of the 6 distributions the *percentage-used* varied at most .05 over the course of the 60 iterations.

[1] Utilizing a 10 second control period, this will limit the number of feedback packets arriving at sender to approximately 12 per second or less than 10 kbps using 100 byte feedback packets.

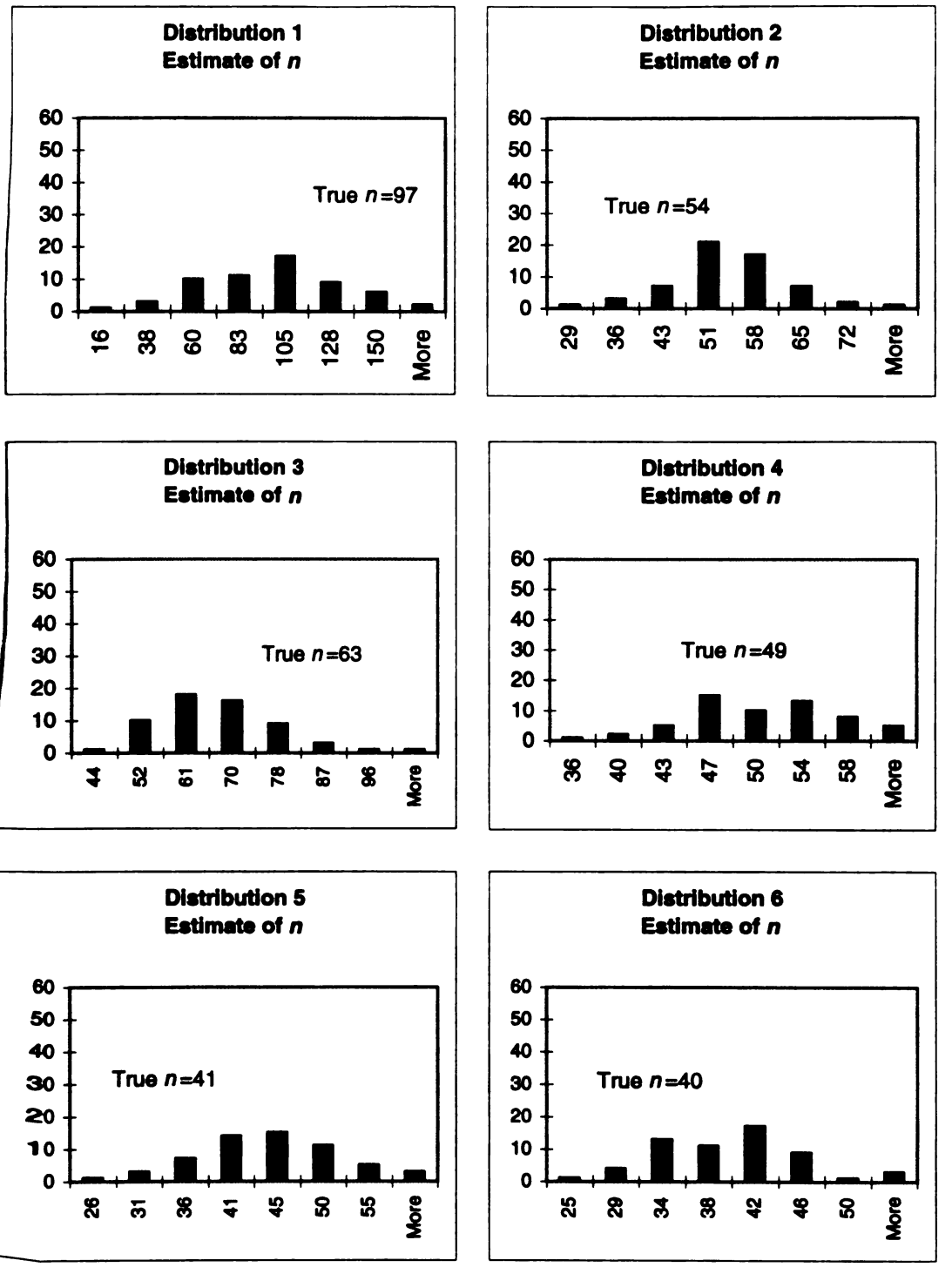


Figure 35: Histograms of the calculation of $\hat{n}$ for the six distributions in Figure 33.

This calculation was done with $\varepsilon = .05$ and $\alpha = .05$.

Figure 37 shows the variations in the metric *percentage-used* for 50 separate distributions. These distributions ranged between slightly clustered to uniformly distributed. The short lines in this graph represents the difference between the optimal *percentage-used* achieved using all receivers' feedback and the minimum *percentage-used* achieved using δ=120 for 100 iterations of our algorithm. As you can see from this graph, our algorithm achieved nearly optimal performance for all distributions with a maximum variation for any of the 50 distributions of .051

6.6 Summary

In this chapter we address the issue of the scalability of the receivers' feedback. Due to the wide variations of receivers' bandwidth distributions we developed an iterative algorithm. This algorithm first estimates a sample size n such that $P(|P-\hat{P}|>\varepsilon)\leq\alpha$, where P is the true *percentage-used* and $\hat{P}$ is calculated based on our sample. The algorithm then samples the receivers' feedback rates and determines the sender's transmission rates. Our analysis showed that this algorithm achieved a $\hat{P}$, which varied at most .051 for the distributions presented.

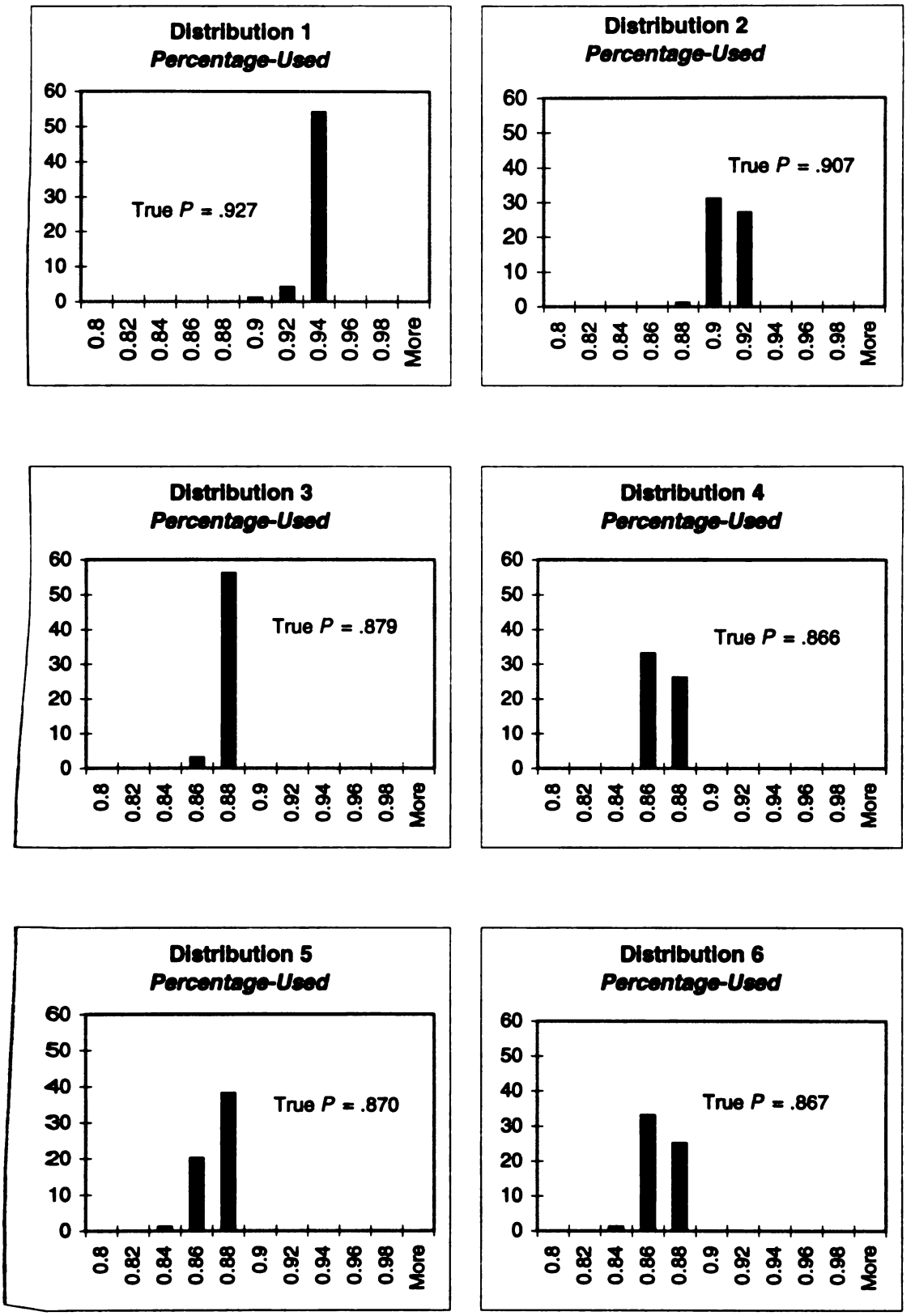


Figure 36: Histograms of *percentage-used* for 60 iterations of our algorithm for the six distributions in Figure 33.

