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.

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.

Bell & Howell Information and Learning
300 North Zeeb Road, Ann Arbor, MI 48106-1346 USA
800-521-0600

Université d'Ottawa · University of Ottawa

Guaranteeing QoS in the Transmission of MPEG-2 Video over IP-Based Networks

By

© Alireza A. Nezhad

A thesis submitted to
The School of Graduate Studies and Research
University of Ottawa
in partial fulfillment of the requirements
for the degree of

Masters of Applied Science
in Electrical Engineering

Ottawa-Carleton Institute for Electrical Engineering
School of Information Technology and Engineering
Faculty of Engineering
University of Ottawa
Ottawa, Ontario, Canada

May 1999

ACKNOWLEDGMENTS

I would like to thank my supervisor, Dr. Luis Orozco Barbosa, for all his guidance and advice throughout this thesis and the rest of my graduate program.

I am also indebted to Dr. Pedro Sanchez for his continuous help and his involvement, especially with the practical aspects of this project.

I would like to thank Nortel Networks Inc. for financially and technically supporting this project.

Finally, a special thank you and my gratitude go to my friend, Mr. Gilles Gagnon whose free-of-expectation help with *mpegstat* and the video decoder made this work possible.

DEDICATION

To my precious wife, Sahar

for her support and her understanding

ABSTRACT

Multimedia applications, have stringent quality of service (QoS) requirements. These requirements are application specific and as far as protocol stacks are concerned, they are layer specific. It is essential to satisfy these requirements at all the system components along the path from a service provider to a service user, starting from and ending at the end points. It is equally important to develop methods to translate QoS requirements of an application into QoS parameters at different layers as well as implement architectures for the global management of QoS.

This thesis includes an analytical study of buffer and jitter requirements in the transmission of digital video as well as an experimental study of QoS requirements and QoS mapping for the transmission of MPEG-2 video over IP-based networks and Ethernet in particular. First, we have examined the buffer constraints of the video server and the video clients in a video on demand system and found the constraints on the channel rate in order to avoid buffer overflow and/or underflow.

In our jitter analysis, we found constraints on the burstiness of the traffic from the point of view of a video client in a video on demand system. Again, the study has focused on determining the lower and the upper bounds on the traffic arrival rate at the client so that the possibility of buffer underflow and/or overflow is avoided. The analysis of the underflow conditions results in a relationship between the maximum acceptable jitter by the user and the minimum frame rate maintained by the network. The overflow conditions provide us with the minimum required buffer size for the client.

In the experimental phase of our work, using a video on demand (VOD) application developed as part of these research efforts, various statistical metrics have been considered in order to characterize the network traffic. Some metrics have proven to reflect QoS as perceived by the end-user of an MPEG-2 video application better than others. The coefficient of variation of inter-arrival times of IP-packets belonging to the video stream seems to be the most appropriate one among the statistics considered in this study.

TABLE OF CONTENTS

TABLE OF FIGURES

LIST OF TABLES

TABLE OF ACRONYMS

ABA	Average Bandwidth Allocation
ATM	Asynchronous Transfer Mode
BEB	Binary Exponential Back-off
B-ISDN	Broadband Integrated Services Digital Networks
CBA	Critical Bandwidth Allocation
CBR	Constant Bit Rate
COV	Coefficient Of Variation
CSMA/CD	Carrier Sense Multiple Access with Collision Detection
CPU	Central Processing Unit
DCT	Discrete Cosine Transform
GOP	Group OF Pictures
HDTV	High Definition Television
IDC	Index of Dispersion for Counts
IDI	Index of Dispersion for Inter-arrivals
IETF	Internet Engineering Task Force
I/O	Input/ Output
IP	Internet Protocols
ISO	International Standards Organization
JPEG	Joint Photographic Experts Group
LAN	Local Area Network
MAC	Medium Access Control
MAD	Mean Absolute Deviation

MCBA	Minimum Critical Bandwidth Allocation
MOD	Multimedia On Demand
MPEG	Moving Picture Experts Group
MVBA	Minimum Variability Bandwidth Allocation
NIC	Network Interface Card
NTSC	National Television Systems Committee
OSI	Open Systems Interconnection
PCRTT	Piecewise Constant Rate Transmission and Transport
PDF	Probability Density Function
PDU	Protocol Data Unit
PS	program Stream
QoS	Quality of Service
RSVP	Resource reservation setup Protocol
RTC	Real Time Clock
RTP	Real Time Protocol
STD	Standard Deviation
TCP	Transmission Control Protocol
TS	Transport Stream
UDP	User datagram Protocol
VBR	Variable Bit Rate
VOD	Video On Demand
WAN	Wide Area Network

CHAPTER I

INTRODUCTION

I.1 Background

Legacy telecommunications infrastructure especially IP-based networks are designed for a small class of applications that are not time-dependent, such as textual electronic mail and file transfer. Even supported interactive applications such as telnet do not have strict time constraints. However, a new generation of real-time applications like audio/video on demand, real audio/video, videoconferencing, etc. has emerged in the past few years. These types of applications usually referred to as *distributed multimedia applications* have completely changed the face of traditional network communications. QoS (Quality of Service) requirements of these applications can not be satisfied by the simple existing protocols. Current protocol stacks such as Open Systems Interconnection (OSI) Reference Model and TCP/IP lack some functionalities needed for multimedia transmission, e.g. negotiation, reconfiguration and maintenance of QoS requirements[1] especially allocation of resources. Real-time systems, unlike conventional text-based services, require continuous access to the network (hence called continuous media). When contention for resources increases, traditional protocols fail to support multimedia communications. The provision of QoS for such applications is an end-to-end task. Performance of all the components on the path including operating systems, protocol stacks, network, I/O devices etc. regarding QoS parameters such as throughput, delay, delay jitter and loss rate should be guaranteed.[2] Inadequacy of even one of these elements results in the failure of a distributed multimedia system. Thus, individual components must all perform in a cooperative manner in order to meet the overall QoS specifications.

---

[1] In section II.1 we will try to clarify on the meaning of some of the terms that are frequently but vaguely used in the context of quality of service such as QoS requirements, QoS parameters etc.

[2] Two main classes of QoS in IETF Integrated Services model are: (i) Guaranteed QoS, where firm delay bounds, lossless queuing and so on are provided, intended for applications with stringent real-time delivery requirements; (ii) Predictive QoS, where no firm quantitative guarantees are provided, suitable for tolerant real time applications [26].

This requires them to work each as a part of an integrated architecture whose goal is to provide users of real-time applications with a satisfactory quality of service.

QoS requirements are application specific but in any case they can be defined as a set of objective (quantitative) and subjective (qualitative) parameters. Objective parameters such as delay and delay variation bounds can be measured. Subjective parameters such as voice quality and video resolution, on the other hand, are perceived by the end-user [1].

Interpretation of quality of service is layer specific. For example, quality of video for the user may be expressed in terms of frame rate, window size, color orientation, etc. but at lower layers, for instance network, these factors bear no meaning. Instead, throughput and loss rate can be used to talk about the quality of service at the network level. Therefore, quality of service at each layer in a QoS architecture must be assessed appropriately.

On the other hand, the role of all the layers in a QoS architecture is to guarantee the quality of service required by an application and eventually an end-user. End-users should be able to communicate their needs to lower layers without having to describe them in terms not understood easily by them, such as bitrate, delay bound, etc. Besides, any two interfaced layers should be able to communicate. Thus, there is need for methods that make the translation of QoS parameters between different layers possible. These methods are called *QoS mapping*.

I.2 Thesis Objectives

Keeping in mind that providing QoS is an end-to-end task, the focus of this project has been on studying the relationship between QoS parameters on the application, transport and network layers in transmission of MPEG-2 video over an IP-based network. Specifically, we have been interested in finding a set of *rules* that would enable us to

make a valid statement about the quality of video as perceived by the end user only by looking at network-level metrics associated with that particular traffic.

We focus our research on a network environment that enables a subscriber to a video service, to request a certain level of quality of service. Under this scenario, the service provider would then have to make sure that the subscriber's requirements are met. Hence, the quality of service should be continually monitored. For that, the service providers would need some criteria enabling them to judge the quality of service as perceived by the end user. Under this scenario, the service provider will base his/ her judgment on the information gathered by looking at the traffic statistics obtained at the network level.

Assuming that such a detection capability is achieved, then some control mechanisms have to be put in place in order to keep the quality of service above the requested level. The block diagram in Figure1 depicts this idea.

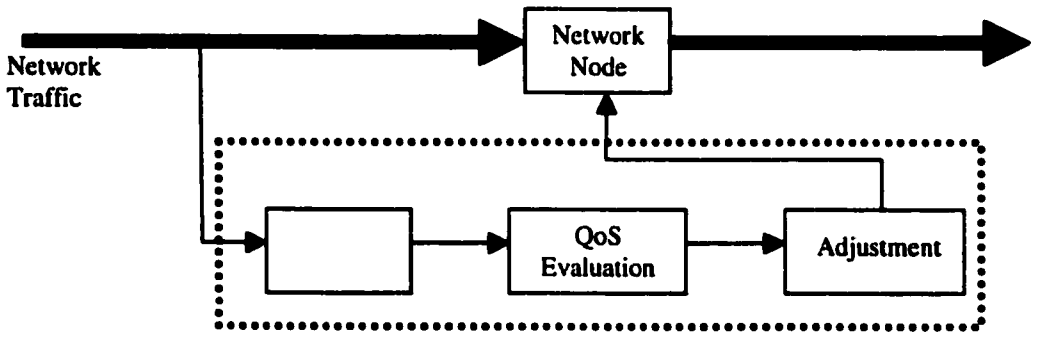

Figure 1. Dynamic QoS control through traffic monitoring

In such a structure, the nodes in a network should be equipped with a "QoS Control Center" enabling them to monitor the traffic on the network and filter different data-streams without interference and according to some criterion such as source or destination address, protocol and so on. Then, using the rules obtained through this study it would evaluate the quality of service and dynamically trigger an adjustment process if needed.

The objective of this thesis is to study QoS mapping relationships for the transmission of MPEG2 video over IP networks and also study other system requirements such as buffer sizes and jitter constraints. We have taken an approach that is a combination of lab work for the first part and analysis for the second.

