INFORMATION TO USERS

This manuscript has been reproduced from the microfilm master. UMI films the text directly from the original or copy submitted. Thus, some thesis and dissertation copies are in typewriter face, while others may be from any type of computer printer.

The quality of this reproduction is dependent upon the quality of the copy submitted. Broken or indistinct print, colored or poor quality illustrations and photographs, print bleedthrough, substandard margins, and improper alignment can adversely affect reproduction.

In the unlikely event that the author did not send UMI a complete manuscript and there are missing pages, these will be noted. Also, if unauthorized copyright material had to be removed, a note will indicate the deletion.

Oversize materials (e.g., maps, drawings, charts) are reproduced by sectioning the original, beginning at the upper left-hand corner and continuing from left to right in equal sections with small overlaps. Each original is also photographed in one exposure and is included in reduced form at the back of the book.

Photographs included in the original manuscript have been reproduced xerographically in this copy. Higher quality 6" x 9" black and white photographic prints are available for any photographs or illustrations appearing in this copy for an additional charge. Contact UMI directly to order.

UMI

A Bell & Howell Information Company
300 North Zeeb Road, Ann Arbor MI 48106-1346 USA
313/761-4700 800/521-0600

On Performance Improvement of Restricted Bandwidth Multimedia Conferencing over Packet Switched Networks

by

Hani H. ElGebaly

B.Sc., Cairo University, 1989
M.Sc., University of Saskatchewan, 1993

A Dissertation Submitted in Partial Fulfillment of the
Requirements for the Degree of

Doctor of Philosophy

in the Department of Computer Science

We accept this dissertation as conforming
to the required standard

Dr. J. Muzio, Supervisor (Dept. of Computer Science)

Dr. E. Manning, Departmental Member (Dept. of Computer Science)

Dr. H. Muller, Departmental Member (Dept. of Computer Science)

Dr. F. ElGuibaly, Outside Member (Dept. of Elect. and Comp. Eng.)

Dr. V. Kumar, External Examiner (Intel Corporation)

Supervisor: Dr. J. Muzio

Abstract

Advances in computer technology such as faster processors, better data compression schemes, and cheaper audio and video devices have made it possible to integrate multimedia into the computing environment. Desktop conferencing evolved as a plausible result of this multimedia revolution. The bandwidth granted for these conferencing applications is restricted in most cases by the speed of the modem device connected to the network.

Poor performance of multimedia conferencing over the Internet can be attributed to two main factors: local and remote induced effects. Local effects are induced by bandwidth sharing between different media components, operating system limitations, or poor design. Remote effects include all Internet related problems such as unfairness, non-guaranteed quality of service, congestion, etc. Both effects are addressed in this study and some solutions are proposed. The primary goal is to maintain audio quality and prevent video from degrading audio performance.

We study characteristics of video and audio traffic sources of conferencing applications following the H.323 set of standards defined by the International Telecommunication Union (ITU). The media traffic uses the Real-time Transport Protocol (RTP) and User Datagram Protocol (UDP) as their transport vehicle over IP network protocol. Tradeoffs involved in the choice of multimedia traffic parameters are presented. Our measurements were carried out on audio and video codecs defined in G.723.1 and H.263 specifications respectively, both drafted by the ITU.

This dissertation investigates traffic multiplexing issues at the host, and the interaction of conferencing media components as they are multiplexed locally in a shared bandwidth transport medium. Lack of appropriate multiplexing algorithms can lead to one or more media components oversubscribing to the shared bandwidth and penalizing other participants. These local effects can contribute significantly to traffic delay or abuse of the network bandwidth. We propose the "bit rate adjuster" (BRA) algorithm and use it

the network bandwidth. We propose the "bit rate adjuster" (BRA) algorithm and use it for regulating media flow. The algorithm compensates for video local effects induced by packet preparation or processing to allow for better audio performance. A new performance qualifier is introduced and used in the evaluation process.

Further on the remote side, we investigate reactive mechanisms used to recover media flow performance degradation caused by shared bandwidth traffic effects. We overview feedback mechanisms based on the Real-time Control Protocol (RTCP). We uncover its limitation on applications connected to the Internet through narrow bandwidth pipes. We propose an alternative approach that predicts and prevents the loss of audio packets before it occurs based on local computation of audio jitter. We also propose a mechanism that recovers audio traffic from jitter and latency effects introduced by the Internet shared medium. These approaches improve the audio performance significantly in multimedia conferencing sessions.

Dr. J. Muzio, Supervisor (Dept. of Computer Science)

Dr. E. Manning, Departmental Member (Dept. of Computer Science)

Dr. H. Muller, Departmental Member (Dept. of Computer Science)

Dr. F. ElGuibaly, Outside Member (Dept. of Elect. and Comp. Eng.)

Dr. V. Kumar, External Examiner (Intel Corporation)

Table of Contents

List of Tables

List of Figures

Acknowledgements

The author would like to express his gratitude to Professor Jon Muzio for all his support throughout the development phases of this thesis. Special thanks to Professor Fayez ElGuibaly for all his inspiring ideas, fruitful discussions and valuable feedback being a member of the supervisory committee. Sincere gratitude is extended to Professor Eric Manning and Professor Hausi Muller for serving as members of the supervisory committee. The author also would like to extend thanks to Dr. Vineet Kumar for serving as an external examiner of this dissertation . In addition, the author would like to express his indebtedness to all coworkers in the conferencing products group of Intel Architecture Labs. Deep appreciation is extended to Mr. Mike Gutmann and Mr. Vijay Rao of Intel Architecture Labs for their support, encouragement and sincere advice. Special thanks and acknowledgement are also extended to Mr. Steve Ing and Mr. Jose Puthenkulam for all the productive discussions during the various stages of the experimental work. Finally, and most of all, the author would like to express special recognition and thanks to his family for all their support and encouragement.

To my mother, my wife and my son.

1. Background and Problem Definition

1.1 Introduction

Advances in computer and communication technology have stimulated the integration of digital audio and video with computing. This integration lead to the development of multimedia applications such as video conferencing and multimedia collaboration. Many of these applications are targeted to run over packet-switched networks such as the Internet. Packet switched networks with non-guaranteed quality of service do not seem suitable for real-time traffic such as multimedia conferencing.

Internet switches do not treat network traffic in a fair way. Packets are routed independently across shared routers and switches. These switches do not pay attention to the type of the packet, packet loss or latency sensitivity.. As parts of the Internet become heavily loaded, congestion can occur. Congestion may lead to buffer overflow and packet loss. It may also lead to packet delay as packets take longer to process. Latency may seem acceptable for some applications such as email and file transfer. For real-time applications, data becomes obsolete if it does not arrive in time. In addition, real-time applications can be quite sensitive to loss. Further, since packets are routed independently across shared routers and switches, transit times may vary significantly. Variation in transit delay is called jitter. Jitter is very disturbing to real-time applications especially to audio playback. It significantly reduces speech intelligibility to the ear and causes choppiness and breakup in the stream. Consequently real-time applications deliver poor quality during periods of congestion over shared bandwidth networks such as the Internet.

Several researchers addressed the Internet problems and proposed several solutions. Most of the solutions aim at either suggesting new infrastructures or proposing modifications to the current Internet TCP/IP-based infrastructure.

New infrastructure enhancement proposes new technologies (e.g., Fiber optics [3], better and faster switch architectures [1], and [21], and new protocol layers (e.g. B-ISDN [11], ATM [51], etc.).

Current Infrastructure enhancement introduces ways of enforcing resource reservation, quality of service bounds and recognition of real-time traffic [19], [41].

Only a few researchers addressed real-time traffic performance over the current Internet infrastructure and ways of improvement [49]. This study is primarily concerned with point-to-point real-time multimedia conferencing over the Internet. The conferencing application link to the Internet has limited bandwidth. The goal is to solve a few media conferencing problems pertaining to a certain class of platforms using current infrastructure and without proposing any alteration to network hardware or protocols.

We believe new technologies will evolve over the years. The deployment of the technology may take some time but eventually it will make it to the implementation phase. Many of these technologies have already been running for years over private networks and exhibiting outstanding performance. Yet, solutions for current networks are essential since they will persist for a while as a vital communication media for many end users. In the next section, we discuss the goals and objectives of this thesis.

1.2 Objectives and Motivation

This study discusses congestion handling schemes in desktop videoconferencing with scarce bandwidth. Focus is set upon media conferencing connections with limited bandwidth (e.g., access to the Internet is provided via modems) and non-guaranteed quality of service (QoS) transport layer such as the Internet. The primary objective is to maintain audio quality and to prevent video from degrading audio performance.

Solutions are provided for overall audio latency and jitter in a conferencing session with the given constraints. These solutions are either local to the host and called "local schemes" or based on feedback (recovery) from (at) the remote conferencing peer and called "remote schemes".

Local schemes address issues such as careful scheduling of video and audio traffic at the host before media components reach the shared bandwidth network layer. The objective is to prevent video from oversubscribing to the available bandwidth without degrading performance of the audio stream. We propose a scheduling algorithm that achieves this goal and provide an elaborate discussion of the advantages and limitations of this approach.

Traditional feedback schemes rely on statistics such as loss, jitter, latency, timestamps, etc. exchanged between network terminal peers. A terminal, upon receiving these statistics will take a decision to adjust the cumulative throughput of one or more media channels r in order to make up for loss, latency or jitter symptoms. This class of feedback schemes is called "Remote Feedback Local Control" (RFLC). Alternatively, terminals can compute these statistics based on the actual stream of media data and provide flow control commands to remote terminals to limit throughput of the degrading media type. This class of feedback schemes is introduced in this thesis and called "Local Feedback Remote Control (LFRC). The objective is to devise a scheme that can predict audio loss occurrence and make appropriate action to prevent it. We uncover a relationship between jitter occurrence and loss and propose a new algorithm to predict audio loss and prevent it.

Media recovery methods at the remote terminal also belong to remote schemes. Examples of such schemes include loss, latency or jitter compensation. We investigate jitter compensation as an example of recovery methods. The goal is to recover audio patterns (interarrival time), generated at the transmitter, at the receiver side to the greatest possible accuracy. We propose an algorithm that achieves this goal and evaluate its performance.

Towards these objectives, this thesis addresses various tradeoffs involved in the choice of media traffic parameters such as packet size versus protocol overhead, maximum packet length versus burstiness, etc. in a heterogeneous traffic mix. The thesis results can be readily applied to any connection-oriented conferencing session with limited bandwidth.

Congestion control of traffic components on non-guaranteed point-to-point multimedia conferencing has hardly been addressed in the literature. The main problem is the non-existence of standard procedures for dealing with congestion on such platforms. The International Telecommunication Union (ITU-T) recently released standards for protocols and connection procedures over packet-based networks but the congestion problem was left as an implementation issue. More work has yet to be established in the area of flow control and QoS satisfaction for different media types. Most current congestion handling schemes for Desktop conferencing over the Internet are ad-hoc and lack methodology. This was a key motivation to address this topic. In the next few sections, we briefly introduce topics such as bandwidth utilization, packet scheduling, and congestion control. These topics are addressed in later chapters.

1.3 Bandwidth and Latency

David Cheriton of Stanford University once said, "*A network link with low bandwidth can be improved by adding several in parallel to make a combined link with higher bandwidth. However, a network link with bad latency can never be improved no matter how much bandwidth is added.*" To understand this problem, it is important to distinguish between speed and capacity. Speed is a measure of distance divided by time while capacity or bandwidth is a measure of bits per second.

1.3.1 Internet Latency Example

Most home communication connections to the Internet are via modems running over telephone lines. Internet Service Providers purchase wholesale bandwidth and share it amongst their subscribers. The bandwidth may get scarce as more and more people get online yet the latency problem is more severe. Here is an example taken from an Internet article [13] showing Internet latency measurement between Stanford University located in California and Massachusetts Institute of Technology located in Massachusetts. Both Universities in this example are connected directly to the Internet:

- Distance from Stanford to MIT is 4320 km.

- The speed of light in vacuum is 3 * 10^8 m/s.
- The speed of light in fiber is 60 percent of the speed of light in vacuum.
- The speed of light in fiber is 180 * 10^6 m/s.
- One-way delay to MIT is 24 ms (round-trip = 48 ms).
- Current ping time from Stanford to MIT over the Internet is 85ms.

This example shows that the current hardware of the Internet as it stands can achieve a data transfer rate of the speed of light + 76 percent (ping time / round-trip delay).

Therefore, this implies that the Internet is doing well. In fact, it is a great achievement to be away from the theoretical optimum by less than a factor of two.

1.3.2 Latency Contributed by Modem Connection

Now suppose the above connection to the Internet is done via modems from both sides. The end-end transmission delay is composed of the speed of light delay, the per-byte transmission time (modem bandwidth), and a fixed software and hardware overhead.

The speed of light in fiber is comparable to the speed of light in copper. Assume their distance from the Internet Service Provider is 18km each.

Hence the speed of light latency = 18000/180*10^{6} = 0.1 ms. The transmission time depends on the transmission rate of the modem. The fixed hardware and software overhead varies between different architectures and platforms. One delay factor, which is attributed to that overhead, is grouping of data. There is a modem wait time as it tries to group data into blocks, perform compression and automatic error correction. To get effective compression and error correction, modems must work on large blocks of data. This means that characters must be buffered until a sufficient block is built for the modem to work on efficiently. Data are not sent until processed by the modem's compression/error correction engine. This adds to the latency of data as it passes through the modem. In addition, the modem does not know the kind of data being sent, and consequently cannot use the best data-specific compression algorithms. For example,

multimedia data are usually compressed before they pass through modems. Modem compression in this case is futile. In fact, it may significantly affect the timeliness of the latency-sensitive data transferred through the modem.

For a typical modem link, the latency due to modem software and hardware overhead is usually about 100ms. The transmission time of 10 characters over a 33kbit/sec modem link time would theoretically be 80 bits / 33000 bits per second = 2.4ms. The actual time taken because of the compression overhead is 102.4ms because of the 100ms latency introduced by the modems at each end of the link.

To enable small chunks transfer, there is a timeout value before the modem starts processing a data block. This way modems can avoid waiting for the large block indefinitely and severely causing huge delays to the peer.

Hence, modem hardware and software overhead is a significant contributor to latency when connected to the Internet via modems through Internet Service providers.

1.3.3 Internet Service Providers Bottleneck

The Internet service provider (ISP) is the means by which the home user and many businesses hook to the Internet. Users connect to their ISPs usually over POTS (Plain Old Telephone Service) lines via modems. An Internet provider in turn must connect to another wholesale Internet provider. The connection of an ISP to the wholesale provider is called the ISP pipe. Most ISPs anticipate relatively low bandwidth activity from their users such as web browsing, reading information, etc. Multimedia conferencing is rarely taken into account. For example, Table 1 shows how an ISP pipe size may map to the number of ISP users [8].

Table 1 does not take into account if users are engaged in multimedia conferencing sessions. Hence, there evolves new requirements for future ISPs in order to satisfy their conferencing clients.

Typically, each conferencing client will chew up at least 15-20kbps of the available bandwidth. This leads to a significant amount of latency and loss experienced during the

conferencing session. Further, most ISPs are oriented to downloading (down streaming). Packet loss is more significant when a conferencing client is sending media data upstream. This loss is due to severe congestion that causes intermediate router buffers to overflow.

In the next section, we discuss some observations noted by network and multimedia researchers since the early trials of integrating real-time components to non real-time networks.

Table 1. ISP Pipeline size versus number of users

Pipeline Size	Number of Simultaneous Dial up users
28.8K modem connection	3
56K DSO leased line	7
64K ISDN connection	8
128K ISDN connection	15
256K Fractional T1	40
512K Fractional T1	100+
Full T1	300+

1.4 Bandwidth Observation in Packet Switched Networks

Congestion defines the situation where performance degrades due to too much offered load [65]. However, it was observed that offering more traffic to a network than its designated capacity is a prerequisite for achieving continuous and full utilization of all its bandwidth [43]. Overloading the network bandwidth usually results in congestion and may ultimately lead to loss of packets because of buffer overflow.

Buffer memory, however, is no longer a scarce commodity. RAM and disk space prices are falling daily. Network bandwidth often turns out to be a more precious resource where

capacity of the transport media is limited. In global switched telephone networks (GSTN), the modem capacity is up to only 56kbps downstream and approximately 33.6kbps upstream. Thus, in networks with limited link capacity, bandwidth is more precious than buffer space. It is better to utilize the buffer space to the maximum extent than to drop packets and later retransmit them. The flow of packets must be able to proceed at its maximum rate whenever there is nothing to stop it. Real-time requirements for multimedia streams also sustain this observation. The buffer space must be roughly at least as large as the desired peak throughput multiplied by the round-trip delay [39]. The round-trip delay is defined as the time the flow control token can travel end-to-end in one direction plus time required for the packet to travel end-to-end in the other direction.

In the next section we overview packet scheduling techniques that are commonly used for dispatching packets over packet-switched networks.

1.5 Packet Scheduling Techniques

In a point-to-point network connection, a packet scheduler chooses multimedia packets to be transmitted over the media transport at specific intervals. Several scheduling disciplines have been addressed previously in the literature [18], [21], and [72]. The most important of these are First-Come-First-Served (FCFS), Static Priority (SP), Earliest-Deadline-First (EDF), Round-Robin (RR), and Weighted Round Robin (WRR).

1.5.1 Scheduling Policies Overview

FCFS schedulers transmit all packets in the order of their arrival. All media types are assigned an identical delay bound. In SP schedulers, all media types are assigned a certain priority, where a lower priority index indicates a higher priority. SP schedulers maintain one FCFS queue for each priority level, always selecting the first packet in the highest-priority FCFS queue for transmission. With EDF scheduling, each media type is assigned a delay bound, where the delay bound may be different for each connection. EDF scheduler selects packets for transmission in increasing order of packet deadlines, where packet deadlines are calculated as the sum of the arrival times of the delay bound of a

packet. RR scheduling circularly scans all queues and picks a packet from each queue that is found ready. RR scheduling equally distributes the entire available throughput amongst all media components that can use it. An extension that gives more power to the basic RR scheduling is to use WRR, where certain queues are visited more than once each scanning cycle in proportion to a prescribed frequency of service weight. Hence, these queues receive a proportionately higher share of bandwidth.

The selection of a particular scheduling mechanism involves a tradeoff between the need to support a large number of connections with diverse delay requirements and the need for simplicity in the scheduling operations. For example, a FCFS is quite easy to implement but it provides one delay bound for all connections. Multimedia components have diverse delay requirements and hence this scheme is not an appropriate scheme despite its simplicity. On the other extreme, an EDF scheduler can support a different delay bound for each connection, yet the scheduling operations are complex. The complexity lies in search engines required for finding the packet with the shortest deadline.

The scheduling scheme must not starve (i.e. ignore forever) any packet present inside the terminal buffers at any time. Further, in a real-time environment, the multiplexing scheme must satisfy the condition that resource requirements of real-time packets must be protected against the demands of non-real time ones. Static priority schemes usually lead to starvation inside the terminal of low priority packets. In other words, higher priority classes tend to monopolize the output links, severely degrading the service of lower priority classes.

In a following chapter, we study and compare FCFS and EDF schedulers for audio and video traffic sources originating from the local host and sharing the same transport bandwidth. Performance qualifiers such as latency, and maximum packet size are collected and analyzed. In the next section, we review some of the common performance evaluation techniques.

1.5.2 Performance Evaluation Techniques

Three techniques are usually considered for the evaluation of multiplexing schemes for multimedia traffic networks: analytical, computer simulation and empirical measurement with hardware test bed [71].

Analytical techniques are highly desirable to evaluate performance with reasonable computational requirements and accuracy. The problem of deriving performance results has been addressed in the literature with more focus on ATM switches e.g., [6], [21], [69], and [26]. However, every analytical technique has certain limitations with respect to the requirements stated. Examples of these limitations are the inability to represent heterogeneous input traffic flows, or extreme percentiles of the delay or queue length probability distributions. Analytical techniques should be developed further to capture these essential considerations.

Simulation methods remain the most flexible for examining network operations and performance under variety of conditions. We use simulation in most of our experiments as our proof of concept. Simulation input is based on either real traffic workload or worst case behavior.

A hardware test bed provides the most detailed system representation and can run in real time. It requires the development of load boxes that can either generate traffic based on statistical models such as those obtained by computer simulation, or can offer traffic loads from real applications. In the former case, the test bed serves as a mean to validate the results estimated by an analytical or simulation method. In the latter case, it provides the ability to test admission control strategies under realistic traffic scenarios.

1.5.3 Bandwidth Utilization

The service supported in a fully evolved multimedia network is expected to produce a wide range of traffic flow characteristics and have a wide range of performance requirements. Individual traffic sources will vary from continuous to extremely bursty. If the sum of peak rates of all sources does not exceed the available bandwidth, then this

mode of operation is termed non-statistical. The strong advantage of non-statistical multiplexing is minimal packet delay and no packet loss due to buffer overflow. Stringent performance requirements can be met in a simple manner by reserving the connection's peak bandwidth requirements and not allowing the total reserved bandwidth to exceed the available link bandwidth. This is the most common approach for bandwidth management of conferencing sessions in the context of H.323 conferencing standard. A gatekeeper can operate as a conference cop and allows call admission only if bandwidth is available. However, bandwidth can not be enforced with this policy in a non-guaranteed QoS shared medium. Media traffic can transiently exceed their allocated bandwidth (e.g. because of variable bit-rate video) causing degradation in performance of other media participants (e.g. constant bit-rate audio). Our scheduling policy discussed in chapter 5 provides some remedy for this situation.

The disadvantage of non-statistical mode is that good bandwidth utilization cannot be achieved when a large proportion of the traffic is bursty. In the next section, we review some of congestion control trends as dictated by general practical experience of academic and industrial leaders in the field.

1.6 Congestion Control

Congestion occurs when the demand is greater than the available resources. It is commonly believed that as resources become cheaper, the congestion problem will be solved [39]. This belief stems from certain myths about the causes of congestion problem.

The first cause of congestion is believed to be a shortage of buffer space. The congestion problem will be solved when memory is cheap enough to allow infinite buffers. Cheaper memory can indeed help the congestion problem but cannot solve it. With infinite memory, the queues can get longer and packets suffer more delay. By the time the packets are dequeued, they would have already become obsolete. In the case of reliable transport, packets would have been timed out and retransmitted leading to more packets on the network and hence adding to the congestion problem. Hence, too much memory can be more harmful than little memory.

The second cause of congestion is believed to be slow links (e.g., low speed modems). The congestion problem is mistakenly believed to be solved with high-speed links and abundant bandwidth. Through the evolution stages of data links, speed has gone up from 300 b/s on telephone links to dedicated 1.5 Mb/s links to 10 Mb/s, 100 MB/s LANs and even a lot more with ATM technology. However, high-speed LANs are connected via low-speed links. Many home users with dial-up slips/PPP are connected to the Internet via less than 56kb/s modems. The point is high-speed links cannot stay in isolation since low-speed links don't go away. Introduction of high-speed networks, however, has increased the range of speeds that have to be managed. This may have a negative effect on the performance. The conclusion is higher link speeds and more bandwidth cannot solve the congestion problem on their own.

The third cause of congestion is believed to be slow processors. It is believed as processors speed increases, protocol tasks will be processed faster and the congestion problem will go away. However, introduction of high-speed processors may increase the mismatch of speeds, leading to more chances of congestion.

All of the above causes and solutions are static. Congestion, on the other hand, is a dynamic problem. The solution to the congestion problem has to be dynamic as well. The explosion of processor speeds, higher speed links and infinite buffers have led to more unbalanced network connections that are causing congestion. In particular, all of the above mentioned causes of congestion such as slow processing, low speed links and packet loss (due to shortage of buffer space) are symptoms of congestion rather than causes. Knowing the symptom, the cause of the problem can be diagnosed and the appropriate (dynamic) solution can be deduced.

1.6.1 Congestion Control Approaches

A resource is congested if the total sum of demands on that resource exceeds its available capacity. Depending upon the number of resources involved, a congestion problem can be classified as a single resource problem or a distributed resource problem [39]. The single resource problem is solved by providing schemes that can coordinate the demands on that

resource. An example is the Internet available bandwidth as the single resource for which audio, video and data sources are competing to share in a point-to-point connection. Bandwidth reservation, adaptive schemes, admission control policies, proper scheduling are all examples of solutions to the problem.

The problem is more difficult if the resource is distributed. An example is a store-and-forward network where links are the resources. User demands have to be limited such that the total demand at each link is less than its capacity. This problem is outside the scope of this study.

Congestion control schemes are classified into two types: preventive and adaptive. These two types require that the users be informed about the load condition in the network connection so that they can adjust the traffic. The preventive approach aims at avoiding congestion occurrence by enforcing stringent admission control policies. This approach prevents new sessions from starting up if their demands (quality of service) are not satisfied. The adaptive approach dynamically asks users to schedule their demands or reduce their loads such that the total demand on the resource is less than its capacity. Congestion schemes studied in this thesis are of the adaptive type.

1.6.2 Main Principles in Handling Congestion

All congestion control schemes require the network to measure the total load on the network and then take some remedial action. The first part is called "feedback", while the second is called "control". A feedback signal is sent from the congested resource to one or more control points that are required to take remedial action. The control point can be the source of the traffic and the action can be reduction of bit-rate flow from that source.

The problem of congestion control is a control problem. The control theory states that the control frequency should be equal to the feedback frequency. The control frequency in this case is the frequency of control actions taken by the congestion control scheme. The feedback frequency is the frequency of feedback information provided to a congested resource. If the control is faster than the feedback, the system will have oscillations and instability. On the other hand, if the control is slower than the feedback, the system will

be slow and tardy to respond to changes. It is important to apply this principle when designing congestion control schemes. The control interval in the network world is mapped to how often feedback messages should be sent and how long to wait before acting.

Another lesson to learn from the control theory is that no scheme can solve congestion that is shorter than its feedback delay [39]. Feedback delay is the time elapsed between the congestion occurrence and the feedback information provided to the congested resource. Short duration congestion can be solved by priority classes, and other similar schemes. Medium duration congestion can be solved by dynamic windows or rate schemes. Long duration congestion is controlled by session control (e.g., admission control) schemes. Infinite duration congestion is treated by installing extra resources.

Since the duration of congestion cannot be determined in advance, it is best to use a combination of schemes operating at different layers. Most of the congestion problems we deal with in this thesis are of either the short or the medium duration type. We study a combination of short and medium duration solutions and evaluate their performance.

1.7 Outline of the Dissertation

Chapter 2 discusses recent advances in desktop conferencing technology and standards. Section 2.1 overviews current encoding and compression technologies applied for audio and video. Section 2.2 provides a general walkthrough over the class of conferencing systems addressed in this study. Section 2.3 overviews the network layers involved in a conferencing session from the ISO seven-layer perspective [65]. Section 2.4 presents the standards approved by the ITU-T for conferencing establishment procedures and streaming.

Chapter 3 addresses traffic characterization of audio and video and tradeoffs involved in multimedia generation in real-time conferencing. Section 3.2 provides an overview of some related work. Section 3.3 presents the methodology used in collecting

measurements. Sections 3.4, 3.5, and 3.6 provide an analysis of video and audio traffic traces and present tradeoffs in the choice of different traffic parameters.

Chapter 4 deals with the problems arising from multiplexing audio and video traffic on the shared bandwidth of the host at the network edge. Section 4.1 presents an analogy with real-time system scheduling problem. Section 4.2 compares First-Come-First-Served (FCFS) scheduling to Earliest-Deadline-First (EDF). Section 4.4 deduces the main reasons for the problem.

Chapter 5 proposes a new algorithm for dealing with audio/video multiplexing problem at the host in an attempt to achieve better latency for audio over the FCFS scheduling case. The algorithm is simulated and analyzed in sections 5.1 through 5.4.

Chapter 6 discusses feedback algorithms for dealing with audio loss problem. Sections 6.1 through 6.4 discuss Real-time Control Protocol (RTCP) mechanisms and present their limitation. Section 6.3 proposes a modification for an RTCP-based congestion control algorithm that is more sensitive to audio performance. Section 6.5 presents a novel scheme for predicting loss using jitter information. Section 6.6 discusses an innovative technique for recovering audio generative pattern at the receiver by smoothing out the jitter effect completely.

2. Desktop Conferencing Technology and Standards

Advances in computer technology, such as faster processors and better data compression schemes have made it possible to integrate audio and video data into the computing environment. A new type of videoconferencing, desktop videoconferencing, has become possible. Unlike room videoconferencing, which requires specially equipped rooms with expensive hardware, desktop videoconferencing can be achieved by adding software and hardware to standard desktop computers.

Desktop videoconferencing has many benefits. One benefit is the convenience of not having to physically move to a special location. Desktop videoconferencing can be used for telecommuting, corporate meetings to cut travel costs, family gatherings, etc. Another benefit is Call Centers deployment. Banks, shopping centers, retail centers can be more efficient and cut a lot of overhead by providing dial-in videoconferencing customer service centers and kiosks for end users.

Desktop videoconferencing systems are significantly less expensive than room videoconferencing systems by at least one order of magnitude [66].

2.1 Technology

This section discusses the enabling technology for desktop videoconferencing. Audio and video are captured from their analog form and stored digitally on the computer. This multimedia data requires massive amounts of bandwidth to transmit. Therefore, compression must take place before data is sent over the communication channels. This must happen in real-time to facilitate communication and interaction.

In this section, an overview of audio and video encoding and compression is discussed. Current multimedia standards available from the ITU (International Telecommunication Union) are also overviewed in a following section.

2.1.1 Audio

The frequency of sound waves is measured in Hertz (Hz), meaning cycles per second. The human ear can typically perceive frequencies between 20 Hz and 20 kHz. Human voice can typically produce frequencies between 40 Hz and 4 kHz [14]. These limits are important factors to remember when discussing digital audio encoding. Desktop videoconferencing systems are typically designed to handle speech quality audio, which encompasses a much smaller range of frequencies than the range perceptible to humans.

Digital audio data is usually described using the following three parameters: sampling rate, bits per sample, and number of channels. The sampling rate is the number of samples per second. Bits per sample are the number of bits used to represent each sample value. The number of channels is one for mono and two for stereo.

2.1.1.1 Audio Sampling

An analog audio signal has amplitude values that continuously vary with time. To encode this signal digitally, the amplitude value of the signal is measured at regular intervals. This is called sampling. According to the Nyquist theory of signal processing, a signal of a certain frequency must have a sampling rate of at least twice that of the highest frequency present in the signal [61]. Using this theory, sampling is loss-less since the original signal can be reconstructed based on these samples. The signal is low-pass filtered to remove any high frequencies that can not be represented by the sampling rate. These unwanted signals are termed aliasing distortion [56].