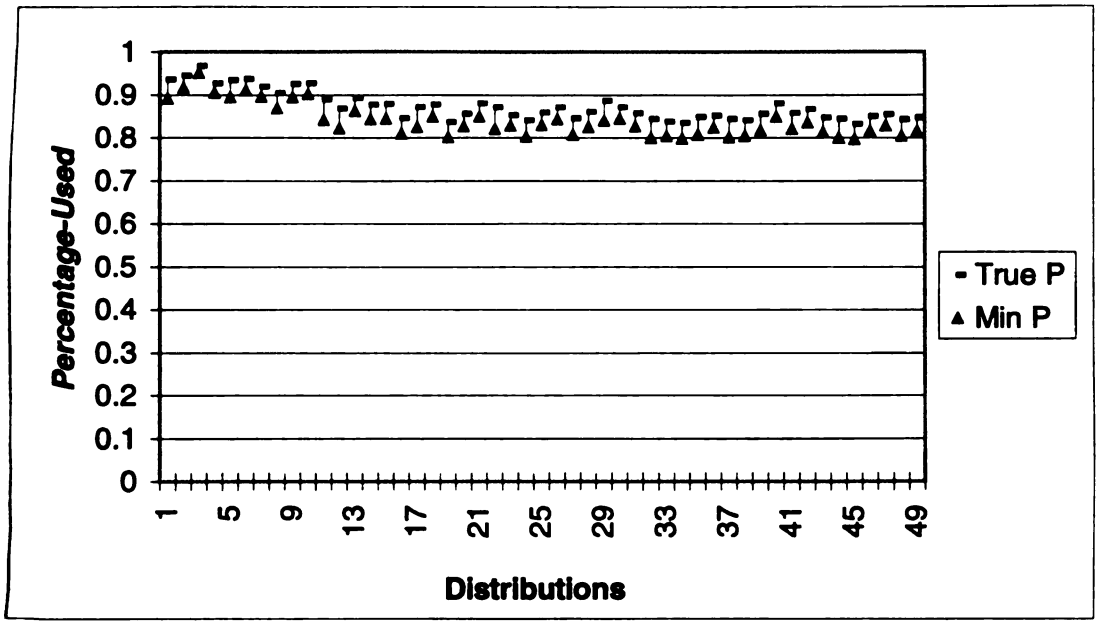


Figure 37: True *percentage-used* versus minimum *percentage-used* for 60 iterations of our algorithm for 50 different distributions of 500 receivers each

Chapter 7

7 Fair Link Sharing with Layered Multicast Video

7.1 Introduction

In chapter 5 we introduced the Layered Multicast Control Protocol (LMCP). As we will discuss in this chapter there are some limitations to this approach. Specifically, the basic LMCP does not fairly share the network between multiple video conferencing sessions. In this chapter we present a modification to the LMCP's control protocol, which utilizes information from the routers to determine the feedback rate for each receiver. This modified control protocol takes into account four competing priorities. These are:

1. Receiver Video Stability: Any type of video control process must maintain a stable rate of reception at the receivers. Adjustments that allow for changes in available resources need to be achieved gracefully. Large and frequent

fluctuations at the receivers will be perceived worse than a smooth lower average rate.

2. Resource Sharing: A video control process needs to adjust gracefully to the addition (and subtraction) of new video sessions and should allow these new sessions an adequate share of the available bandwidth.
3. Maximize the Displayable Video: While the previous goals have to do with the interaction between video applications, the control process needs to pick transmission rates that maximize the video displayed at its receivers based on the entire range of resource requirements of receiver set.
4. Maximizing Network Utilization: In order to deliver the highest quality video possible over the available bandwidth, it is important to maximize the utilization of the network links.

The modified LMCP presented this chapter is an end-to-end approach for controlling a multicast video conferencing application. This approach builds on the positives of the LMCP by allowing the sender to stripe the video across multiple multicast groups and modifying its transmission rates to maximize the displayable video at the receivers. In addition, we have made modifications to both the sending and receiving components of the algorithm in order to allow the control algorithm to share the available resources fairly while maintaining stability at the receivers.

The remainder of the chapter is organized as follows. In section 2 we discuss some of the limitations of both the RLM and LMCP approaches. Following this we introduce our

new router-based scheme that improves the stability and sharing of the LMCP approach. In section 4 we discuss the actual router implementation. We then provide simulation results of this new approach.

7.2 Limitations of the RLM and LMCP approaches

There are two major disadvantages to the RLM approach. First, as discussed in chapter 5, its control process relies solely on the receivers and therefore does not adjust its transmission rate to maximize the video displayed at the receivers. Second, due to its fixed transmission rates, high receiver rates require a large number of multicast groups. The LMCP approach presented in chapter 5 removes these limitations. One limitation that is common to both approaches is their inability to share the network fairly between multiple video sessions. This limitation comes from the underlying RLM implementation and is identified in [31].

The basis of this problem is the fact that the RLM approach is focused on finding stable receiver rates based on the interaction of receivers in a particular video session and not between video sessions. Therefore, receivers who have maintained a stable reception rate over time hold their rate even in the face of short-term congestion. Conversely, receivers experiencing packet loss during a join-experiment are forced to immediately drop the newly joined multicast group. These two factors combine to cause the RLM approach to favor existing video sessions over newer ones.

In order to overcome this limitation we researched the effect of utilizing RED based routers in conjunction with our LMCP approach. One benefit of RED based routers is that all receivers downstream from a congested router will experience the same percentage of packet loss regardless of their reception rate. We utilized this information in determining which receivers should drop their highest multicast connections and in how the receivers' feedback rate is determined.

We exploited the additional information obtained for utilizing RED based routers in two ways. First, we modified the algorithm for determining when a receiver should drop its highest multicast group during congestion. Instead of prioritizing existing connections over new ones, we forced all receivers to drop based on a calculated threshold. This threshold was set based on a function of time and the receiver's current reception rate. Since all receivers experience the same percentage of loss, this approach caused the higher receivers to drop more quickly.

Second, we modified the process for estimating the receivers' feedback rate. The new feedback calculation utilized the fact that the only time a receiver would drop a multicast group is when it was one of the high-end receivers. Therefore, the actual rate the receiver was receiving prior to leaving the multicast group could be considered as the receivers' maximum rate and was used as the receivers' new feedback. By utilizing adds and drops we allowed our receivers to *probe* the network for their fair share of the link.

Our simulation results identified a number of flaws with this type of approach. The simulations discussed here consisted of set of sources (between 4 and 6) transmitting to 4 receivers each. All the video sessions shared a 1.3 Mbps link. Our findings from these simulations may be summarized as follows:

1. While we were able to force higher-end receivers to drop their highest multicast group, due to a long delay between the dropping of a multicast group and the clearing of the network congestion, there could be a number of lower receivers dropping their links needlessly. The longer the leave latency the more receivers reached their drop threshold. These extra drops had two negative effects. First they caused the receivers to experience a large number of fluctuations in reception rates. Second, they caused these receivers to underestimate their feedback rates.
2. The method of *probing* using adds and drops to determine the receiver's fair share of the link caused large oscillations in the receiver's displayable video. (See Figure 38)
3. The addition of a new video session exasperated the problems caused by this *probing*, resulting in additional fluctuations in the receivers displayable video.