I.3 Video over IP/Ethernet

Although high-speed networks such as B-ISDN/ATM seem to be the media of choice for video transmission (MPEG2 in particular), studies such as [40,43-53] have shown that video transmission over a medium such as Ethernet is feasible. The results in [40] have shown that performance is actually limited by display technologies and TCP/IP protocol. The authors conclude that improvements of compression schemes and communication protocols will soon make video transmission over TCP/IP-based networks (specially the Internet) a reality. One of the main parameters affecting the overall quality of service is the size of PDU (Protocol Data Unit) at the transport and network layer. As mentioned in [42], the longer the packets the less the number of attempts to access the medium which in turn means fewer collisions in environments like Ethernet and consequently shorter delays. On the other hand, as noticed in [41] longer packets cause longer packetization delay which is greatly crucial in real-time applications. It is therefore clear that the optimal packet size has to take into account the packetization delay and access mechanism to the underlying channels.

F.A.Tobagi et al. [43,44] have performed an evaluation of 10Base-T and 100Base-T Ethernet carrying video, audio and data traffic. In their simulations, they considered a protocol architecture consisting of the application, transport and MAC protocol mechanisms. In their conclusions, they acknowledge that the loss rate alone does not give a clear indication of the quality of service perceived by the end-user. The syntax and rate being used by the video encoding scheme play a major role.

In Ethernet, large delay variations (jitter) can be caused by the capture effect of the Binary Exponential Back-off (BEB) algorithm [49,51]. When two nodes collide, the winning node is likely to continue succeeding over the second node, because the second node's back-off range keeps increasing while the first one has reset its back-off range to its minimum. So, BEB algorithm favors the winning node. New algorithms have been proposed to solve this problem [51,52]. Also, a modification of CSMA/CD has been proposed in [53] in order to improve the performance of Ethernet in carrying multimedia traffic. This protocol has reportedly improved the overall utilization of the network as well.

S.Deng et al. [45] have investigated the performance of a residential Ethernet network in providing multimedia applications. They have found that jitter for a residential Ethernet is negligible since collisions are resolved soon enough. K.M.Nicholas [46] has noticed the effectiveness of a play-out buffer at the receiver in improving the delay performance of Ethernet.

In [47], a simulation study using data traffic traces has been conducted to evaluate the performance of a video-telephony application over Ethernet. This study shows that trace-driven traffics compared to Poisson traffic models, produce highly variable packet delays and a higher packet loss rate. In order to compensate for these effects, a new delay control scheme based on a timed packet-dropping algorithm is suggested. Simulations have shown that improvements in real-time loss rates can be achieved.

In [50], feasibility of H.261 encoded video transmission over Ethernet has been investigated. According to the performed simulations, a large number of real-time video streams with acceptable delay and loss performance can be supported at channel utilization of up to 50-60%.

In [39], Mellaney et al. have explored the impact of parameter translation over a system consisting of two Ethernets interconnected by an ATM network. One of their

main findings was that the existence of various networks has a significant impact over the system performance.

I.3 Thesis Organization

On the course of this project, a test-bed was developed in DCNlab at Nortel's Kanata site in Ottawa including a video transmission application that imitated a video on demand system and performed application level measurements. Due to its vast usage in LANs, Ethernet was selected as the physical layer. In the following sections we will elaborate on the specifics of this test-bed.

In this thesis, the findings of our experiments have been included along with some analytical study of jitter that, unfortunately, was not linked to an experimental work due to some practical limitations. This thesis has been organized as follows. In chapter I, we provided a background on the QoS issues and concepts in real-time applications. We also outlined the objectives of the thesis as well as a justification of its goals and methods. Chapter II concerns multimedia communications and quality of service requirements of real-time applications and many works that have been done in this area. In particular, we have focused on stored video applications (consistent with the development of our VOD software) and their quality of service requirements, namely loss and jitter. Chapter III describes the set-up for the experimental part of our project. Chapter IV defines the performance metrics of interest as well as the measurement tools used during the experiments. This chapter also provides a detailed analysis of the experimental results and insights of how the information obtained from the measurements can be used as indicators of the QoS perceived by the end-users. Finally, chapter VI draws our conclusions and possible future extensions of these research efforts.

CHAPTER II

MULTIMEDIA COMMUNICATIONS

In the past few years, the demand for incorporating audiovisual applications into traditional distributed text-based and graphics-based applications has been increasing, giving rise to a new type of services called *distributed multimedia communications.* These types of applications consist of time-critical data and have stringent requirements that did not exist in older types of media. For example, in a videoconferencing application a delay of more than 250ms in video may render that piece of information unusable [50]. A delay jitter (variation in delays of pictures) of greater than a certain level may manifest itself as a discontinuity in the video. On the other hand, most of these applications are more resilient to losses because the lost data causes only some degradation in quality, which may not be critical except in some special cases such as remote medical applications.

The notion of QoS guarantee addresses methods for satisfying user-specified requirements. Distributed multimedia applications can be categorized as conversational (e.g. videoconferencing) or presentational (e.g. video-on-demand). The application type has a significant implication on the user's QoS requirements. For example, conversational applications, due to their interactive nature, are delay sensitive while presentational applications are mostly loss sensitive because they are expected to offer high quality.

Providing QoS in a distributed multimedia system involves three steps:

1. Assessing user-level QoS requirements in subjective terms meaningful to human e.g. frame-rate, resolution and window-size for video.
2. Mapping these requirements onto QoS parameters of different components on the path from a sender to a receiver, for example, bandwidth of the network, packet loss rate of the transport layer or CPU-usage in the end system.

3. Putting in place a mechanism (architecture), enabling the system to provide negotiation capability between different components and layers of the protocol stack, to ensure the overall QoS in an end-to-end fashion.

In this project, our emphasis has been on the second step.

II.1 QoS Terminology

Despite its vast usage, the term "quality of service" and related concepts have never been uniformly and precisely defined. In this document, we follow the formal definitions given by the ISO/ITU-T standards document for quality of service [31].

Quality of service: A set of qualities related to the provision of an (N)-service, as perceived by an (N)-service user.

QoS characteristics: Some aspect of quality of service that can be quantified ...It is defined independently of the means by which it is represented or controlled.

QoS requirement: QoS information that expresses part or all of a requirement to manage one or more QoS characteristics; ...when conveyed between entities, QoS requirement is expressed in terms of QoS parameters.

QoS parameters: A vector or scalar value relating to QoS that is conveyed between entities.

QoS guarantees and mechanisms: Meeting a QoS requirement may require the use of mechanisms for QoS establishment, QoS monitoring, QoS alert, QoS maintenance, QoS control or QoS inquiry.

QoS management function: The general term for a function designed to meet a QoS requirement.

II.2 QoS Architectures

Although it has been known for a long time that guaranteed QoS is an end-to-end issue, until recently most of QoS-related researches concentrated on individual layers.

Especially, many methods were developed for providing QoS at the physical and network layers (resource reservation protocols, differentiated services, priority queuing,...). Fortunately, in the past few years there has been a growing interest in the end-to-end aspect of QoS. There is an abundance of work on QoS architectures. A thorough review of some of these studies can be found in [3]. In this article, elements of a generalized QoS framework are identified and layer-specific quality of service management is discussed. In [4] Vogel et al present a functional relationship between different layers of a distributed system that intends to provide QoS guarantees. This study considers protocols at all levels of the architecture, namely, physical-layer, network, transport layer and application protocols.

The research presented in [20] suggests a QoS metric that takes into account the role of different parameters affecting the quality of service. In [8] Nahrstedt and Smith develop the model of a "QoS Broker" within an end-point that helps a real-time application specify its requirements and translate these requirements into resource allocations through negotiation with the operating system and the network. Alfano [9] has designed a cooperative multimedia environment where users can remotely share applications in real-time and communicate audio-visually at the same time. This system includes a QoS mapper that translates user QoS requirements into parameters for the media service and into QoS parameters for the underlying resources such as host and network resources.

One of the earliest end-to-end treatments of QoS can be found in [11]. In this article, a "Resource Negotiation Protocol" has been implemented that allows QoS negotiations from a MOD (Multimedia on Demand) server application to the storage system, network and the client. Admission control and scheduling concepts have been used to make this protocol able to work through all these components.

[13] and [16] were among the first to introduce a layered model for QoS management. [13] also addresses the dynamic nature of QoS. Dynamicity means the possibility of changing QoS requirements based on availability of resources, e.g.

degrading a colored video to a b/w video. The work in [16] is based on the architecture proposed in the OSI Reference Model extension work. However, both of these studies fail to provide a methodology for mapping QoS requirements of the application into QoS parameters of the communication subsystem. This issue was dealt with in a similar model introduced in [12].

In [14] a functional and computational architecture is designed in a QoS driven way for a news-on-demand system as a case study. In their experiments, the authors have used the concept of QoS interfaces to handle QoS negotiations. Then, different QoS negotiation protocols have been investigated.

[18] introduces the notion of distributed QoS management agents, which has been also the motivation force behind our own study. In this architecture, application-oriented QoS agents are distributed throughout the network and the end systems, constantly monitoring the QoS level and communicating among each other. Some advantages of a distributed QoS management process are [18]:

1. An easier and more precise localization of the cause of QoS problems.
2. Better knowledge of local situations.
3. A lower complexity for a single QoS agent.
4. An increase in possible actions.

In [10], Bom et al have designed a multimedia architecture with QoS control that not only handles the translation of user-level QoS parameters into system-level parameters but also adapts the level of requested QoS according to the network and end-system conditions. In other words, if the amount of available resources is not enough to continue transmission with high quality, it will reduce the requirements of the application (e.g. exchange a colored video for a black and white) and in some cases it may consult the application for critical decisions such as termination of transmission. Authors have recognized the inadequacy of TCP as a transport protocol for continuous media transmission as the result of the following:

1. The "slow start" and congestion control mechanisms used in TCP impose a transmission rate slower than what is needed by the application.
2. Retransmissions in TCP are useless in the context of real-time applications, most of the time, and cause more congestion and loss.

Therefore, RTP (Real Time Protocol) [30] has been adopted in this system and many others. RTP owes its superior performance to the following properties [10]:

1. It does not have an error correction mechanism. Instead, it has a control protocol called RTCP (Real Time Control Protocol) that exchanges information about some monitored parameters such as loss rate and jitter between the sender and the receiver.
2. It uses Application Layer Framing (ALF): So, the unit of communication has a meaning to the application.
3. It has Integrated Layer Processing (ILP): This approach reduces the processing overhead of several layers.