Using Nyquist theory, 8 kHz is a sufficient sampling rate to capture the range of human voice (40 Hz to 4 kHz). Further 40 kHz is a sufficient sampling rate to capture the range of human hearing (20 Hz and 20 kHz). In practice, typical sampling rates range from 8 kHz to 48 kHz [55].

2.1.1.2 Audio Quantization

Sampled values representing the amplitude of the signal at the sample time are quantized into a discrete number of levels. The number of levels depends on how many bits are used

to store the sample value. For digital audio, this precision usually ranges from 8 bits per sample (256 levels) to 16 bits per sample (65536 levels) [55]. Quantization induces error into the data because no matter how many bits of precision are used; it is impossible to represent an infinite number of amplitude values with a finite number of increments [56].

Uniform pulse code modulation (PCM) encoding is an encoding method where the quantizer values are uniformly spaced. Uniform PCM is an uncompressed audio-encoding format, however some other PCM formats such as μ-law or A-law PCM use quantizer values that are logarithmically spaced, effectively achieving a degree of compression.

2.1.1.3 Digital Audio Compression Techniques

Uncompressed digital audio can require a large amount of bandwidth to transmit. There are many techniques used to compress digital audio. Some of the techniques commonly used in desktop videoconferencing systems are described below. Typically, these techniques can achieve real-time compression and decompression in software or inexpensive hardware. Some techniques apply to general audio signals and some are designed specifically for speech signals.

The first technique overviewed is the A-Law/μ-Law algorithm. With PCM encoding methods, each sample is represented by a code word. Uniform PCM uses uniform quantizer step spacing. By performing a transformation, the quantizer step spacing can be changed to be logarithmic, allowing a larger range of values to be covered with the same number of bits. There are two commonly used transformations: μ-law and A-law. These transformations allow 8 bits per sample to represent the same range of values that would be achieved with 14 bits per sample uniform PCM. This translates into a compression ratio of 1.75:1. Because of the logarithmic nature of the transform, low amplitude samples are encoded with greater accuracy than high amplitude samples.

The μ-law and A-law PCM encoding methods are formally specified in the International Telecommunication Union - Telecommunication Standardization Sector (ITU-T) Recommendation G.711 [36], "Pulse Code Modulation (PCM) of voice frequencies." The

μ-law PCM encoding format is common in North America and Japan for digital telephony with the Integrated Services Digital Network (ISDN). The A-law PCM encoding format is common with ISDN in other countries [55]. G.711 is one of the audio standards specified in H.320 (standard for multimedia conferencing over ISDN). It is also the mandatory audio codec for the H.323 standard (procedures for multimedia conferencing over non-guaranteed LAN) [29]. Note that at 8 kHz, 8 bits per sample, and 1 channel, μ-law or A-law PCM requires a bandwidth of 64 kb/s.

The second technique for audio compression is called adaptive differential pulse code modulation (ADPCM). PCM encoding methods encode each audio sample independently from adjacent samples. However, usually adjacent samples are similar to each other and the value of a sample can be predicted with some accuracy using the value of adjacent samples. For example, one simple prediction method is to assume that the next sample will be the same as the current sample. The ADPCM encoding method computes the difference between each sample and its predicted value and encodes the difference (hence the term "differential") [55]. Fewer bits (typically 4) are needed to encode the difference than the complete sample value.

Encoders can adapt to signal characteristics by changing quantizing or prediction parameters (hence the term "adaptive"). ADPCM typically achieves compression ratios of 2:1 when compared to μ-law or A-law PCM [14]. ADPCM encoders differ in the way the predicted value is calculated. They also differ in how the predictor or quantizer adapts to signal characteristics.

Many desktop videoconferencing systems use ADPCM encoding methods. The ITU-T has several recommendations defining different ADPCM methods (G.721, G.722, G.723.1, G.726, and G.727).

2.1.2 Video

Video is a sequence of still images. When presented at a high rate, the sequence of images (frames) gives the illusion of fluid motion. For instance, in the United States, movies are presented at 24 frames per second (fps) and television is presented at 30 fps.

Video input for desktop videoconferencing may come from a camera, VCR, or other video device. An analog video signal must be encoded in digital form so that it can be manipulated by a computer. To understand digital encoding, it helps to comprehend some background information about analog video, including basic color theory and analog encoding formats.

2.1.2.1 Video Encoding Techniques

Analog video is digitized so that a computer can manipulate it. Each frame of video becomes a two-dimensional array of pixels. Uncompressed images and video are too large to deal with and compression is needed for both storage and transmission. Two of the most important compression measures, are compression ratio (ratio of compressed data bits to original data bits) and bits per pixel (the number of bits required to represent one pixel in the image).

Video compression is typically lossy, meaning some of the information is lost during the compression step. This is acceptable because encoding algorithms are designed to discard information that is not perceptible to humans or information that is redundant. There are some basic techniques common to most video compression algorithms, including color space sampling and redundancy reduction.

Color space sampling is an effective technique used to reduce the amount of data that needs to be encoded. Color can be represented by two components: luminance and chrominance. Luminance, denoted by the symbol Y, represents the brightness information in a color space. Chrominance is composed of two components representing color differences, identified as U and V. The notation YUV is often used generically to refer to a color space represented by luminance and two color differences. If an image is encoded

in YUV space, the U and V components can be sub-sampled because the human eye is less sensitive to chrominance information. Redundancy reduction is another technique used to decrease the amount of encoded information. Intra-frame encoding achieves compression by reducing the spatial redundancy within a picture. This technique works because neighboring pixels in an image are usually similar. Inter-frame encoding achieves compression by reducing the temporal redundancy between pictures. This technique works because neighboring frames in a sequence of images are usually similar.

Discrete Cosine Transform (DCT) *encoding transforms a signal* from the time domain to the frequency domain taking advantage of the fact that visual sensitivity drops with increasing frequency. This transform is widely used in video encoding codecs such as H.261, H.263, JPEG, and MPEG. DCT is an *O(N log N)* complexity algorithm. It provides excellent energy compaction of images by packing transform coefficients toward the low frequency end of the spectrum, and hence eliminates unwanted high frequency components. The basic steps of DCT-based encoding are shown in Figure 1.

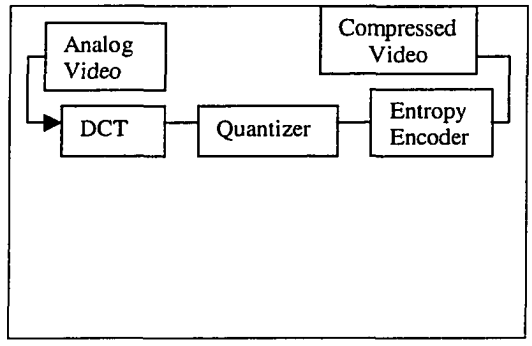

Figure 1. DCT encoding steps.

The first step of the encoding process is to perform a DCT on 8x8 blocks of component samples. This step transforms the information into the frequency domain. The output of this step is 64 DCT coefficients. The DCT has the effect of concentrating most of the information in the 8x8 block into the upper left-hand corner. The average value of the block, called the DC component, is the upper left-hand coefficient. The remaining

coefficients are called AC coefficients. No information is lost during the DCT step. The data is merely transformed to another domain and can be recovered by performing an inverse DCT.

The next step of the encoding process is Quantization. An 8x8 Quantization matrix divides the DCT coefficients. This matrix is designed to reduce the amplitude of the coefficients and to increase the number of zero value coefficients [2]. The Quantization step is the lossy step of the encoding process.

After Quantization, a bit stream is formed from the block. The DC coefficient is encoded as the difference between the current DC coefficient and the DC coefficient of the previous block. The AC coefficients are encoded in a zigzag sequence from the upper left of the block to the bottom right as shown in Figure 2. The AC coefficients are run-length encoded and entropy encoded. The run-length encoding removes long runs of zero valued coefficients. The entropy encoding (Huffman encoding) encodes the information efficiently based on statistical characteristics. Patterns that are statistically more likely to occur are encoded with shorter code words.

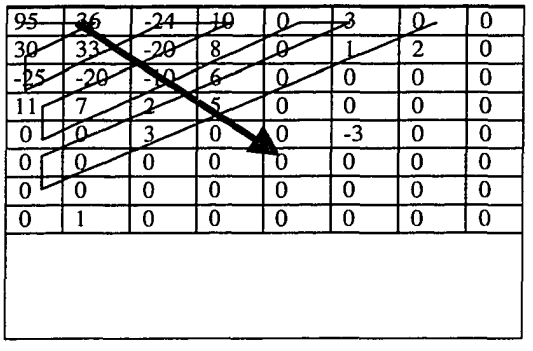

Figure 2. Encoding of DCT coefficients in a zigzag sequence

The decoding process is the reverse of the encoding process. The decoder must have access to the same quantization and entropy tables that were used to encode the data.

High compression ratio results in poor image quality. A user-configurable quality parameter is usually available that allows a compression versus quality tradeoff.

2.1.3 Communication Channels

There are different types of communication channels available to transport desktop videoconferencing data. These channels can be classified as either circuit-switched or packet-switched. Circuit-switched communication is a method of data transfer where a path of communication is established and kept open for the duration of the session. A dedicated amount of bandwidth is allocated for the exclusive use by the session. When the session is terminated, the bandwidth becomes available for other sessions. On the other hand, in Packet-switched communication information is divided into packets, each of which has an identification and destination address. Packets are sent individually through a network. Packets may take different routes and may arrive at their destination at different times and out-of-order depending on the network condition. No dedicated bandwidth circuit is set up as with circuit-switched communication. Bandwidth must be shared with others on the network. Each type of channel has advantages and disadvantages when evaluating its suitability for transporting desktop videoconferencing data.

Desktop conferencing data types are mainly audio, video and interactive data. Generic data is not sensitive to delay but is sensitive to reliability. An example is a data file that is sent over a network. It does not matter how long the file takes to get to its destination, but the information in the file is expected to be correct. Audio data is sensitive to delay and reliability. Audio data must arrive at a constant rate with little variance for it to be intelligible. Audio loss is not acceptable in most cases. Still image data is not sensitive to delay and may not be sensitive to reliability. Incorrect image data may not be noticeable in the form of visual quality, but delivery time is not crucial. Video data is sensitive to delay and large delays will be obvious by a jerky picture. Uncompressed video data is not sensitive to reliability since if one frame is lost it will immediately be replaced by another frame. However, compressed video, which uses intra-frame and inter-frame encoding

techniques, is sensitive to reliability since redundancy has been removed and the effects of data loss may propagate. This is an important aspect to consider when sending video data across unreliable communication channels. Some video compression techniques compensate for this sensitivity to data loss by periodically sending complete information about a frame.

Desktop videoconferencing can involve all the data types discussed so far. Typically, Audio and video are exchanged between conference participants. Types of data that can be exchanged include files, whiteboard, chat and application sharing. Some of this data requires reliable transmission while some requires timely transmission.

2.2 Desktop Videoconferencing Systems

This section describes features and characteristics of some desktop videoconferencing systems available today. This is not meant to be a comprehensive survey, yet an overview of the most important products that are currently available.

There are three major platforms for desktop videoconferencing products: Intel-based (or compatible) personal computers running Microsoft Windows, Apple Macintosh computers, and Unix-based workstations running the X Window System.

Products are evolving towards conformance to the emerging videoconferencing interoperability standards. The ITU-T is leading a major effort to make interoperability possible across different videoconferencing systems running on different platforms. Videoconferencing companies conduct interoperability tests with each other as development phases evolve. Competitors may exchange new features information amongst each other in order to escalate interoperability efforts and not break backward compatibility.

All systems require hardware that captures and digitizes audio and video. Most systems have a graphical user interface that assists in making connections to other parties, usually utilizing the paradigm of "placing a telephone call." Many products allow storing

information about other parties in a phone book. Systems commonly have controls to adjust audio volume, picture contrast, etc.

Many systems allow an easy way to transfer files among participants. Some systems allow application sharing, which enables a participant to take control of an application running on another participant's computer. The usefulness of application sharing is often demonstrated with an example of sharing a spreadsheet or word processor program to facilitate group collaboration. Desktop videoconferencing systems use different control and transport protocols for different communication platforms. Some of these platforms are described in the following subsections.

2.2.1 POTS Conferencing

Plain Old Telephone Service (POTS) is the basic telephone service that provides access to the public switched telephone network (PSTN). This service is widely available but has low bandwidth (typical modem speeds are 28.8 kb/s or 33.6 kb/s). Few desktop videoconferencing products operate at these rates. H.324 [31] is the standard approved by the ITU-T for videoconferencing over POTS. It involves a set of protocols that deal with terminal capability negotiation, media encoding, and multiplexing. H.245 [32] is the chosen protocol for exchanging control information between conference participants, channel signaling and flow control. H.223 [34] is the protocol that multiplexes media components from different channels into a single packet and sends it over the POTS line. The preferred codecs for H.324 conferencing are H.263 [33] for video and G.723.1 [27] for audio.

2.2.2 ISDN Conferencing

ISDN (Integrated Services Digital Network) is a digital service. There are two access rates defined for ISDN, Basic Rate Interface (BRI) and Primary Rate Interface (PRI). Basic Rate Interface provides two data channels of 64 kb/s (B-channels) and one signaling channel of 16 kb/s (D-channel). Many desktop videoconferencing products in the market utilize ISDN BRI. However, problems exist with access to ISDN because it is

not available in all areas. Even when it is available, it can be non-trivial to configure correctly. Primary Rate Interface provides 23 or 30 B channels of 64 kb/s and one D channel of 64 kb/s. ISDN PRI is expensive and therefore not really applicable for desktop videoconferencing.

Because ISDN channels offer 64 kb/s of bandwidth, standards and compression algorithms have been designed around that number. ITU-T recommendation, H.320 [29], describes the procedures for multimedia conferencing over ISDN.

2.2.3 Internet conferencing

LANs provide connectivity among a local community. The Internet connects LANs to other LANs. The protocol developed to interconnect various networks is called the Internet Protocol (IP). Two transport layer protocols were developed with IP, TCP and UDP. TCP (Transmission Control Protocol) provides a reliable end-to-end service by using error recovery and reordering. UDP (user datagram protocol) is an unreliable service making no attempt at error recovery [4].

Desktop videoconferencing applications that operate over the Internet primarily use UDP for video and audio data transmission. TCP is not practical because of its error recovery mechanism. If lost packets were retransmitted, they would arrive too late to be of any use.

TCP is used by videoconferencing applications for other data that is not time sensitive such as protocol information. Both LAN and Internet conferencing use procedures of ITU-T recommendation H.323 that is described in section 2.4.

2.3 The Infrastructure

This study focuses on Internet conferencing. In this section we overview the protocol layers that carry traffic components. Networking protocols are normally developed in layers, with each layer responsible for a different facet of communication. The protocol suite of interest to our study is TCP/IP. TCP/IP is considered a 4-layer system [64] as shown in Figure 3.

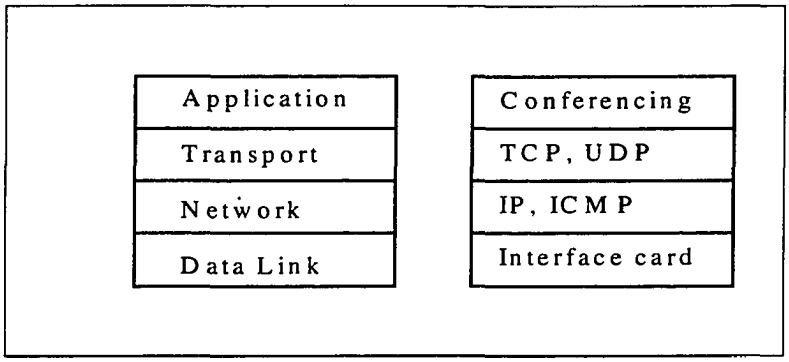

Figure 3. The four layers of the TCP/IP protocol suite

The application layer handles the details of user interface, video and audio encoding, packet preparation, etc. in a conferencing application. The transport layer provides a flow of data between two hosts for the application layer above and a good example is UDP, TCP and RTP protocols. The network layer handles the movement of packets around the network and IP routing protocol falls into this category. The data-link layer handles sending and receiving IP datagrams. It also handles hardware interfacing with the physical layer. A good example for the data-link layer is the PPP protocol that interfaces network drivers on a local host to the Internet service provider gateway to the Internet. The protocol hierarchy used for conferencing over the Internet using H.323 set of standards is shown in Figure 4. These protocols are described in the following subsections.

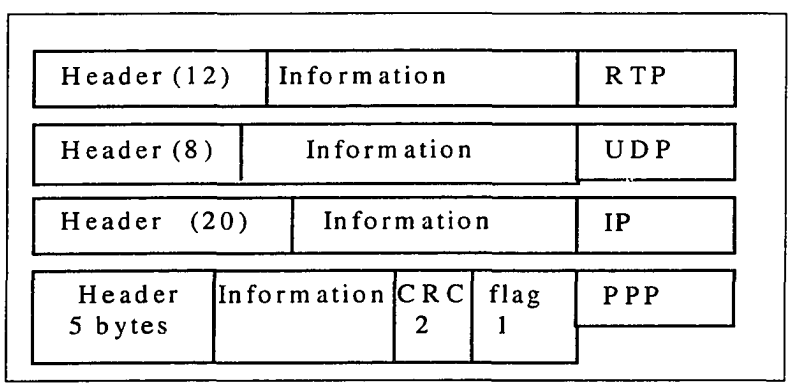

Figure 4 Multimedia Conferencing Protocol layers

2.3.1 The MTU Size

The maximum transmission unit (MTU) is defined as the maximum limit on the size of a data-link layer frame. A typical MTU value is 1500 for Ethernet frame encapsulation. As discussed in 2.3.3 if the datagram sent by IP is larger than the link-layer MTU, IP layer performs fragmentation, breaking the datagram up into smaller pieces, so that each fragment is smaller than the MTU size. When two hosts are communicating with each other across multiple networks, each inter-network link can have a different MTU size. The path MTU is defined as the smallest MTU of any data link that packets traverse between the two hosts. The path MTU between two hosts need not be constant. It depends on the route being used at any time. Also routing need not be symmetric (the route from host A to host B may not be the same as from B to A). Hence, the path MTU may not be the same in the two directions. Methods to discover the path MTU were studied in [53].

Table 2 lists some typical MTU values taken from [53].

Table 2 Typical maximum transmission units (MTUs)

Network	MTU (bytes)
Hyperchannel	65535
16 Mb/s token ring (IBM)	17914
4 Mb/s token ring (IEEE 802.5)	4464
Ethernet	1500
IEEE 802.3/802.2	1492
X.25	576
Point-to-point (low delay)	296

2.3.2 The RTP Protocol

RTP [49] is a real-time protocol for multiplexing multimedia components participating in a conference. It does not offer any form of reliability nor does it ensure real-time delivery. The protocol is real-time in the sense that it provides functionality suited for carrying real-time content, e.g., a timestamp and control mechanisms for synchronizing different streams with different timing properties.

RTP assigns timestamps for multimedia packets such that the timing order is restored at playback side. It also assigns sequence numbers in order to be able to detect losses. The RTP packet header size is 12 bytes. It carries information about payload type, source identifier, time stamp, and sequence number. Although RTP is an unreliable protocol, it provides hooks for adding reliability through fault tolerance, congestion reactive algorithms, and other reliability schemes. Reliable protocols such as TCP are inadequate for handling real-time traffic since by the time lost packets are discovered, one or more round-trip delays would have already elapsed. Such waiting periods may be extremely disturbing to the media stream audibility. Further, reliable protocols may have huge headers that render bandwidth utilization extremely poor. Bandwidth in general is an expensive resource especially for users interfacing to the Internet via dialup modems through ISPs.

2.3.3 The UDP protocol

User datagram protocol (UDP) is a simple transport layer protocol. The simplicity is because each UDP packet written by an application produces exactly one IP datagram to be sent. UDP provides no reliability. The application needs to worry about the size of the resulting IP datagram to ensure it does not exceed the maximum MTU (maximum transmission unit) size discussed in section in 2.3.1. If the size of an IP packet exceeds the maximum MTU size, the packet is fragmented. In order to avoid fragmentation overhead, the packet should be less than the smallest MTU size of all networks in the path MTU between the sender and the receiver. Fragmentation can take place either at the

originating sender or at an intermediate router. When an IP datagram is fragmented, it is reassembled at the final destination. The fragmentation and re-assembly process is transparent to the transport protocol layer. One reason for avoiding fragmentation process is that if a fragment is lost, the whole packet has to be retransmitted. UDP is not capable of error recovery by retransmission and hence the whole packet will be lost. It is thus undesirable to send large UDP packets over the network.

Figure 5 shows the structure of a UDP packet header. The UDP header is composed of eight bytes. It carries sender and receiver port numbers. UDP uses the destination port number to demultiplex incoming data from IP. The header also has a packet length to indicate the length of the header and data of a UDP packet. Finally, the 2-bytes checksum covers header and data of a UDP packet.

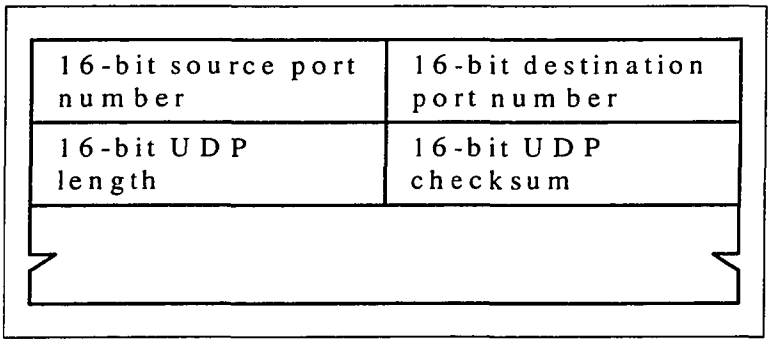

Figure 5. UDP header

2.3.4 The IP protocol

A UDP packet is transported to the IP layer to be routed on the network. Each IP datagram consults a routing table. Information about destination IP address, next hop router IP address (for an indirect route), or local interface IP address (for a direct route) and a pointer to a local interface to use is stored in the routing table. A search for the routing table is made for every IP datagram that the system generates or forwards. Several searching mechanisms and discovery messages were developed for routing tables. A good discussion on the topic can be found in [64].

IP does not maintain any state information about successive datagrams. Each datagram is handled independently from all other datagrams. This means that IP datagrams can be delivered out of order. The IP header is 20 bytes. It contains information about header length, total length, fragmentation, duration of a packet in the network, source and destination IP address.

2.3.5 The PPP protocol

This is the most popular protocol used by Internet service providers to connect their customers to the Internet via serial modems. The format of the PPP protocol [61] is similar to ISO HDLC standard data link control protocol. The protocol header contains information about the payload protocol type, CRC to detect errors in the frame, and a flag for detecting frame boundaries.

2.4 Internet Conferencing Standards

The ITU-T developed a set of standards for conferencing over local area networks. These standards are aggregated under a mother standard umbrella termed H.323 Recommendation. Recommendation H.323 [29] describes terminals, equipment, and services for multimedia communication over local area networks, which do not provide a guaranteed quality of service (QoS) including the Internet. H.323 terminals and equipment may carry real-time voice, video, data or any combination. H.323 procedures encompasses a series of other protocols such as H.225 [35] (setup and media multiplexing), H.245 [32] (logical channel signaling procedures), H.261 [37] and H.263 [33] video codecs, and G.711 [36], and G.723.1 [27] audio codecs. Figure 6 shows the various components of an H.323 terminal.

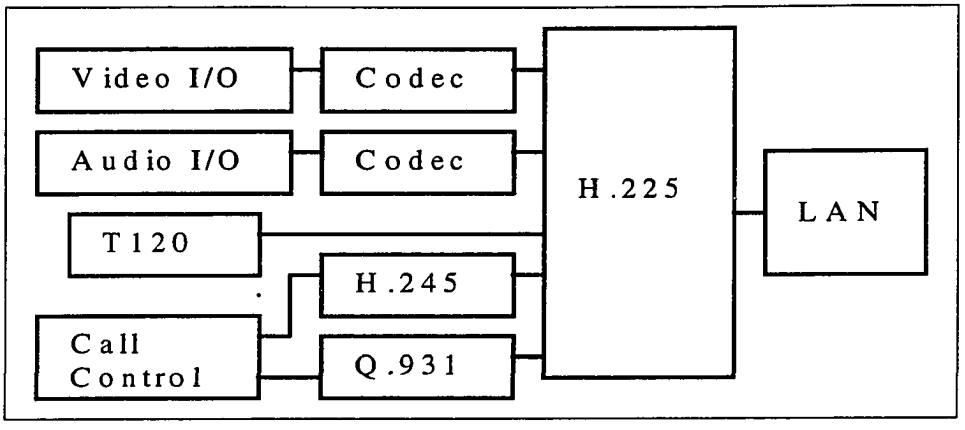

Figure 6 H.323 terminal architecture

2.4.1 Call Control

H.323 terminals use H.245 and Q.931 control messages to signal the beginning and termination of a conferencing session as well as to provide out of band flow and mode control signaling. An H.323 terminal uses Q.931 signals for session alerting, setup and bandwidth reservation. H.245 procedures are used for negotiating terminal capabilities, master/slave determination, media logical channels, mode requests, and other session flow control including custom and proprietary features between H.323 terminals. Both video and audio logical channels are unidirectional while a data channel is bi-directional .

The control channel is established over TCP/IP to make it reliable. Any loss in control data is unacceptable and the TCP retransmit engine handles it. The packetization used to encapsulate control information should conform to Recommendation H.225 [35].

2.4.2 Media Streaming

Media streams are packetized and transported using H.225 procedures. The protocol used for packetizing all media streams is the real-time protocol (RTP). Media packets use UDP as the selected transport layer protocol because of its low overhead. However, no reliability mechanisms are provided for audio and video streaming.

Analog audio is captured via an input device and passed to an audio codec (e.g. G.723.1) to be digitized and compressed. Each digitized packet is then passed to the RTP module to be modulated with a time stamp, a sequence number and a source identifier. The RTP packet is then passed to the network socket interface where UDP, IP, and PPP information are added. Figure 7 shows audio stream architecture and audio stream flow from its analog format until it reaches the Internet in an H.323 conferencing context.

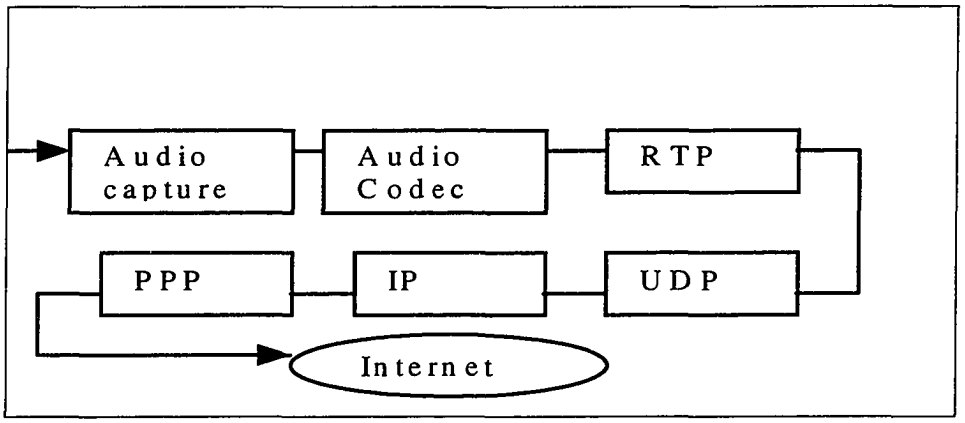

Figure 7. Audio stream control flow

Video streaming is similar to audio. Video is captured from its analog format via a video capture card and then passed to a video codec (e.g., H.263) to be digitized and compressed. The digitized video packet is then fed into a fragmentation module that divides the large video packet into small fragments. This process prevents sending video packets larger than the MTU size of the data link layer. Video fragments are then fed into the RTP module to add time stamps, source identification and sequence numbers. Multiple fragments belonging to the same packet shall have the same time stamp. Video RTP packets are then passed to network socket interface where UDP, IP and PPP headers are added before being dispatched to the Internet. Figure 8 shows the control flow of video from its analog format until it reaches the Internet in the context of H.323 conferencing.

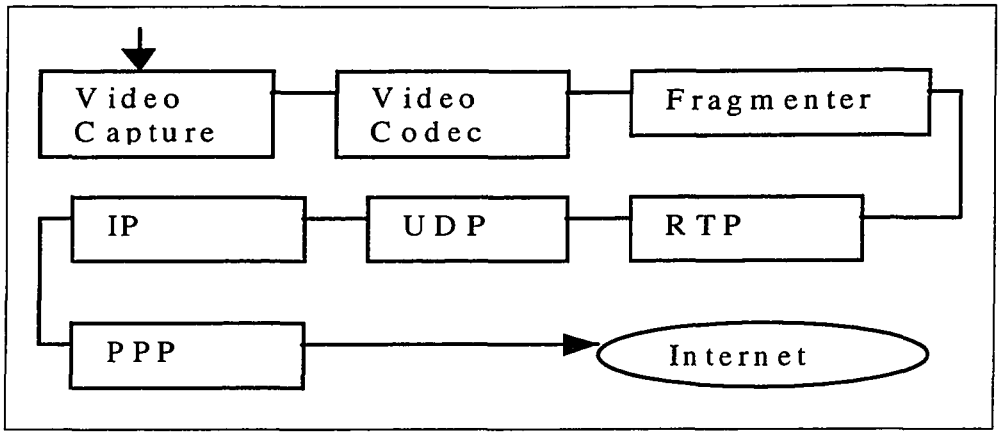

Figure 8. Video stream control flow

Real-time data such as file transfer, chat, interactive whiteboard application, and application data sharing use T.120 protocol. T.120 is a reliable protocol that ensures accurate delivery of data content but provides no guarantee for real-time constraints.

2.5 Concluding Remarks

Desktop conferencing and IP telephony is expected to evolve tremendously in the next couple of years. The technology allows excellent compression rates and quality for media streams. The ITU-T H.323 procedures represent the enabling conferencing standard for this technology. It is granted a lot of focus by different study groups to prefect it for all IP conferencing and telephony purposes. The current infrastructure of platforms and transport protocols encourages more companies to invest in desktop conferencing development and deployment.

The current most preferred codecs available to home users conferencing terminal are ITU-T recommendations H.263, and G.723.1 for video and audio respectively. These codecs can perform well over a narrow bandwidth connection, which is typically less than 28.8 kb/s for home users connecting via modems to their ISPs.