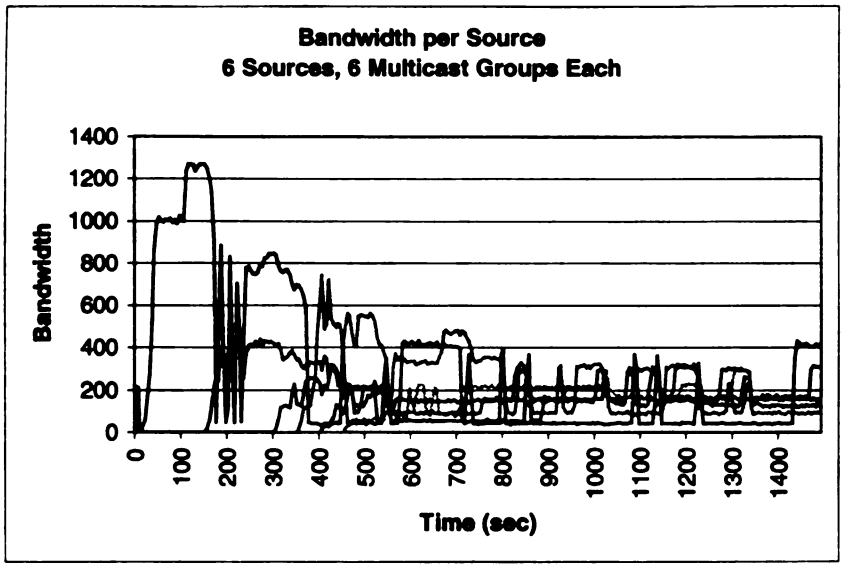


Figure 38: Six Video Sessions Sharing a 1.3Mbps Link

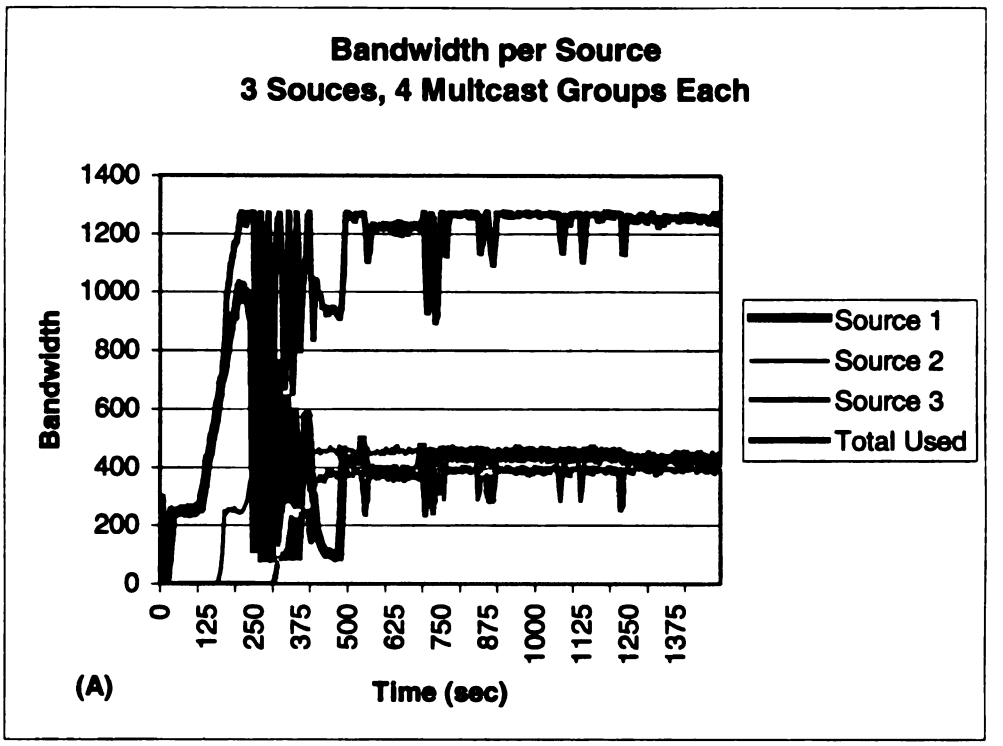
4. The number of multicast groups a receiver belongs to is independent of their reception rate. Therefore, while a high-end receiver may drop its highest multicast group, its new reception rate may still be too high. This allows a higher-end receiver to maintain a larger than fair share of the network link. As shown in Figure 39-(A,B), source 3 controls a larger share of the link when we move from four multicast groups (Figure 39-A) to six (Figure 39-B). In Figure 39-B, source 3 is receiving 4 multicast groups, while sources 1 and 2 are only receiving on 3. Therefore, when congestion occurs and the sources drop their highest group, source 3 still maintains more than its fair share of the link.

7.3 LMCP with Router Support

As discussed in the last section, the basic Layered Multicast Control Protocol does not allow multiple video conferencing sessions to share the network links fairly. In addition,

modifications to allow for both fair link sharing and maximizing network utilization cause large fluctuations in the amount of video received at the receivers. In this section we introduce a router-based solution that will allow all the links to be shared fairly while maintaining stability at the receivers.

There are three components that comprise this solution. As in the original LMCP approach, both the sender and receiver will be involved in determining the appropriate transmission rate (see Figure 40). In addition, this approach requires support from the network routers. The basic idea is for the sender to multicast periodically a special control packet to all its receivers. The amount of time between control packets is a system wide constant and is called the control period. The control packet contains one field, called the minimum fair-share rate. This field is initially set to a value of -1.



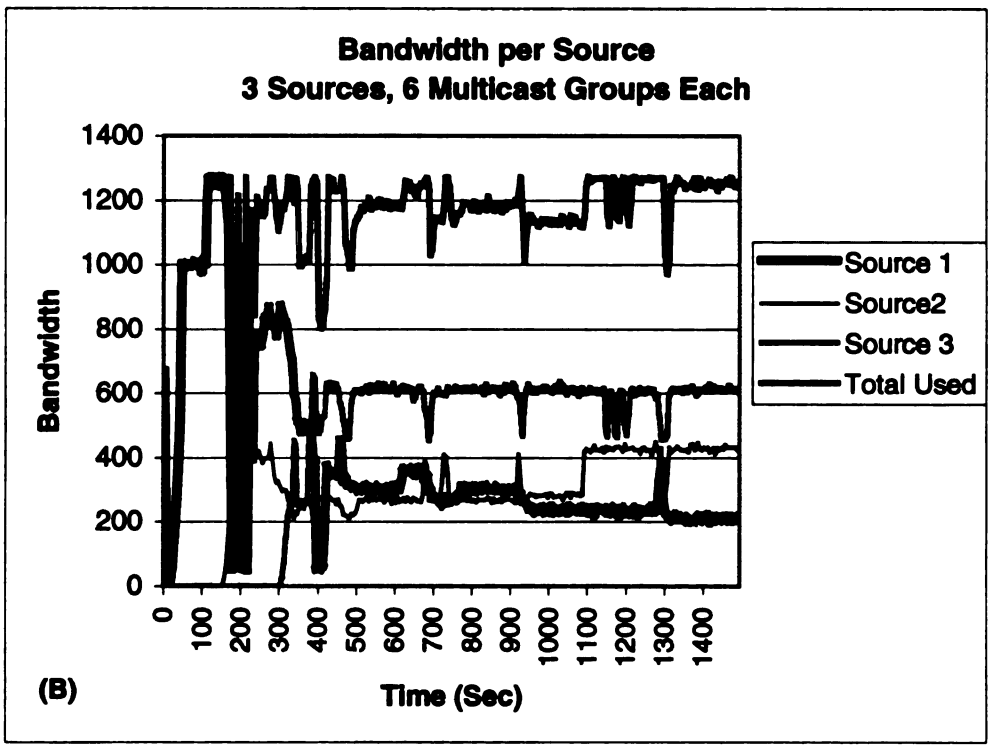


Figure 39: Three Sources Sharing a 1.3Mbps link

The purpose of this control packet is to inform the receivers of the minimum fair share rate between themselves and the source. To do this the routers must utilize this packet in two ways. First, the router maintains a count of the number of control packets that have passed through the router during a control period. This count is maintained for each outgoing link. At the end of a control period the router calculates for each outgoing link its fair share rate, where this rate is defined as the link rate divided by the control packet count. Second, the router compares the fair share rate calculated in the last control period to the fair share rate contained in the control packet. If the link's fair share rate is less than the value contained in the control packet, the control packet rate is replaced with the link's fair share rate. The control packet is then forwarded down the link. Upon arriving at the receiver, this control packet contains the minimum fair share rate on the path between the receiver and the source. We call this rate the receiver's *fair-share bandwidth*.