RTP is a generic protocol, independent of media being transported.

Finally, [19] presents a transport performance QoS parameter set which enables applications to describe their required performance properties and guarantees. An adaptive transport system is designed using which, applications can configure their required transport service individually. A transport service is the service needed for transferring messages end-to-end through the network without considering network technology. In fact, this model addresses the shortcomings of conventional transport services. In this framework, performance of a stream is characterized by parameters denoting the size, interval, delay, jitter and loss behavior of its Transport Service Data Units (TSDU). In other words, quality of service of underlying components is reflected in performance QoS parameters of the transport service.

II.3 QoS Mapping

An important aspect of end-to-end QoS provision concerns the mapping of QoS between different layers. Lower level QoS parameters (bitrate, data loss rate, packet delay

bounds...) are often meaningless to end-users. However, in order to be able to provide the required QoS to a particular user, these lower levels need to understand user-level and application-level requirements and properly setup the network resources, e.g., CPU, switching elements, protocol mechanisms and so on. Therefore, a mapping scheme between user-level specifications and parameters of all protocol layers is essential.

In the QoS architectures mentioned previously, although the functional relationships between separate layers are explained and required QoS mapping mechanisms are implemented there is not enough information about the rules according to which, QoS parameters have to be translated. In other words, it is not clear in what language, different entities of a distributed multimedia system should communicate. However, there have been some efforts in this direction by other researchers. In [5], the authors have suggested and verified a simple relationship between application-level and network-level parameters such as PDU (Protocol Data unit) maximum rate, PDU maximum loss rate, and PDU maximum size, among others. Banerjee [7] presents a simulation study of application-level performance based on cell-level QoS parameters in ATM networks. In [9], a QoS mapping study for JPEG video and audio has been conducted using a group of 10 observers. In these experiments, a 5-level scale for user-perceived QoS suggested in [2], has been used, where 5 represents the best quality. The results of mapping user-level QoS requirements (according to this scale) into application level QoS requirements (in terms of frame rate for video and coding scheme for audio) and also a translation of these requirements into network parameters (bandwidth) and CPU usage are provided in a tabular form.

Voran and Wolf [6] have found an objective video assessment technique that can be considered a QoS mapping method between application level and user-level. This technique is based on normalized energy difference between sobel-filtered video frames for the measurement of spatial impairments in a video, and the first order frame difference sequence for the measurement of temporal impairments.

In [38], Cuenca et al. have explored the impact of losses of various pieces of information in MPEG-2 video streams. They have concluded that the cell loss rates alone do not give a clear indication of the QoS perceived by the end users. In [23], Cobley and Davies explore the issues related to the mapping of QoS parameters in an IP/ATM environment. As part of their experimental work, they analyze the interconnection of an IP/ATM system architecture. The main objective of their study has been to explore the effect of the parameter translation into the QoS perceived by the application.

II.4 Stored Video

An example of a presentational multimedia application is transmission of a video sequence that has been previously encoded and stored. This application can be, for example, a video-on-demand (VOD) system. As was mentioned before, transmission delay in such an application is not very important. For instance, it is not too disturbing to receive each picture of a movie a few seconds after it has been transmitted as long as all the pictures experience the same amount of delay. In this case, the display of the movie starts a few seconds after it has been requested. However, if during the playback of the movie some pictures experience longer delays due to network congestion or other reasons, a discontinuity in the video may be noticed. On the other hand, if for some reason such as fast frame transmission rate (the term *frame* may be used to refer to a video picture throughout this document), some pictures take a shorter time to be received, the receiver will have to drop them or keep them in a buffer until their designated display time. It has to be noted that video is normally displayed at a fixed rate, e.g. 30 frames per second for the NTSC system. Because no buffer has an infinite size, some pictures may be lost due to overflow at the receiver's buffers. The random variation of delay in multimedia applications is called *delay jitter.*

Similarly, data loss may happen in the sender side. Suppose that for some reason such as network congestion, the transmission rate of the pictures is less than the rate at which they are retrieved from the storage device and sent to the network interface.

Remember that the sender is required to maintain a minimum frame rate, which is the display rate of the video. In this case, the buffer at the sender may overflow.

Thus, the two factors affecting QoS when delivering stored video can be identified to be: 1) delay jitter and 2) packet losses in the buffers at both ends. Numerous techniques for reducing jitter and loss in real-time applications have been proposed in the literature [27-37]. At the core of all these methods, some principles are notable. One is employing buffers at the receiver to encounter jitter. The others are using buffers along with clever traffic smoothing, scheduling and flow control schemes at the sender and network nodes to reduce jitter and loss rate. For stored video, due to prior knowledge of frame sizes, some special techniques can be applied [32,37].

In the following three sections we will provide an analytical approach to understanding the loss and jitter constraints of a stored video transmission application. Our models although tailored to suit our experimental system, explained later, are generic and can be applied to a wide range of similar system configurations. Moreover, the results obtained from our analysis may be applied to real-time applications where buffering of at least one frame at the sender is possible without the operation of the system being disrupted due to the delay introduced by this buffering process.

Throughout this work, in the analysis and the experiments, as explained in the thesis objectives, we have tried to explore relationships between application and user level parameters (e.g. frame sizes and jitter) to network parameters (e.g. throughput and loss rate). The results of this study can be used to develop QoS mapping rules or be employed in a QoS architecture to configure communications subsystem such as network resources. Especially, the results of our analytical study can be used in bandwidth negotiations with network management.

II.5 Buffer Constraints

In this section, we analyze the constraints on the network throughput for a stored video transmission application imposed by the sizes of the buffers at the sender and the receiver. Figure 2 depicts a general representation of such a system.

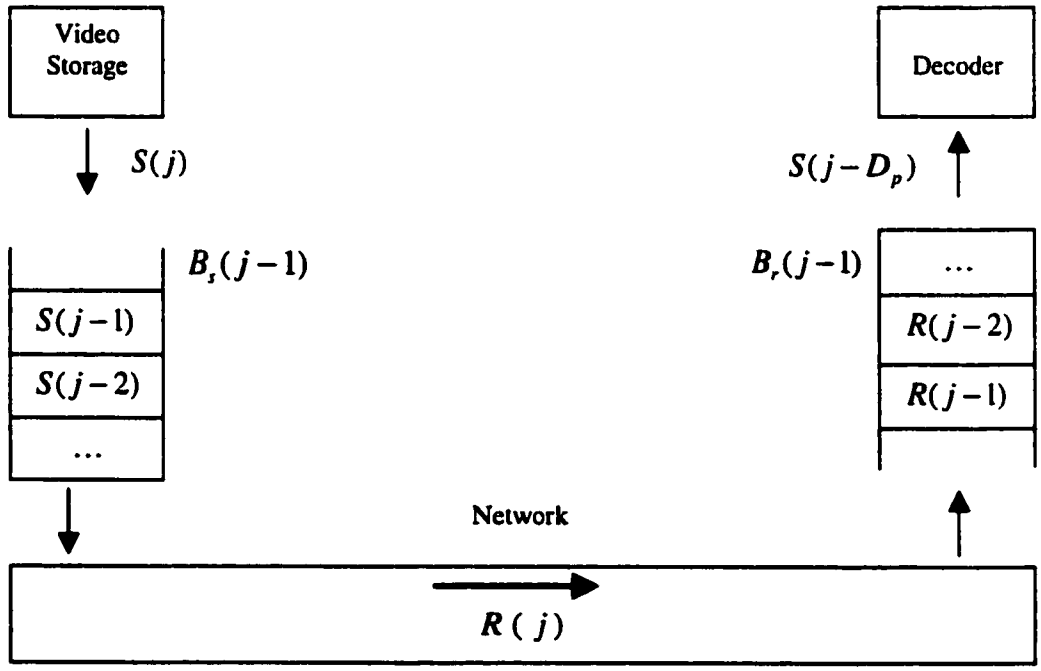

Figure 2. System components in a stored-video transmission application

Our analysis is carried out over discrete time intervals of equal lengths, T. During the j^{th} interval, i.e. $[(j-1)T, jT)$ an arbitrary amount of data, $S(j)$ bits, is retrieved from the video storage and transferred to the sender's buffer. To be consistent with our experimental set-up, let T be a picture interval, for example, 33 msec for the NTSC system. In this case, $S(j)$ will be equal to the size of the j^{th} picture in the video clip. Depending on the network conditions, available bandwidth and sender buffer occupancy during the j^{th} time interval, $R(j)$ bits may be transmitted to the client. $R(j)$ may be smaller or greater than $S(j)$ and it is certainly smaller than the bandwidth of the

network. If $C(j)$ is the mean channel bitrate during the interval then $R(j) = C(j)T$. Obviously, if no data is lost in the network, $R(j)$ bits are delivered to the client's buffer in the same time interval if transmission delay of the channel is ignored or instead the receiver's clock has a shift equal to the transmission delay relative to the sender's clock.

Let $B_s(j)$ and $B_r(j)$ denote the buffer occupancy at the sender and the receiver ends, expressed in bits, at the end of the j^{th} time interval, respectively. It is a common practice to start decoding the video only after having received a given number of frames, say p. This is done to reduce the effect of jitter during the playback of the video. We will refer to the client's buffer as the play-out buffer. Let the playback delay be D_pT. Thus, in the j^{th} interval, the decoder will be decoding $(j - D_p)^{th}$ picture.

From the definitions given above it is easy to see that:

$$B_s(j) = \sum_{i=1}^{j} S(i) - \sum_{i=1}^{j} R(i) \qquad j > 0 \qquad \text{(II.5-1)}$$

and

$$\begin{cases} B_r(j) = \sum_{i=1}^{j} R(i) & 0 < j \le D_p \\ B_r(j) = \sum_{i=1}^{j} R(i) - \sum_{i=1}^{j-D_p} S(i) & j \ge D_p \end{cases} \qquad \text{(II.5-2)}$$

For $j \ge D_p$, from these two equations we have:

$$B_r(j) = \sum_{i=1}^{j} S(i) - \sum_{i=1}^{j-D_p} S(i) - B_s(j) = \sum_{i=j-D_p+1}^{j} S(i) - B_s(j) \qquad \text{(II.5-3)}$$

and

$$B_r(j) = \sum_{i=1}^{j} R(i) - B_s(j - D_p) - \sum_{i=1}^{j-D_p} R(i) = \sum_{i-j-D_p+1}^{j} R(i) - B_s(j - D_p) \qquad \text{(II.5-4)}$$

(II.5-3) and (II.5-4) can be used at the sender side to keep track of the receiver's buffer occupancy based on previous and present information. These formulas can be useful in bandwidth renegotiations. They help estimate the required level of network throughput in each time interval.