3. Traffic Characterization

3.1 Introduction

A study of multimedia performance over the Internet requires accurate representation of the workload model. Measurements of multimedia conferencing applications provide a snapshot of real workloads. This chapter presents a characterization for video and audio traffic transported over the Internet by video conferencing applications following the H.323 set of standards defined by the ITU.

In this chapter, our main objective is to study multimedia components generation and packetization at the host. We are not concerned with the Internet infrastructure or store and forward technology deployed. The traces collected in this study are done at the network edges. They provide information about packet length, inter-arrival time, jitter, protocol overhead, and burstiness. The collected traces also represent local induced effects on the conferencing traffic.

3.2 Previous Work

Researchers have long used traffic measurements to gain insight on network behavior. Kleinrock was one of the early pioneers to address network traffic modeling and measurement even before the foundation of the Internet [45], [46], [47], and [48]. In the mid-1980's, Marshall and Morgan [52] measured characteristics of local-area networks that supported hosts accessed by terminals and diskless workstations. Jain and Routhier [40] derived a new model of traffic, the packet train, based on local-area network observations. Many studies addressed the characteristics of audio, video and data traffic in real-time and non-real-time environments. Broadband research contributed significantly to the characterization of traffic models [17].

The Modulated Markov model [23] is a popular model for interactive multimedia sources. Various models emerged from that model. The Burst-Silence model is quite realistic for audio and interactive data sources . To model this ON/OFF periodic process,

both ON and OFF period traffic are examined. Figure 9 shows a two-state Markov process model. This model represents the ON and OFF states of a source. During ON State, a source periodically sends a packet with probability P_{ON}. The probability of sending a cell in OFF state is P_{OFF}. Figure 9 also illustrates the sending sequence in both ON and OFF states. The source stays in the ON or OFF states for a duration approximated by a Poisson distribution. The mean time of the ON and OFF states are $1/\alpha$ and $1/\beta$ respectively. Hence, the transition rates from ON to OFF State and from OFF to ON are α and β, respectively as shown in Figure 9. The ON/OFF model is used for modeling voice sources. The Two-State Markov process is used to represent the talk spurt and silent states. The source in Burst State emits cells during a constant cell interval T_B. No cells are emitted in the silent state. T_B is between 5 ms and 25 ms according to the encoding rate [60].

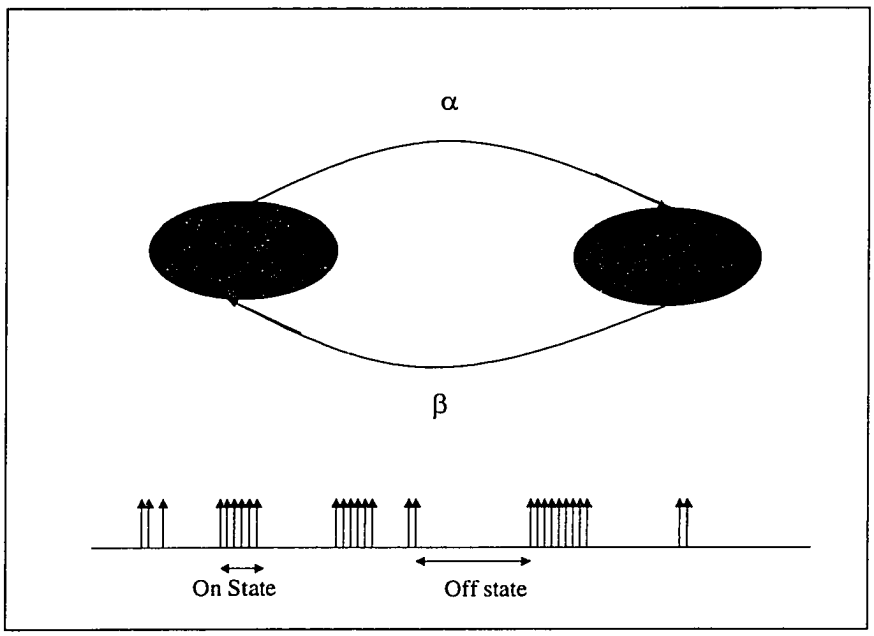

Figure 9. Two State Markov Process

For N multiplexed voice sources, an N state Markov process is a good approximation. In this model, the data rate changes between N rates. The rate increment A and the transition

rate α and β are chosen to match the mean, variance, and auto-covariance function of the multiplexed sources. Figure 10 shows N states Markov process that represents N multiplexed voice sources.

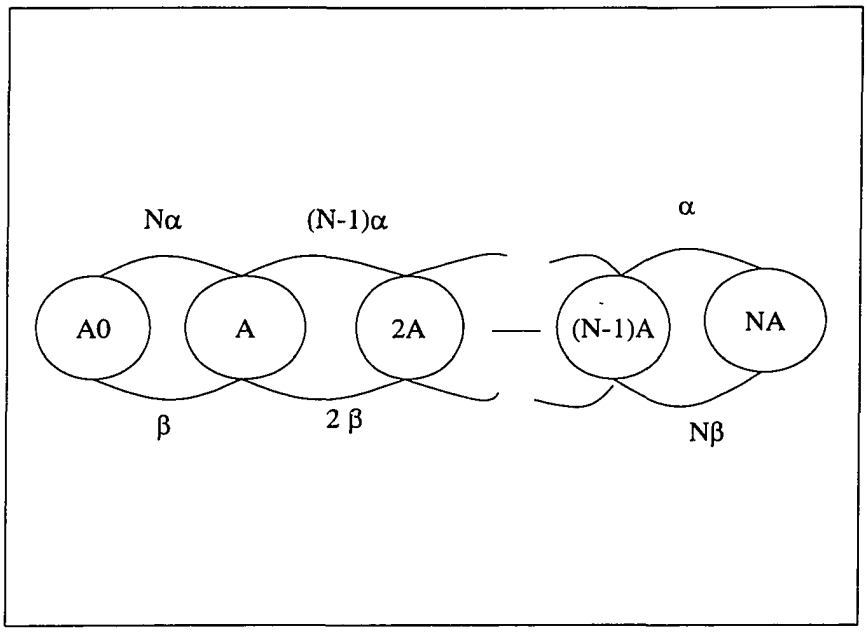

Figure 10. N states Markov process

A two-dimensional Markov process can be used for modeling video traffic, such as, broadcast television, video conferencing and video telephone. The $N_l * N_2$ Markov model is shown in Figure 11. The data rates of the multiplexed sources may alternate between fixed-rate levels. There are two data rates that constitute the main building components of the model: a high rate depicted as A_h, and a low rate depicted as A_l. Different combinations of these two rates produce a wide range of discrete data rates. The maximum data rate that can be achieved is $N_1 * A_l + N_2 * A_h$.

Those models proved to be quite efficient in capturing multimedia traffic, yet practical experience in some cases proved otherwise. One reason is that compression and encoding techniques research and development are usually ahead of traffic models. New video and

audio codecs are developed and deployed in the market each year by industrial firms. These codecs may exhibit different traffic characteristics than those captured by traffic models. Real workloads of traffic sources usually provide accurate assessment of the network problem under investigation.

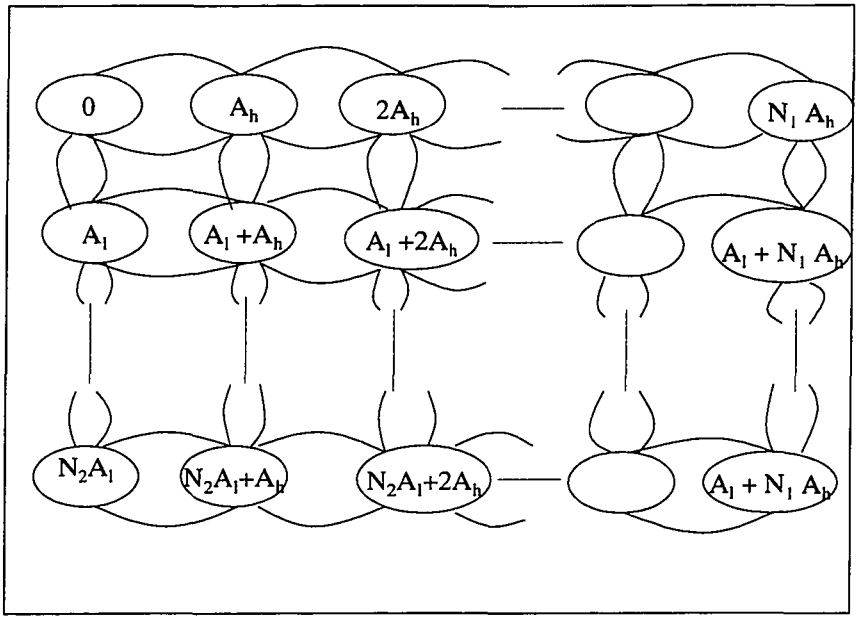

Figure 11. A Two-dimensional Markov process

In this study, we rely on data collected from the traffic source directly as opposed to a particular theoretical model.

3.3 Measurement Methodology

This study addresses the characterization of multimedia traffic flowing through the Internet at the transport layer. We measure traffic at the sending and the receiving host. Measurements at the sending host provide an understanding of local induced effects on the multimedia traffic mix. Many of these effects are primarily due to operating system limitations, bandwidth sharing or lack of accurate local performance qualifiers for the choice of traffic parameters. These local effects can contribute significantly to traffic delay, jitter, or abuse of network bandwidth and congestion. On the other hand,

measurements of the traffic at the receiving host provide an insight into the Internet modulation effect on the traffic mix. The Internet effects are manifested as traffic loss, delay, and jitter.

In this chapter, we address the various tradeoffs involved in generating audio and video packets. We focus our study on audio and video codecs defined in the G.723.1 specification and the H.263 specification respectively. Although the mandatory codecs for H.323 standard are G.711 and H.261, most of the H.323 terminals that connect to the Internet via modems through Internet Service Providers use G.723.1 and H.263 codecs as their preferred conferencing codecs. G.723.1 has lower bit-rate (5.3/6.3 kb/s) than G.711 (64 kb/s). Hence, it is more suitable for transmission over low bandwidth modems. H.263 has better quality than H.261 and only backward compatibility with other terminals (e.g. ISDN) mandates the inclusion of H.261 codec in an H.323 terminal.

We conducted the measurements using an Intel H.323 stack. Intel stack captures more than 80% of the OEM (Original Equipment Manufacturer) and Web download market. The Intel stack is capable of generating G.723.1 audio at both rates, variable bit-rate H.263 video, and T.120 data. It is expected that the collected traces provide an accurate behavior picture for the majority of the H.323 terminals in the market.

It is important to capture the worst case behavior of the network such that the remedy solutions proposed can have wide applicability. It is also important to observe the most congested time of the day over the Internet. We used a variety of comparable Internet service providers for establishing conferencing sessions.

The collected measurements are applied to a traffic scheduler. The scheduler accepts audio and video packets, and schedules them for transmission using the scheduling algorithm of interest. During the simulation run, measurements of packet sizes, interarrival time, latency, and jitter are computed. The scheduler is depicted in Figure 12.

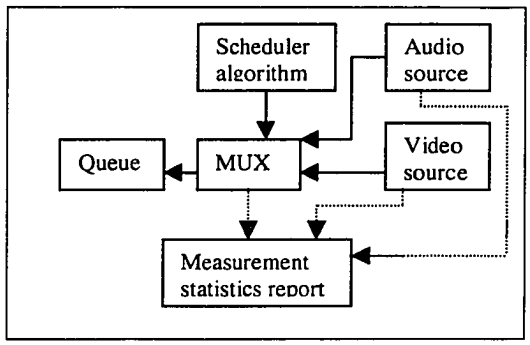

Figure 12. Architecture of the traffic scheduler simulator

The scheduler is a multithreaded application. Each traffic source has its own thread. The scheduler also has its separate thread. Additional traffic sources such as T.120 data or other video or audio codecs can be easily hooked to the scheduler.

3.4 Trace Context and Format

3.4.1 Audio

G.723.1 [27] is a dual rate speech coding standard, which operates in low bit rate while maintaining high perceptual quality. G.723.1 is intended to standardize the audio portion of videoconferencing/telephony over public telephone (POTS) networks and is part of the overall ITU H.324 [31] standard. It is also recommended as the preferred speech codec for ITU H.323 [29] conferencing standard over the Internet when the access link to the Internet has limited bandwidth (e.g., ISP connection using a modem). H.323 recommendation is supported by Microsoft, Intel and many other leading companies as the standard for communication on the Internet.

The G.723 codec has two bit rates associated with it. These are 5.3 and 6.3 kb/s. The higher bit rate has better quality. Both rates are a mandatory part of the encoder and decoder. It is possible to switch between the two rates "on the fly."

A quality test called Mean Opinion Score (MOS) is used to rate the quality of different speech coders. On this scale, a score of 4.0 is considered "toll" quality, the quality of

speech heard through a normal telephone line. In tests performed by a company in Texas called Dynastat Inc., G.723.1 received a rating of 3.98 [63] - only 2 hundredths of a point away from full toll quality. Thus, G.723.1 can enable voice communications through the Internet as well as in other applications requiring compression with voice quality equal to that experienced over a regular telephone.

G.723.1 encoder provides one frame of audio every 30 ms. The audio frame size is 20 bytes of PCM samples for the low rate (5.3 kb/s) and 24 bytes for the high rate (6.3 kb/s). Each application decides the number of frames per audio packet before the conference starts. These values are negotiated during the call establishment and the least number prevails. Both applications are forced to use this number as the maximum number of frames incorporated in an audio packet.

Increasing the number of frames per audio packet improves the bandwidth utilization and decreases network packet overhead. On the other hand, it introduces additional delay to the audio playback since a packet has to wait for all the audio frames to be accumulated before sending it across. This host-induced delay can be even more harmful especially with the timeliness requirement of real-time audio telephony.

Each audio packet will have an RTP header (12 bytes) that carries time-stamps, payload type, and synchronization source identifier. In addition, a UDP header is needed to carry UDP information about the packet (8 bytes). Then an IP header of 20 bytes is also needed to carry routing information. Finally, since we are strictly constrained in bandwidth as we are expected to connect to the Internet via Internet Service Providers, an additional 5 bytes[1] of PPP header is also required. Table 3 shows the number of frames per packet for G.723.1 (5.3 kb/s) audio against induced latency, and packet header overhead.

Table 4 shows the number of frames per packet for G.723.1 (6.3 kb/s) as a function in the induced latency, and packet header overhead. It is observed from both tables that as we

---

[1] Most implementations negotiate to omit the constant *address* and *control* fields and to reduce the size of the protocol field from 2 bytes to 1 byte and hence a total of 5 bytes (1 protocol + 2 CRC + 2 flags)

increase the number of frames, the packet overhead will decrease while host-induced latency will increase. The overhead decrease is explained by more information bytes (audio frames) being included in a packet with a fixed header (RTP+UDP+PPP+IP). The latency increases as the number of frames per packet increases. This is expected since more audio frames require more time to be captured and buffered. Hence, a tradeoff exists between packet overhead and local latency for audio packet transfer. We show in a later section the optimal value for the number of audio frames included in one packet for low and high audio rates.

Table 3 G.723.1 (low bit-rate) audio packet information

No. of Frames	Packet Size (bytes)	Protocol Overhead (%)	Delay (ms)
1	20	69.2	30
2	40	52.9	60
4	80	36	120
6	120	27.3	180
8	160	21.95	240
10	200	18.36	300
12	240	15.79	360
14	280	13.85	420
16	320	12.32	480
18	360	11.11	540
20	400	10.1	600
21	440	9.3	660
22	480	8.6	720

Table 4 G.723.1 (High bit-rate) audio packet information

No. of Frames	Packet Size (bytes)	Protocol Overhead (%)	Latency (ms)
1	24	65	30
2	48	48	60
4	96	32	120
6	144	32	180
8	192	24	240
10	240	16	300
12	288	14	360
14	336	12	420
16	384	10	480
18	432	9.5	540
20	480	8.6	600
22	528	7.8	660
24	576	7.2	720

3.4.2 Video

The H.263 codec [33] is designed for wide range of bit-rates (10kb/s - 2 Mb/s). The codec supports five different resolutions. In addition to QCIF and CIF that were supported by H.261 there is SQCIF, 4CIF, and 16CIF. SQCIF is approximately half the resolution of QCIF. 4CIF and 16CIF are four and sixteen times the resolution of CIF respectively. The support of 4CIF and 16CIF means the codec could then compete with other higher bit-rate video coding standards such as the MPEG standards.

H.263 is a scalable codec, i.e., some performance and error recovery options can be sacrificed for lower bit-rate. The basic codec includes four optional negotiable options included to improve performance: Unrestricted Motion Vectors, Syntax-based Arithmetic Coding, Advance Prediction, and forward and backward frame prediction similar to MPEG called P-B frames.

In the Unrestricted Motion Vectors mode, motion vectors are allowed to point outside the picture. The edge pixels are used as prediction for the "not existing" pixels. A significant gain is achieved if there is movement across the edges of the picture, especially for the smaller picture formats. Additionally, this mode includes an extension of the motion vector range so that larger motion vectors can be used. This is especially useful in case of camera movement.

In the Arithmetic Coding mode, arithmetic coding is used instead of variable length coding. The reconstructed pictures will be the same, but significantly fewer bits will be produced.

In the Advance Prediction mode, overlapped block motion compensation (OBMC) is used for the luminance part of P-pictures explained in the next paragraph. Four 8x8 vectors instead of one 16x16 vector are used for some of the macro-blocks in the picture. The encoder has to decide which type of vectors to use. Four vectors use more bits, but give better prediction. The use of this mode generally gives a considerable improvement.

Finally, A PB-frame consists of two pictures being coded as one unit. The name PB comes from the name of picture types in Recommendation H.262 where there are P-pictures and B-pictures. Thus, a PB-frame consists of one P-picture that is predicted from the previous decoded P-picture and one B-picture which is predicted from both the previous decoded P-picture and the P-picture currently being decoded. The name B picture was chosen because parts of B-pictures may be bidirectionally predicted from the past and future pictures. With this coding option, the picture rate can be increased considerably without increasing the bit-rate much.

A hybrid of inter-picture prediction to utilize temporal redundancy and transform coding of the remaining signal to reduce spatial redundancy is adopted. The decoder has motion compensation capability, allowing optional incorporation of this technique in the codec. Half pixel precision is used for the motion compensation.

It is important to understand how the video source generates video data in order to analyze video-generated traffic. Video pictures are fragmented into multiple video fragments. Each fragment forms a video packet. Each packet is sent across the network after adding RTP, UDP, IP, and PPP headers. Video is different from audio in the sense that the generated bit-rate can be variable while the audio bit-rate is always constant. The ITU standard left this issue for the application to decide whether to use a fixed or variable bit-rate for video [29]. Another important difference between video and audio is that video can be fragmented while audio frames are not. t A video frame can be quite large and cannot fit in a single packet. A Fragmentation module will divide a video frame into multiple fragments. Each video fragment will form a video packet that is sent across the network. There is an upper limit on the maximum fragment size used by the fragmentation process.

A third major difference between audio and video is the burstiness of the video source due to fragmentation, which causes multiple fragments to cluster in a small interval of time. Video variable bit-rate also contributes to this burstiness. This burst of packets can lead to packet loss, latency or at least some amount of jitter that is generally unfavorable.

In this section, we focus on the characterization of generated video traffic after fragmentation and the implied tradeoffs involved when enforcing different maximum fragment sizes for video packets.

Smaller fragment size implies smaller latency while bandwidth efficiency decreases because of larger packet overhead. Yet, larger fragment size implies larger latency while maintaining good bandwidth utilization. As the maximum fragment size limit increases, larger video packets will occur more often. These packets will require more time to be transmitted and hence the video latency increases.

An experiment was conducted to study the video packet sizes (number of bytes per packet) for different maximum fragment limits as generated by our video source. We set the maximum fragment size to four different values (128, 256, 512, and 750 bytes). We ran four experiments, each with a different maximum fragment limit and collected the corresponding video packet sizes generated after the fragmentation module. Figure 13 shows how video packets sizes (number of bytes per packet) are distributed for different maximum fragment sizes. The figure shows that on using a maximum fragment size of 750 bytes, 70% of the video packets had a size in the range of 50-250 bytes. Less than 10% of the total generated video packets were more than 512 bytes. For a fragment size of 512 bytes, 75% of the packet had sizes between 50-250 bytes. Decreasing the fragment size further to 256 bytes, more that 60% of the packets were in the range of 150-250 bytes. For a 128 maximum fragment size, the video packets sizes were equally distributed between 50-100 bytes, 100-150 bytes ranges. In general, the majority of the packets for all maximum fragment sizes were less than 256 bytes. This can imply that the maximum fragment size choice should be in the range of 256 - 512 bytes.

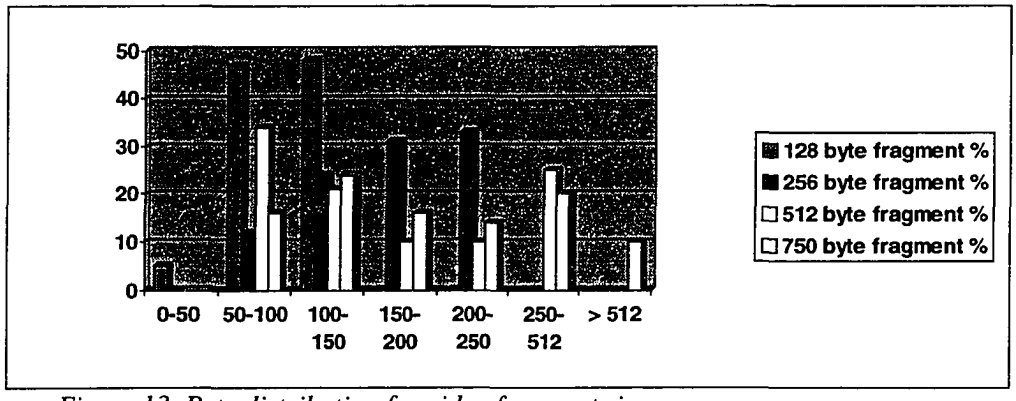

Figure 13. Byte distribution for video fragment sizes

3.5 Determining Traffic Parameters

3.5.1 Audio Traffic

It has been demonstrated that the choice of the number of frames per packet affects both local latency and protocol overhead (i.e., bandwidth utilization). We are primarily concerned with the latency induced by the packet preparation manager that buffers the captured audio frames and synthesizes them into audio packets. This latency is computed before the packet is actually sent on the network, and it only accounts for local capture and packet preparation delay. The choice of the number of frames per audio packet should indicate a compromise between local latency and packet overhead.

Figure 14 shows the percentage packet overhead for both high and low bit-rate G.723.1 audio codec. The packet overhead decreases as the number of frames per packet increases. Intuitively, high bit-rate audio has slightly less overhead compared to low bit-rate packets.

In Figure 15, the number of audio frames per packet is plotted for both rates against normalized packet overhead and latency. The latency is normalized against the maximum latency value exhibited by the largest audio packet (24 audio frames per packet) used in our experiment. The packet overhead is normalized against the maximum overhead exhibited by an audio packet. An audio packet with only one audio frame has the maximum packet overhead and minimum latency as given in Figure 14. As the number of audio frames per packet increases, the packet overhead decreases.

The latency increases as the number of frames per packet increases. This is expected since more audio frames require more time to be captured and buffered. The intersection of the latency and the overhead curves provides a compromised point that has acceptable latency and l affordable packet overhead. The number of frames of choice based on this plot is between seven and eight for both G.723.1 audio bit-rates. This value should be used by terminals that target a compromise between packet efficiency and latency for audio packets.

Sometimes, the audio local latency is reduced at the expense of packet overhead, in order to achieve latencies acceptable to users. In fact, a few applications are willing to sacrifice some protocol overhead for achieving better audio latency. Some H.323 terminals use a value of three or four for the number of frames per audio packet in order to reduce this latency.

Protocol header compression has been an active research area for the past couple of years especially after the maturity of the protocols and standards that drive audio and video streaming over the Internet. Researchers have investigated RTP, IP, UDP and PPP header compression. Jacobson and Casner [10] proposed an approach for compressing RTP, UDP and IP headers to be used over Low-Speed Serial connections to the Internet. Examples for these links include low speed (e.g. 28.8 kb/s) modem connections to the Internet through Internet Service Providers. Their approach seems ideal for better bandwidth utilization of the media streams since the protocol overhead is significantly reduced. Companies such as Intel and Microsoft are currently working on the deployment of protocol header compression approaches inside the network interface infrastructure in order to improve multimedia conferencing over the Internet for terminals connected via low speed links.

Latency and packet overhead plots can again be used to deduce a compromised number of frames per audio packet if header compression schemes are applied at any protocol layer. IPv6 header compression draft [15] provides a good baseline for a discussion about header compression. The techniques used there can reduce the 28 bytes of IP/UDP header to about six bytes. With the remaining five bytes of PPP overhead and 12 bytes for RTP, the total header overhead can be about halved but still exceeds the size of one G.723.1 frame.

Figure 16 shows a plot of the packet overhead versus latency after applying IP/UDP header compression. The curve suggests that the compromised number of audio frames is almost five for low bit rate audio and six for high bit-rate audio.

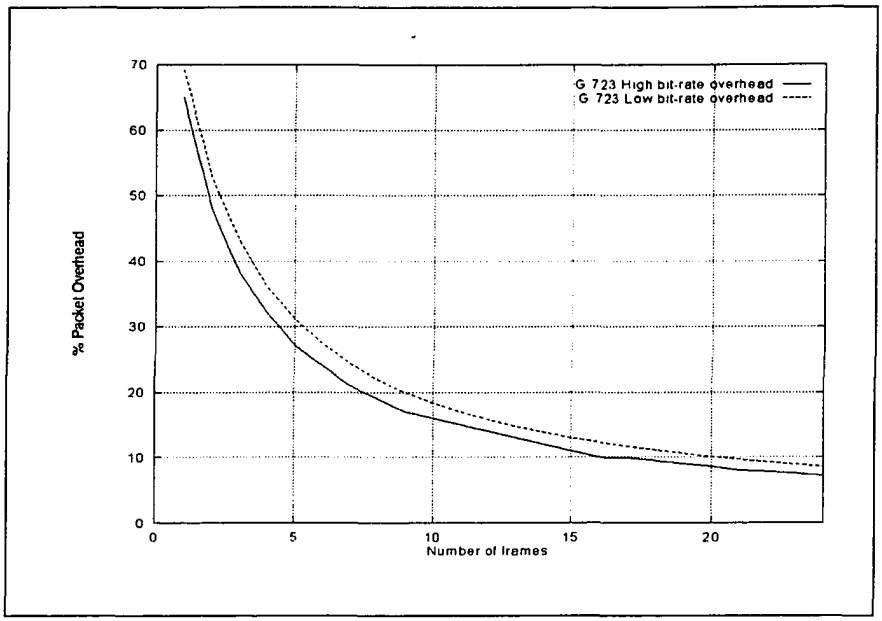

Figure 14. Packet overhead for lo and hi G.723.1 audio

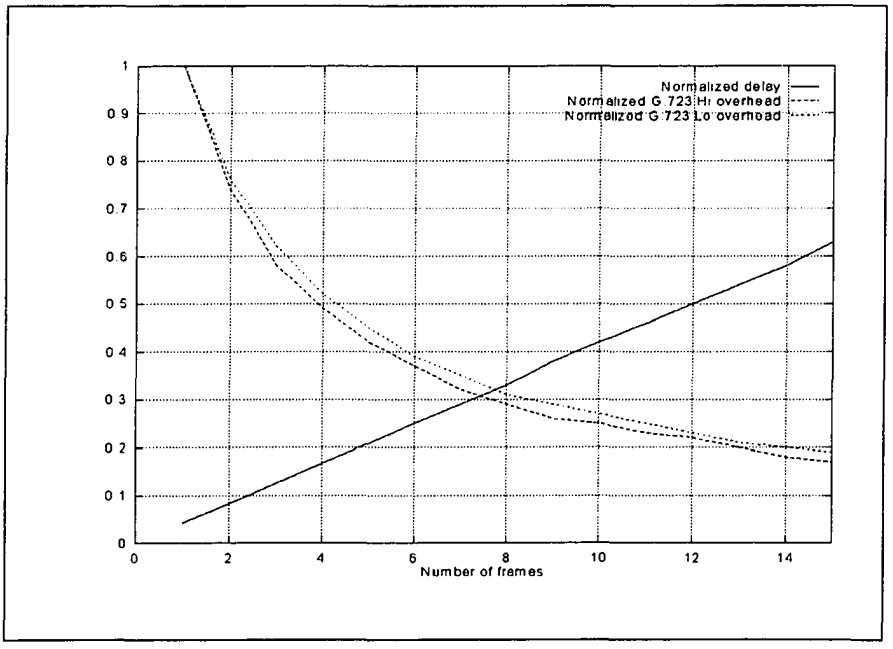

Figure 15. Packet overhead versus latency for uncompressed header

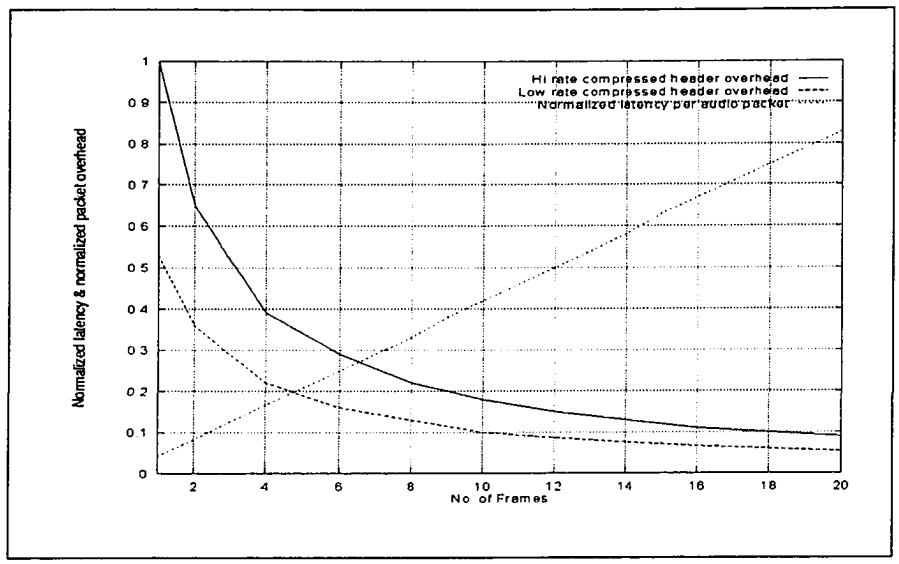

Figure 16. Packet overhead versus latency with compressed IP/UDP headers

3.5.2 Video Traffic

3.5.2.1 Packet size

The choice of an appropriate maximum fragment size is another major decision factor as a video source parameter. The maximum fragment size contributes to the latency, network overhead and generated traffic. As the maximum fragment size increases, larger video packets are generated. Large packets lead to increased latency and less protocol overhead. Reducing the maximum fragment size leads to more packets being generated for the same number of video frames and hence more traffic and vice versa. Further, bandwidth may be affected because large video packets are segmented by the IP layer into several smaller packets if they exceed the MTU size. All fragment sizes selected for this study are chosen to be lower than the maximum MTU size approved by IETF for Ethernet which is 1500 bytes.