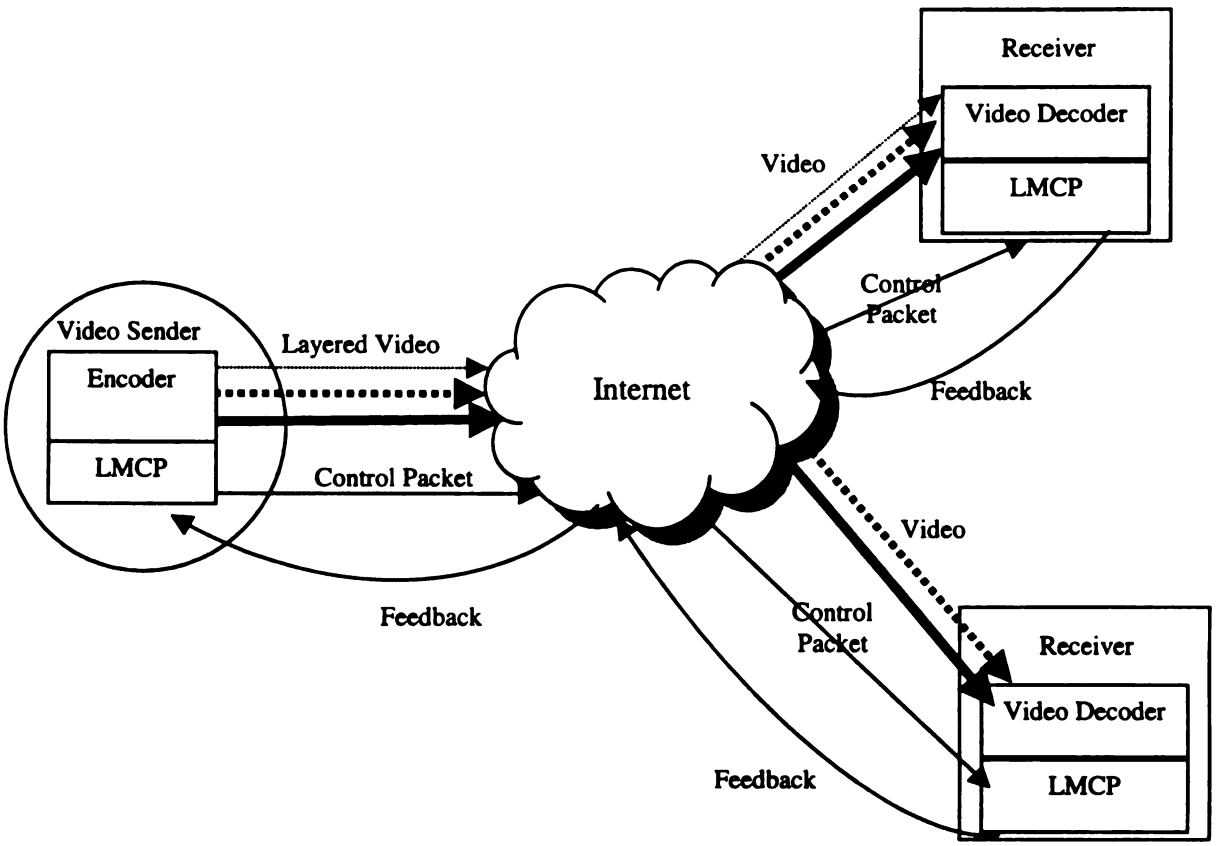


Figure 40: Modified Layered Multicast Control Protocol

In order to support this new feedback approach, significant changes are necessary to the receiver algorithm in the basic LMCP protocol. There are two disadvantages to the current approach. First, this approach gives priority to existing sessions over newer video sessions. Second, its use of packet loss to probe for link availability does not scale well to high-speed links, which are supporting a large number of video sessions. Our modified receiver's algorithm utilizes the receiver's fair-share bandwidth in determining how many of the source's multicast groups to join. The basis of this approach is that a receiver will only receive at a rate that is at or below their fair-share bandwidth.

Therefore, a receiver will join a multicast group only if the new reception rate will be below its fair-share rate. In addition, a receiver will leave a multicast group if the current reception rate is greater than their current fair-share rate.

7.4 Router Implementation

In order for the router implementation to be feasible it must be simple to implement with low overhead. This is accomplished in two ways. First, the process for calculating a link's fair-share bandwidth rate is separated from the processing of the each control packet. This is done by allowing the router to calculate each link's rate once per control period, and utilizing the calculated rate during the succeeding period. This streamlines the processing of the control packets and reduces the overhead on the router. Second, the actual processing of the control packet consists of incrementing the link's control count, comparing the link's previously calculated fair-share rate to the one contained in the packet and replacing the rate in the packet if necessary. A key advantage to this approach is that no per-source state information is needed on the routers and therefore, this approach scales well as the number of sources increases. In addition, it is completely independent of the number of receivers participating in the video session.

One difficulty with this approach is that the fair-share rate is dependant on an accurate count of the number of sources sharing a link. Due to packet jitter it is possible for control packets to be delayed across control periods. Therefore, a control packet from one source may be not counted in one control period and counted twice in the next control period. These fluctuations in count can have an impact on the fair-share

bandwidth rate in two ways. First, when a control packet is pushed into the succeeding control period the number of sources will be under-counted. This will cause the fair-share rate to increase erroneously. As the receivers adjust their transmissions rates based on this rate, the link may become congested. This is the more serious of the two problems. Second, on an over-count, the link's fair-share rate will decrease. As the senders adjust to this erroneous decrease the receivers will notice a change in the quality of the video delivered.

We have implemented a two-part solution to this problem. First, we delay any change, either increase or decrease, in a link's count until it has persisted for two control periods. If the change is due to packet jitter, there will most likely be an under-count in one period and an over-count in the following period. In appendix C we show that the probability that we under-count by one or more due to jitter is 0.14 for 100 sources. The probability that we either over-count or count correctly in the next period is 0.99. Therefore this approach filters out the majority of the fluctuations due to jitter.

Second, we utilize the fact that the probability of having more than a small number of packets affected by jitter at any one time is very small. We show in appendix C that the probability that that we under-count by more than 1 is less than 0.01 and that 98% of the time our count will fall between ± 1 for 100 sources. Therefore we are able to filter out small jitters (± 1) by not modifying the count until an increase or decrease of more than 1 is encountered. In order to minimize the effects of this smoothing for legitimate changes, the router uses count + 1 in its rate calculation. This decreases the chance of loss at the

receivers in the event of a legitimate increase in count due to the addition of a new source. It should be noted that this second solution is only necessary as the number of sources increases. For smaller numbers of sources the first solution is all that is necessary.

7.5 LMCP Simulation

In order to analyze the interaction of the router based feedback with the sender's control algorithm; we have implemented them in the *ns* [29] simulation package. In this section we give an overview of some of the key details of the *ns* implementation, discuss the changes required to support our protocol, and then present the performance results.

7.5.1 NS Implementation

Our ns simulation was built upon our LMCP [46] implementation and the RLM implementation found in the *ns* package. In addition to the changes necessary to the network routers in order to support the sender's control packets, changes were necessary in both the receiver and sender's RLM code. Two changes were necessary to compensate for the delay in recognizing new session imposed by the router based solution. First, the routers were modified to utilize only 90% of their capacity in their bottleneck rate calculations. The remaining capacity was utilize during the start up of new sessions and minimized the effect of these new sessions on existing sessions. Second, receivers were initially started up using only the lowest multicast group. Only after 5 control periods are receiver's able to add additional multicast groups. This delay gives the existing video sessions time to adjust their transmission rates based on the network feedback. The final

change to the receiver's code concerned processing the sender's control packet. The receiver's code was to modify to receive the fair-share bandwidth rate in sender's control packet. This rate, along with a receiver ID, was then periodically transmitted as feedback to the sender via a control channel.

The implementation of the sender's side of the LMCP algorithm required two modifications in order to allow it to work in a simulation environment. First the sender was modified to transmit a control packet every control period. Second, in order to provide service to all receivers, the sender's lowest multicast channel was locked into transmitting at *min_rate*[1]. This is necessary in order to prevent the control algorithm from picking a transmission rate that would prevent some low end users from receiving any of the video channels. While this channel was included in the optimal calculation of transmission rates, it was not allowed to vary.

7.5.2 Simulation Results

Figure 41 shows the network utilized in our first simulation. This figure shows a network with three video sources transmitting over a shared 1.3 Mbps link. On the receiving side there are 6 receivers per source that are connected to the 1.3 Mbps by non-shared 2 Mbps links. In this set of simulations we are interested in three things. First, we examine how fairly the link is being shared among the three sources. Second we look at the stability of the feedback being generated by the routers and third, we look at how the receivers are

---

[1] For our purposes we set *min_rate* to 32Kb/s. In any specific video application, the setting of this rate needs to take into account the video coder's limitations and the type of receivers the sender wishes to support.

impacted by this approach. Figure 42-(A,B,C,D) shows the results of this simulation. For simplicity, Figure 42-(B,C,D) show results for only Source 1.

Figure 42-A, shows how the three sources share the 1.3Mbps link. The start-up of the three sources is staggered, with the sources starting at 40, 130 and 330 seconds into the simulation respectively. As can be seen in Figure 42-A, the addition of the new sources causes the shared rate to drop. Figure 42-D shows the feedback rate as sent back by the six receivers to Source 1. This feedback rate is stable and only fluctuates when a new source begins to share the link. The effects on the receivers of the additional sources and adjustments in transmission rates can be seen in Figure 42-C. This figure shows that even as the new sources are added, the receivers see a graceful adjustment in their received rate.