For a lossless (no overflow) and continuous (no underflow) transmission, two sets of conditions should be satisfied:

$$\begin{cases} 0 \le B_s(j) \le B_s^{phys} \\ 0 \le B_r(j) \le B_r^{phys} \end{cases} \qquad \text{(II.5-5)}$$

Where B_s^{phys} and B_r^{phys} are the physical sizes of the sender's and the receiver's buffers, respectively. Note that the lower bound in the first set of conditions in (II.5-5) is not really a constraint as buffer length can not be less than zero; rather it only states that the channel rate $R(j)$ does not need to be greater than $B_s(j-1)+S(j)$ because:

$$B_s(j) = B_s(j-1) + S(j) - R(j)$$

Another thing that is worth pointing out is the lower bound in the second set of these conditions. Note that we have not excluded an empty play-out buffer as a condition for not having pauses in the video. Remember that $B_r(j)$ is the buffer length at the end of the j^{th} interval.

Combining the two sets of conditions in (II.5-5) and using (II.5-3) one obtains:

$$0 \le \sum_{i=j-D_p+1}^{j} S(i) \le B_s^{phys} + B_r^{phys} \qquad ;\ j \ge D_p \qquad \text{(II.5-6)}$$

Since (II.5-5) is valid for any interval, from (II.5-4) it follows:

$$\sum_{i=j-D_p+1}^{j} R(i) = B_r(j) + B_s(j-D_p) \le B_r^{phys} + B_s^{phys} \qquad ;\ j \ge D_p$$

From there, an upper-limit on the channel rate is found to be:

$$R(j) \le B_r^{phys} + B_s^{phys} - \sum_{i=j-D_p+1}^{j-1} R(i) \qquad ; \; j \ge D_p \qquad \text{(II.5-7)}$$

From (II.5-6), we get:

$$B_s^{phys} + B_r^{phys} \ge \underset{j}{Max}\left\{ \sum_{i=j-D_p+1}^{j} S(i) \right\} \qquad \text{(II.5-8)}$$

This formula gives a minimum for the required overall physical size of buffers at the sender and the receiver.

(II.5-1) can be written as:

$$R(j) = \sum_{i=1}^{j} S(i) - \sum_{i=1}^{j-1} R(i) - B_s(j) \qquad ; \; j \ge 0$$

Then, applying (II.5-5) results:

$$\sum_{i=1}^{j} S(i) - \sum_{i=1}^{j-1} R(i) - B_s^{phys} \le R(j) \le \sum_{i=1}^{j} S(i) - \sum_{i=1}^{j-1} R(i) \qquad ; \; j \ge 0 \qquad \text{(II.5-9)}$$

Similarly, applying (II.5-2) in (II.5-5) results:

$$0 \le R(j) + \sum_{i=1}^{j-1} R(i) - \sum_{i=1}^{j-D_p} S(i) \le B_r^{phys} \qquad ; \; j \ge D_p$$

$$\sum_{i=1}^{j-D_p} S(i) - \sum_{i=1}^{j-1} R(i) \le R(j) \le B_r^{phys} + \sum_{i=1}^{j-D_p} S(i) - \sum_{i=1}^{j-1} R(i) \qquad \text{(II.5-10)}$$

Combine (II.5-9) and (II.5-10) to obtain:

$$\begin{cases} R(j) \le Min\left\{\sum_{i=1}^{j} S(i) - \sum_{i=1}^{j-1} R(i),\ \sum_{i=1}^{j-D_p} S(i) + B_r^{phys} - \sum_{i=1}^{j-1} R(i)\right\} \\ R(j) \ge Max\left\{\sum_{i=1}^{j} S(i) - \sum_{i=1}^{j-1} R(i) - B_s^{phys},\ \sum_{i=1}^{j-D_p} S(i) - \sum_{i=1}^{j-1} R(i)\right\} \end{cases} ;\ j \ge D_p \qquad \text{(II.5-11)}$$

By applying (II.5-1) to the $(j-1)^{th}$ time interval, these constraints can be transformed into:

$$\begin{cases} R(j) \le Min\left\{S(j) + B_s(j-1), B_r^{phys} + B_s(j-1) - \sum_{i=j-D_p+1}^{j-1} S(i)\right\} \\ R(j) \ge Max\left\{S(j) + B_s(j-1) - B_s^{phys}, B_s(j-1) - \sum_{i=j-D_p+1}^{j-1} S(i)\right\} \end{cases} ;\ j \ge D_p \qquad \text{(II.5-12)}$$

(II.5-11) or (II.5-12) and (II.5-7) can be used to determine required channel rate for each picture interval based on picture sizes up to that picture, physical sizes of buffers and server's buffer length in the last picture interval. Given the physical sizes of the buffers, these conditions dictate the way the channel rate should be selected for a given sequence of picture sizes. However, this approach for video transmission, most of the time is not applicable. For instance, in a VOD service large groups of users with various types of connections and limitations have to be served. Under this scenario, users have to adapt their rates to the channel conditions, i.e. available bandwidth.

There are two solutions for this problem.

1. To use scalable video encoding schemes that allow the rate of video to be adapted to the network resources [24].
2. To jointly select the source rate and the channel rate [25].

We believe that these two techniques can be combined. In appendix B, we briefly explain the principles of the first method. In appendix C, we analyze possible ways to optimize the system operation using the second method.

II.6 Jitter Constraints

In this section, we offer an analysis of burstiness and jitter and their relationship from the point of view of a video receiver. Video is usually decoded and displayed at a fixed rate in terms of frames per second. Therefore, we choose a video frame as the unit of reference. When an estimate of channel bitrate is needed, one may consider the average frame size or the maximum frame size in the video sequence. Remember that for stored video frame sizes are known in all the sequence, in advance.

Since the receiver decodes the frames at the same rate they are generated at the sender, any failure in properly delivering them is due to low level of network performance. For example, in the case of channel congestion some frames will be delivered to the receiver later than their target arrival time. In other words, video throughput is less than what is required. On the other hand, when the congestion is resolved, the network will be able to quickly deliver any amount of data that is available in the sender's buffer from past without considering the target frame rate. This phenomenon is especially true in Ethernet due to "Capture Effect" [49,51] and was clearly noticed in our experiments. Thus, the traffic arriving from the network is usually bursty. The most important step in the analysis of jitter is to characterize this burstiness. Based on the nature of video transmission requirements, it is appropriate to consider the two extreme cases of sustained bandwidth starvation (underflow condition) and momentary high data influx (overflow condition).

Define $A(t_1,t_2)$ to be the number of video frames arriving in the interval $[t_1,t_2)$. Then $A(a_p,a_m) = m - p$ is the number of video frames that arrive before a_m including the p^{th} frame where $a_i (i = 1,2,...,M)$ is the arrival time of the i^{th} frame at the client and M is

the number of pictures in the sequence; see figure 3. However, the techniques[3] used here can be applied at different points in the path from the server to the client.

We assume that the decoder's buffer (play-out buffer) is pre-filled with p frames before it starts to playback the video. Frames are decoded at a fixed rate of $\mu = 1/T$ frames per second after $t = a_p$ and T is the picture-interval. d_m is the display (or departure) time of the m^{th} frame.

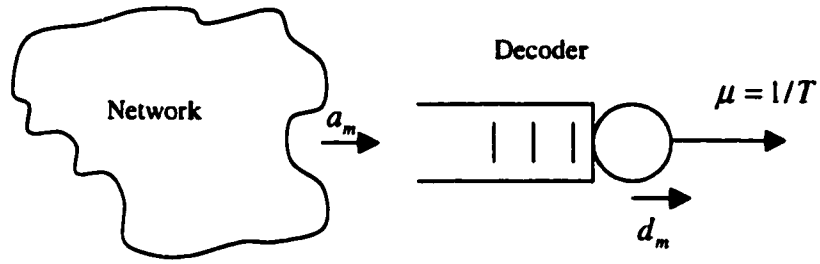

Figure.3 Video decoder as a fixed-rate, discrete-time server

We will characterize the burstiness of the arriving traffic by:

1. Minimum sustainable frame rate $\rho_{\min} = Min[X_{p+1},...,X_M]$ where $X_m = \frac{A(a_p,a_m)}{a_m - a_p} = \frac{m-p}{a_m - a_p}$ is the average frame rate up to time a_m after decoding has started.
2. Maximum instantaneous frame rate $\rho_{\max} = Max[Y_{p+1},...,Y_M]$ where $Y_m = \frac{1}{a_m - a_{m-1}}$.

Burstiness of the traffic before playback time is not important.

[3] These techniques have been inspired by [21], in which, delay jitter of arriving cells at an ATM switch has been analyzed.

By definition, for any $m > p$:

$$\begin{cases} \rho_{\min} \le \dfrac{m-p}{a_m - a_p} \\ \rho_{\max} \ge \dfrac{1}{a_m - a_{m-1}} \end{cases} \qquad \text{(II.6-1)}$$

When the incoming traffic has these constraints, we say that it conforms to $(\rho_{\max}, \rho_{\min})$ and use the notation $S_{in} \approx (\rho_{\max}, \rho_{\min})$[4].

Lemma 1: If $S_{in} \approx (\rho_{\max}, \rho_{\min})$ then for $m > p$:

$$a_m \le a_p + \frac{m-p}{\rho_{\min}}$$

Proof

From (II.6-1) this is obvious.

Note: For $m \le p$, $a_m \le a_p$.

Lemma 2: If $S_{in} \approx (\rho_{\max}, \rho_{\min})$ then for $m > p$:

$$a_m \ge a_p + \frac{m-p}{\rho_{\max}}$$

Proof

From (II.6-1) it can be seen that:

$$a_m \ge a_{m-1} + \frac{1}{\rho_{\max}} \qquad \text{(II.6-2)}$$

By applying (II.6-2) recursively, this lemma can be proven. QED.