Histograms of the video traffic are shown in the following figures. The choice of a maximum fragment size should capture most of the data clustering. In addition, since video packets share bandwidth with audio packets, video packet sizes should not be large

or frequent to the extent that audio packets are blocked behind them for an unfavorable amount of time.

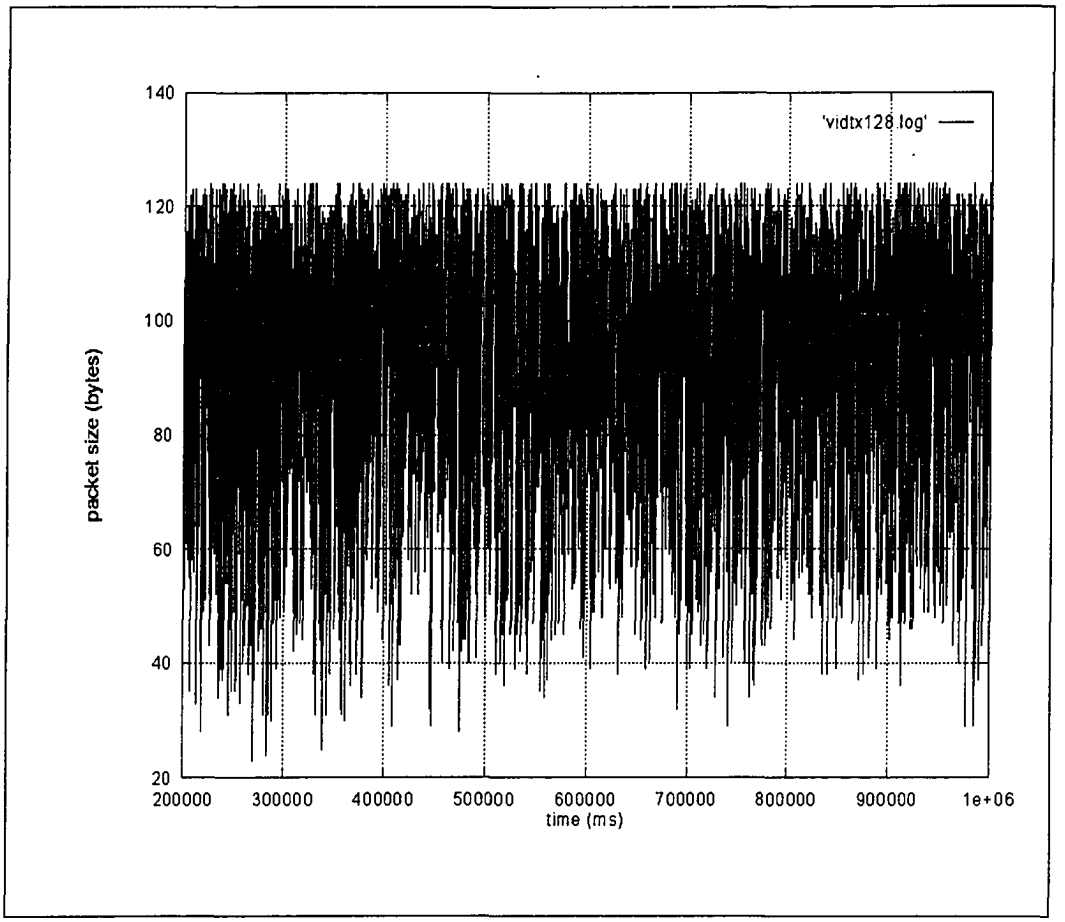

Figure 17. Video Histogram for a max fragment size of 128 bytes

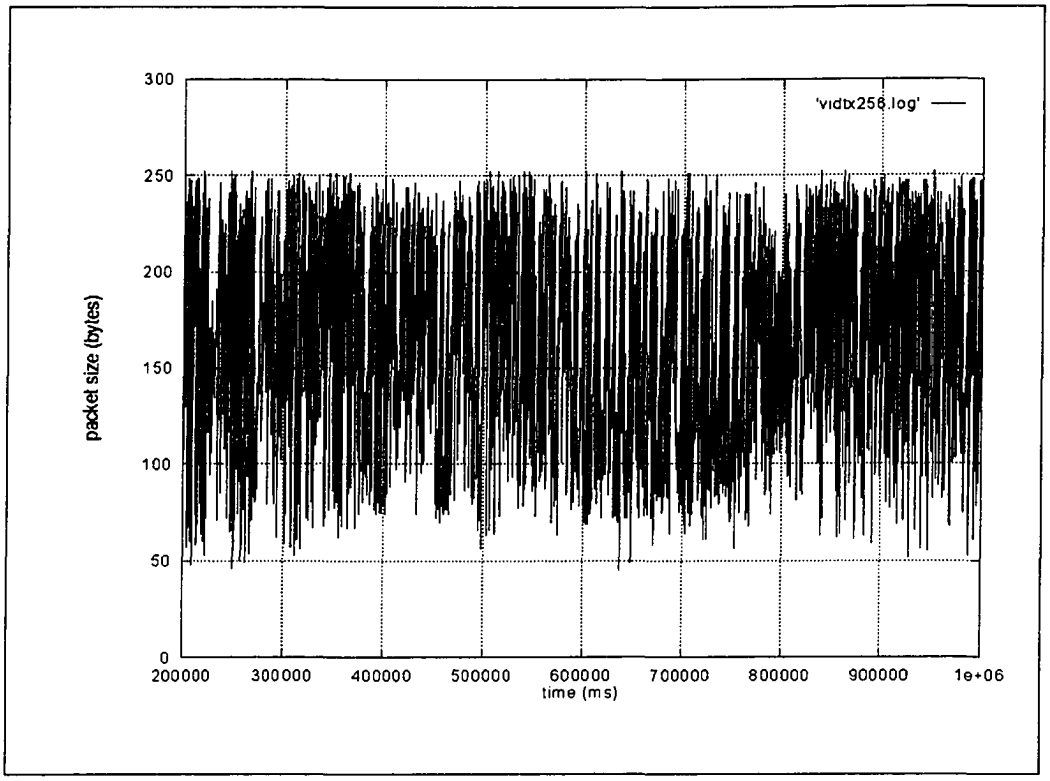

Figure 18. Video Histogram for a max fragment size of 256 bytes

Our choice for video bit-rate is 10 kb/s, which is a typical value used by ISP-based Internet video conferencing terminals. The video stream is QCIF and each trace lasted for 15 minutes. The trace shows the video fragments generated from the H.263 codec after the RTP header is added. Measurements are taken before any UDP or IP packetization.

Figure 17 shows a plot of the video packet sizes against time with the maximum fragment size set to 128 bytes. The packets varied in size from as small as 25 bytes to the maximum value allowed which is 128 bytes. The plot seemed too crowded as more packets are generated to make up for the small value of the maximum fragment size. Figure 18 shows a plot of the video packet size against time where the maximum fragment size is set to 256 bytes. Packets varied in size from as high as 256 bytes to as low as 50 bytes. Figure 18 appears to be less crowded than Figure 17. Figure 19 shows a

plot of the video packet size against time with the maximum fragment size set to 512 bytes. The packet sizes ranged from 75 bytes to 512 bytes. The larger swing in the packet size range allowed for a less crowded histogram compared to Figure 17 and Figure 18.

Finally Figure 20 shows a histogram of the video packets when the maximum fragment size is set to 750 bytes. The video packets had a full swing from 80 bytes to 750 bytes but the majority of the packets were below 500 bytes.

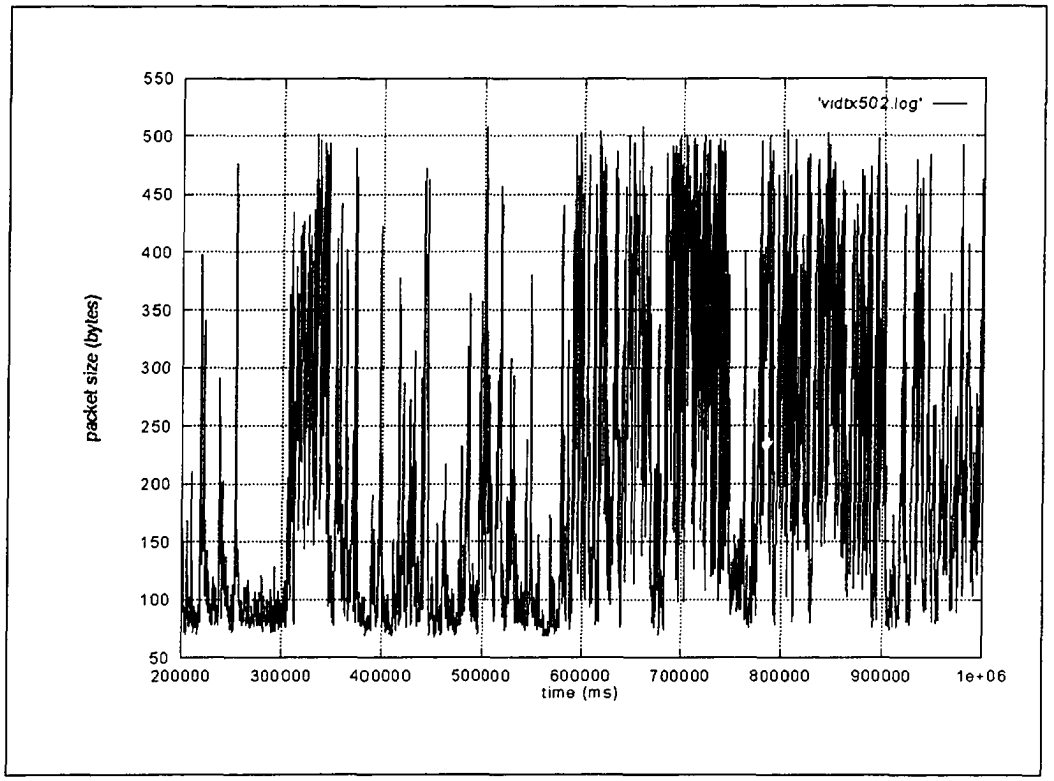

Figure 19. Video Histogram for a max fragment size of 512 bytes

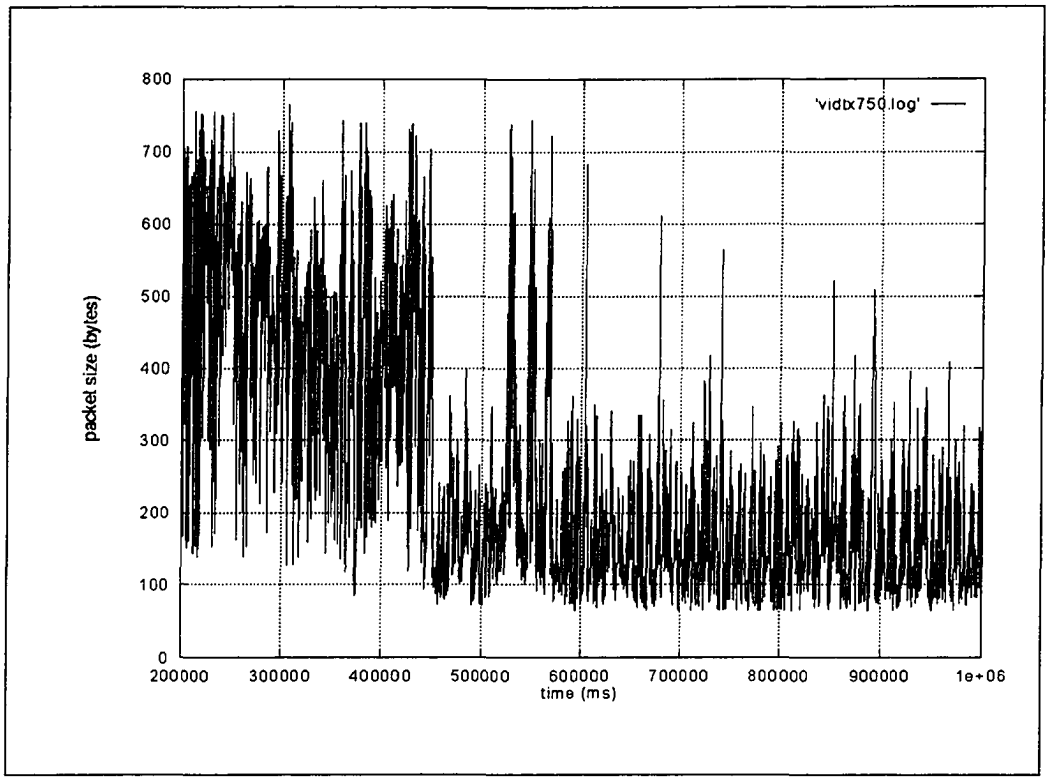

Figure 20. Video Histogram for a max fragment size of 750 bytes

Table 5 shows the maximum fragment size and the corresponding mean, standard deviation and number of generated fragments of the video traffic source.

Table 5 Mean and standard deviation for different fragment sizes

Max Frag	Mean size	Standard Dev	Mean arrival	Frag no.
750	245.378238	179.167286	342.562824	2888
500	191.643815	127.200162	275.026562	3680
256	169.383407	51.358614	224.404204	4464
128	94.581922	21.460085	132.568063	6382

It is shown from Table 5 that the smaller the maximum fragment size, the smaller the mean size of generated fragments and consequently the smaller the delay (smaller inter-arrival time). However, by examining the number of fragments generated for each maximum fragment size in Table 5, we note that as the maximum fragment size decreases, the number of generated fragments increases. This is expected, however, since the number of video frames to be fragmented and formed into packets is almost fixed for all maximum fragment sizes with which we experimented. In addition, as the maximum fragment size decreases, the packet overhead increases, and the bandwidth utilization decreases. Normally, it doesn't take any longer to send and receive fewer but larger video packets than it does to send and receive more but smaller video packets, if the same amount of video data is played. In fact, the extra packet-handling overhead may cause the reverse. However, larger video packets can cause audio latency as the bandwidth is shared, and this is the major concern.

Table 5 shows the mean, maximum and standard deviation of video packet sizes. It also shows the mean interarrival time. These measurements can be used to derive a mathematical model for the H.263 video source. This topic is postponed for future research.

3.5.2.2 Inter-arrival Time and Jitter

Another important measure we can deduce from data provided by these graphs is the mean inter-arrival time. This measure provides an estimate for video packet latency induced locally at the host. We compute the rate of change of the inter-arrival time, which is a measure of video packet jitter. As we decrease the maximum fragment size, it is expected that the number of packets generated within the same interval of time to increase. The measurements for inter-arrival time and jitter for maximum fragment sizes of 128, 256, 500 and 750 bytes are shown in the figures below.

Figure 21 shows a plot of the interarrival time and jitter of video packets against time where the maximum fragment size is set to 128 bytes. It is observed that many packets are generated with variant interarrival time and hence the jitter between packets is

significant. As we increase the maximum fragment size to 256 bytes (as in Figure 22), the jitter is less significant as smaller number of packets with lesser delay variations are generated. Increasing the maximum fragment size further to 512 bytes decreases the jitter significantly as shown in Figure 23. Finally, Figure 24 shows very few jitter occurrences as the maximum fragment increases to 750 bytes.

Generally, it is observed that jitter increases as the maximum fragment size decreases. This observation is explained by the fact that more packets are generated for the smaller maximum fragment size during the same time interval. More packets with a variety of interarrival times result in greater likelihood of inter-arrival time variation and hence jitter.

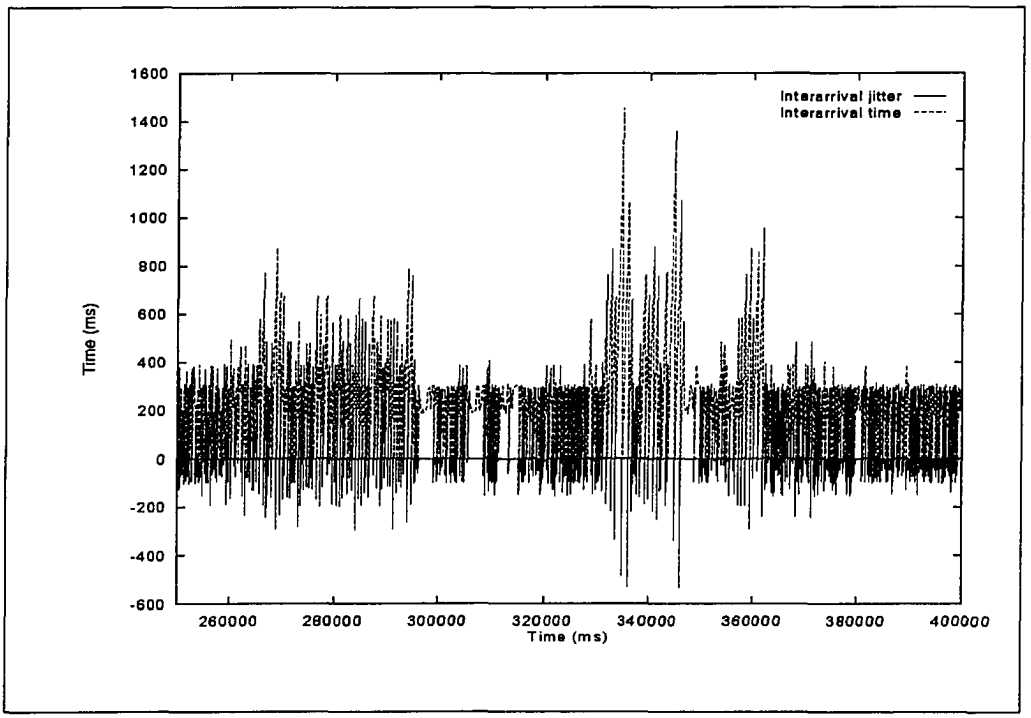

Figure 21. Interarrival time and jitter for video maximum fragment size of 128 bytes

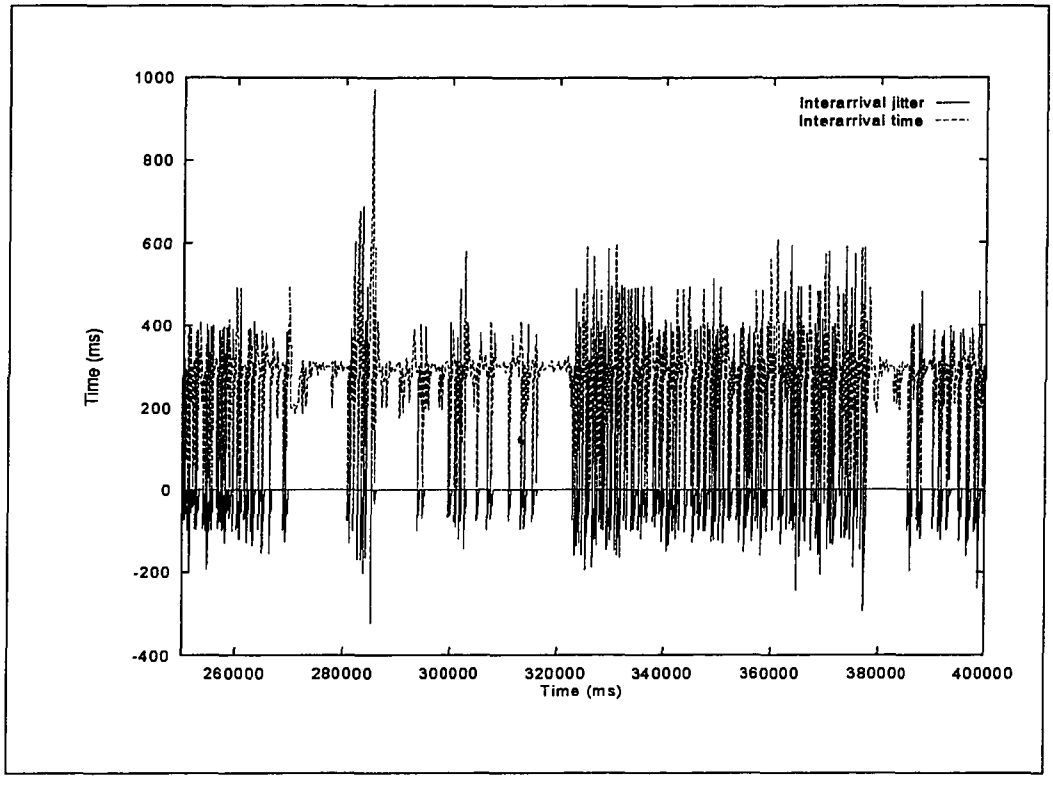

Figure 22. Interarrival time and jitter for video maximum fragment size of 256 bytes

Theoretically, as the maximum fragment size decreases, the sizes of the packets are normalized and jitter should be reduced. However, the increase in jitter in this case is also due to packet handling overhead in the transport stack from the increased number of packets.

An intriguing relation was observed from both the inter-arrival time and jitter curves. Therefore, we attempted to discover a mathematical relation between jitter and interarrival time from the given traces. In all four plots, jitter *J(t)* can be approximated as a function of the inter-arrival time *I(t)* after removing a DC component. The inverse effect can be embedded in a multiplier.

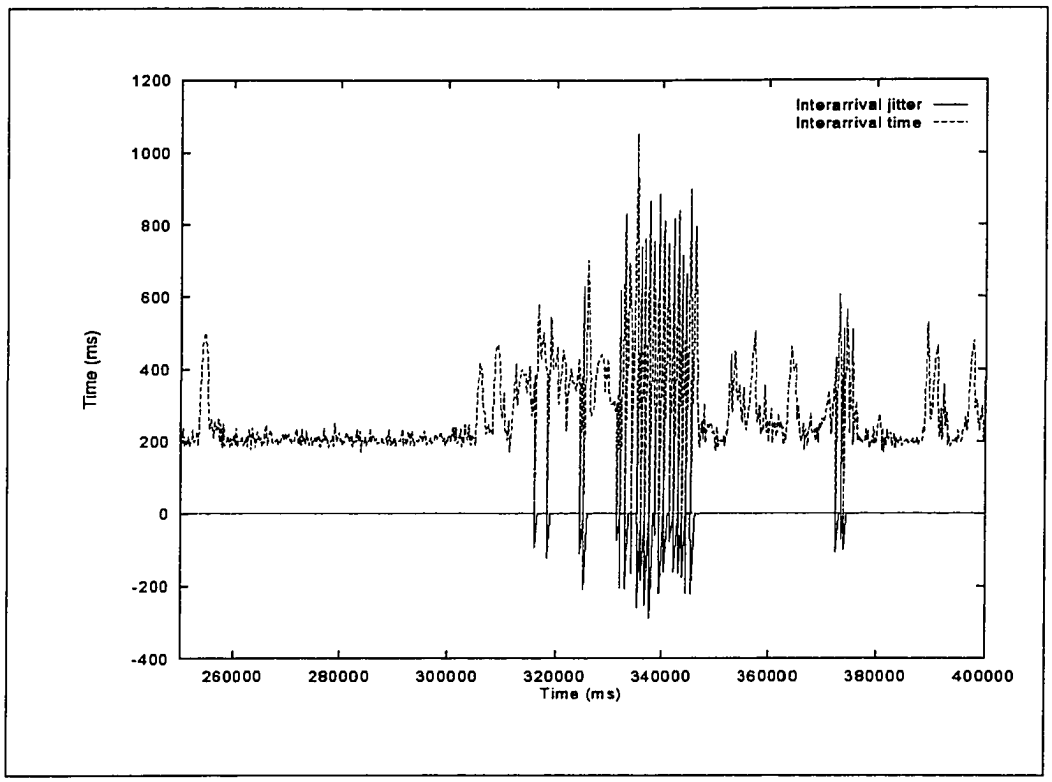

Figure 23. Interarrival time and jitter for video maximum fragment size of 500 bytes

Equation 3-1 shows the relation between inter-arrival time *I(t)*, and the jitter *J(t)*.

$$J(t) = \partial I(t) / \partial t \qquad 3\text{-}1$$

The observation derived from I(t) and J(t) curves can be transformed to the following equation:

$$I(t) = \alpha * J(t) + G \qquad 3\text{-}2$$

Where α, G are constants.

From equations 3-1and 3-2, we can deduce the following equation:

$$I(t) = \alpha * \partial I(t) / \partial t + G \qquad 3\text{-}3$$

Solving for *I(t) using Laplace transform*:

$$I(s) = \alpha s I(s) + G / s$$

$$I(t) = G - G e^{t/\alpha} \qquad 3\text{-}4$$

Hence, the inter-arrival time of video packets is exponential. This result conforms with previous mathematical attempts to model network traffic arrival [23], [40]. This result together with the data in Table 5 can be used to derive a mathematical model for the video traffic generated by an H.263 video codec source. This topic is outside the scope of this study and will be considered in future work.

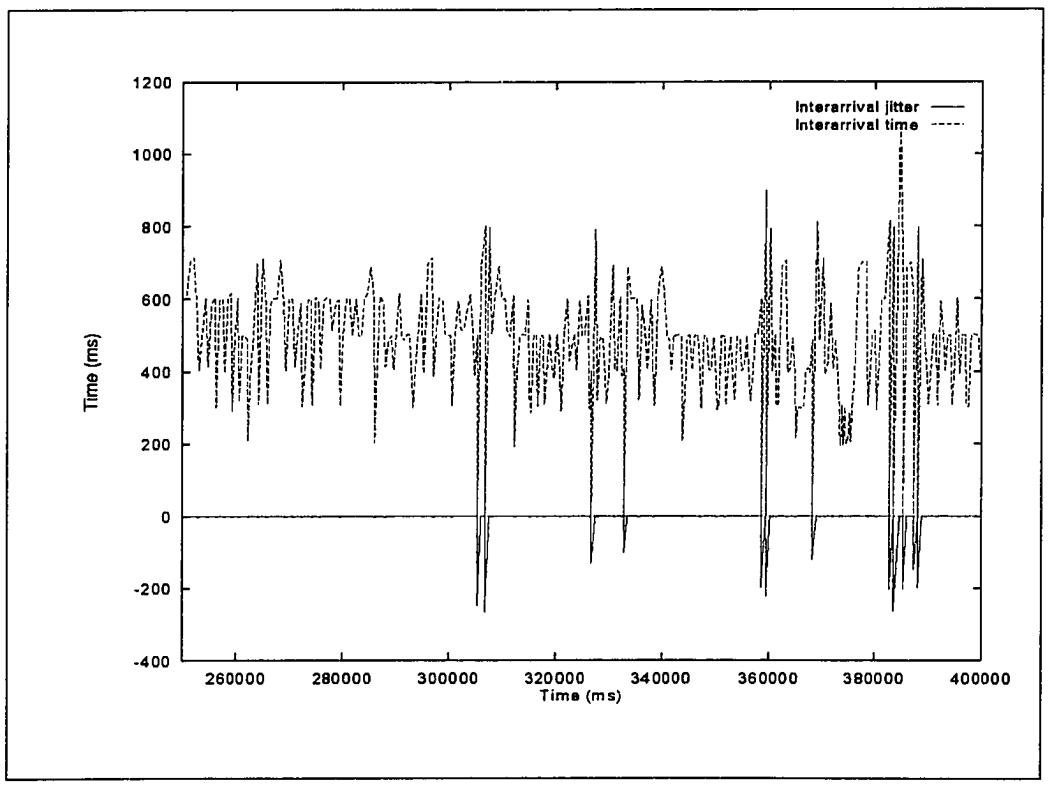

Figure 24. Interarrival time and jitter for video maximum fragment size of 750 bytes

3.6 Video Traffic Synthesis for other Bit-rates

So far, we have been assuming a fixed bit-rate for both Audio and Video traffic. Audio traffic is constant because of the nature of the G.723.1 codec. Video can have variable rate depending on the bit-rate controller that controls the rate of flow. A cumulative average bit-rate of a certain fixed value can be maintained throughout the session or can be varied. A video codec always has a maximum bit-rate limit.

Given a matrix of timestamps and corresponding packet bytes, it is possible to scale the inter-arrival time of packets to produce the required bit-rate according to the following formula:

$$BRS = (\sum_i P_i / T_i) / Br_{new}$$

BRS is the bit-rate scalar to be multiplied by all inter-arrival times between two consecutive packets in a certain interval which is the duration of the given matrix. P_i is the number of bytes of packet with sequence number i. T_i is the inter-arrival time between packet i-1 and packet i. Br_{new} is the new bit-rate to be synthesized. The new matrix M_{new} composed of timestamps, and number of bytes pair can be reproduced for the new bit-rate.

3.7 Concluding Remarks

Real video traffic workload is collected and used as video source input for scheduling algorithms studied in this research. Worst case G.723.1 audio traffic is used as the audio source input. The measurements collected during traffic analysis are packet length, inter-arrival time, jitter, and packet overhead.

Analysis of audio traffic revealed a tradeoff between the audio latency and the network bandwidth utilization. Increasing the number of frames decreases the packet overhead and increases the network utilization. On the other hand increasing the number of frames per packet increases audio latency at the receiver. The number of frame value that achieves a

compromise between latency and packet overhead is approximately 7 frames per packet for uncompressed header. This value will add latency overhead to the audio packet of 210 ms. Analysis of audio packets with compressed UDP and IP headers revealed compromised numbers of frames of 5, and 6 for low and high bit-rate audio respectively. These numbers induce a local delay on the audio packet of 150 ms and 180 ms for low and high bitrates, respectively.

Analysis of the video traffic revealed that mean packet size is less than 250 bytes for all fragment size limits. The standard deviation of packet size for all fragment sizes of choice is less than 180 bytes. The majority of the video packet sizes clustered in the range below 250 bytes. This conclusion suggests that a maximum fragment size between 256 and 512 bytes is an appropriate choice.

It was also shown that as the maximum fragment size increases, the mean inter-arrival time between video packets increases, the number of generated fragments decreases, and the inter-arrival jitter decreases and vice versa. This conclusion is explained by the fact that more packets are generated for the smaller maximum fragment size during the same time interval. . The packet-handling overhead in the transport stack by the RTP fragmentation module is a major contributor to this jitter effect.