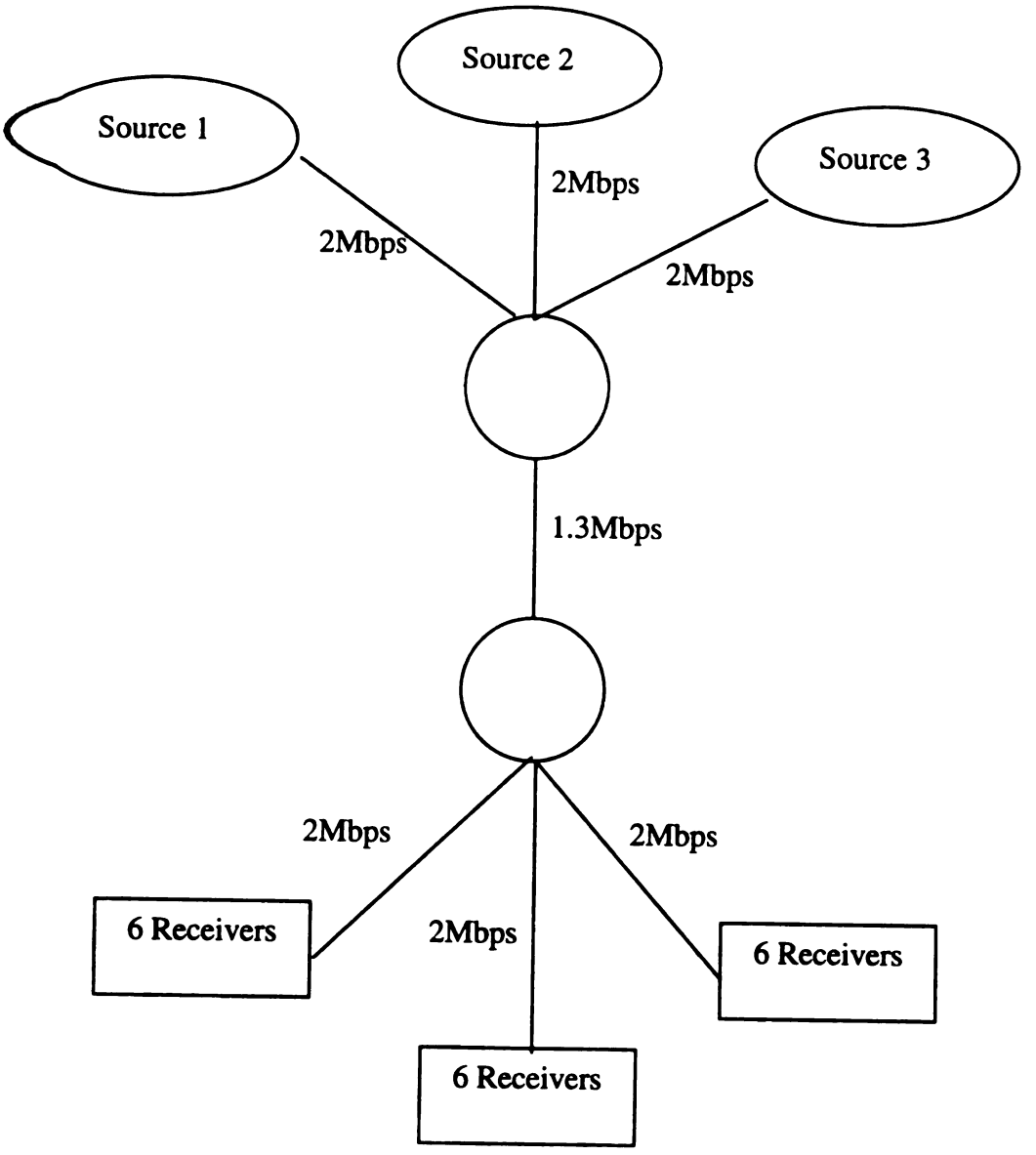


Figure 41: Network for First Simulation

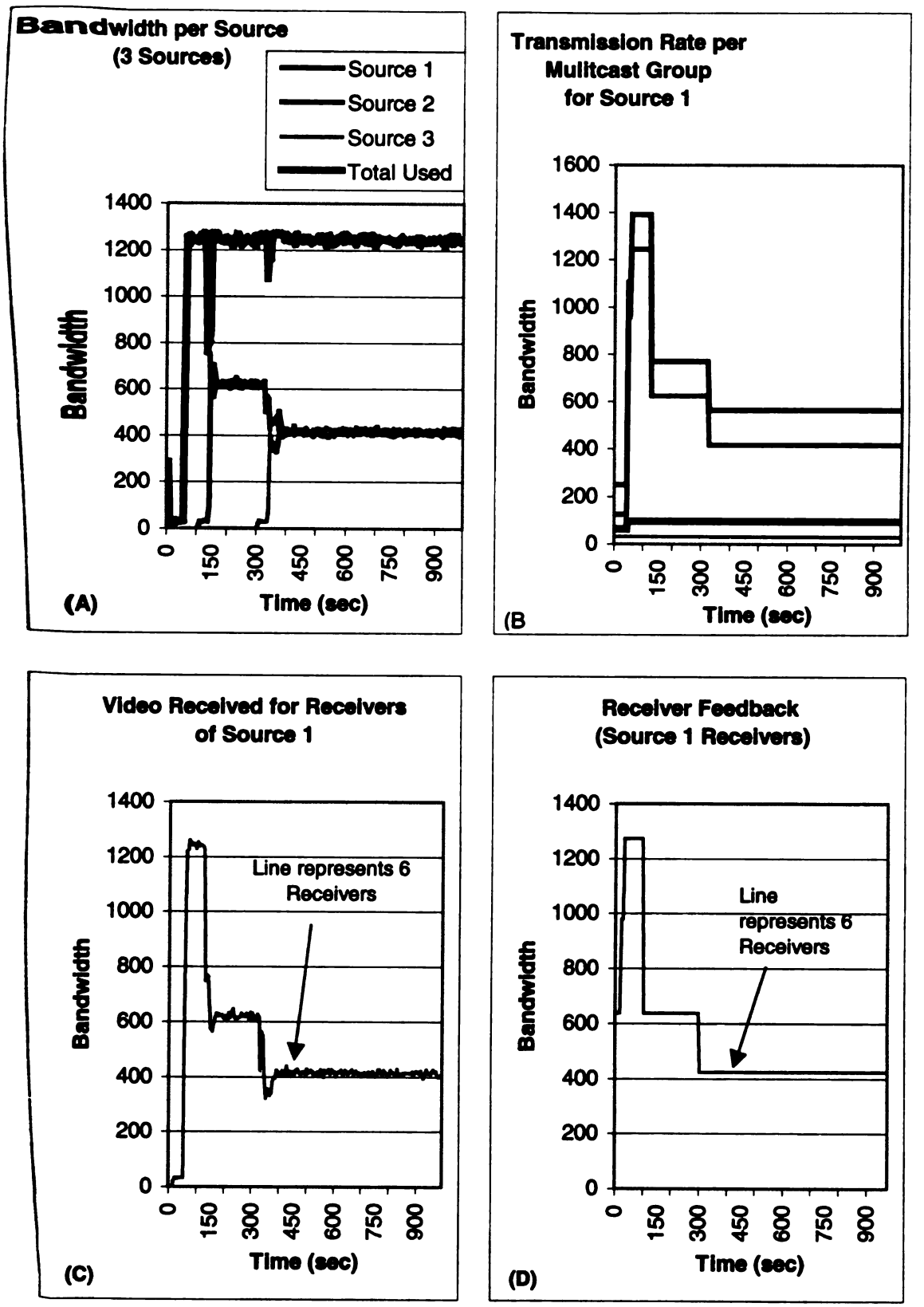


Figure 42-(A,B,C,D) Results for Simulation 1

Figure 43 illustrates the network used in our second simulation. This network consisted of 30 video sessions. Each video session is made up of one sender and six receivers. This simulation was run for 20 simulated minutes. In this simulation, we were interested in looking at the impact of fluctuating the number of video sessions over an extended