By combining Lemma 1 and Lemma 2, range of variation of arrival time of the m^{th} frame can be found to be:

[4] A similar characterization of traffic and a complete network calculus based on that can be found in [22]. We have applied these techniques to a video traffic.

$$< a_m >= \frac{m-p}{\rho_{\min}}\left(1-\frac{\rho_{\min}}{\rho_{\max}}\right)$$

Define burstiness of frame $m > 1$ to be the difference between its target and its actual inter-arrival time at the play-out buffer relative to its previous pictures:

$$b_m = a_m - a_{m-1} - T$$

Burstiness of a frame can have positive or negative values depending on whether it arrives after or before the target inter-arrival time, respectively.

Lemma 3: If $S_{in} = (\rho_{\max}, \rho_{\min})$ then for $m > p$:

$$(m-p)\left[\frac{1}{\rho_{\max}} - \frac{1}{\rho_{\min}}\right] + \frac{1}{\rho_{\min}} - \frac{1}{\mu} \le b_m \le (m-p)\left[\frac{1}{\rho_{\min}} - \frac{1}{\rho_{\max}}\right] + \frac{1}{\rho_{\max}} - \frac{1}{\mu}$$

Proof

This is easy to see from Lemma 1 and Lemma 2 and the definition of burstiness. QED.

Note that if $\rho_{\max} = \rho_{\min} = \rho$ then $b_m = \frac{1}{\rho} - \frac{1}{\mu}$ and in particular if $\rho = \mu$ (i.e. video arrives at the rate it is decoded) then $b_m = 0$. If $\rho \neq \mu$, then b_m is a constant, which is either always positive or always negative as shown in figure 4.

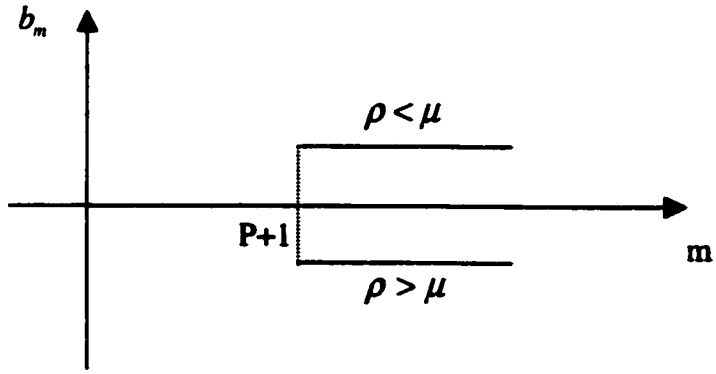

Figure.4 Burstiness as a function of frame number

It is easy to see that the range of variation of b_m is:

$$< b_m >= \frac{2(m-p)-1}{\rho_{\min}}(1-\frac{\rho_{\min}}{\rho_{\max}})$$

$\frac{\rho_{\min}}{\rho_{\max}}$ can be used as a measure of burstiness of the traffic.

The following relationships are obvious:

$$\begin{cases} d_m = Max[d_{m-1}, a_m] + \frac{1}{\mu} & ; \text{for } m \geq 2 \qquad \text{(II.6-3)} \\ d_1 = a_p + \frac{1}{\mu} & \qquad \text{(II.6-4)} \end{cases}$$

When $a_m \geq d_{m-1}$ (which happens only for $m > p$), there is a gap between the display time of the (m-1)th frame and the arrival time of the mth frame. Let this gap be i_m.

$$\begin{cases} i_m = a_m - d_{m-1} & ; \ a_m > d_{m-1} \\ i_m = 0 & ; \ a_m \leq d_{m-1} \end{cases}$$

Let $I(i,j)$ be the sum of i_m's between the ith and the jth frame; then $I(i,j) = \sum_{n=i}^{j} i_n$.

In all the analyses that follow, we will assume that $\mu \geq \rho_{\min}$, otherwise video would be decoded strictly slower than it arrives and every frame is present before its target decoding time, thus jitter does not exist but buffer length will always grow without bound. In that case, the decoder would be a server with a mean service rate slower than the mean arrival rate, because the mean arrival rate is obviously larger than the minimum sustained arrival rate.

Jitter is defined as the variation of frame delays at the display time. If all the frames in a video sequence experience the same amount of delay in the path, jitter is zero for every frame because their arrival times at the client are equal to their generation times

plus a common end-to-end delay. In other words, each frame arrives T seconds (target inter-arrival time) after its previous frame. In that case, inter-departure times will also be equal to T (target inter-departure time) since they are decoded one every T seconds. Therefore, jitter can be defined as the deviation of inter-departure time of two consecutive frames from the target inter-departure time. The jitter of the mth frame can be written as:

$$j_m = d_m - d_{m-1} - \frac{1}{\mu} \qquad ; \text{for } m > p$$

For $m \leq p$ clearly $j_m = 0$.

For $m > p$:

$$j_{m-1} = d_{m-1} - d_{m-2} - \frac{1}{\mu}$$

Using recursion;

$$\sum_{n=p+1}^{m} j_n = d_m - d_p - \frac{m-p}{\mu} = d_m - \left[a_p + \frac{p}{\mu}\right] - \frac{m-p}{\mu}$$

Then;

$$d_m = a_p + \frac{m}{\mu} + \sum_{n=p+1}^{m} j_n$$

$$d_{m-1} = a_p + \frac{m-1}{\mu} + \sum_{n=p+1}^{m-1} j_n$$

From the definition of jitter;

$$j_m = d_m - \left[a_p + \frac{m-1}{\mu} + \sum_{n=p+1}^{m-1} j_n\right] - \frac{1}{\mu}$$

Thus;

$$j_m = d_m - \left[a_p + \frac{m}{\mu} + \sum_{n=p+1}^{m-1} j_n\right]$$

Let D_m denote the delay of the mth frame by the play-out buffer. Then for all m:

$$D_m = d_m - a_m - \frac{1}{\mu} = (d_m - d_1) - (a_m - a_p)$$

For $m \leq p$; $d_m = a_p + \frac{1}{\mu} + \frac{m-1}{\mu}$ and

$$D_m = d_m - a_m - \frac{1}{\mu} = a_p + \frac{m-1}{\mu} - a_m$$

For $m > p$; according to definitions given above:

$$D_m = j_m + d_{m-1} - a_m$$

$$b_m = a_m - a_{m-1} - \frac{1}{\mu}$$

from there;

$$D_m = j_m - b_m + d_{m-1} - a_{m-1} - \frac{1}{\mu} = j_m - b_m + D_{m-1}$$

So;

$$D_{m-1} = j_{m-1} - b_{m-1} + D_{m-2}$$

By recursion:

$$D_m = \sum_{n=p+1}^{m} j_n - \sum_{n=p+1}^{m} b_n$$

In a video-on-demand application, once the decoding has started, pictures have to arrive at a fixed rate of μ, but even the pictures that do not meet this deadline are used. In other words, in VOD as a presentational application, there is no reason to drop late pictures. In these applications, delay jitter is the important factor not the delay itself. In fact, that is why we can have buffers at the server and the client. These buffers are used to buffer pictures before and after congestion at the server and the client, respectively.

However, in conversational applications, in order to keep the synchronization between the server and the client, any late picture is dropped and having big buffers does not make sense. The present analysis is applicable to conversational applications by substituting p=1 or another small value depending on the delay tolerance of the application.

Now, we are ready to relate the incoming traffic to the outgoing traffic.

Theorem 1 : If $S_{in} \approx (\rho_{\max}, \rho_{\min})$ then $S_{out} \approx (\mu, \rho_{\min})$ for all m.

Proof

Lower bound:

If $a_m \geq d_{m-1}$; (II.6-3) $\Rightarrow$ $d_m = a_m + \frac{1}{\mu} \geq d_{m-1} + \frac{1}{\mu}$

If $a_m < d_{m-1}$; (II.6-3) $\Rightarrow$ $d_m = d_{m-1} + \frac{1}{\mu} \geq d_{m-1} + \frac{1}{\mu}$

Hence

$$\mu \geq \frac{1}{d_m - d_{m-1}}$$

Therefore, for any $m \geq 2, \mu \geq Y_m^{out}$.

Upper bound:

We use induction.

Step1: If $a_2 \leq d_1$ (e.g. when p>1) $\Rightarrow$ $d_2 = d_1 + \frac{1}{\mu} \leq d_1 + \frac{1}{\rho_{\min}}$.

If $a_2 > d_1$ (only when p=1) $\Rightarrow$ $d_2 = a_2 + \frac{1}{\mu}$

From Lemma 1, m=2, p=1 $\Rightarrow$ $a_2 \leq a_1 + \frac{1}{\rho_{\min}}$ $\Rightarrow$ $d_2 \leq a_1 + \frac{1}{\mu} + \frac{1}{\rho_{\min}}$

But $d_1 = a_p + \frac{1}{\mu} = a_1 + \frac{1}{\mu}$

So

$$d_2 \le d_1 + \frac{1}{\rho_{\min}}$$

Hence

$$\rho_{\min} \le \frac{1}{d_2 - d_1}$$

Step2: If $d_m \le d_1 + \frac{m-1}{\rho_{\min}}$ we show that $d_{m+1} \le d_1 + \frac{m}{\rho_{\min}}$.

If $a_{m+1} \le d_m \quad \Rightarrow \; d_{m+1} = d_m + \frac{1}{\mu} \le d_1 + \frac{m-1}{\rho_{\min}} + \frac{1}{\mu} \le d_1 + \frac{m}{\rho_{\min}}$

If $a_{m+1} > d_m \quad \Rightarrow \; d_{m+1} = a_{m+1} + \frac{1}{\mu}$

From Lemma 1, for m>p we have $a_{m+1} \le a_p + \frac{m}{\rho_{\min}}$ and for $m < p$ obviously $a_{m+1} \le a_p$; thus for $m > 1$:

$$d_{m+1} \le a_p + \frac{m}{\rho_{\min}} + \frac{1}{\mu}$$

(II.6-4) $\Rightarrow \quad d_{m+1} \le d_1 + \frac{m}{\rho_{\min}}$

Therefore;

$$\rho_{\min} \le \frac{m-1}{d_m - d_1}$$

So, for $m > 1$, $\rho_{\min}$ is less than any $X_m^{out} = \frac{m-1}{d_m - d_1}$. QED.