4. Multiplexing Video and Audio Packets

In this chapter, we study the interaction of conferencing media components as they are multiplexed on the host to be delivered to a peer terminal. Clearly, without careful manipulation of good scheduling algorithms, there is the possibility of performance loss or degradation locally.

Media components are generated by their respective sources and then funneled through a shared communication medium. Each media component should subscribe to a bandwidth amount sufficient for meeting its quality of service. Lack of appropriate multiplexing algorithms can lead to one or more media components oversubscribing to the shared bandwidth and penalizing other participants.

4.1 Analogy with Real-time Scheduling

Video conferencing is a real-time application where all tasks have to be performed correctly and in a timely fashion. The function of a scheduling algorithm is to determine, for a given set of tasks, whether a schedule for executing the tasks exists such that the timing, precedence, and resource constraints of the tasks are satisfied. Task scheduling in real-time systems can be static or dynamic. A static approach calculates schedules for tasks off-line and it requires complete prior knowledge of tasks' characteristics. A dynamic approach determines schedules for tasks on the fly and allows tasks to be dynamically invoked. Static approaches are inflexible despite their low run-time cost. When new tasks are added to a static system, the entire schedule has to be recalculated. In contrast, dynamic approaches involve higher run-time costs, but, because of the way they are designed, they are flexible and can easily adapt to changes in the environment.

The timing constraints of a task are specified in terms of one or more of the following parameters:

- The arrival time A_i: The arrival time of task *i*.

- The worst case computational time, C_i: The computation time of the task.
- The deadline, D_i: The time by which a task must finish.

In the case of a Audio/Video conferencing, a task is analogous to the arrival of x buffers that need to be delivered to the network. The arrival time is their time-stamp after finishing encoding and protocol field processing and ready to be transmitted. The deadline is the time by which this data will become obsolete because of excessive delay.

Since desktop conferencing shares one communication line as the communication media, the order of dispatching packets is important. Scheduling real-time video, audio, and data is analogous to scheduling a set of non-preemptable periodic tasks over a uniprocessor. The main reason for prohibition of preemption is that each media packet delivery cannot be interrupted until accomplished. The tasks are periodic since a similar pattern of media components repeats itself every interval of time.

This problem has been extensively addressed in the literature and we shall see in the next few sections how some of the proposed algorithms can perform with respect to our AV (audio/video) scheduling problem.

4.2 Multiplexing Audio and Video on First Come First Served Basis

A typical example for this multiplexing scheme is if the scheduler dispatches all media types through a FIFO queue. All packets are serviced in the order they arrive without paying attention to their priority or timeliness. In this example, if the video data is bursty, a train of video packets can be generated during an audio silent (inter-arrival) period. Consequently, audio packets will be blocked for the duration of the video packet train. This duration may be sufficiently long to disturb the continuity of the audio speech and make it incomprehensible.

To illustrate this example we simulated the generation of both media types according to the models we presented in the previous chapter. We have chosen a worst case audio source that continuously generates audio packets of four frames each every 120 ms.

Assume that the resulting packets from both sources are served by a FIFO queue scheduler. We are interested in the number of video bytes that reside between audio packets during periodic silence intervals. If too many video bytes accumulate between two consecutive audio packets, severe delay (and jitter) may be experienced for the blocked audio packet. The number of video bytes interleaving audio packets is assumed to be a performance qualifier for the different scheduling algorithms addressed in this dissertation.

Computing the mean number of bytes from the plot depicts the host extra inter-arrival time penalty added to audio packets in terms of video bytes. Further, knowing the maximum number of video interleaved bytes (VIB) shows the maximum penalty incurred on the inter-arrival time of audio packets that will show at the receiver. We can also deduce the mean audio jitter by computing the rate of change of the interleaved video packets over the experiment interval.

Figure 25 and Figure 26 show a plot of the number of video bytes residing between consecutive audio packets against time using an *H.263* video source of maximum fragment size 256 and 500 respectively. Intuitively, a smaller maximum video fragment size may lead to better audio performance, however, our experience proved otherwise. It is observed that the mean interleaved video bytes between audio packets for a video maximum fragment size of 256 is larger than the mean of the corresponding trace for a maximum fragment size of 500 counter to the intuition. Hence, reducing the video fragment size alone will not help the audio performance. To the contrary, it may lead to irregular spacing in time between audio packets at the receiver, which renders audio playback unintelligible.

The conclusion is that the lack of good scheduling mechanisms can lead to an appreciable amount of unnecessary performance penalty on the host locally before reaching the network. These local effects disturb the quality and the clarity of the audio session and it is obviously desirable to eliminate them. We investigate another scheduling algorithm, "Earliest Deadline First" (EDF), in an attempt to solve the audio local latency problem.

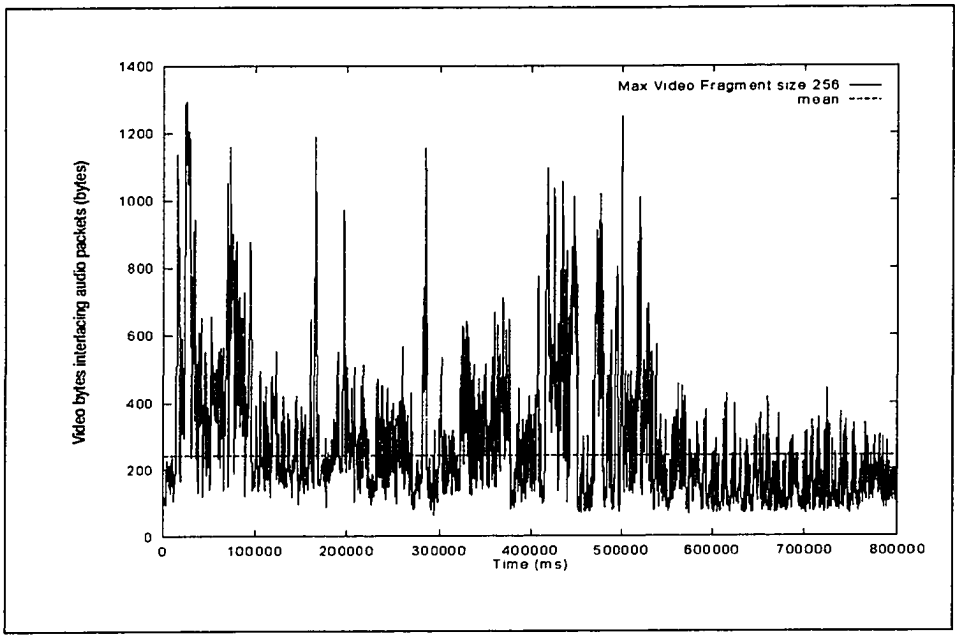

Figure 25 VIB for FCFS (max video fragment size 256)

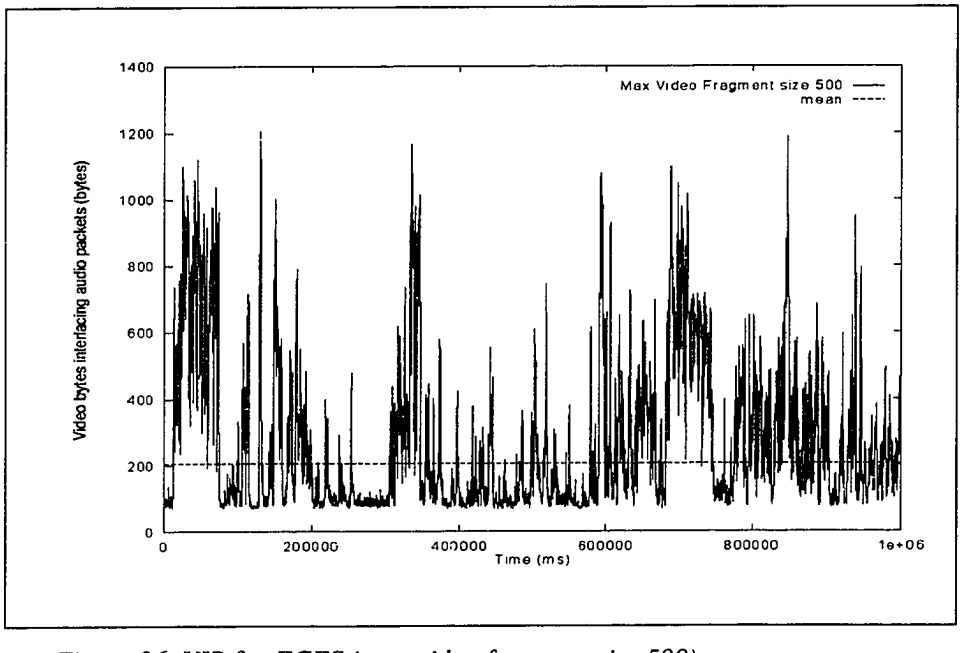

Figure 26. VIB for FCFS (max video fragment size 500)

4.3 Earliest Deadline First

The Earliest Deadline First (EDF) algorithm is applied on the same audio and video samples we used in the previous chapter. We assume worst case audio where a packet of four audio frames arrive every 120 ms. We also assume maximum size for video fragments of 500, and 256 bytes. We measured the number of interleaved video bytes between audio packets. Figure 27 shows an indicator for the audio inter-arrival time based on the number of video bytes in between. EDF showed a slight improvement in audio inter-arrival time, as video bytes were more scattered than the FCFS algorithm. It is shown that the maximum number of video bytes interleaving audio packets is 1200 bytes. This value can be converted into seconds by dividing by the video bit-rate.

We also measured the latency in ms for the whole pattern and the results are shown in Figure 28. Latency is defined as the time where audio or video packets are kept idle in the outgoing queue while the scheduler is busy. It neither takes the encoding time nor the dispatching time into account. The maximum latency was found to be 514 ms.

A latency comparison was conducted with the FCFS scheduler and the result is depicted in Figure 29. It is shown that EDF has a better latency compared to FCFS, although the result is still below expectation and some investigation need to be done. The latency comparison for maximum fragment size of 256 is omitted since the results turned out to be almost identical to the case of maximum fragment size of 500.

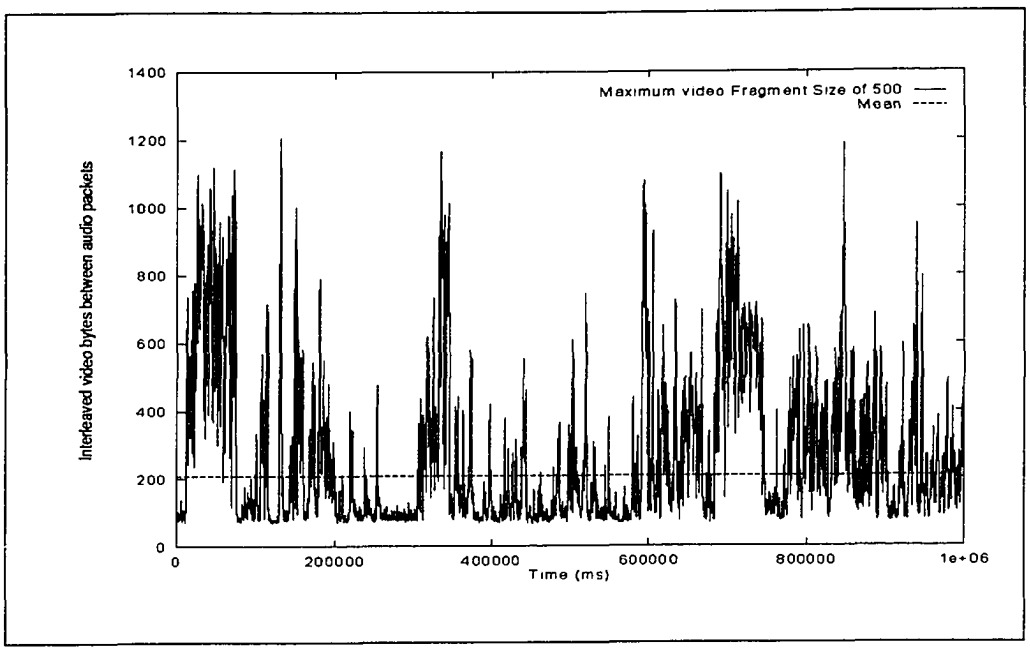

Figure 27. Interleaved video bytes between audio packets (max frag size of 512)

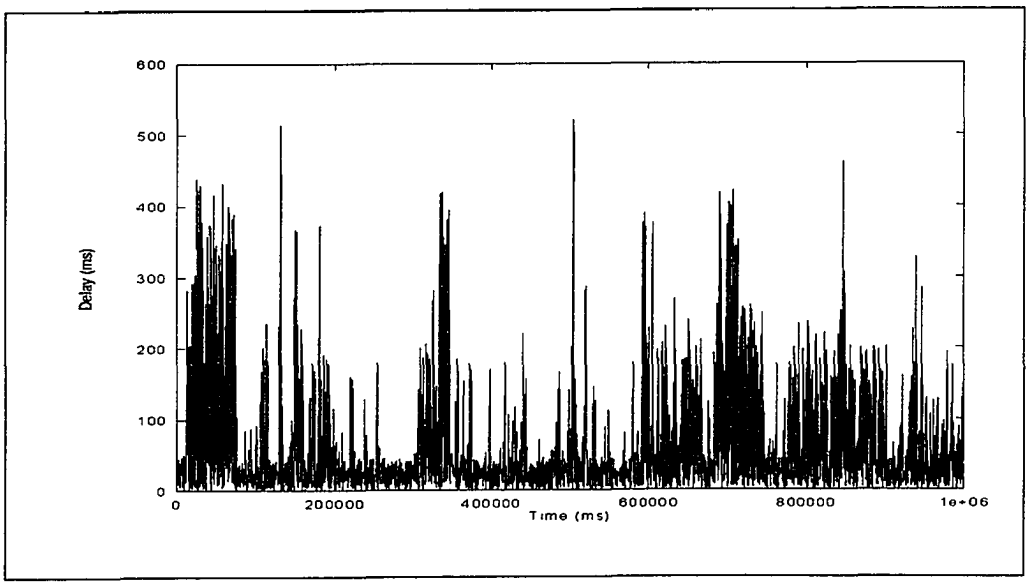

Figure 28. Latency experienced by EDF scheduler (max frag size is 512 bytes)

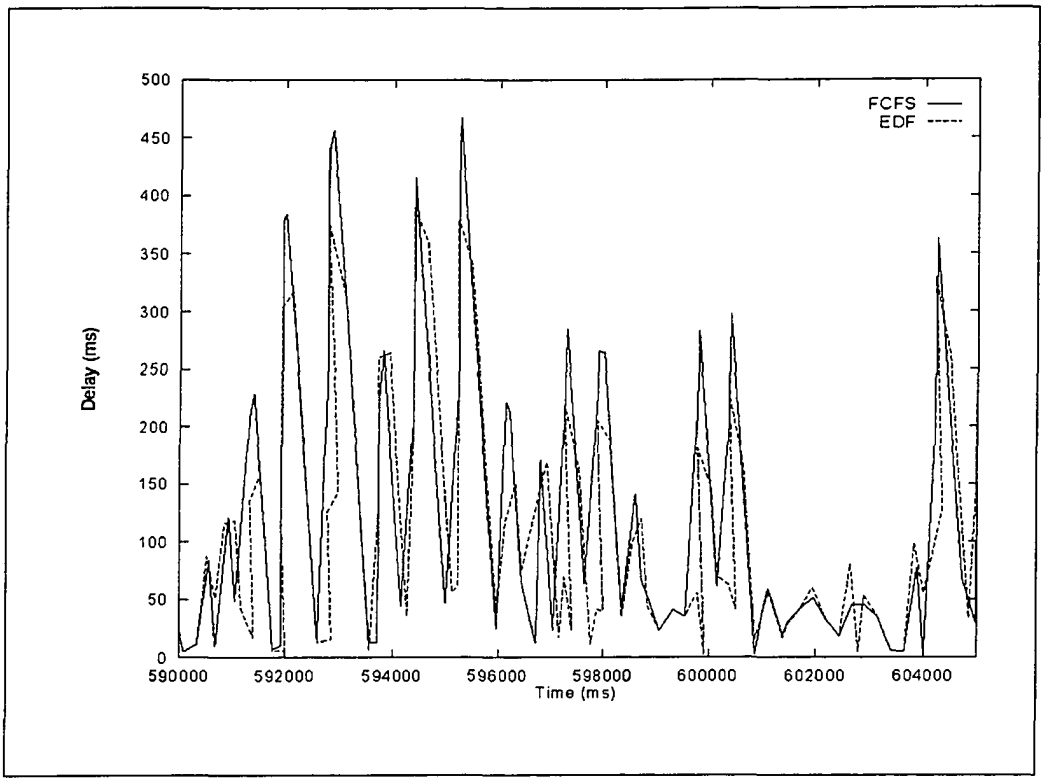

Figure 29. Latency comparison between EDF and FCFS (maximum fragment size =512 bytes)

4.4 The Origin of the Problem

Both EDF and FCFS experienced local latency that is outside the scope of the encoder at the output. Variable gaps between audio packets were also observed in both cases, which lead to audio jitter. Audio jitter makes audio comprehension difficult and needs to be eliminated. There are two reasons for this problem. The first reason is directly related to the variable size of the video packet or fragment. The second reason is the burstiness of the video source. The first problem can be controlled by fixing the size of the video packets, which can be inefficient. The second reason is related to the burstiness of the video source. Burstiness can be caused by the encoder, the packet fragmenter, or both.

The remedy for the second problem is a traffic regulator that controls the flow of video packets and maintains quality of service for all media types on the network.

The second problem is addressed and our proposed solution is the "Bit-Rate Adjuster" (BRA) scheme to be described in the next chapter.

4.5 Concluding Remarks

Scheduling real-time video, audio, and data is analogous to scheduling a set of non-preemptable periodic tasks over a uniprocessor. FCFS and EDF schedulers are used for our experiments as audio/video multiplexers using the simulator that is developed for this purpose.

The EDF scheduler showed improvements over the FCFS scheduler in terms of audio latency. However, the results are still below expectations. Both EDF and FCFS experienced local audio latency outside the scope of the audio encoder at the output. Variable gaps between audio packets were also observed in both cases that lead to audio jitter.

The reasons for local audio latency and jitter experienced by the different schedulers, besides packet assembly overhead, are video fragment variable size, and burstiness of the video source. The burstiness can be solved by regulating the video traffic rate to the maximum negotiated bit rate as suggested in the next chapter.

A new performance measurement is introduced in this chapter. This measurement is the number of interleaved video bytes scheduled between audio packets (VIB). This measurement indicates the amount of video bytes blocking audio packets and adding to their latency in a shared bandwidth environment. This measurement has been used as a performance qualifier in the comparison between FCFS and EDF schedulers.

5. The Bit-rate Adjuster

Traffic in an H.323 connection is always constrained by an upper limit on the bit-rate of each component. This maximum bit-rate is negotiated during connection startup using H.245 procedures. Each conference participant shall abide by the maximum limit publicized by itself and its peers. Hence, one good solution for the burstiness of video traffic is a regulator that controls and adjusts the bit-rate throughout the session.

5.1 The Bit-rate Adjuster Algorithm

The bit-rate adjuster (BRA) monitors the video traffic rate and ensures that the video bit-rate never exceeds the maximum negotiated by the terminal. The bit-rate adjuster is positioned after the fragmentation module as shown in Figure 30 in order to compensate for any irregularities caused by either the encoder or the fragmentation process.

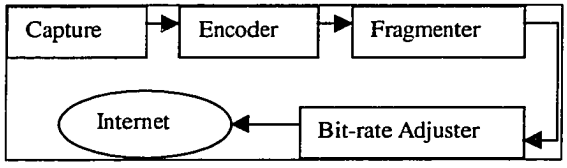

Figure 30. Bit-rate adjuster architecture

The algorithm for the bit-rate adjuster works as follows:

- Compute the bit-rate, Br_{cur}, for current packet, P_{cur}, and previous packet, P_{prev}.
- If Br_{cur} is greater than maximum Bit-rate for this traffic source, Br_{max}, delay P_{cur} by T_{comp}.
- T_{comp} is computed as

$$((\text{BytesOf}(P_{cur}) + \text{BytesOf}(P_{prev})) / Br_{cur}) - ((\text{BytesOf}(P_{cur}) + \text{BytesOf}(P_{prev})) / Br_{max})$$

This algorithm can work on any number of traffic sources participating in a conferencing session. It proved to have a significant improvement for audio intelligibility that is the number one priority in conferencing.

In the next section we show how the BRA algorithm improved the delay as well as the jitter experienced by the audio/video session.

5.2 Bit-rate Adjuster Simulation

We applied the bit-rate adjuster algorithm on a video source using multiple maximum fragment sizes. The resulting video stream is multiplexed with a worst case, G.723.1 audio source. We assume each audio packet carries four frames and a packet is generated every 120 ms. The video stream has variable bit-rate with a maximum of 10 kb/s.

The main objective is to discover how this algorithm can improve the audio delay and jitter without sacrificing a lot of video performance.

5.2.1 Audio Inter-Packet Gap

We start by looking at the number of video bytes that are scheduled in the gap between audio packets. This measure provides an estimate of how the audio will be jittered locally even before reaching the network. The goal is to spread video bytes as much as possible so that audio bytes are neither delayed nor jittered at the transmitter before reaching the network. We applied the BRA algorithm over the same video trace for maximum fragment sizes of 128, 256, 512, and 750. Plots of the number of video bytes lying between audio packets (VIB) against time are depicted in the figures below. Figure 31 shows the VIB against time for a maximum fragment size of 128 bytes. The figure shows the video bytes heavily clustered in short periods. The number of interlaced video bytes was in the range of 50-200 bytes with few overshoots exceeding 400 bytes. Figure 32 shows the VIB against time for a maximum fragment size of 256 bytes. Compared to Figure 31, the number of interlaced video bytes are more scattered in time but the overshoots are higher. The majority of the interlaced video bytes were below 400 bytes with few VIB overshoots exceeding 500 bytes. Figure 33 shows the VIB against time for

a maximum fragment size of 512 bytes. The VIB occurrences are more scattered in the time frame with most of the interlaced video bytes being less than 500 bytes. Finally, Figure 34 shows the VIB against time for a maximum fragment size of 750 bytes. Again, we discovered lesser occurrence of interlaced bytes and higher overshoots because of larger video packets. So generally, it is observed that bytes tend to cluster more as the fragment size decreases. On the other hand, the overshoots of the number of inter-gap video bytes are higher with larger maximum fragment sizes.

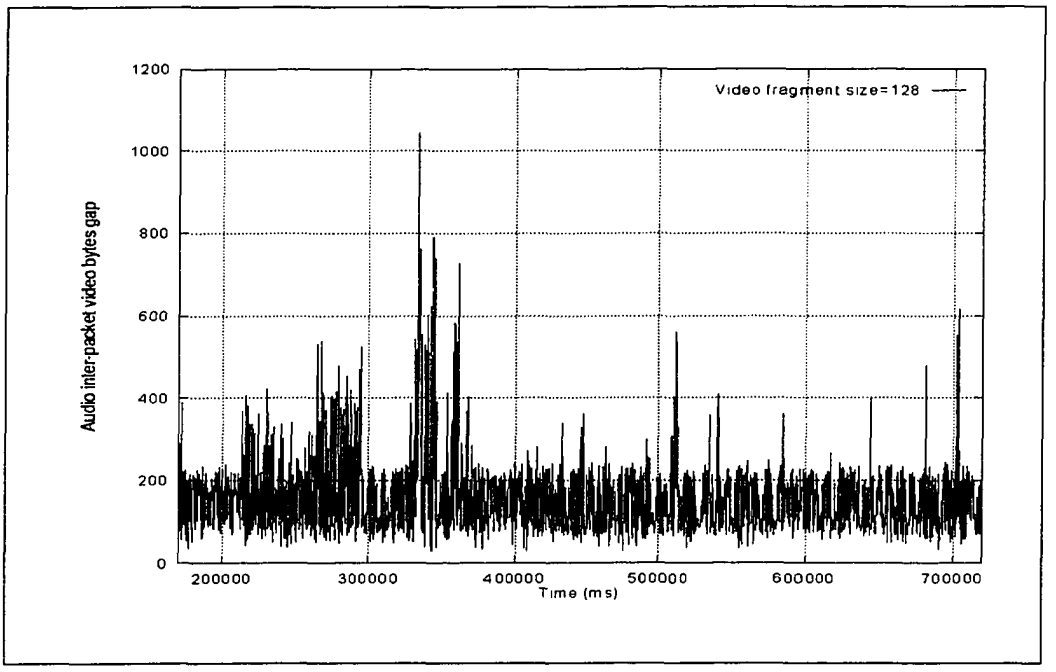

Figure 31. Bit Rate Adjuster for video fragment size of 128 bytes

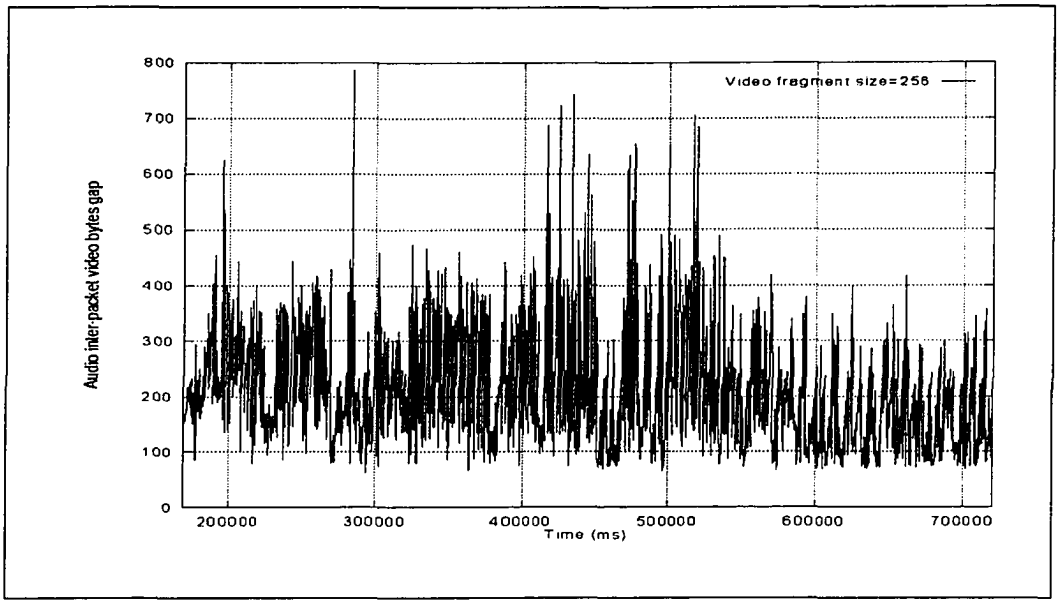

Figure 32. Bit Rate Adjuster for video fragment size of 256 bytes

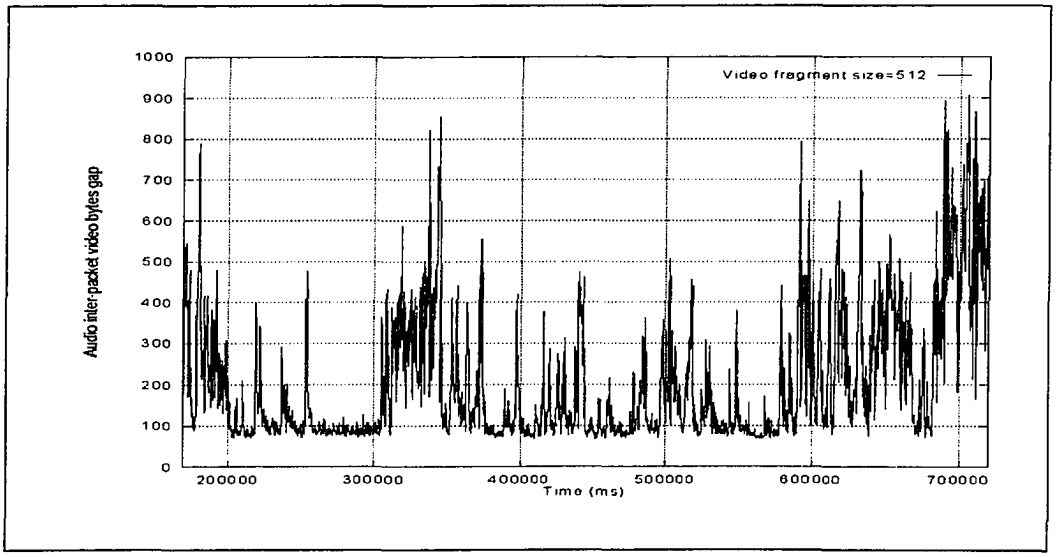

Figure 33. Bit Rate Adjuster for video fragment size of 512 bytes

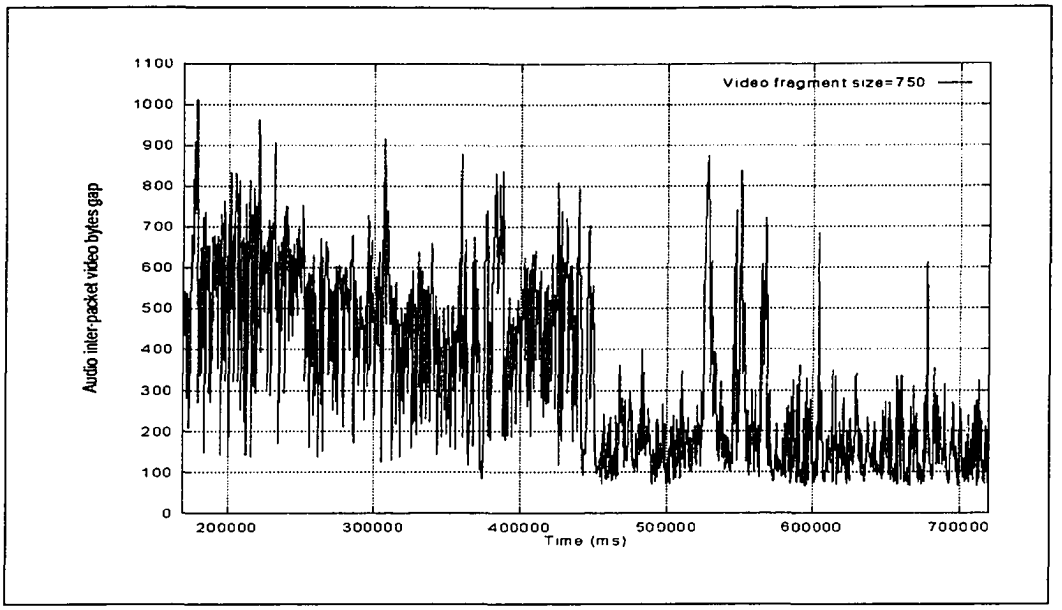

Figure 34. Bit Rate Adjuster for video fragment size of 750 bytes

5.2.2 BRA Delay Analysis

The bit-rate adjuster requires delaying video packets in order to maintain the maximum bit-rate announced by the transmitter. This delay penalty incurred on video packets is computed. Plots are constructed for video fragment size of 128, 256, 512, and 750. The plots are shown in the figures below. Figure 35 shows the delay (in ms) of the video packets for a maximum fragment size of 128 bytes. The figure shows a few video packets that required delay adjustment, however, most of the introduced latency was below 300 ms. Figure 36 shows the delay of the video packets for a maximum fragment size of 256 bytes. The number of video packets that required delay adjustment dropped compared to Figure 35, yet the delay value introduced to video packets was higher.