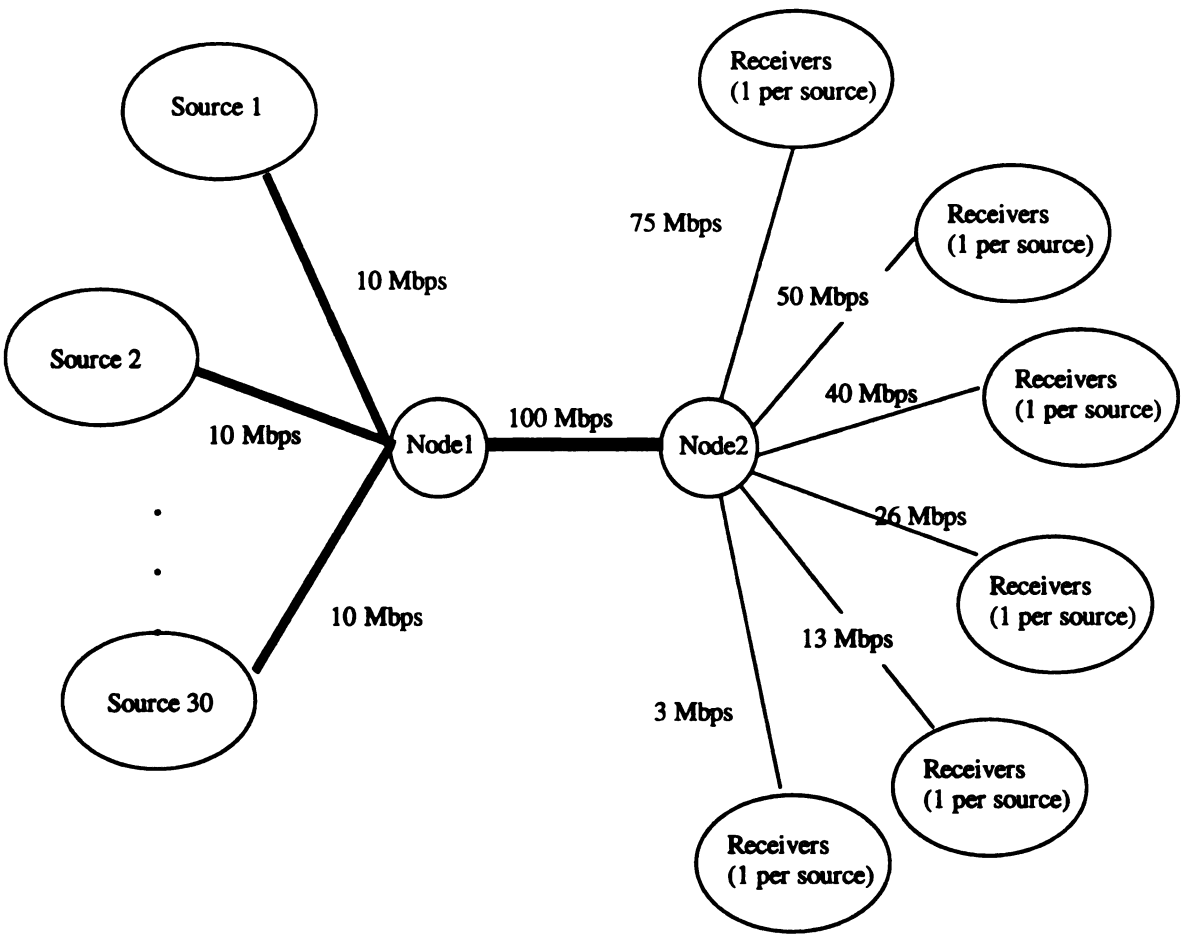


Figure 43: Network for Second Simulation

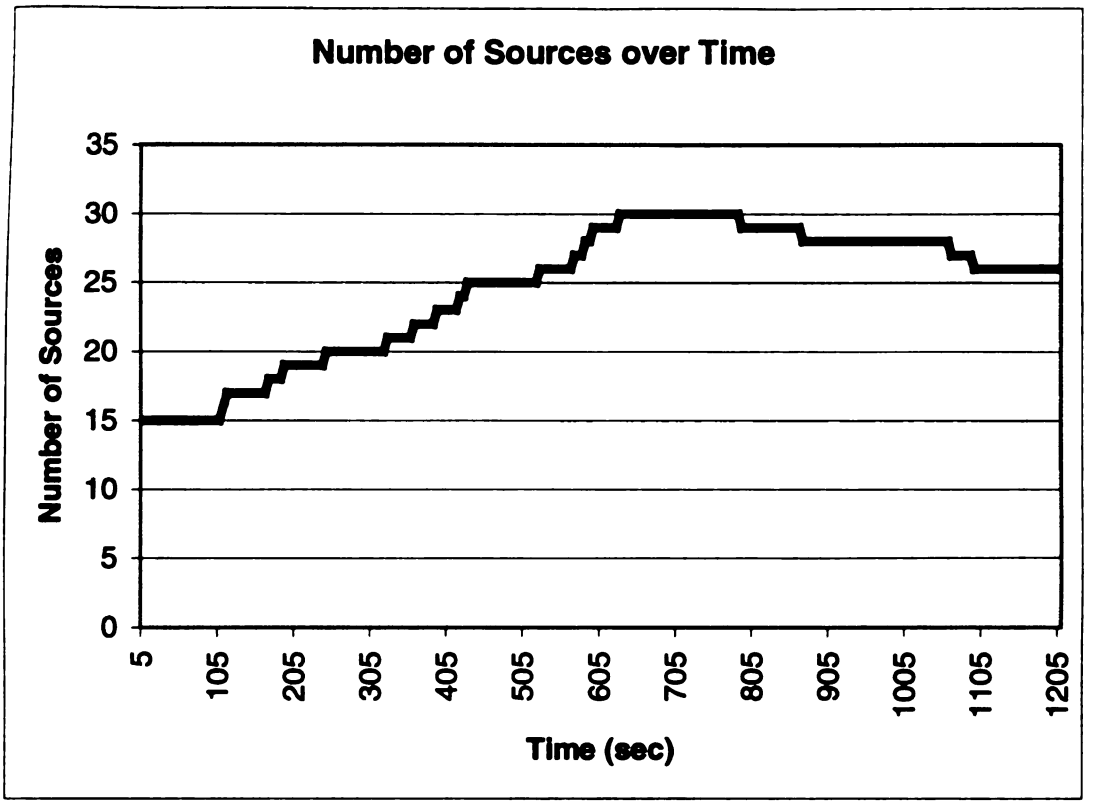


Figure 44: Number of Video Sessions over Time

period of time. Figure 44 shows how the number of active video sessions varied over time. As can be seen in this figure, fifteen of the sessions ran for the entire simulation. The remaining fifteen sessions started at random times and four of these sessions terminated prior to the completion of the simulation.

Figure 45-(A,B,C,D) contains the results of this simulation. For simplicity Figure 45-(B,C,D) contain results for only one video session. Figure 45-A shows the bandwidth utilized by each of the 30 sessions over the 100 Mbps link between node 1 and node 2. This figure shows that the bandwidth is disturbed evenly between the video sessions. Figure 45-(B,C,D) shows the results for one of the video sessions. Figure 45-D graphs

the feedback rate generated by the six receivers in this video session. Figure 45-B shows how this feedback impacts the sources transmission rates. Figure 45-C shows the bandwidth received at the six receivers for this video session. As these figures show, our router-based technique allows the video session to fairly share the available bandwidth while gracefully adjusting the senders' transmission rates to maximize the video displayed at the receivers.

7.6 Summary

In this chapter we introduced an end-to-end approach for controlling multicast video conferencing applications. This approach built upon our previous work on the Layered Multicast Control Protocol. This new approach takes into account four competing priorities. These are 1) Receiver video stability, 2) Network resource sharing, 3) Maximizing the displayable video at the receivers, and 4) Maximizing network utilization.

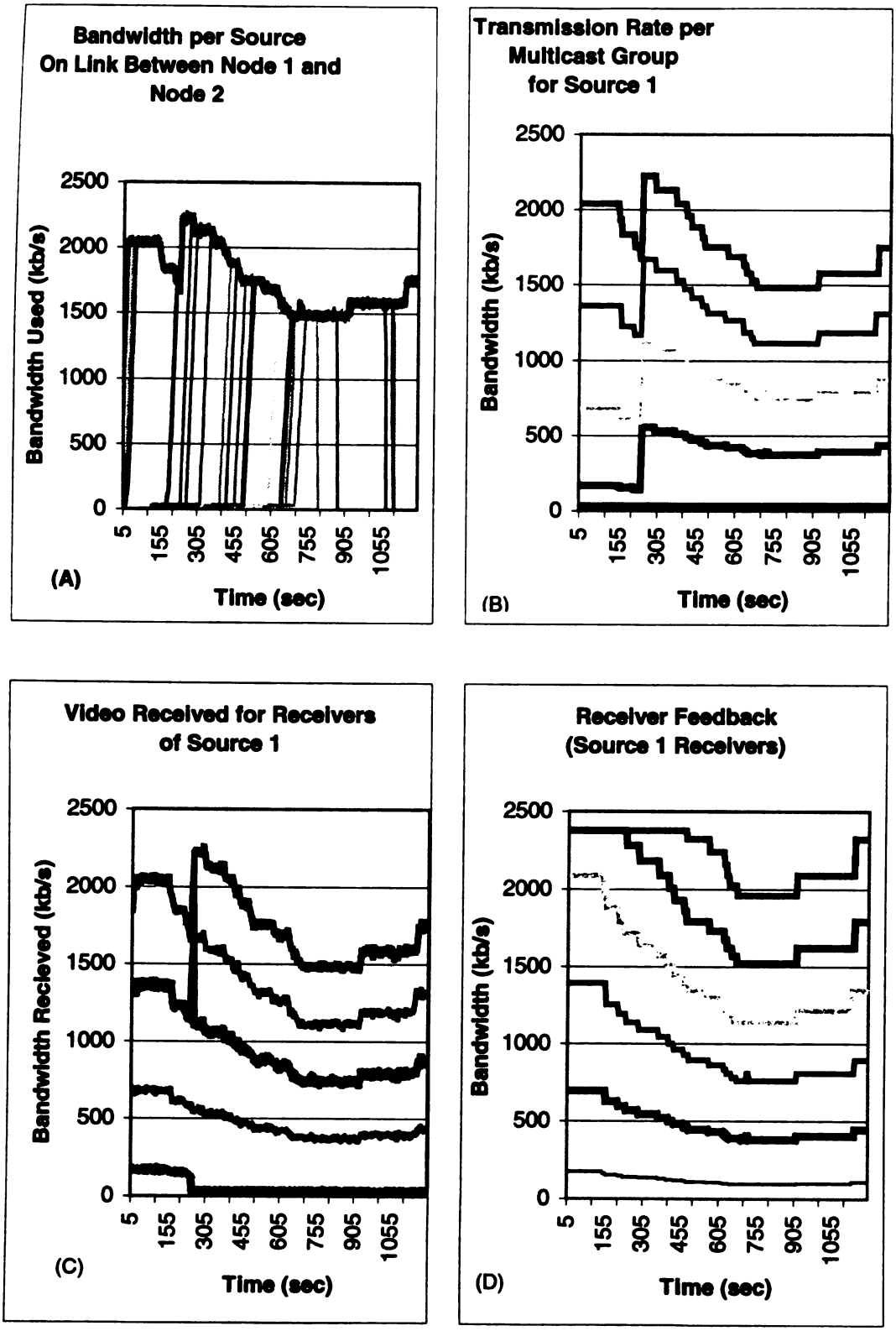


Figure 45-(A,B,C,D): Results for Simulation Two

This new approach has three components. As in the basic LMCP approach, both the sender and receiver are actively involved in determining the appropriate transmission rates. In addition, support in the routers allows the receivers to determine their fair share of the available network bandwidth. As discussed, this stateless approach requires minimal processing in the routers and scales well to large numbers of video sessions. The results from our simulations show that the LMCP approach combined with router support allows competing video sessions to adjust their transmission rates to fairly share the available network resources. In addition, we showed that the transmission rate adjusts gracefully to the addition and subtraction of video sessions.

Chapter 8

8 Conclusion and Directions for Future Research

This chapter summarizes the major contributions of this dissertation and describes our plans for continuing our research on multicast video conferencing.