Theorem 2: If $S_{in} \approx (\rho_{\max}, \rho_{\min})$ then:

$$0 \le j_m \le \frac{1}{\rho_{\min}} - \frac{1}{\mu} \qquad ; m > 1 \qquad \text{(II.6-5)}$$

$$\frac{m-1}{\mu}-\frac{m-p}{\rho_{\min}} \le D_m \le \frac{m-1}{\rho_{\min}}-\frac{m-p}{\rho_{\max}} \qquad ; m > p \qquad \text{(II.6-6)}$$

Proof

Since $S_{out} = (\mu, \rho_{\min})$ for all $m > 1$;

$$d_1 + \frac{m-1}{\mu} \le d_m \le d_1 + \frac{m-1}{\rho_{\min}}$$

So;

$$d_1 + \frac{m-2}{\mu} \le d_{m-1} \le d_1 + \frac{m-2}{\rho_{\min}}$$

By the definition of jitter;

$$0 \le j_m \le \frac{1}{\rho_{\min}} - \frac{1}{\mu}$$

This makes sense because in the output, frames can only be late for display and never early. Besides, maximum jitter is $j_m = \frac{1}{\rho_{\min}}$ that happens if $\mu >> \rho_{\min}$.

Also;

$$\frac{m-1}{\mu} \le d_m - d_1 \le \frac{m-1}{\rho_{\min}} \qquad \text{(II.6-7)}$$

For $m > p$, according to the definition of delay:

$$\frac{m-1}{\mu} - a_m + a_p \le D_m \le \frac{m-1}{\rho_{\min}} - a_m + a_p$$

By using lemma1 and lemma2, (II.6-6) is easily obtained. Furthermore, since for $m \le p$ we have $a_m \le a_p$ the upper bound of (II.6-6) is valid for all m. QED.

(II.6-5) can be used to estimate $\rho_{\min}$ for a given level of acceptable jitter. If this value is $(j_m)_{\max}$, then:

$$\rho_{\min} = \frac{1}{\frac{1}{\mu} + (j_m)_{\max}} \qquad \text{(II.6-8)}$$

Lemma 4: For all m;

$$d_m \geq a_m + \frac{1}{\mu}$$

$$d_m \geq d_{m-1} + \frac{1}{\mu}$$

Proof

If $a_m \geq d_{m-1} \quad \Rightarrow \quad d_m = a_m + \frac{1}{\mu} \geq a_m + \frac{1}{\mu}$, $\qquad d_m \geq d_{m-1} + \frac{1}{\mu}$

If $a_m < d_{m-1} \quad \Rightarrow \quad d_m = d_{m-1} + \frac{1}{\mu} \geq a_m + \frac{1}{\mu}$, $\qquad d_m \geq d_{m-1} + \frac{1}{\mu}$ QED

Theorem 3: For all m; $D_m \geq 0$.

Proof

Obvious from the definition of delay and lemma4. QED.

Consequently;

$$\sum_{n=p+1}^{m} j_n \geq \sum_{n=p+1}^{m} b_n$$

Theorem 4: If $S_{in} \approx (\rho_{max}, \rho_{min})$ and $\mu = \rho_{min}$ then;

$$j_m = 0 \qquad \text{; for all m}$$

$$D_m \geq \frac{p-1}{\mu} \qquad \text{; for m>p}$$

Proof

This can be proven using Theorem 2 and makes sense, intuitively, because the video sequence is displayed at the same rate as the minimum arrival rate. However, buffer length at no time will be decreasing.

Theorem 5: If $S_{in} \approx (\rho_{max}, \rho_{min})$ and $\rho_{min} \leq \mu \leq \rho_{max}$ then for $m > p$:

$$D_m \leq \frac{m-1}{\mu} - \frac{m-p}{\rho_{max}}$$

Proof

First we show

$$a_m \geq a_{m-1} + \frac{1}{\rho_{max}} + i_m \qquad \text{(II.6-9)}$$

$$d_m = d_{m-1} + \frac{1}{\mu} + i_m \qquad \text{(II.6-10)}$$

- If $a_m \geq d_{m-1}$, for m>p we have $i_m = a_m - d_{m-1}$;

$$a_m = d_{m-1} + i_m \geq a_{m-1} + \frac{1}{\mu} + i_m \geq a_{m-1} + \frac{1}{\rho_{max}} + i_m$$

and

$$d_m = a_m + \frac{1}{\mu} = d_{m-1} + \frac{1}{\mu} + i_m$$

- If $a_m \le d_{m-1}$, we have $i_m = 0$. Thus;

$$d_m = d_{m-1} + \frac{1}{\mu} = d_{m-1} + \frac{1}{\mu} + i_m$$

Lemma 2 $\Rightarrow$ $$a_m \ge a_{m-1} + \frac{1}{\rho_{\max}} + i_m .$$

Now we show:

$$a_m \ge a_p + \frac{m-p}{\rho_{\max}} + I(p,m) \qquad \text{(II.6-11)}$$

$$d_m = d_1 + \frac{m-1}{\mu} + I(p,m) \qquad \text{(II.6-12)}$$

From (II.6-9);

$$a_{m-1} \ge a_{m-2} + \frac{1}{\rho_{\max}} + i_{m-1} \quad , \quad a_m \ge a_{m-1} + \frac{1}{\rho_{\max}} + i_m$$

$$\Rightarrow \quad a_m \ge a_{m-2} + \frac{2}{\rho_{\max}} + I(m-1,m)$$

By applying (II.6-9) recursively;

$$a_m \ge a_p + \frac{m-p}{\rho_{\max}} + I(p,m)$$

Similarly, with recursion, (II.6-12) can be proven.

From (II.6-11), (II.6-12) we have;

$$d_m - a_m \le d_1 + \frac{m-1}{\mu} + I(p,m) - a_p - \frac{m-p}{\rho_{\max}} - I(p,m)$$

Thus;

$$D_m \le \frac{m-1}{\mu} - \frac{m-p}{\rho_{\max}}$$ QED

From there, the maximum value of delay can be found to be:

$$D_{\max} = \frac{M-1}{\mu} - \frac{M-p}{\rho_{\max}} = D_M$$

When M, μ and one of $\rho_{\max}, p$ are known, the unknown parameter can be found for a given maximum tolerable delay. From maximum delay we can obtain the minimum physical size of the play-out buffer in terms of number of frames:

$$B_{phys}^{d} = \frac{D_{\max}}{T} = \left[\frac{M-1}{\mu} - \frac{M-p}{\rho_{\max}}\right]\mu \qquad \text{(II.6-13)}$$

II. 7 Summary

In this chapter, we investigated fundamental requirements of video transmission. In the case of stored video, we found that (II.5-12) reflects overflow/ underflow constraints of the buffers at the sender and the receiver. Besides, in (II.5-8) we obtained a minimum overall size for these buffers.

The main results of our jitter analysis are (II.6-8) and (II.6-13). From the former, we can find an underflow avoidance condition on the minimum sustained network throughput given the maximum acceptable jitter by the viewer. The latter can be used to find a minimum for the physical size of the buffer at the video client in order to avoid overflows. Using this value and the minimum overall buffer size in (II.5-8), the minimum size of the sender's buffer can be found.

In these formulas, $\rho_{\min}$ and $\rho_{\max}$ can be thought of as hypothetical limits on the burstiness of a successful video traffic. Any video transmission process must remain between these boundaries. If at some point during the transmission, for example, the

sustained minimum frame rate is evaluated to be less than ρ_{min}, a request for bandwidth increase must be issued.

CHAPTER III

EXPERIMENTAL WORK

In our experiments, we considered the combined effect of delay jitter and data loss on the quality of an MPEG-2 video, transmitted over a dedicated 10 Mbps Ethernet. Studying delay jitter independently from the loss, in such an environment, is not possible. Because, if MAC frames carrying the video data belonging to a certain video frame are delayed long enough to create jitter, they are most of the time dropped by NIC at the sender due to random back-off algorithm. Note that the only reason for delay in transmission of a MAC frame is a loaded channel that can cause collisions and therefore make that packet subject to random back-off algorithm.

The only way to study the effect of jitter, solely, on QoS is to have a lossless full-duplex Ethernet. However, with that configuration, nodes can not be directly connected to the channel. Instead, a switch should be used. In that case, buffers of the switch must have enough capacity in order to have a lossless transmission in presence of some level of background traffic. Unfortunately, in our project we could not prepare such an environment.

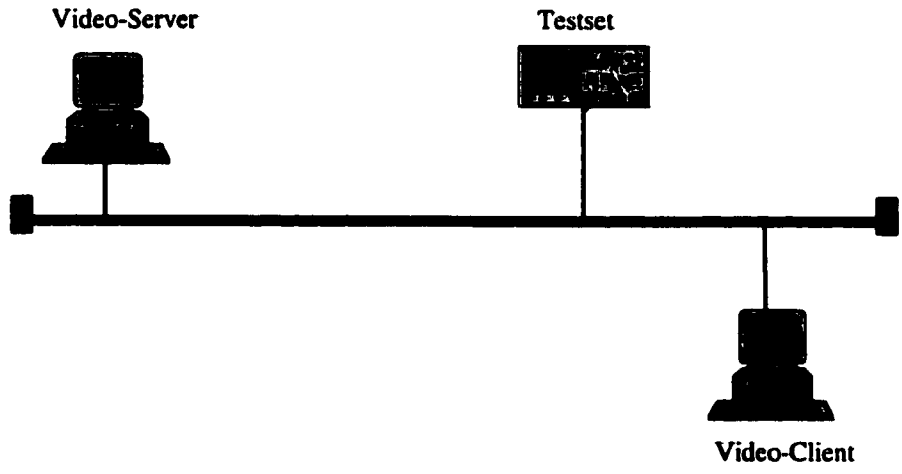

Figure 5. Experimental Set-up

Figure 5 depicts the set-up for the experimental phase of the project. The server, the client and the test-tool were all connected to an isolated 10BASE-T Ethernet. In these experiments, subjective rating of video quality and loss rate were used as user-level and application-level QoS parameters, respectively. Also, inter-arrival times of MPEG frames and IP-packets were measured in order to be used in finding appropriate application-level and IP-level QoS metrics, respectively. All these items were measured under different load conditions and their relationships were investigated.