Figure 37 shows the delay of the video packets for a maximum fragment size of 512 bytes. The delay introduced to video packets dropped significantly compared to smaller maximum fragment size curves. Finally, Figure 38 shows the delay of the video packets for a maximum fragment size of 750 bytes. It is observed that the video delay clusters

more as the fragment size decreases. This implies that as fragment size decreases, more packets tend to violate the negotiated bit-rate. This is mainly due to the packet-handling overhead incurred by the fragmentation process. Hence, more packets need to be adjusted. Intuitively, it is thought that in order to get around the large video packets blocking problem for audio, smaller video fragment sizes should be used. Our observation contradicts this thought.. On the other hand, the delay overshoots are higher for larger maximum fragment sizes. This is expected since with larger fragment sizes, the fragments contain more bytes and take more time to adjust. However, the maximum overshoot starts decreasing again for higher maximum fragment sizes. Video fragments with maximum fragment size of 750 bytes have smaller maximum overshoot than both 500 and 256 cases. The maximum delay penalty for all fragment sizes never exceeded 800 ms.

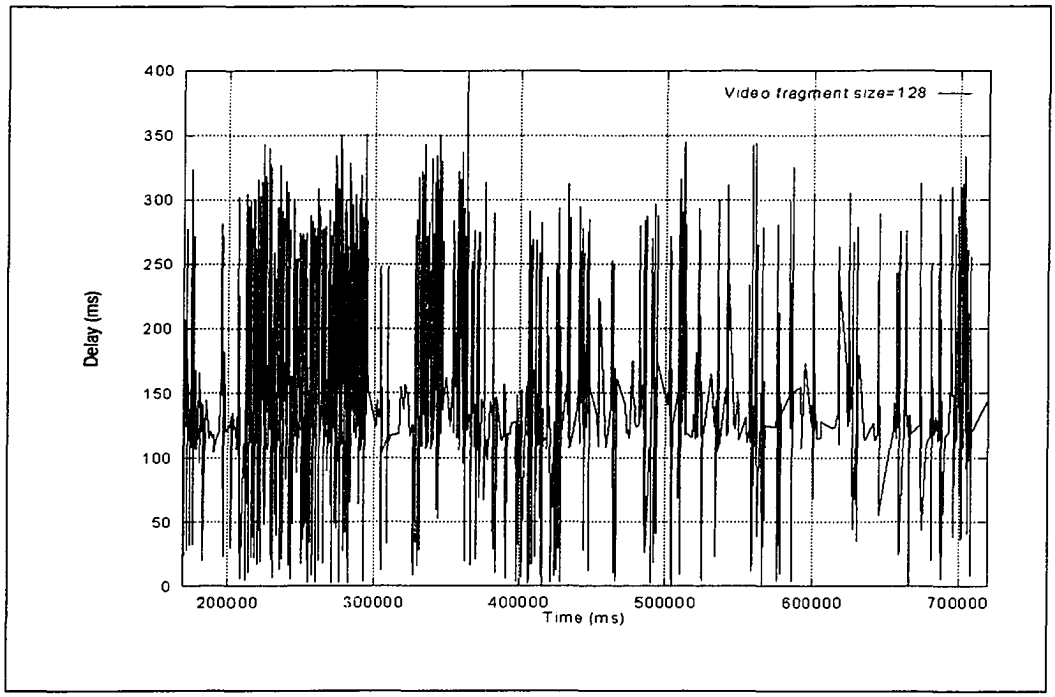

Figure 35. BRA delay for maximum fragment size of 128

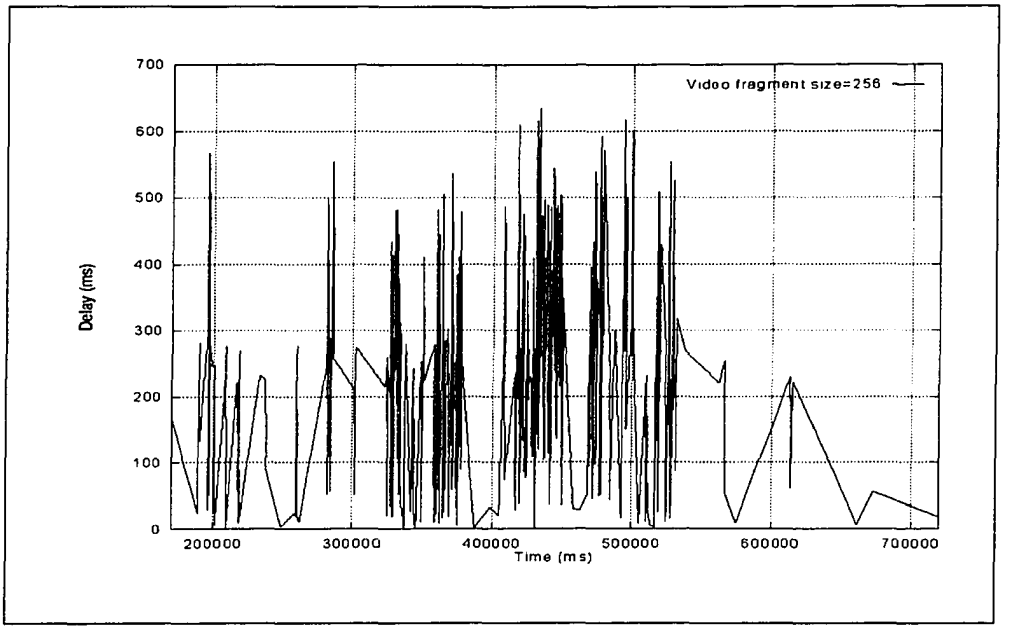

Figure 36. BRA delay for maximum fragment size of 256

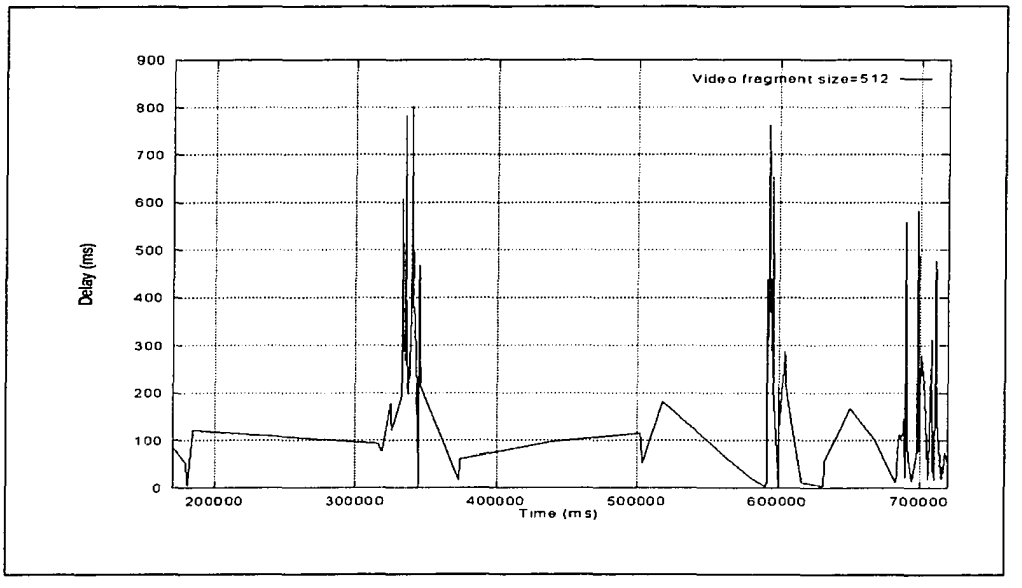

Figure 37. BRA delay for maximum fragment size of 512 bytes

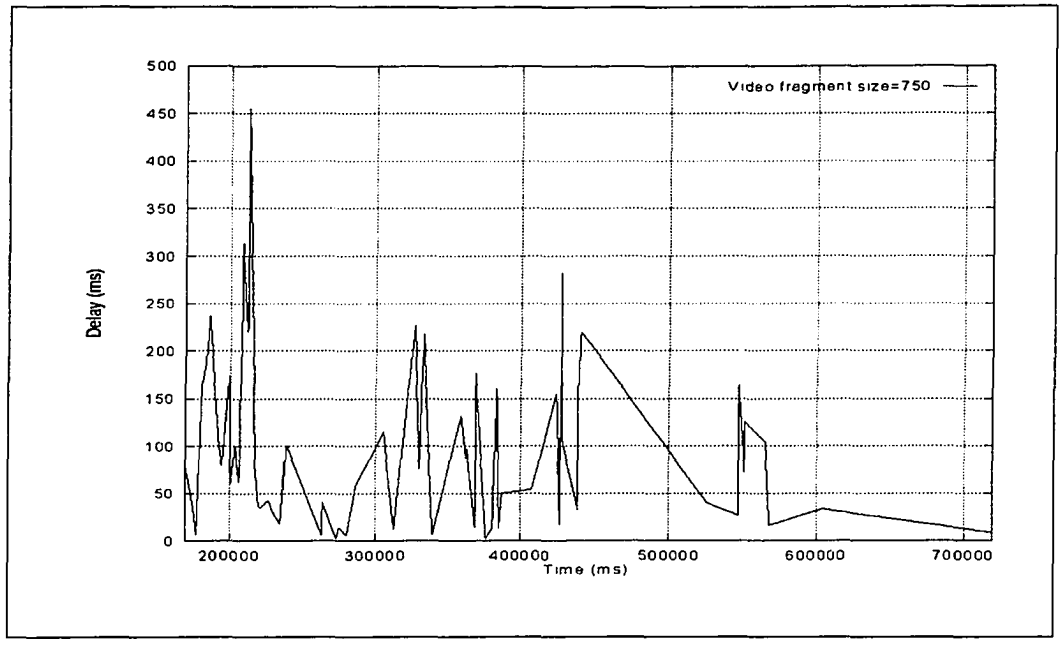

Figure 38. BRA delay for maximum fragment size of 750 bytes

5.3 Effect of Maximum Fragment Size on BRA Algorithm

It has been shown with FCFS and EDF algorithms that the maximum fragment size for video affects the performance of the algorithms. In this section, we study this effect on the BRA algorithm.

The first aspect that we consider is the number of video packets adjusted for different fragment sizes. Figure 39 shows a plot of the percentage of video packets adjusted for the BRA algorithm for different fragment sizes. It is observed that the smaller the fragment size, the more the tendency for video packets to cluster in short intervals of time. Hence, more packets need to be adjusted for smaller fragment size. The curve drops steeply as the maximum fragment size increases.

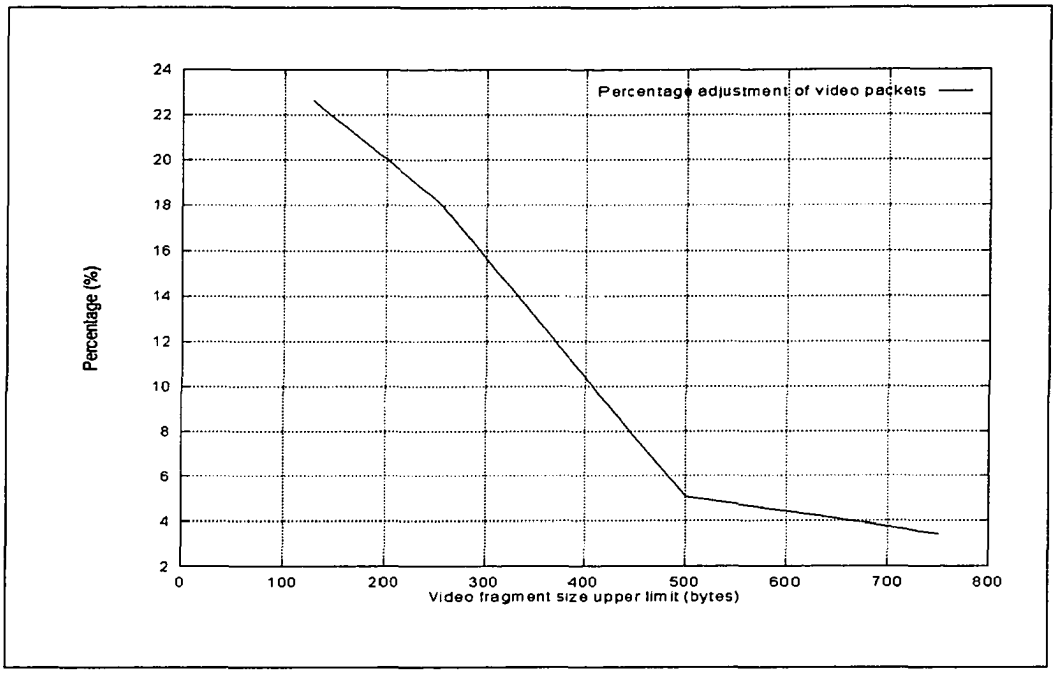

Figure 39. Percentage of video packets adjusted for BRA

We also measured the number of video bytes interleaving audio packets (VIB). This measure is a good indicator for local audio latency. Since the bandwidth for both video and audio is shared, video packets can cluster in the outgoing queue causing audio packets to accumulate and miss their subscribed share of the bandwidth. Figure 40 shows the maximum, mean and standard deviation of video bytes occupying audio inter-packet gap. The graph shows that the mean rises steadily as the maximum fragment size increases and then stabilizes or falls slightly between fragment sizes of 256 – 500 bytes. The mean starts rising again afterwards. The maximum inter-packet bytes reach a minimum around 256 bytes and then starts rising again. Two interesting observations are deduced from this graph. The first observation is that the number of video bytes is steady during the 256 – 500 bytes fragment size span. The second observation is that the maximum number of bytes also reaches a local minimum during that span. Both observations strongly suggest that the choice of maximum fragment size for video

packets should be between 256 and 500 bytes. This choice can significantly reduce the local latency suffered by audio packets because of video packets are oversubscribing to the shared bandwidth.

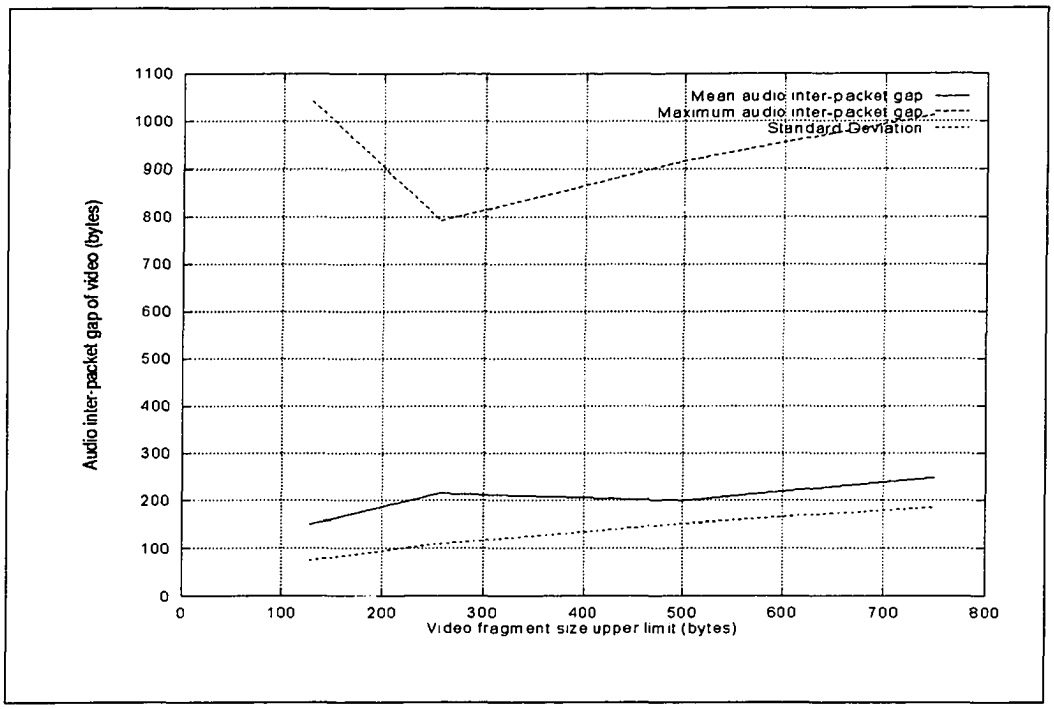

Figure 40. Interleaved video bytes between audio packets for BRA algorithm

A third important measurement is the delay penalty incurred on video packets. This measurement describes the side effects of the BRA algorithm. It is thus a detrimental factor in the scope and applicability of the algorithm. Figure 41 shows the maximum, mean and standard deviation of the video packets delay penalty because of the BRA algorithm. It is observed that the mean delay penalty was less than 250 ms for all maximum fragment sizes. Another observation is that the mean delay increases as the fragment size increases. It then starts falling off after fragment size of 256 bytes. The explanation for this behavior is that for small fragment sizes, more packets tend to violate the publicized maximum bit-rate and hence require adjustment. One important advantage

for the BRA algorithm is that it satisfies the delay requirements for audio packets at the expense of less important video packets.

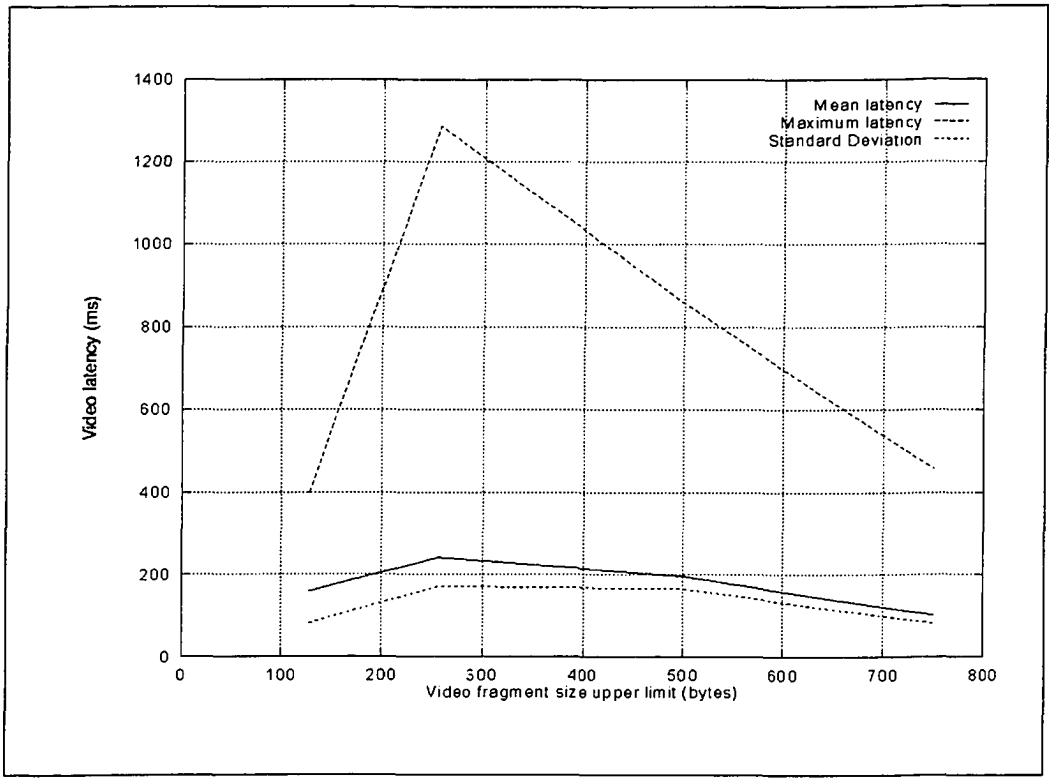

Figure 41. Delay penalty for the BRA algorithm

5.4 BRA Versus FCFS

A comparison was conducted between the two algorithms to decide if any significant improvement is achieved by applying the BRA algorithm as opposed to the simple FCFS algorithm. We compare the number of interleaved video bytes between audio packets for both cases. Figure 42 shows a comparison between the maximum number of VIB bytes for both algorithms. The BRA algorithm shows a better performance over FCFS in that aspect. The figure shows a significantly lower maximum number of interlaced video bytes in the audio gap for BRA against FCFS. The maximum delay was cut by almost 50% for small fragment sizes. Both curves had sharp declines of VIB bytes going from a

maximum fragment size of 128 bytes to 256 bytes. This result strongly suggests that the smaller upper limit on the video fragment size does not imply a better bandwidth share and latency for audio. As the maximum fragment size increased from 128 bytes to 256 bytes, the VIB improvement gap of the BRA algorithm over the FCFS algorithm slightly decreased. The gap narrowed at a higher rate as the maximum fragment size decreased from 256 to 750 bytes. The lowest maximum for the BRA algorithm was achieved at a maximum fragment size of 256 bytes. The BRA improvement over FCFS at this fragment size limit scored almost 40%.

Figure 43 shows a comparison of both VIB means. The BRA showed some improvement over the FCFS. The mean number of bytes occupying the audio gap improvement for BRA reached almost 28% for small fragment sizes compared to the FCFS algorithm. The BRA algorithm maintained a consistent VIB improvement gap between maximum fragment sizes of 128 and 256 bytes. The gap started to decrease as the maximum fragment size increased from 256 bytes to 512 bytes. The improvement was not significant when the maximum fragment size for video increased beyond 512 bytes. The maximum improvement of the BRA algorithm over the FCFS achieved when the maximum fragment size is 256 bytes. The improvement scored almost 20% better. Figure 44 shows a comparison of standard deviation of both algorithms. BRA showed less deviation than the FCFS algorithm.

Given the discussion above, and the data shown in the figures, we can conclude that the BRA algorithm demonstrated significant performance improvements over the general FCFS case.

5.5 Concluding Remarks

The bit-rate adjuster (BRA) monitors the video traffic rate and ensures that the video bit-rate never exceeds the maximum negotiated rate by the conferencing terminal. Hence, video traffic burstiness can be controlled.

The smaller the maximum fragment size the more the tendency of video packets to recur and cluster in short intervals of time. Hence, more packets need to be adjusted by the BRA algorithm for smaller video fragment limits.

The BRA algorithm showed better performance over FCFS. The improvement reached 28% for mean interleaved video bytes between audio packets for small fragment sizes. The improvement gap was significant between maximum fragment sizes of 128 bytes and 256 bytes and then started to shrink thereafter. The lowest maximum of interleaved video bytes was achieved when the maximum video fragment size was set to 256 bytes. The incurred delay penalty on the algorithm was also not significant. It was discovered that the mean delay penalty never exceeded 250 ms for all video fragment sizes experimented.

In order to economize on audio inter-packet gap using the BRA algorithm, it has been shown that best performance is achieved using a video fragment size limit greater than 256 bytes and smaller than 512 bytes. A midway value for the maximum fragment size is recommended.

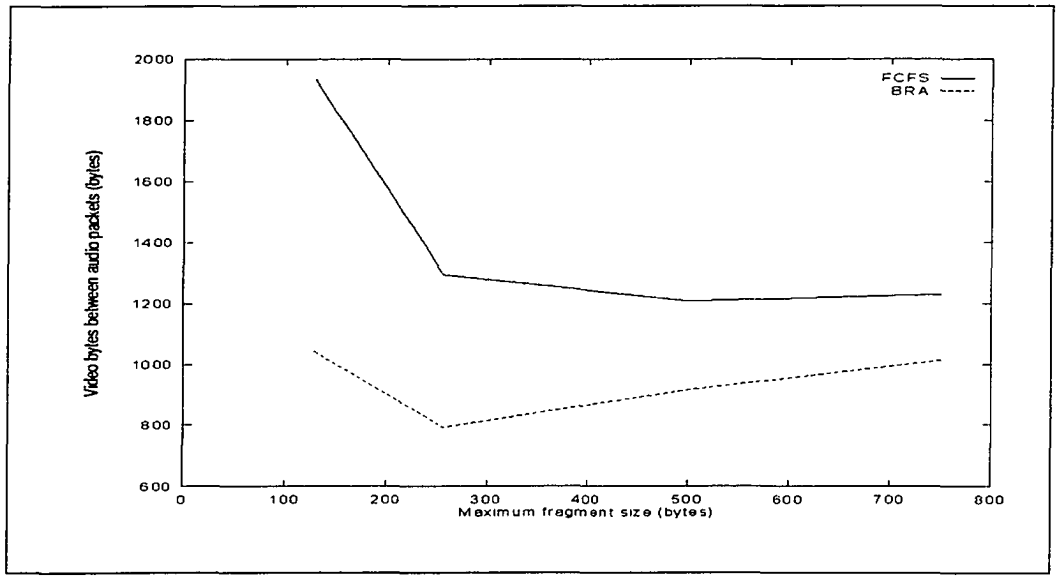

Figure 42. Comparison for maximum number of VIB between FCFS and BRA

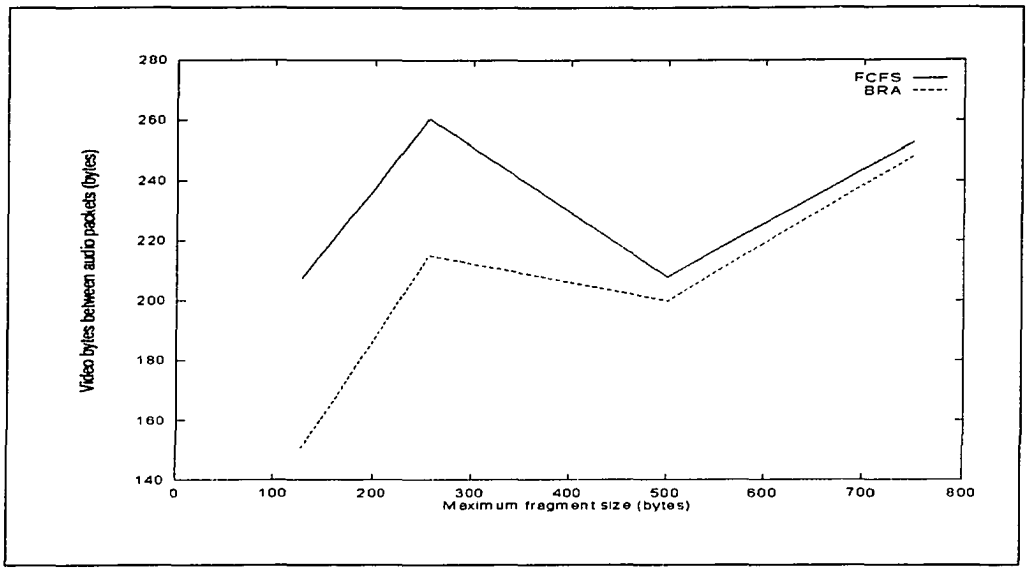

Figure 43. Comparisons for means of audio inter-packet video bytes

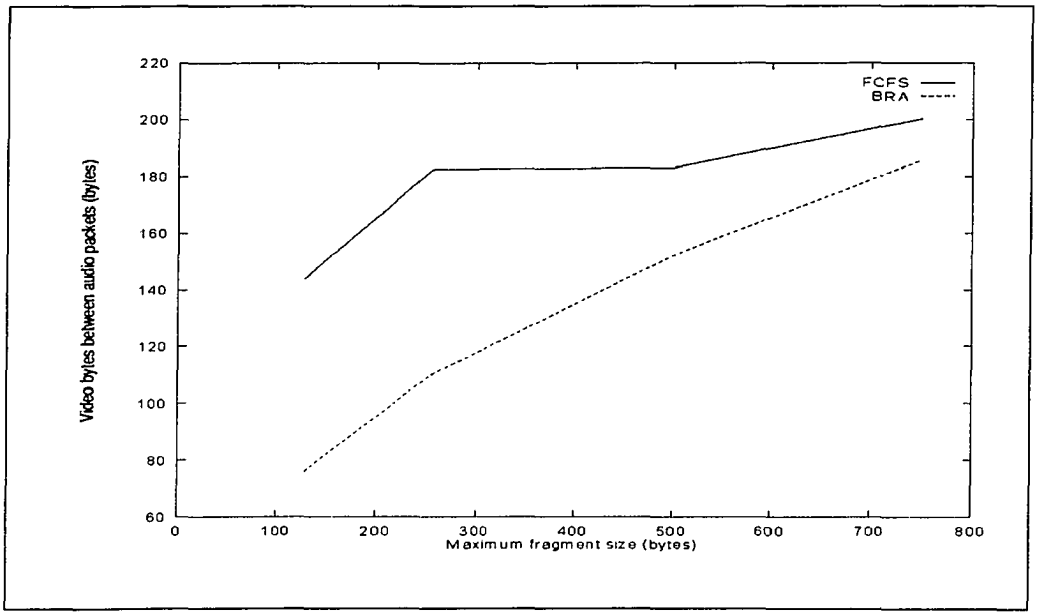

Figure 44. Standard deviation of number of audio inter-packet video bytes comparison

senders generate a sender report. A sender report contains information useful for media synchronization as well as cumulative counters for packets and bytes sent. This information allows receivers to estimate the sender's data rate. Receivers generate receive reports for all video and audio sources they have heard from. These reports contain information on the highest sequence number received, the number of packets lost, a measure of the inter-arrival jitter, and timestamps needed to compute the round-trip delay between sender and receiver issuing the report.

RTCP packets also contain a unique identification for their original sender via an SDES (source description) packet. This packet may contain user name, canonical name, email, etc.

6.3 RTCP Feedback Congestion Technique

Busse et al. [9] introduced a dynamic QoS control mechanism for multimedia applications based on RTCP feedback information. Their technique is based on receiver reports delivered from receiving end applications to the source. A new algorithm using a modification for this approach is introduced in this section. The algorithm is based on using audio receiver report to adjust video bandwidth. The objective is to provide audio with enough bandwidth to function with an acceptable QoS by sacrificing some of the video bandwidth.