8.1 Contributions

In this thesis we investigated whether the Internet can support the transmission of multicast videoconferencing. Initially, we studied transmission techniques for variable bit rate encoded video. We showed that our *Pattern Smoothing* algorithm significantly improves network utilization and reduces the effects of loss due to statistical multiplexing in the network.

One limitation of the *Pattern Smoothing* algorithm is that it assumes homogenous network capacity. The *Target Bandwidth* algorithm was developed to work in a heterogeneous environment. This algorithm controls the transmission rate of a single video stream for a single multicast group. Utilizing feedback from the receivers and a

unique maximization function this algorithm maximizes the revivers' displayable video while limiting the worst-case loss at the low-end receivers.

The *Layered Multicast Control Protocol* was developed to control the transmission rates of video being striped across multiple multicast groups. This algorithm utilizes both the sender and receivers in the control process in order to maximize the video displayable at the receivers. In addition to studying an $O(n^3)$ optimal algorithm, we developed a non-optimal divide and conquer algorithm. While the complexity of this approach is $O(\binom{2k}{k} * kn)$ which is exponential in the number of multicast groups, for reasonable k ($k < 7$) and $n >> k$, this approach has a significant runtime performance advantage over the optimal algorithm. We showed through our new performance metric, *percentage-used*, that this approach significantly outperforms the basic RLM approach.

In addition to addressing the scalability of the sender's processing through the use of the non-optimal divide and conquer algorithm, we addressed the scalability of the receivers' feedback. We developed an algorithm to utilizing sampling in order to limit the amount of feedback utilized by the sender in determining its transmission rates. We showed that for many distribution our performance metric, *percentage-used* is asymptotically normal and therefore many be estimated using a sufficiently large sample size. We showed that for many distributions a sample size of 120 gives excellent performance. We then presented a dynamic algorithm for determining a sample size n. This dynamic algorithm allows the sender's control algorithm to pick transmission rates which achieve a *percentage-used* to within 5% of its actual value with a confidence interval of 95%.

Finally, we developed a modified version of the LMCP. This new protocol utilized a unique router-based process to determine the receivers' fair-share of the network bandwidth. The sender's component of the LMCP was modified to take into account four competing priorities. These are 1) Receiver video stability, 2) Network resource sharing, 3) Maximizing the displayable video at the receivers, and 4) Maximizing network utilization. Our simulation results show that this approach allows the network to be shared fairly while adjusting gracefully to fluctuations in the link utilization.

8.2 Future Work

As discussed, there are four competing priorities that must be addressed when developing a multicast videoconferencing control algorithm. In our modified LMCP, we prioritized resource sharing and receiver stability over maximizing network utilizing. In contrast the basic LMCP approach prioritized link utilization over sharing and receiver stability. In the future we will look at approaches, which provide a compromise between these two extremes. By reducing the requirement that the link be shared completely fairly we hope to be able to achieve a higher network utilization on non-bottleneck links.

APPENDIX

Appendix

Appendix A: The Optimal Dynamic Programming Algorithm

The purpose of this algorithm is to find a set of transmission rates, T, which maximizes the bandwidth received at the *n* receivers. This is built upon the fact that finding the maximum rate received with i channels with a maximum transmission rate of t_i can be found by finding the set of maximum rates received for i-1 channels for all transmission rates $t_{i-1} < t_i$.

So let $T(i, t_i)$ be the maximum bandwidth received at the *n* receivers using i channels with the highest channel set to t_i. This can be calculated as follows::

$$T(i, t_i) = \begin{cases} Max\ T(i-1, t_{i-1}) + diff(t_i, t_{i-1}) & \text{if } i > 1 \\ Value(t_i) & \text{if } i = 1 \end{cases}$$

Where:

```
Diff(t_i,t_i-1) {
        //This function calculates the additional benefit of adding the ith channel

        diff = 0;
        for (j = 0; j < n; j++)
                if (r_j > t_i)
                    diff = diff + (t_i - t_i-1)
        return (diff);
}
```

```
Value(t_i) {
        // This function calculates the bandwidth received using only one channel

        value = 0;
        for (j = 0; j < n; j++) {
                if (r_j > t_i) value = value + t_i;
        }

        return (value);
}
```

Appendix B: The divide and Conquer Algorithm

This non-optimal algorithm works by dividing up the set of possible transmission rates into buckets and then picking the buckets that produce the highest rate received. This set of buckets is then divided and the buckets, which produce the highest rate, are picked again. This process continues until only *k* buckets of size 1 remain. For the purpose of our discussion the set of possible transmission rates is the set R.

- **Initialization:**
    1. Using *bucket_size* = (max(R) – min(R)) / (2**k*),
       divide the rates into 2*k* buckets
        1. min(R) through min(R) + *bucket_size*
        2. min(R) + *bucket_size* + 1 through min(R) + *bucket_size* * 2

           ...

        2k) min(R) + *bucket_size* * 2 *(k-1) + 1 through max(R)
    2. Calculate the number of rates in set R which falls into each bucket
    3. If any bucket is empty split the largest remaining bucket into two buckets
    4. Continue step 4 until all buckets have some rates or all buckets are of size 1

5. Let *num-buckets* be the number of buckets with 1 or more rates
6. Let the *bucket-rate* of each of the buckets be the minimum rate in the bucket

- **Calculate**:

7. Using the *bucket-rate* of each bucket evaluate all $\binom{num-buckets}{k}$ possible combinations and find the set of buckets which produces the maximum bandwidth received.

- **Divide:**

8. Selecting the k buckets which produced the maximum bandwidth received in step 7, divide the number of rates in each of these buckets into two buckets giving us a total of $2k$ buckets (note: all other buckets and therefore the rates in them are dropped)
9. If any bucket is empty split the largest bucket into two buckets
10. Continue step 9 until all buckets have 1 or more rates or all buckets are of size 1
11. Let *num-buckets* be the number of buckets with 1 or more rates
12. Let the *bucket-rate* of each of the buckets be the minimum rate in the bucket
13. If the number of buckets is greater than k go to step 7.
14. Let the set T be the *bucket-rates* of the k buckets

Appendix C: Probability that a control packet is delayed

In this appendix we derive the probability that a control packet is delayed into the next control period due to network jitter.

- Let X = the time in the control period a packet is due to arrive (without jitter). Note that X is uniformly distributed over (0, *Control Period*).
- Let J = the amount of jitter experienced by a packet. Jitter is measured as the delay experience between the scheduled arrival time and the actual arrival time of the packet. Note that J is exponentially distributed with parameter λ
- Let CP = the length of time for one control period (10 seconds for our simulations)

Then

$$P(X+J > CP) = \int_0^{CP} \frac{1}{cp} \int_{CP-x}^{\infty} \lambda e^{-\lambda y} dy dx = \frac{1-e^{-1-\lambda}}{CP*\lambda} \cong \frac{1}{CP*\lambda}$$

We estimated λ experimentally utilizing an automated ping process on seven US based Internet sites. This process was run approximately 4 times an hour for each site over a two-day period. Ten samples points were collected every run. We then used the methods of moments estimate:

$$\frac{1}{\lambda} = \frac{\sum_{i=11}^{N} Jitter_i}{N} = .014654, \text{ our estimate of } \lambda = 68.2374$$

Let the *control period (CP)* = 10 seconds:

$$P(X+J > 10) = \frac{1}{10 \times 68.23} = .0014$$

Using .0014 as the probability that one control packet is delayed we can determine the probability that more than one packet is delayed using the binomial distribution.

	Number Of Sources		
	N = 10	N =100	N = 500
P(Number Delayed > 0)	0.0139	0.1307	0.504
P(Number Delayed > 1)	0.0001	0.0089	0.156
P(Number Delayed > 2)	0.0000	0.0004	0.034
P(Number Delayed > 3)	0.0000	0.0000	0.006
P(Number Delayed > 4)	0.0000	0.0000	0.001
P(Number Delayed > 5)	0.0000	0.0000	0.000

Table 8: Probability of Delayed Control Packets

BIBLIOGRAPHY

Bibliography

[1] E. Amir, S. McCanne, and H. Zhang, *An Application-Level Video Gateway.* In Proceedings of ACM Multimedia, pp. 255-265, San Francisco, CA, November, 1995.

[2] A. Bakic, M. Mutka, and D. Rover, *BRISK: A Portable and Flexible Distributed Instrumentation System.* In Proceedings of International Parallel Processing Symposium/Symposium on Parallel and Distributed Processing, pp. 387-391, April, 1999.

[3] J. Bolot and T. Turletti, *A Rate Control Mechanism for Packet Video in the Internet.* In Proceedings of INFOCOM, pp. 1216-1223.

[4] J. Bolot, T. Turletti, and I. Wakeman, *Scaleable Feedback Control for Multicast Video Distribution in the Internet.* In Proceedings of SIGCOMM, pp. 58-67, University College London, London, U.K., September, 1994.

[5] I. Busse, B. Duffner, and H. Schulzrinne, *Dynamic QoS Control of Multimedia Applications Based on RTP.* In Proceedings of First International Workshop on High Speed Networks and Open Distributed Platforms, St. Petersburg, Russia, June, 1995.

[6] B. Cain, S. Derring, and A. Thyagarajan, *Internet Group Management Protocol, Version 3,*, November, 1997, IETF, pp. 1-26.