III.1 Elements of the system

III.1.1. Video server station

A 300 MHz Pentium-II with 128 MB-RAM was used as the video server. Debian GNU/Linux 2.0.33 was chosen as the operating system due to its superior networking capabilities and available open source kernel, which makes it ideal for lab environments. Two SCSI hard disks with a capacity of 6.2 GB/each were installed on this machine which contained the VOD server application and the MPEG-2 bit-streams. The video server was equipped with a 10 Mbps Ethernet adapter.

III.1.2. Video client station

A 300 MHz Pentium-II machine, similar to the server with 128 MB-RAM and running Debian GNU/Linux operating system, was used as the client. The client took advantage of a Vela 2000-0206 MPEG-2 video decoder board [61]. A 3.4GB IDE hard disk was installed in this machine to prevent possible SCSI device conflicts with the video decoder, which had a generic SCSI interface.

III.1.2.1. MPEG-2 Hardware Decoder

A VELA 2000-0206 ISA SCSI-II MPEG-2 video/audio decoder board [61] developed by Vela Research Inc. was installed in the client machine. This

decoder has been designed to handle compressed digital video from NTSC resolution MPEG-1 (352 x 240 pixels) to CCIR –601 (ITU-R-601) resolution MPEG-2 (740 x 480 pixels) and half D-1 (352 x 480 pixels). It also supports PAL resolutions of 704 x 576, 352 x 576 and 352 x 288 pixels.

The decoder has a SCSI-II fast/wide, single-ended (16-bit) interface. All commands and data to the decoder are transferred via the SCSI bus and the ISA interface is used only to power up the board. This decoder supports MPEG-2 program streams and transport streams at data rates of up to 15 Mbps. Thirty 64-KB buffers can hold up to 1.92 MB of MPEG data for playback. The video output of the decoder was fed into a color TV monitor as composite video. Audio was sent to the TV monitor through a balanced stereo output.

The decoder can perform flow control over the input video by using the SCSI Disconnect-Reconnect commands. For a host adapter, the decoder was connected to an Adaptec AHA-2940 Ultra-wide SCSI-II PCI bus card.

The Vela decoder has been originally designed to function under Windows NT. Therefore, the device driver application for Linux had to be developed as part of the project as well as a prototype video transmission application. The device driver application, after pre-loading 16-29 buffers of the decoder issues a playback command and then continues to send 64-KB blocks of video data to the decoder.

III.1.3. Testset

InterWATCH 95000 (IW95000) ATM/LAN/WAN protocol analyzer developed by GN-Nettest [62] was used to measure the traffic sent over the network. However, a piece of program had to be written in order to calculate the MAC-frame inter-arrival times as well as the mean, variance and distribution based on timestamps obtained from the tool. In addition to its wide range of capabilities for ATM networks, IW95000 has many LAN-related features. The following are some of its main features:

- Graphical user interface based on Solaris 2.5.1.

- 7-layer protocol analysis.
- LAN I/O modules support 2 ports, which can be, configured as dual Ethernet. This allows the testset to both, monitor and generate traffic using the same module.
- Non-intrusive (monitoring and performance analysis) and intrusive (traffic generation) tests and measurements.

The server, the client and the test-tool were all connected to an isolated 10BASE-T Ethernet with a star topology (see Figure 5).

III.2 Video Clips

In order to understand the behavior of Ethernet when carrying MPEG-2 bitstreams, the video sequences in Table 1 were used in our experiments.

Sequence	Modality	Q - Factor	Bit Rate (Mbps)	Encoding Rate (frames/sec)
Hook10q.mpg	VBR	10		24
Hook20q.mpg	VBR	20		24
Martin10q.mpg	VBR	10		24
Martin20q.mpg	VBR	20		24
4HD1AFFP.MPG	CBR		4	30
4HD1AFFS.MPG	CBR		4	30
6HD1AFFT.MPG	CBR		6	30
6HD1AFFP.MPG	CBR		6	30
8FD1FOEP.MPG	CBR		8	30
8FD1FOES.MPG	CBR		8	30

Table 1. MPEG-2 Video Sequences

All CBR bit-streams are almost 60 seconds long while VBR bit-streams are only 10 seconds long. So, VBR sequences were concatenated 6 times to make them almost equal in duration to the CBR sequences. The number of bits per picture for these sequences were obtained using an application called "mpegstat" (see Figures 6 through 15). Notice the difference between a VBR and a CBR video. For the CBR sequences, the running mean can be easily visualized from the figures while for a VBR sequence, the mean varies depending on the contents of the video sequence.

III.3 Background Traffic

IW95000 was used to inject background load into the network. This background traffic is actually a continuous stream of fixed-size (64 bytes)[5] Ethernet frames that imposes a specified utilization of the channel bandwidth. In fact, the traffic generator attempts to transmit at the specified load by adjusting the gap between generated packets, following Ethernet MAC specifications. For any level of load, this gap is a different, but fixed, value.[6]

Experiments were conducted for 0%, 75%, 95%, 96%, 97% and 98% background load. The reason for choosing these values was the following; we noticed that for loads below 95%, background traffic had virtually no effect on the MPEG traffic. On the other hand, for 98% load the quality of video was so poor that it deemed totally unacceptable and tests for higher loads were not necessary. Indeed, with that level of load most of the time the video could not even be displayed. The reason for this, seems to be the loss of the first packet that contains MPEG-2 Sequence-Header-Code. Similarly, sometimes the client application would not properly end because the last packet, which is used to mark the end of transmission, was lost.

Using VOD system, video sequences were transferred from the server to the client under different load conditions.

---

[5] According to [46], 80-90% of packets on a typical Ethernet have a length of 64 bytes.
[6] As noted in [49], this results in the worst case throughput in Ethernet.