On receiving audio RTCP receiver report (ARR) of a remote terminal, the conferencing manager performs the following steps:

1. Analyze the statistics reported by the ARR for loss.
2. Determine network Congestion State. A network can be in an unloaded, loaded or congested state. The loss value obtained from ARR determines current network Congestion State. This value is used to decide whether to increase, hold or decrease the video bit-rate of the terminal.
3. Adjust video bit-rate according to the network state analysis. The user has the freedom to specify minimum and maximum video bandwidth allowed. The user also

has the freedom to specify audio parameters that determine the network different states and adjust the video bit-rate accordingly.

Parameter values for determining the current state of the network are shown in Figure 45. λc is the lowest audio packet loss value beyond which the network becomes congested and video bit-rate has to decrease by δc. λu is the highest audio packet loss value below which the network is lightly loaded and video bit-rate will increase by δu. During the loaded state, the video bit-rate is unchanged.

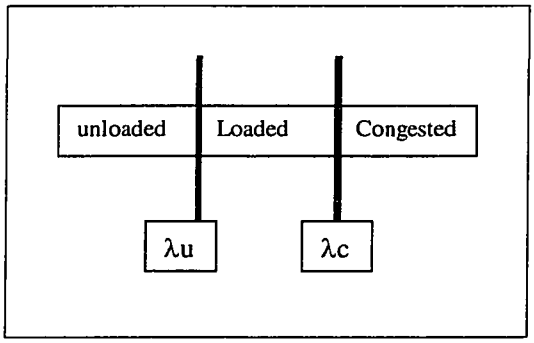

Figure 45 Different Network States

We implemented the above algorithm on two Intel Videophone terminals. Both terminals were connected over the Internet via different Internet service providers. The maximum video bandwidth was set to 25 kb/s. Both terminals use G.723.1 as their preferred audio codec and H.263 as their preferred video codec. An audio interview file that lasted for approximately 10 minutes was used as the audio input sample. We chose δc = 1 kb/s, δu= 0.5 kb/s, λc = 45, λu= 30. The experiment was carried out during a known Internet rush hour to capture the worst congestion case. Both terminals were running full-screen video CIF and low bit-rate audio.

Figure 46 shows the relationship between video bandwidth and audio packet loss as the session proceeds. It is observed that as audio loss increases beyond the congested state, video bit-rate decreases quickly and consequently audio loss starts to improve. Video bit-rate starts to rise again whenever audio loss falls below the loaded state. The choice of

parameters that determines state of the network is user dependent. Our choice was based on statistics collected after running multiple sessions using the same environment conditions.

There are several disadvantages for this approach. One disadvantage is the lack of criteria to determine accurate estimation of network state parameters. Internet home users may have relaxed requirements for the parameters that may not be acceptable for business users. The user has to rely on statistics for each particular case. Another disadvantage is the slow response of the algorithm to network audio loss. This observation may lead to more loss than expected to happen before the terminal reacts by decreasing the bit-rate. A third disadvantage is that a possible QoS oscillation may happen where video bit-rate and audio loss follow a sinusoidal waveform. This phenomenon will make audio conversation unintelligible due to large peak losses. Video on the other hand will also suffer from poor performance due to the reduced bit-rate.

One solution for the oscillation problem was suggested in [9]. A filter is used to smooth the statistics and to avoid QoS oscillations. The smoothened loss rate λ is computed by the low pass filter $\lambda \leftarrow (1 - \alpha)\,\lambda + \alpha\, b$, where b is the new audio loss value. Increasing α increases the influence of the new loss value while decreasing α increases the influence of the previous value. In section 6.5, we introduce another innovative approach that may not suffer from the same drawbacks.

6.4 RTCP Feedback Limitations

In order to keep RTCP overhead to a minimum, thereby allowing better utilization for data packets, RTCP packet transmissions are limited. In general, no more than five percent (5%) of the total bandwidth of a media stream can be allocated to RTCP functionality [49]. On slow links with limited bandwidth, the inter-arrival time between successive RTCP receive reports may exceed seven seconds. This interval is quite long and may lead to slower response of adaptive schemes to network conditions as depicted in section 6.3. A more responsive approach is required to effectively control the flow of all media streams.

Another limitation for RTCP feedback approach is the possible QoS oscillations phenomenon discussed in section 6.3. Long inter-arrival interval between reports together with bad choice of control parameters both contribute to this phenomenon.

A third drawback of the RTCP feedback approach is related to its periodic nature. Congestion or packet loss is a sporadic event. Hence, it has to be treated by another non-periodic event. In fact, network congestion caused by audio loss, particularly in real-time applications, is short-lived and non-correlated [5]. Thus, reaction to loss should be prompt and instantaneous. In addition, waiting until loss happens and reacting afterwards may not be the best approach. A better approach is to provide a scheme that can predict loss and informs the media terminal to slowdown in order to avoid loss from happening. We propose a new approach that can predict loss for the incoming audio stream and advises the sender to decrease its video bit-rate. This approach is discussed in the following section.

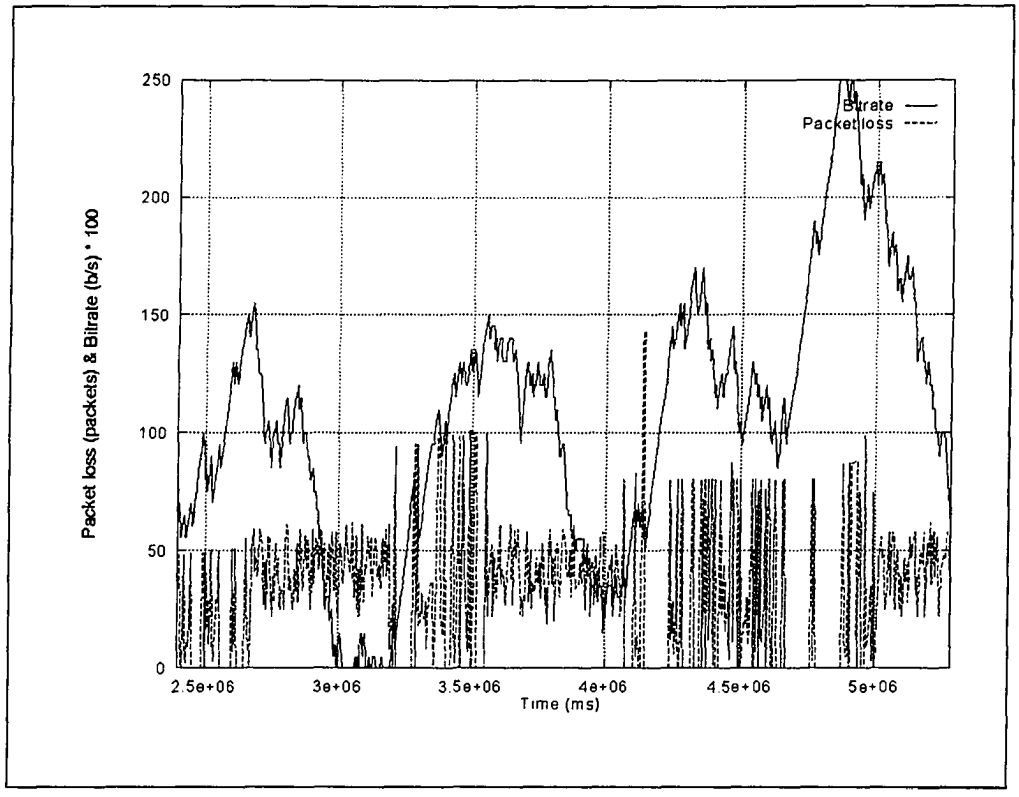

Figure 46. Video bit-rate versus audio loss

6.5 Loss Prediction Using Jitter Feedback

6.5.1 Relationship Between Packet Loss and Jitter

Another experiment was conducted using similar input, and network conditions as the RTCP feedback experiment case discussed in section 6.3. We measured audio packet loss, inter-arrival time between packets and inter-arrival jitter at the receiver. The packet loss is measured as the difference between sequence numbers of packets arriving at the receiver. If the difference is greater than one then loss occurred. The inter-arrival jitter is measured as the difference in packet spacing at the receiver compared to the sender for a pair of packets as suggested in [59]. This is equivalent to the difference in the "relative transit time" for the two packets. The relative transit time is the difference between a

packet's RTP timestamp and the receiver's clock at the time of arrival. This value is measured in milliseconds.

Figure 47 shows a plot of the interarrival time according to the receiver clock and audio packet loss both against time. Figure 48 shows an extension in the time domain for the same plot. It is observed from the plots that packet losses peaks occurred often shortly after an abrupt change in the inter-arrival time of pervious packets. Smaller changes in the packet interarrival times were followed by no or small packet losses. The more significant the change in the interarrival time, the higher the loss value as shown in the figures. This was a motivation to pursue this observation further and find out if there is any relationship between audio inter-arrival jitter and packet loss.

We computed the audio inter-arrival jitter at the receiver. The jitter value can be positive or negative. Figure 49 shows a plot of the relationship between the inter-arrival jitter and packet loss. The relationship was extended in the time domain over five figures for proof of concept. It is observed that the negative audio jitter peaks are of larger amplitude and occur more frequently. The reason is that the interarrival time between packets at the receiver is often larger than that on the source since the source is usually controlled by a constant bit-rate regulator. It is also observed that the jitter at the receiver is of positive value. The reason is that the packet interarrival time at the receiver can be lower than that of the source. Since the interarrival value at the sender is usually maintained at a constant value (in our case 120 ms, since we pack four frames per audio packet), the peak positive value of the jitter is not expected to exceed this value. There may be very few exceptions to this rule as shown in Figure 49. One exception to that rule is when an error occurs at the source causing it to violate the constant bit-rate rule. Another exception is during the case of silence periods at the source where packet interarrival time may be longer. These exceptions explain the very few occurrences of positive high values for the jitter

Figure 49 shows small overshoots for packet loss following abrupt rises in audio jitter of previous audio packets. Figure 50 shows increases for packet loss overshoots as the jitter magnitude increases for previous audio packets. Both curves showed consistent results

with packet loss moving up and down based on the amount of jitter of the previous packets. The can thus deduce from both figures that abrupt rises in jitter value may result in packet losses after a short interval of time. Figure 51 also strongly sustained this observation. The relationship between audio packet loss and jitter implies that jitter computation at the receiver can predict the possibility of packet loss in the near future. Hence, by carefully monitoring inter-arrival time at the receiver, it is possible to reduce packet loss significantly by warning the sender.

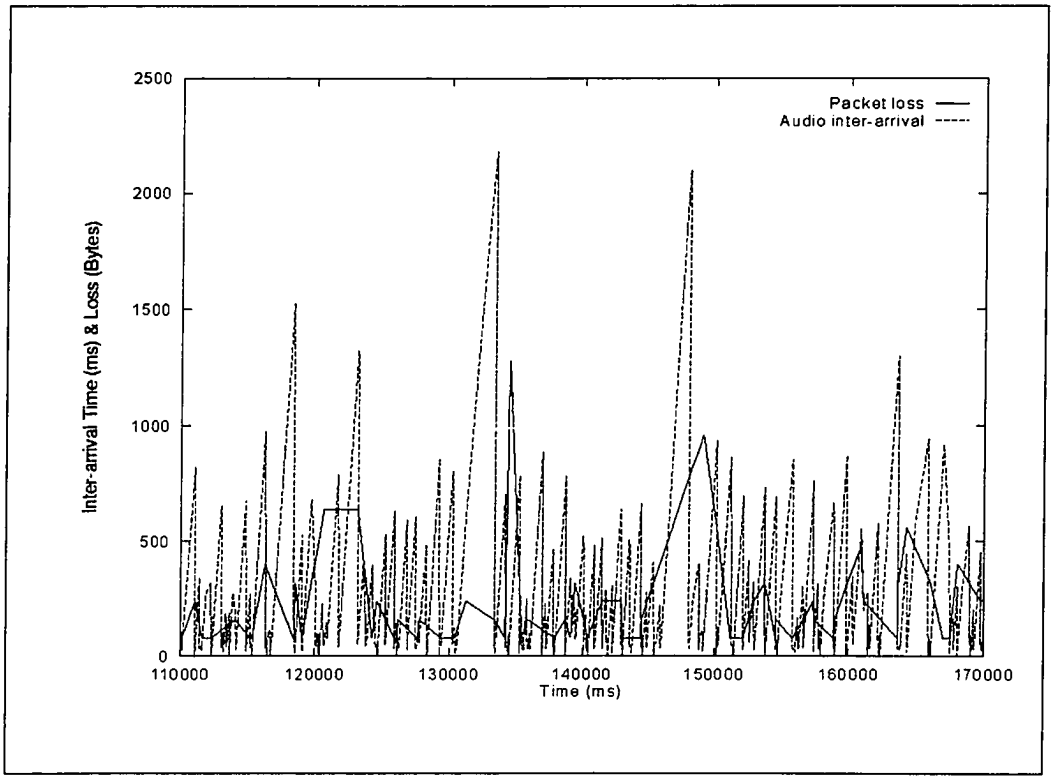

Figure 47. Audio receiver inter-arrival time and packet loss

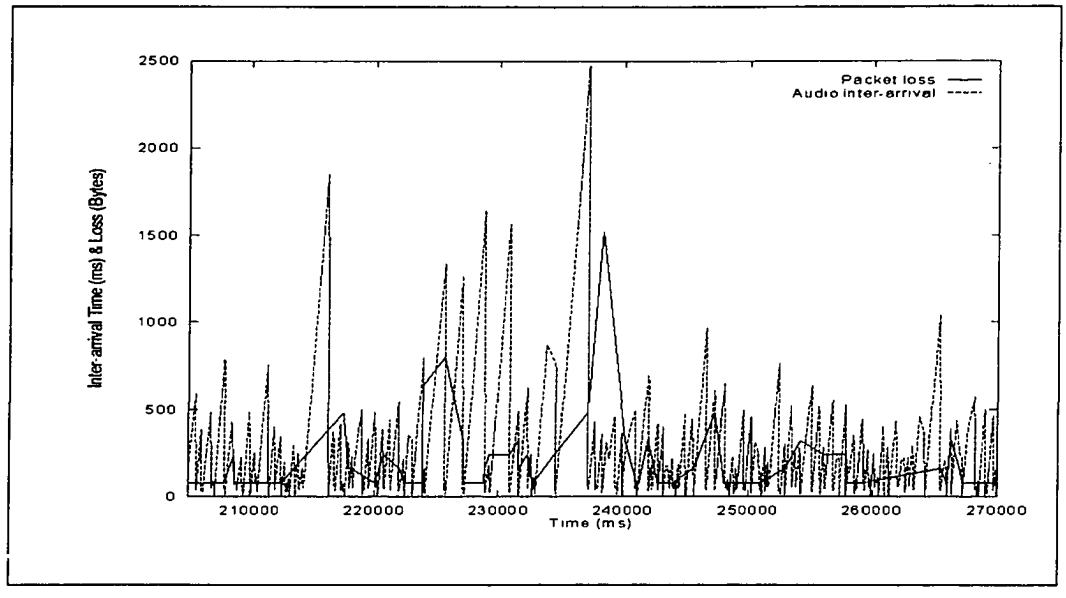

Figure 48. Audio receiver inter-arrival time and packet loss

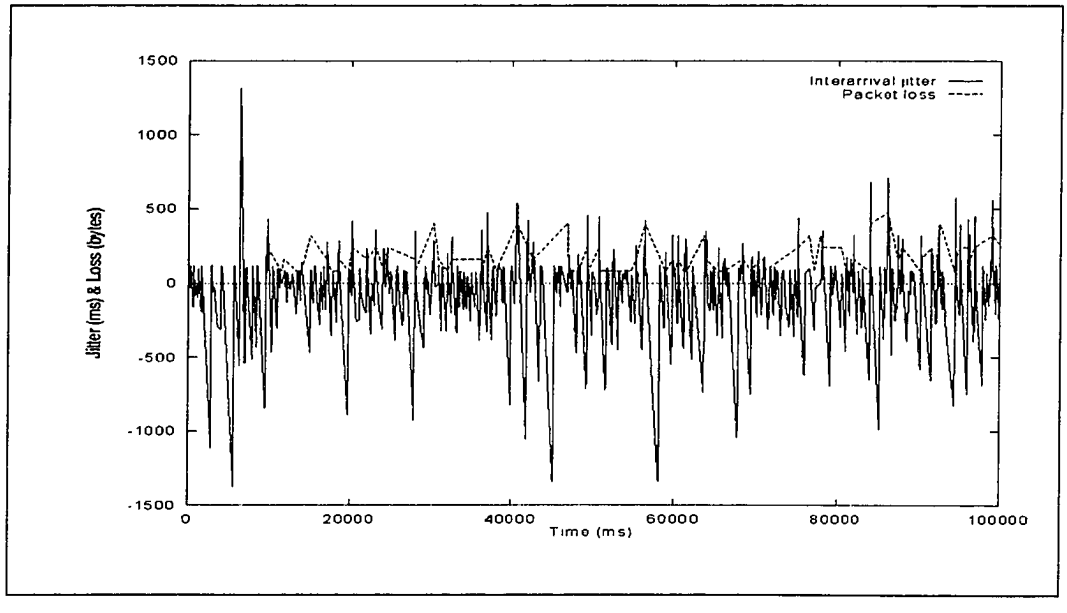

Figure 49. Inter-arrival jitter and packet loss

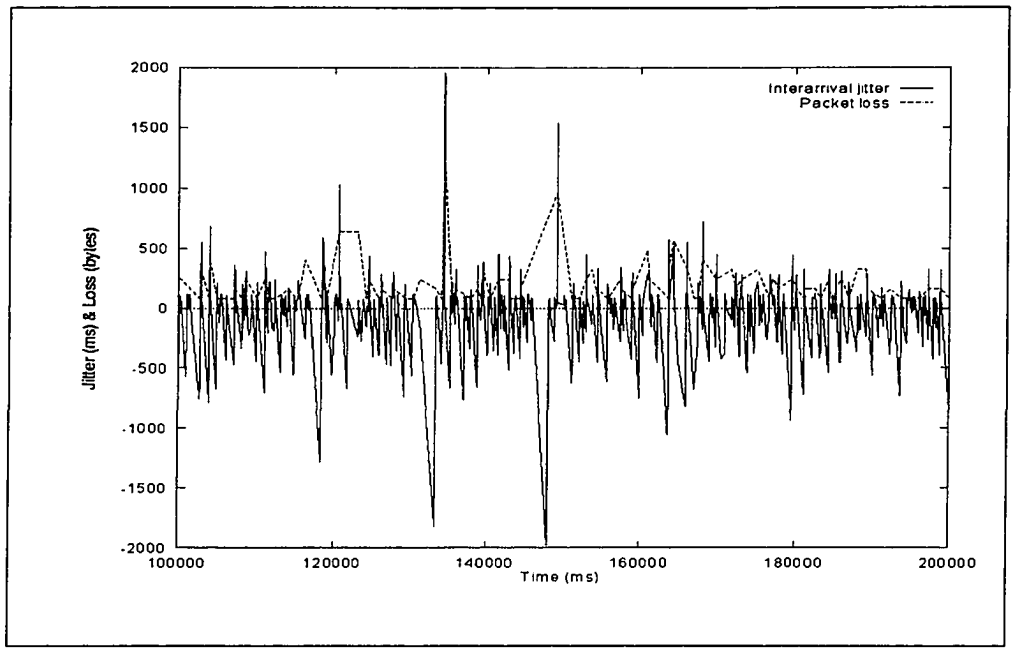

Figure 50. Inter-arrival jitter and packet loss

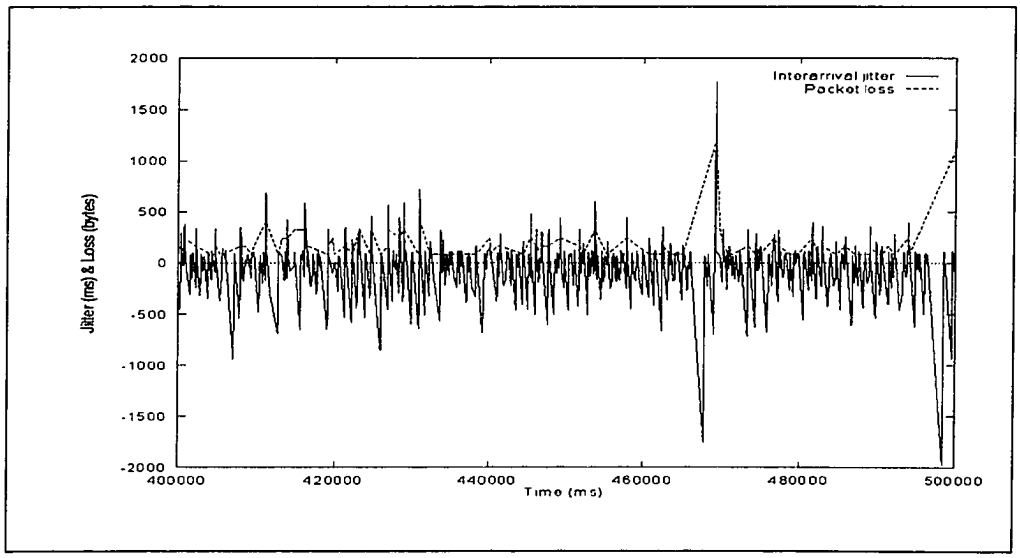

Figure 51. Inter-arrival jitter and packet loss

6.5.2 The Loss Prediction Algorithm

These measurements were performed for audio inter-arrival jitter and loss only and it is yet to be seen if the same relationship applies to other real-time media components. This result motivated the development of another algorithm that can predict the occurrence of packet loss. The algorithm works as follows:

- Compute packet inter-arrival jitter J_i for each audio packet i at the receiver.
- If the value of J_i exceeds a maximum value J_{max}, the network is in the congested state and loss may occur. Send a flow control command to the sender to limit the video bit-rate.
- If the value of J_i drops below a minimum value J_{min}, the network is unloaded and video bit-rate can be restored to default value. Send a flow control command to the sender to increase the video bit-rate.
- If the value of J_i falls between J_{max} and J_{min} , the video bit-rate will not be adjusted.

This algorithm controls the sender based on the loss predictability of the receiver. Figure 52 shows the number of jitter warnings and the percentage of unpredicted loss both as a function in J_{max}. The jitter warnings are normalized against the maximum number of warnings occurred during the experiments and J_{max} is measured in units of the standard deviation of the inter-arrival jitter. It is observed from the figure that the optimal value for J_{max} that allows the lowest value for the number of generated jitter warnings and provides prediction closure for a majority of packet loss is 1.8 * jitter standard deviation.

It is possible to optimize on the number of generated flow control commands by following certain heuristics. One possible heuristic is to limit the number of generated jitter flow control packets within a certain interval of time provided that the last computed jitter was not preempted. Preemption means that a new jitter value was

computed with a larger absolute value than the last computed jitter value that caused the video flow control command to occur. Likewise, stabilization in the jitter computation at the receiver can prompt the receiver to issue flow control commands to raise the video flow rate.

This algorithm is more responsive than the RTCP-based feedback algorithm. It is also possible to implement using standard protocol messages such as H.245 'Flow Control'. The algorithm is also more efficient than the RTCP algorithm since flow control is generated on demand as opposed to generating huge statistics packets periodically. Further, it is based on local feedback and remote control (LFRC) as opposed to remote feedback and local control (RFLC) (e.g., RTCP-based algorithms). LFRC is more effective when alien terminals are interoperating since remote control is more binding than remote feedback. The reason is that terminals are required to obey commands issued from other terminals but can ignore indications.

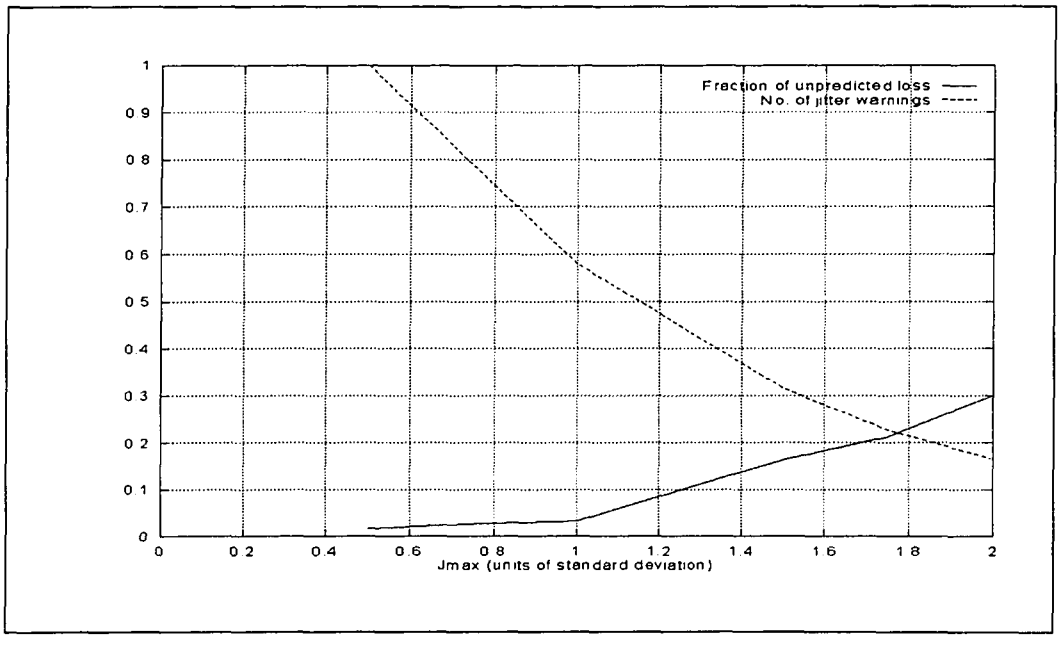

Figure 52. Relation between loss, jitter indications and Jmax

6.6 Jitter Compensation

Audio jitter occurs over the Internet during multimedia conferencing due to several factors. The first contributing factor is host-induced jitter. This occurs because of interference of several media components (audio, video, and data). Multiplexing all components together induces a jitter effect at the host even before reaching the network as discussed in chapters 3 and 4.

The second contributing factor to audio jitter is network-induced jitter. This factor is due to heterogeneous switches, routers and links speeds along the connection path. Bursty traffic occurring at any point in the path can cause momentary accumulation of packets in switch queues. This phenomenon causes significant jitter at the peer end as shown in section 6.3.

Figure 55 shows a characterization for audio jitter at the originating point and at the receiver. It is observed that audio inter-arrival time at the receiver vary significantly from the corresponding sender timestamps. Sender inter-arrival time occasional deviation from the nominal value of 120 ms[1] is explained by silence interrupts separating talk spurts.

The goal of the proposed technique is to recover the original audio jitter at the receiver to the maximum possible accuracy.

6.6.1 Jitter Compensation Algorithm

This method aims to recover the original pattern of audio samples at playback time. It can be described as follows:

(1) Check the number of audio frames per packet used for the conference, *AsduSize*.

(2) Monitor RTP packet headers for start of talk-spurt marker bit in the audio stream at the receiver. Each RTP packet header has a marker bit that signals end of silence.

---

[1] This is assuming we pack 4 audio frames of G.723 per audio packet. The G.723 codec has a constant bit-rate that generates 20 bytes every 30 ms.

(3) On detecting beginning of a talk spurt TS_k, compute interarrival jitter $|J_i|$ between successive audio packets. Record the maximum inter-arrival jitter for the whole talk spurt M_k. Compute inter-arrival time at the receiver for each packet I_i. M_k is used during the following talk spurt.

(4) Construct new arrival times A_i for the audio packets such that the interarrival time between packets is retained constant. $A_i = M_k +$ AsduSize * Gc * i where Gc is the granularity[1] of the audio codec compression and i is the packet number.

(5) On start of the talk spurt TS_{k+1}, map the arriving packets to the new constructed timeline A_i by applying a delay D_i to each packet in sequence in the talk spurt before playback where

$D_i = A_i - \text{sum}(I_i)$; if $D_i < 0$ *then* $D_i = 0$; where sum(Ii) is the summation of interarrival times in the talk spurt starting from packet 0 till packet i.

(5) Goto step (3).

This dejittering technique eliminates jitter effect by streamlining audio sampling at specific playback points in time. The inter-arrival time between audio packets played at the receiver is constant during every talk spurt. Audio generation pattern is restored during playback and user does not sense packet jitter. Figure 53 shows how the algorithm performs. The latency overhead introduced is monitored in order not to exceed an upper limit. It is also possible to pass the maximum inter-arrival jitter, M_k, through a low pass filter as the one described in section 6.3 for better results. The new value of M_k is derived

---

[1] The granularity of audio codec compression Gc is the time taken between successive audio buffers compression. Gc is 30 ms for G.723 High and Low rates.

from the following formula $M_k \leftarrow (1 - \alpha) M_k + \alpha\, b$ where b is the new value for the maximum computed inter-arrival jitter.

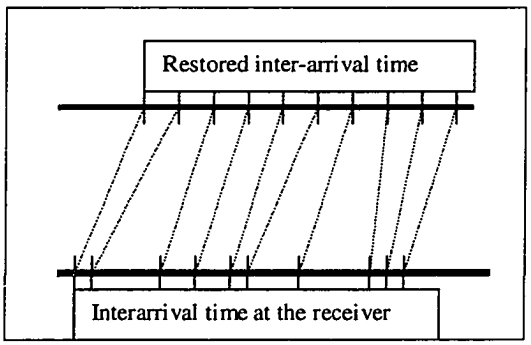

Figure 53. Redistribution of interarrival time by dejitter algorithm

6.6.2 Numerical Example

In order to verify the dejitter algorithm described in the previous section we provided numerical example in this section. Table 6 shows audio packets arriving at the receiver and then treated by the jitter compensation. It is assumed the G_c=30ms, *AsduSize=4* frames/packet. P(i) is the audio consecutive packets. Previous I(i) is the interarrival time of packets in the previous talk spurt. Previous J(i) denotes the differences in the interarrival time during the previous talk spurt. Current I(i) is the interarrival time between audio packets during the current talk spurt. D(i) is the delay incurred on the arriving audio packets because of the dejitter algorithm. A_i represents the new timeline constructed at the receiver to compensate for the jitter effect. Restored I(i) is the interarrival time between packets restored at the receiver after applying the dejitter algorithm. From Previous J(i) row we deduce that the maximum jitter for this example in the previous talk spurt was found to be 50 ms. We construct A_i given this information as shown in the A_i row. Applied D_i is then computed by subtracting previous interarrival time of packets until packet i from A_i.

The table shows that the algorithm restored the interarrival time of packets at the receiver to a constant value (G_c * AsduSize). This significantly helps the intelligibility of the audio playback at the receiver. Figure 54 shows how the interarrival time is mapped to the new scale Ai in order to compensate for the jitter.