[7] S. Cheung, *et al.*, *On the Use of Destination Set Grouping to Improve Fairness in Multicast Video Distribution.* In Proceedings of INFOCOM, pp. 553-560, San Francisco, CA, March, 1996.

[8] C. Compton and D. Tennenhouse, *Collaborative Load Shedding for Media-Based Applications.* In Proceedings of International Conference on Multimedia Computing and Systems, Boston, MA, May, 1994.

[9] W. Conover, *Practical Nonparametric Statistics.* 1998: John Wiley & Sons, Inc.

[10] S. Derring, *Multicast Routing in Internetworks and Extended LANs.* In Proceedings of SIGCOMM, pp. 88-101, August, 1998.

[11] H. Erikson, *MBONE: The Multicast Backbone.* Communications of the ACM, , pp. 54-64.

[12] K. Fall, T. Pasquale, and S. McCanne, *Workstation Video Playback Performance with Competitive Process Loads.* In Proceedings of Fifth International Workshop on Network and OS support for Digital Audio and Video, pp. 179-182, Durham, NH, April, 1995.

[13] W. Feng and S. Sechrest, *Critical Bandwidth Allocation for Delivery of Compressed Video.* To appear in Computer Communications, .

[14] W. Feng and S. Sechrest, *Smoothing and Buffering for Delivery of Prerecored Compressed Video.* IS&T/SPIE Multimedia Computing and Networking, February, 1995.

[15] W. Fenner, *Internet Group Management Protocol, version 2,* , February, 1996, Internet Engineering Task Force, Internet Draft RFC2236.

[16] J. Fernandez and M. Mutka, *A Burst-Oriented Traffic Control Framework for ATM Networks.* In Proceedings of ICCCN, September, 1995.

[17] S. Floyd and V. Jacobson, *Random Early Detection Gateways for Congestion Avoidance.* In Proceedings of IEEE/ACM Transactions on Networking, pp. 397-413, August, 1993.

[18] S. Floyd, *et al., A Reliable Multicast Framework for Light-weight Sessions and Application Level Framing.* IEEE/ACM Transactions on Networking, December, 1997, 5(6), pp. 784-803.

[19] D. Gall, *MPEG: A Video Compression Standard for Multimedia Applications.* Communications of the ACM, April, 1992, 34(4), pp. 46-58.

[20] R. Gerber and L. Gharai, *Experiments with Digital Video Playback,* , 1995, University of Maryland Technical Report.

[21] S. Gupta and C. Williamson, *A Performance Study of Adaptive Video Coding Algorithms for High Speed Networks,* , University of Saskatchewan Technical Report, DR-94-4.

[22] D. Hoffman and M. Speer, *Hierarchical Video Distribution over Internet Style Networks.* In Proceedings of International Conference on Image Processing, September, 1996.

[23] H. Kanakia, P. Mirshra, and A. Reibman, *An Adaptive Congestion Control Scheme for Real-Time Packet Video Transport.* In Proceedings of SIGCOMM, pp. 20-31, San Francisco, CA, September, 1993.

[24] H. Kanakia and P. Mishra, *Packet Video Transport in ATM Networks with Single-bit Feedback.* ACM Multimedia Systems, , pp. 370-380.

[25] S. Lam, S. Chow, and D. Yau, *An Algorithm for Lossless Smoothing of MPEG Video.* In Proceedings of SIGCOMM, pp. 281-293, 1994.

[26] M. Lucas, B. Dempsey, and A. Weaver, *MESH: Distributed Error Recovery for Multimedia Streams in Wide-Area Multicast Networks.* In Proceedings of IEEE International Conference on Communications (ICC'97), June, 1997.

[27] M. Macedonia and D. Brutzman, *MBONE Provides Audio and Video Across the Internet.* IEEE Computer, April, 1994, pp. 30-36.

[28] S. McCanne, *Scalable Compression and Transmission of Internet Multicast Video* Ph.D. thesis, University of California at Berkeley, Computer Science, Berkeley, CA, 1996.

[29] S. McCanne and S. Floyd, *The LBNL Network Simulator,* , Lawrence Berkeley Laboratory.

[30] S. McCanne and V. Jacobson, *VIC: A Flexible Framework for Packet Video.* ACM Multimedia, .

[31] S. McCanne, V. Jacobson, and M. Vetterli, *Receiver-Driven Layered Multicast.* In Proceedings of SIGCOMM, pp. 117-130, Stanford, CA, August, 1996.

[32] S. McCanne and M. Vetterli, *Joint Source/Channel Coding for Multicast Packet Video.* In Proceedings of IEEE International Conference on Image Processing, pp. 25-28, Washington, DC, October, 1995.

[33] S. McCanne, M. Vetterli, and V. Jacobson, *Low-Complexity Video Coding for Receiver-Driven Layered Multicast.* IEEE Journal of Selected Areas in Communications, August, 1997, 16(6), pp. 983-1001.

[34] P. Pancha and E. Zarki, *Bandwidth Requirements of Variable Bit Rate MPEG Sources in ATM Networks.* In Proceedings of INFOCOM, pp. 902-909, March, 1992.

[35] P. Pancha and M. Zarki, *MPEG Coding for Variable Bit Rate Video Transmission.* IEEE Communications, May, 1994, pp. 54-66.

[36] S. Paul, *et al., Reliable Multicast Transport Protocol (RMTP).* IEEE Journal on Selected Areas of Communications, April, 1997, 15(3), pp. 407-420.

[37] S. Pejhan, M. Schwartz, and D. Anastassiou, *Error Control Using Retransmission Schemes in Multicast Transport Protocols for Real-Time Media.* IEEE/ACM Transaction on Networking, June, 1996, 4(3), pp. 413-327.

[38] D. Rover, *et al., The Application of Software Tools to Complex Systems.* IEEE Concurrency, 1998, 6(2), pp. 40-54.

[39] J. Salehi, *et al., Supporting Stored Video: Reducing Rate Variability and End-to-End Resource Requirements through Optimal Smoothing.* In Proceedings of SIGMETRICS, May, 1996.

[40] H. Schulzrinne, *Issues in Designing a Transport Protocol for Audio and Video Conferences and other Multiparticipant Real-Time Applications, ,* May, 1994, Internet Engineering Task Force.

[41] H. Schulzrinne, *et al., RTP: A Transport Protocol for Real-Time Applications, ,* January, 1996, Internet Engineering Task Force, RFC1889.

[42] N. Shacham, *Multipoint Communication by Hierarchically Encoded Data.* In Proceedings of INFOCOM, pp. 2107-2114, Florence, Italy, May, 1992.

[43] J. Shapiro, *Embedded Image Coding Using Zerotrees of Wavelet Coefficients.* In Proceedings of IEEE Transactions on Signal Processing, pp. 3445-3462, December, 1993.

[44] H. Smith, Mutka, M., Rover, D., *A Feedback Based Rate Control Algorithm for Multicast Videoconferencing.* Journal of High Speed Networks, 1998, pp. 259-279.

[45] H. Smith, Mutka, M., *Pattern Smoothing for Compressed Video Transmission.* In Proceedings of IEEE International Conference on Communications (ICC'99), pp. 1335-1339, Monreal, Canada, June, 1997.

[46] H. Smith, M. Mutka, and E. Torng, *Bandwidth Allocation for Layered Multicasted Video.* In Proceedings of International Conference on Multimedia and Computer Systems, Florence, Italy, June, 19999.

[47] D. Towsley, J. Kurose, and S. Pingali, *A Comparison of Sender-Initiated and Receiver-Initiated Reliable Multicast Protocols.* IEEE Journal on Selected Areas of Communications, April, 1997, 15(3), pp. 398-406.

[48] T. Turletti and C. Huitema, *Videoconferencing on the Internet.* IEEE/ACM Transactions on Networking, June, 1996, 4(3), pp. 340-351.

[49] K. Varadhan, D. Estring, and S. Floyd, *Impact of Network Dynamics on End-to-End Protocols: Case Studies in TCP and Reliable Multicast.* IEEE/ACM Transaction on Networking, February, 1998.

[50] L. Vicisano, L. Rizo, and J. Crowcroft, *TCP-Like Congestion Control for Layered Multicast Data Transfer.* In Proceedings of INFOCOM, pp. 996, San Francisco, CA, March/April, 1998.

[51] M. Vishwanath and P. Chou, *An Efficient Algorithm for Hierarchical Compression of Video.* In Proceedings of IEEE International Conference on Image Processing,, Austin, TX, November, 1994.

[52] A. Waheed, *et al., Modeling, Evaluation, and Adaptive Control of an Instrumentation System.* In Proceedings of IEEE Real-Time Technology and Applications Symposium (RTAS'97), pp. 100-110, Montreal, Canada, June, 1997.

[53] G. Wallace, *The JPEG Still Picture Compression Standard.* Communications of the ACM, April, 1991, 34(4), pp. 30-44.

[54] L. Zhang, *et al., RSVP: A New Resource ReSerVation Protocol.* IEEE Network, September, 1993, pp. 8-18.

[55] Z. Zhang, *et al., Smoothing, Statistical Multiplexing and Call Admission Control for Stored Video.* Journal on Selected Areas in Communications, August, 1997.