Table 6 Numerical example for the dejitter algorithm

Time(ms)	P(0)	P(1)	P(2)	P(3)	P(4)	P(5)	P(6)
Previous I(i)	0	50	100	140	160	130	110
Previous J(i)	0	50	50	40	20	-30	-20
Current I(i)	0	100	150	80	90	180	130
New A_i	50	170	290	410	530	650	770
Restored I(i)	0	120	120	120	120	120	120
Applied D_i	50	70	40	80	110	50	40

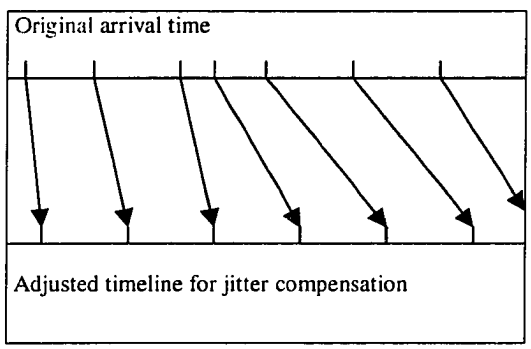

Figure 54. Jitter compensation algorithm example

6.6.3 Dejitter Algorithm Analysis

Audio interarrival time was recorded at the receiver side to study the jitter effect and compensation suggested by the dejitter algorithm. Data were collected during an Internet rush hour. Further, full video CIF was allowed to run side by side with audio to contribute to the jitter effect such that we achieve the worst case condition.

Figure 56 shows a plot of the mean and the maximum jitter per talkspurt. All maxima were below 900 ms with a mean of 490 ms. It is observed that the mean jitter per talkspurt is significantly lower than the corresponding maximum. This implies that using the maximum jitter per talkspurt in the dejitter algorithm can be overly pessimistic. An alternative will be to apply a delay equivalent to a multiplier of the previous talkspurt mean.

Figure 57 shows the penalty incurred on the audio packets at the receiver side because of the dejitter algorithm. During each talkspurt, delay will start on accumulating as each audio packet is mapped to its projected playback time according to the dejitter algorithm. The delay adds gradually following a saw-toothed curve. It is observed that short spurts have shorter delays and vice versa. Long talkspurts are not very likely during interactive conversations. Since the arrival data sample was taken at the worst possible conditions, it is expected that during hours of regular load the delay will be much better. The dejitter algorithm will playback all received audio data at regular intervals equal to that produced at the original source. As jitter effect is eliminated, audio quality will significantly improve.

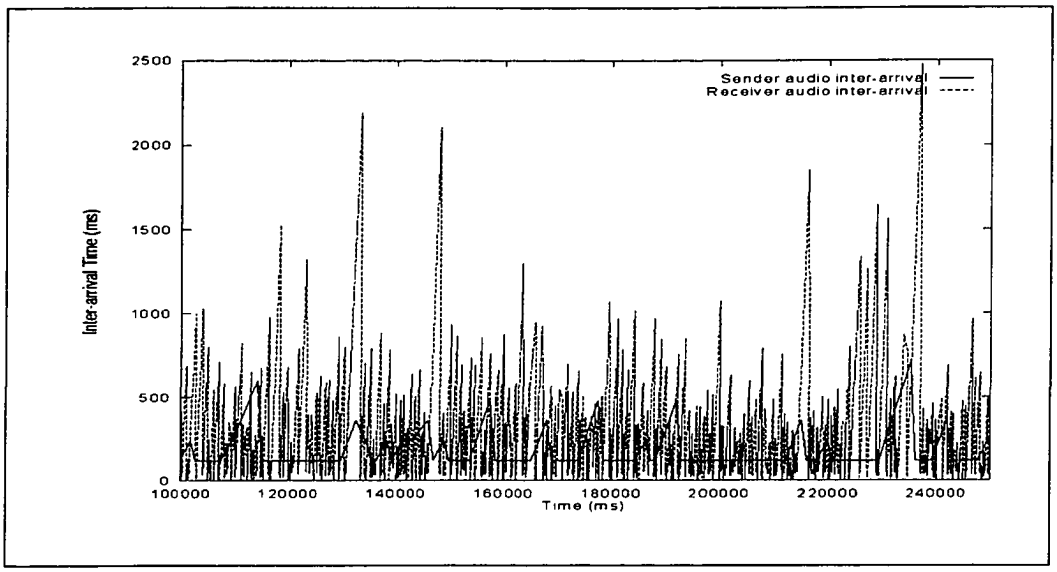

Figure 55. Interarrival time at sender and receiver for audio packets

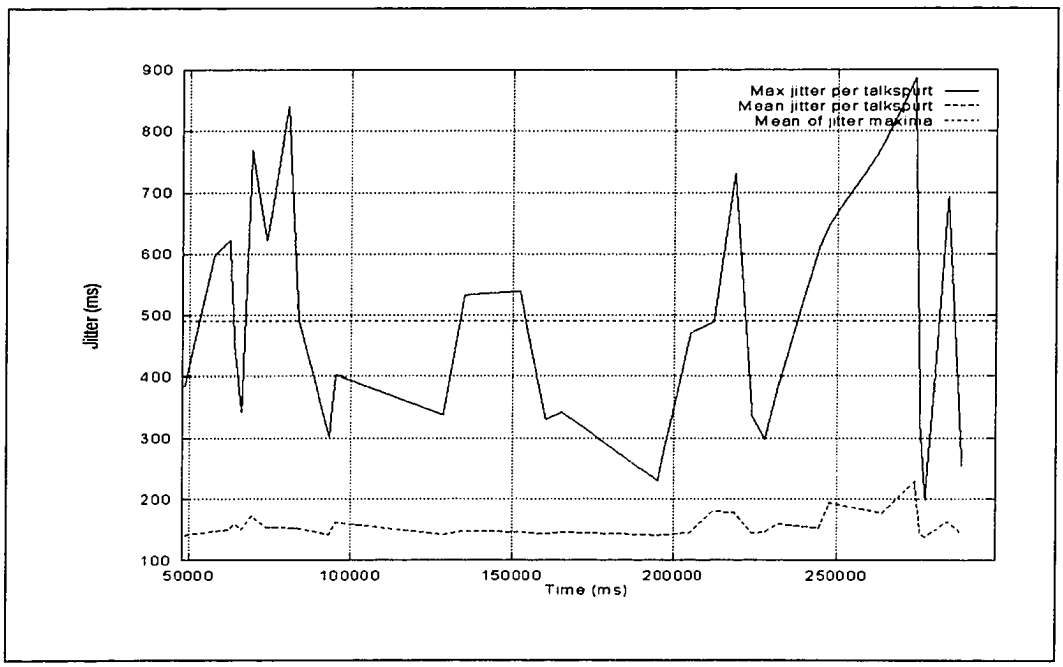

Figure 56. Mean and maximum computed jitter per talkspurt

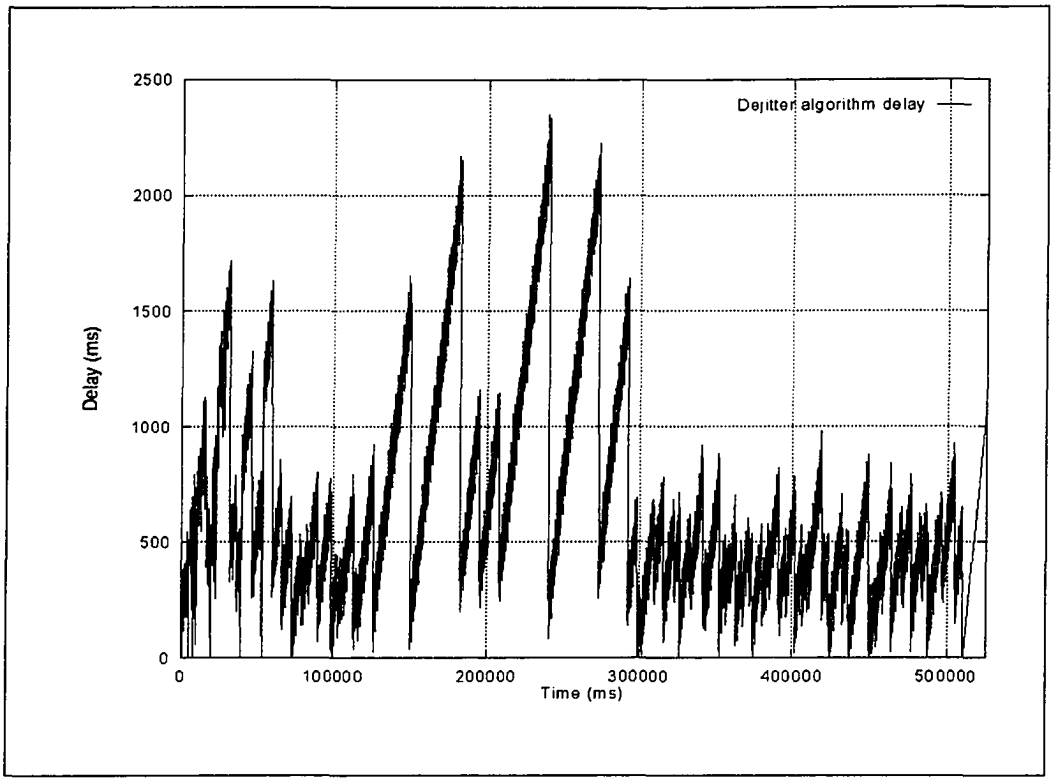

Figure 57. Delay penalty incurred on dejitter algorithm

6.7 Concluding Remarks

Multimedia conferencing applications can be rate controlled over Internet by continuously providing feedback information about its performance over the network. Typical useful statistics include packet loss, delay, jitter, and timestamps for media synchronization. RTCP is an associated protocol with RTP designed to handle the delivery of monitoring service of the RTP protocol. It provides feedback information periodically on the quality of media packets including timing information.

We introduced an RTCP-based feedback algorithm that recovers from audio loss by controlling video bit-rate. The algorithm adapted well to network condition, however the response was rather slow and tardy since the feedback cycle is larger than 6 seconds.

We presented the limitation of RTCP-based algorithms for bit rate control. We then attempted to discover a relationship between audio loss and packet interarrival jitter. We computed the jitter ant the packet loss at the receiver and plotted their relation with time. It was discovered that most of the significant packet losses were preceded by abrupt change in packet interarrival time or jitter. We proposed a new algorithm that can predict audio loss based on jitter computation at the receiver. We evaluated the performance of the algorithm to assess its efficiency. The algorithm predicted 96.7 % of all losses during a conferencing session. The optimal value for the jitter threshold that allows the lowest number of generated jitter warnings and provides prediction closure for a majority of packet loss was found to be 1.8 * jitter standard deviation. The loss prediction algorithm is more efficient than RTCP-based feedback algorithms since it does not require periodic occupation of the bandwidth. In addition, the algorithm is enforcing control as opposed to RTCP-based algorithms that only provide feedback indications.

We also proposed a new algorithm for making up for the audio jitter at the receiver by recovering audio original pattern generated by the encoder at the sender's side. We explained by example the performance of the algorithm. The dejitter algorithm can eliminate the jitter effect induced by the network effectively. The algorithm contributes significantly to the improvement of the audio quality as the jitter effect is significantly eliminated.

7. Conclusion and Future Work

7.1 Conclusion

This dissertation addressed the problem of host and network induced effects on the media components quality of service in a conferencing session. We showed how traffic parameters could affect the performance of audio and video. We compared a few common scheduling algorithms (FCFS and EDF) and arrived at the conclusion that they are not sufficient to solve the audio-video multiplexing problem alone. As we strived to solve the audio-video multiplexing problem at the host, we introduced a new performance qualifier for the audio performance. This qualifier is the number of interleaved video bytes between audio packets (VIB) due to multiplexing. We introduced an algorithm called the bit-rate adjuster (BRA) that proved to achieve better performance in terms of audio delay and jitter. We also studied congestion handling using feedback algorithms. We proposed and evaluated two feedback-based algorithms to handle audio loss. We also proposed and evaluated a scheme to eliminate network-induced jitter at the receiver. We arrived at the following conclusions:

- Packet size is an important traffic parameter that effects end to end latency and jitter. Audio packet size is determined by the number of audio frames (N_{af}) bundled in an audio packet. Video packet size is limited by the maximum size of a video fragment (M_{vf}).
- The smaller the number of audio frames per packet, the smaller the audio latency and the larger the packet overhead (i.e, the less the bandwidth utilization).
- For audio packets, a good choice for N_{af} that achieves a compromise between latency and packet overhead is 7-8 frames per packet.
- For audio packets, if IP/UDP header compression schemes are used, the choice of Naf can be reduced to 5-6 frames per packet.

- For video packets as the fragment size M_{vf} decreases, the number of generated video fragments increases. In addition, packet overhead increases and bandwidth utilization decreases. The mean size of all video packets was below 250 bytes for all fragment sizes used. A M_{vf} of 750 bytes exhibited a mean of 245 bytes with a deviation of 179.

- The choice of M_{vf} affects audio performance in the sense that audio packets may get stuck during the session behind huge video fragments. Despite the intuition, simulation results showed the mean interleaved video bytes between audio packets (VIB) for a video M_{vf} of 256 bytes is larger than the mean of the corresponding trace for an M_{vf} of 500 bytes. Further, as the M_{vf} decreases, audio latency may get even worse.

- Scheduling real-time video, audio, and data is analogous to scheduling a set of non-preemptable periodic tasks over a uniprocessor. We showed by simulation how Earliest Deadline First (EDF) algorithm improved local performance over First Come First Served (FCFS) case.

- Lack of a good scheduling mechanism can lead to an appreciable amount of unnecessary performance penalties on the host traffic locally before reaching the Internet. The most important penalty addressed is the variable time gap introduced between audio packets by video packets. This gap causes latency and jitter to the audio stream, which may result in more severe consequences such as unintelligible audio playback, or loss.

- Two reasons are identified as direct causes for the audio time gap problem. The first reason is the variable size of the video packet or fragment. The second reason is the burstiness of the video source. The second reason is solved by the bit-rate adjuster algorithm (BRA) while the first reason is left for future research.

- The BRA scheme showed significant improvement over FCFS algorithm. The improvement reached 28% for small fragment sizes. The artificial delay side effect

imposed by the algorithm was in fact less than expected. The mean delay was less than 250 ms for all fragment sizes.

- In order to economize on audio inter-packet gap using the BRA algorithm, it has been shown that best performance is achieved using a video fragment size limit greater than 256 bytes and smaller than 512 bytes.
- RTCP feedback algorithms are used to control audio loss over network connections indirectly. We introduced and evaluated a method that uses audio receive reports (ARR) to control video bit-rate in order to improve audio loss and performance. This method is a modification to the RTCP-based dynamic QoS algorithm introduced by Busse et al in order to accommodate limited bandwidth links.
- RTCP feedback algorithms have their limitations over slow links. Of the most important limitations are slow feedback response over slow links (interarrival time between RTCP reports may exceed 7 seconds), and QoS oscillations with performance going up and down periodically.
- Jitter can be used as an early predictor for network audio loss. This new algorithm showed 96.7 % prediction of all losses during the conferencing session for a jitter threshold of 0.6 the standard deviation. The optimal value that minimizes the number of jitter warnings generated and detects the maximum number of losses is 1.8 times the jitter standard deviation value.
- Jitter can be compensated at the receiver by applying a scheme that eliminates audio inter-arrival glitches. The algorithm restores audio generated pattern by adaptively introducing some computed delay value at the receiver.

7.2 Future Work

Further work is required in the area of fault tolerant protocols that provides forward error correction functionality at the RTP level. Some Internet drafts have been suggested and extra work is required by standard committees and various study groups to clear the ambiguity of the deployment of such algorithms in H.323 endpoints.

Interactive data such as games, whiteboard and application sharing has not been addressed in this study. Adding interactive data to the conference and balancing resources amongst different media components on desktop conferencing platforms is a challenging problem that requires investigation.

Further work is also required to integrate the jitter predictor algorithm introduced in this dissertation with multilayer video codecs (e.g., H.263+). Multilayer codecs provide essential signal and optional attributes. The receiver can choose to economize on its incoming bandwidth by reducing one or several layers of the optional attributes.

The Jitter compensation algorithm used the maximum jitter of the previous talk spurt. Sometimes this value can be overly pessimistic. Additional work is required to smooth the value of the maximum jitter by averaging over a set of historical values as explained in Section 6.6.1.

8. References

[1] Ahmadi H., and Denzel W., "A Survey of modern high Performance Switching techniques," *IEEE Journal on Selected Areas in Communications*, Vol. 7, No. 7, pp. 1091- 1103, September 1989.

[2] Aravind R., "Image and Video Coding Standards," *AT&T Technical Journal*, Vol. 72 No. 1, January/February 1993, pp. 67-89.

[3] Ballart R., "SONET: Now It's the Standard Optical Network," *IEEE Communications Magazine*, pp. 8-15, March 1989.

[4] Bertsekas D. and Gallager R., "*Data Networks,*" Prentice-Hall, Inc., 1992.

[5] Bolot J., Vega-Garcia A., "The Case for FEC-Based Error Control for Packet Audio in the Internet," *to appear in ACM Multimedia Systems 1998.*

[6] Brunell H., "Queuing Behavior of Statistical Multiplexers with Correlated Inputs," *IEEE Transactions on Communication*, Vol. 36, pp. 1339-1341, December 1988.

[7] Burgin J., and Dorman D., "Broadband ISDN Resource Management: The Role of Virtual Paths," *IEEE Communications Magazine*, pp. 44-48, September 1991.

[8] Burke C., Busch J., "*How to build an Internet Service Company,*" systems 2000+ Publishing, 1996.

[9] Busse I., Deffner B., Schulzrinne H., "Dynamic QoS Control of Multimedia Applications based on RTP," *Computer Communications*, January 1996.

[10] Casner S., Jacobson V., "Compressing IP/UDP/RTP Headers for Low-Speed Serial Links," *Internet Draft, draft-ietf-avt-crtp-03.txt*, July 1997.

[11] CCITT, "*Recommendations on BISDN,*" Geneve, January 1990.

[12] Chen J., and Guerin R., "Performance Study of an Input Queueing Packet Switch with Two Priority Classes," *IEEE Transactions in Communications*, Vol. 39, No. 1, pp. 117-126, January 1991.

[13] Cheshire S., "Bandwidth and Latency," http://www.d.umn.edu:81/~mchrist8/bw+lt.html

[14] Daniel A., "Introduction To Video And Audio Compression Techniques," Course Notes 6, *ACM Siggraph* 1994, Orlando FL, July 1994.

[15] Degermark M., Nordgren B., Pink S., "Header Compression for IPv6," *Internet-Draft*, draft-degermark-ipv6-hc-03.txt, July 1997.

[16] Dertouzos M., "Control Robotics: The Procedural Control of Physical Processes," *In Proceedings of the IFIP* Congress, 1974.

[17] Dittmann L., and Jacobson S.B., "Statistical Multiplexing of Identical Bursty Sources in an ATM network," *In Proceedings of Globecom'88*, Hollywood, pp. 1293-1297, November 1988.

[18] ElGuibaly, F., "Design and Analysis of Arbitration Protocols," *IEEE Transactions on Computers*, Vol. 38, No. 2, pp. 161 – 171, February 1989.

[19] Ferrari D., "Real-time Communication in an Internetwork," *Journal of High-Speed Networks,* Vol. 1, pp. 79-103, January 1992.

[20] Ferrari D., and Verma D., "Buffer Allocation for Real-time Channels in Packet Switching Networks," *Tech Rep. TR-90-022, International Computer Science Institute*, Berkeley, CA, June 1990.

[21] Ferrari D., and Verma D.C., "A Scheme for Real-Time Channel Establishment in Wide-Area Networks," *IEEE Journal on Selected Areas in Communications*, Vol. 8, No. 3, pp. 368-379, April 1990.

[22] Gianatti S., and Pattavina A., "Performance Analysis of Shared-buffered Banyan Networks under Arbitrary Traffic Patterns," *In Proceedings of Infocom'93*, San Francisco, pp. 943-952, March 1993.

[23] Habib I., and Saadawi T., "Multimedia Traffic Characteristics in broadband networks," *IEEE Communications Magazine*, pp. 48 – 54, July 1992.

[24] Hluchyj M. and Karol M., "Queuing in high-performance Packet Switching," *IEEE Journal on Selected Areas in Communication*, Vol. 6, No. 12, pp. 1587-1596, December 1988.

[25] HORN W., "Some Simple Scheduling Algorithms," Naval Research Logistics Quarterly, 21, 1974.

[26] Hui J.Y., "Resource Allocation for Broadband Networks," *IEEE Journal of Selected Areas in Communication*, Vol. 6, pp. 1593-1608, December 1988.

[27] ITU-T Recommendation G.723.1 (1996) – "Dual Rate Speech Coders for Multimedia Communication Transmitting at 5.3 & 6.3 kb/s".

[28] ITU-T Recommendation H.223 (1996) – "Multiplexing Protocol for Low Bit-rate Multimedia Communication".

[29] ITU-T Recommendation H.320 (1993) – "Narrow-band ISDN Visual Telephone systems and Terminal Equipment".

[30] ITU-T Recommendation H.323 (1996) – "Terminal for Low Bit-rate Multimedia Communication over Non-guaranteed Bandwidth Networks."

[31] ITU-T Recommendation H.324 (1996) – "Terminal for Low Bit-rate Multimedia Communication."

[32] ITU-T Recommendation H245 (1995) – "Control Protocol for Multimedia Communication."

[33] ITU-T Recommendation H263 (1996) – "Video Coding for Low Bit-rate Communication."

[34] ITU-T Recommendation H223 (1995) – "Multiplexing Protocol for Low Bit-rate Multimedia Communication."

[35] ITU-T Recommendation H.225 (1995) – "Media Stream Packetization and Synchronization on Non-Guaranteed Quality of Service LANs."

[36] ITU-T Recommendation G.711 (1988) – "Pulse Code Modulation (PCM) of Voice Frequencies."

[37] ITU-T Recommendation H.261 (1993) – "Video Codec for audiovisual services at p X 64 kb/s."

[38] Jacobson V., "Congestion Avoidance and Control," *ACM Computer Communication Review*, Vol. 18, pp. 314-329, Aug 1988.

[39] Jain R., "Congestion Control in Computer Networks: Issues and Trends," *IEEE Network Magazine*, pp. 24-30, May 1990.

[40] Jain R., and Routhier S., "Packet Trains: Measurements and a New Model for Network Traffic," *IEEE Journal on Selected Areas in Communications*, Vol. 4., No. 6, September 1986.

[41] Karlsson G. and Vetterli M., "Packet Video and its Integration into the Network Architecture," *IEEE Journal on Selected Areas in Communication*, Vol. 7(5), pp. 739-752, June 1989.

[42] Karol M., Hluchyj M., and Morgan S., "Input Versus Output Queuing on a Space-Division Packet Switch," *IEEE Transactions on Communications,* Vol. COM-35, No. 12, pp. 1347-1356, December 1987.

[43] Katevenis M., "Fast Switching and Fair Control of Congested Flow in Broadband Networks," *IEEE Journal on Selected Areas in Communication*, Vol. 5, No. 8, pp. 1315-1327, October 1987.

[44] Katevenis M., Sidiropoulos S., and Courcoubetis C., "Weighted Round-Robin Multiplexing in a General Purpose ATM Switch Chip," *IEEE JSAC special issue on Large Scale ATM Switching Systems for BISDN*, Fall 1991.

[45] Kleinrock, L. "Models for Computer Networks," Conference Record, IEEE International Conference on Communications, Boulder, Colorado, pp. 21-9 to 21-16, June 1969.

[46] Kleinrock, L., "Analytic and Simulation Methods in Computer Network Design," AFIPS Conference Proceedings, Vol. 36, Spring Joint Computer Conference, Atlantic City, New Jersey, May 1970, AFIPS Press, Montvale, New Jersey, pp. 569-579, 1970.

[47] Kleinrock, L. and W. Naylor, "On Measured Behavior of the ARPA Network," AFIPS Conference Proceedings, Vol. 43, National Computer Conference, Chicago, Illinois, May 1974, AFIPS Press, Montvale, New Jersey, pp. 767-780, 1974.

[48] Kleinrock, L., "Performance Models and Measurements of the ARPA Computer Network," ONLINE 72 Conference Proceedings, Vol. 2, Uxbridge, Middlesex, England, September 1972, Online Computer Systems Ltd., Brunel University, England, pp. 61-85, 1972. Also in Computer Communication Networks, Proceedings of the NATO Advanced Study Institute Series, Noordhoof, Leyden, The Netherlands, pp. 63-87, 1975.

[49] Kumar S., and Agrawal D., "On Design of a Shared-Buffer based ATM Switch for Broadband-ISDN," *In Proceedings of 13th International Pheonix Conference on Computers and Communications*, Arizona, pp. 377-383, April 1994.

[50] LBNL, "Audio Conferencing Tool (VAT)," http://www-nrg.ee.lbl.gov/vat/.

[51] Le boudec J-Y., "The Asynchronous Transfer Mode: a tutorial," *Computer Networks and ISDN Systems*, No. 24, pp. 279-309, 1992.

[52] Marshall W.T. and Morgan S. P., "Statistics of Mixed Data Traffic on a Local Area Network," *Computer Networks and ISDN Systems*, Vol. 10, No. 3-4, October-November 1985.

[53] Mogul J., and Deering S., "Path MTU Discovery," *RFC 1191*, April 1990.

[54] Ohnishi H., Okada T. and Noguchi K., "Flow Control Schemes and Delay/Loss Tradeoff in ATM Networks," *IEEE Journal on Selected Areas in Communication*, Vol. 6(9). pp. 1609-1616, December 1988.

[55] Pan D., "Digital Audio Compression," *Digital Technical Journal*, Vol. 5 No. 2, Spring 1993, pp. 28-40.

[56] Pohlmann K., "*Principles of Digital Audio,*" Howard W. Sams & Co., 1985.

[57] Poynton C., "Frequently Asked Questions about Colour," May 28 1995, http://www.inforamp.net/~poynton/Poynton-colour.html.

[58] Schulzrinne H., "RTP Profile for Audio and Video Conferences with Minimal Control," *RFC 1890*, January 1996.

[59] Schulzrinne H., Casner S., Frederick R., and Jacobson V., "RTP: A Transport Protocol for Real-Time Applications," RFC 1889, January 1996.

[60] Sen P., Maglaris B., Rikll N. and Anastassiou D., "Models for packet switching of variable bit-rate," *IEEE Journal of Selected Areas in Communication,* Vol. 7, No. 5, pp. 865 – 869 June 1989.

[61] Shannon C., "A Mathematical Theory of Communication," Bell System Journal, Vol. 28, pp. 656-715, October 1949.

[62] Simpson W., "The Point-to-Point Protocol (PPP)," RFC 1548, December 1993.

[63] Smith, K., and Brushteyn D., "Internet Telephony and Voice Compression," http://www.kgw.tu-berlin.de/~mengel/SpeechTech/HLTsurvey.html

[64] Stevens W., "*TCP/IP Illustrated Volume 1: The Protocols,"* Addison-Wesley Longman Publishers, 1994.

[65] Tanenbaum A., "*Computer Networks*," Prentice Hall, 1981.

[66] Taylor K. and Tolly K., "Desktop Videoconferencing: Not Ready for Prime Time," *Data Communications*, Vol. 24 No. 5, April 1995, pp. 64-80.

[67] Tobagi F.A., "Fast Packet Switch Architectures for Broadband Integrated Services," *Proceedings of the IEEE*, Vol. 78, pp. 133-167. January 1990.

[68] Turletti T., Parisis S., and Bolot J., "Experiments with a Layered Transmission Scheme over the Internet," *to appear in Infocom'98.*

[69] Verbiest W., Pinnoo L., and Voeten B., "The Impact of the ATM Concept on Video Coding," *IEEE Journal of selected Areas in Communication*, Vol. 6, pp. 1623-1632, December 1988.

[70] Witte E., "A Quantitative Comparison of Architectures for ATM Switching Systems," *wucs-91-47 Technical Report*, Washington University, October 1991.

[71] Woodruff G.M., and Kositpaiboon R., "Multimedia Traffic Management Principles for Guaranteed ATM Network Performance," *IEEE Journal on Selected Areas in Communications*, Vol. 8, pp. 437-446, April 1990.

[72] Zhang, Q. and Ferrari D., "Rate-Controlled Static-Priority Queuing," *In Proceedings of IEEE Infocom'93*, pp. 227-236, April 1993.

9. Appendix

9.1 A.1 Symbols and Abbreviations

μ-law	PCM audio encoding technique
A-law	PCM audio encoding technique
ARR	Audio Receive Report
ATM	Asynchronous Transfer mode
B-ISDN	Broadband – integrated Services Digital Network
CIF	Common Intermediate Format (video picture size)
Dev	Deviation
EDF	Earliest Deadline First
FCFS	First Come First Served
Frag	Fragment
G.711	ITU-T Recommendation: Pulse Code Modulation (PCM) of Voice Frequencies
G.723.1	ITU-T Recommendation: Dual Rate Speech codec for multimedia telecommunications transmitting at 6.4 and 5.3 kbit/s
H.223	ITU-T Recommendation: Multiplexing Protocol for Low Bit-rate Multimedia Communication
H.225	ITU-T Recommendation: Media Stream Packetization and Synchronization for Visual Telephone Systems on Non-Guaranteed Quality of Service LANs
H.245	ITU-T Recommendation: Control of communications between visual telephone systems and terminal equipment.

H.261	ITU-T Recommendation: Video CODEC for audiovisual services at p X 64 kbit/s
H.263	ITU-T Recommendation: Video CODEC for narrow telecommunications channels.
H.320	ITU-T Recommendation: Narrow-band ISDN visual telephone systems and terminal equipment
H.323	ITU-T Recommendation: Line transmission of non-telephone signals over packet based multimedia communication systems.
H.324	ITU-T Recommendation: Terminal for low-bitrate multimedia communication.
IP	Internet Protocol
ITU-T	Telecommunication standardization sector of the International Telecommunication Union
LFRC	Local Feedback Remote Control
Max	Maximum
MOS	Mean Opinion Score
M_{vf}	maximum video fragment size
N_{af}	number of audio frames in an audio packet.
PCM	Pulse Code Modulation
PPP	Point-to-Point Protocol
QCIF	Quarter CIF
QoS	Quality of Service
RFLC	Remote Feedback Local Control

RTCP	Real-time Control Protocol
RTP	Real-time Protocol
SQCIF	Sub Quarter CIF
TCP	Transport Communication Protocol
UDP	User Datagram Protocol
VIB	Video bytes interleaving audio packets.