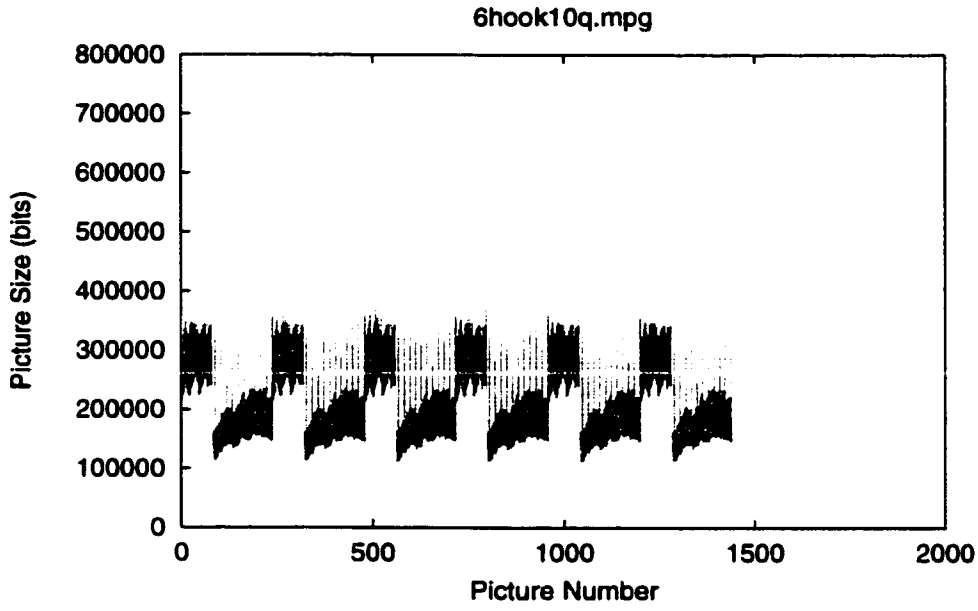

Figure 6. Picture Sizes: hook10q.mpg, VBR, Q=10, 24fps

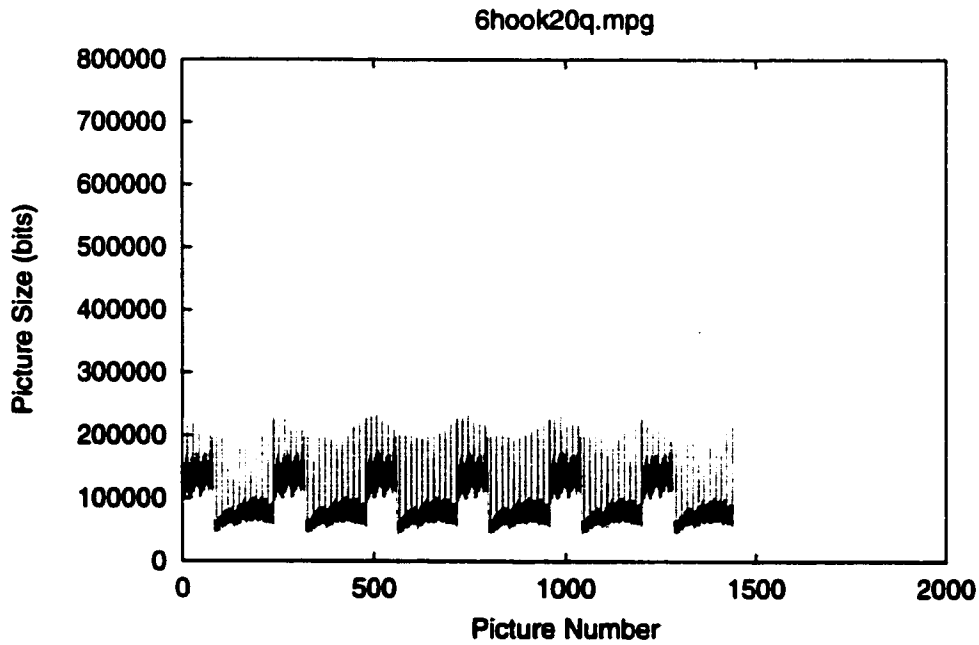

Figure 7. Picture Sizes: hook20q.mpg, VBR, Q=20, 24fps

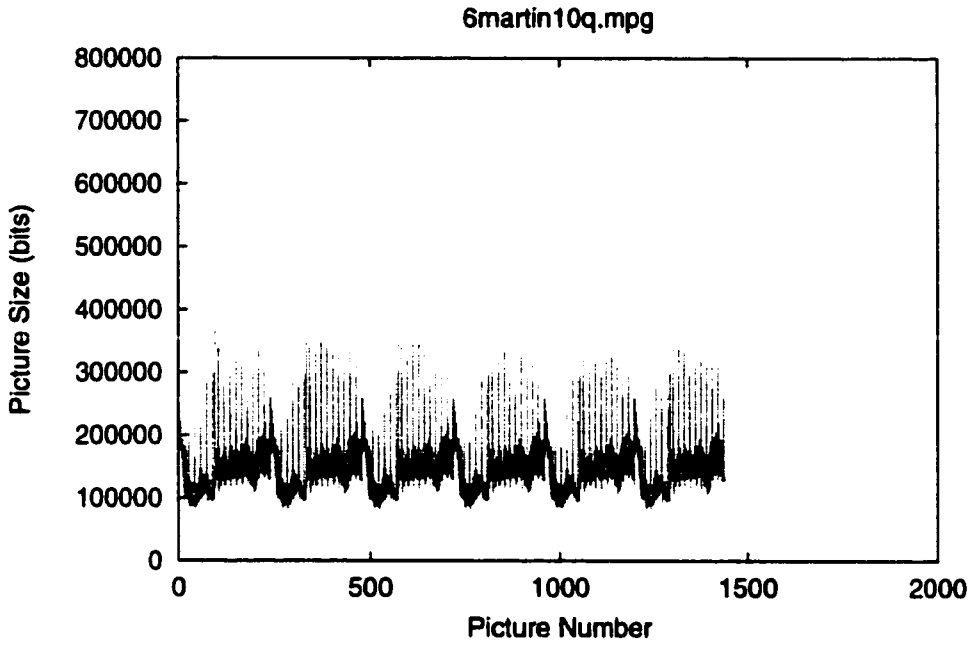

Figure 8. Picture Sizes: martin10q.mpg, VBR, Q=10, 24fps

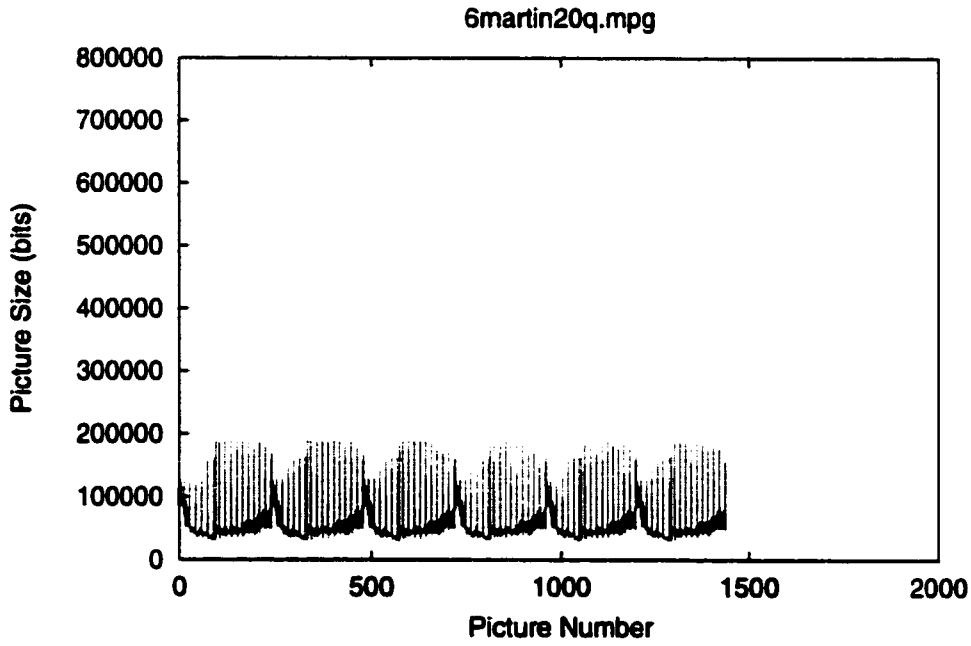

Figure 9. Picture Sizes: martin20q.mpg, VBR, Q=20, 24fps

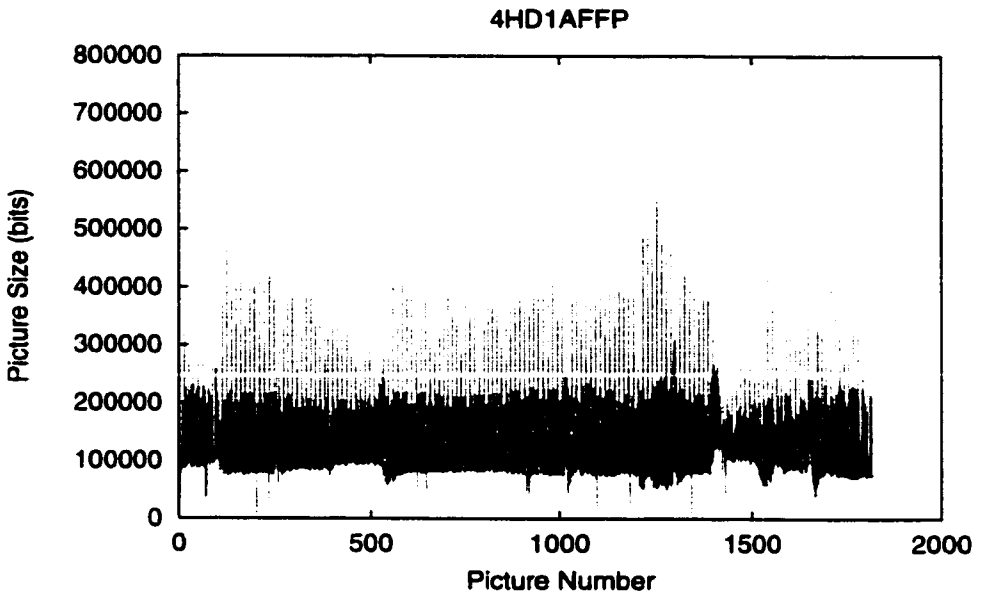

Figure 10. Picture Sizes: 4HD1AFFP.MPG, CBR, 4Mbps, 30fps

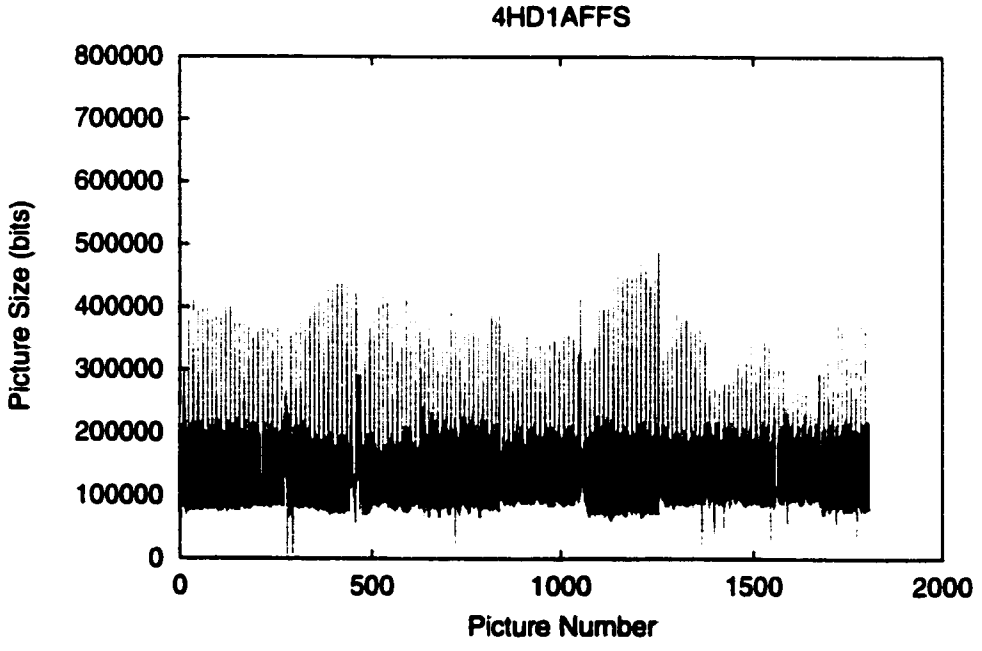

Figure 11. Picture Sizes: 4HD1AFFS.MPG, CBR, 4Mbps, 30fps

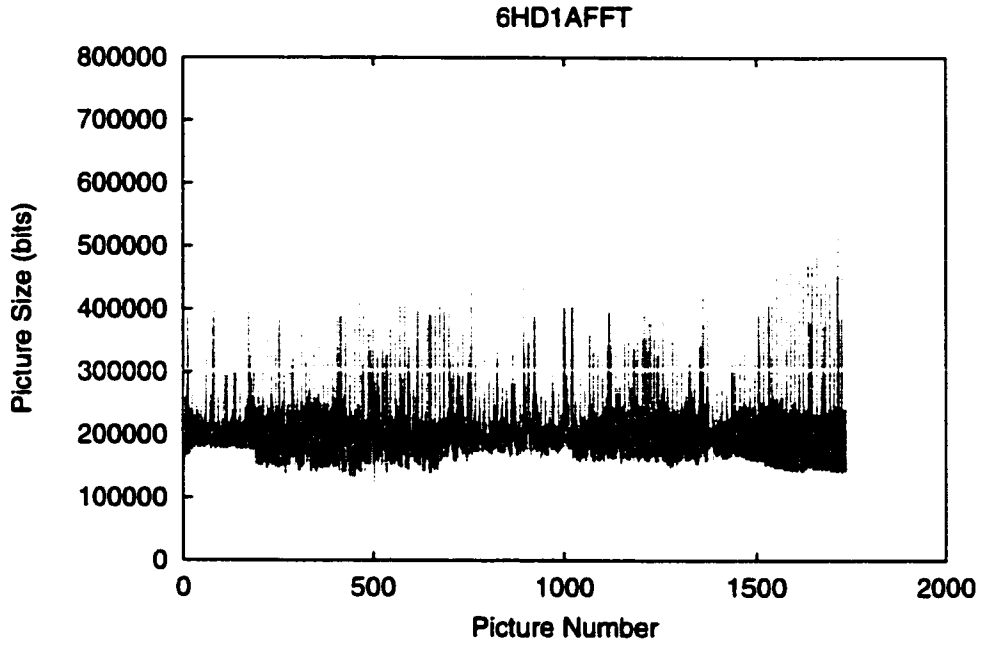

Figure 12. Picture Sizes: 6HD1AFFT.MPG, CBR, 6Mbps, 30fps

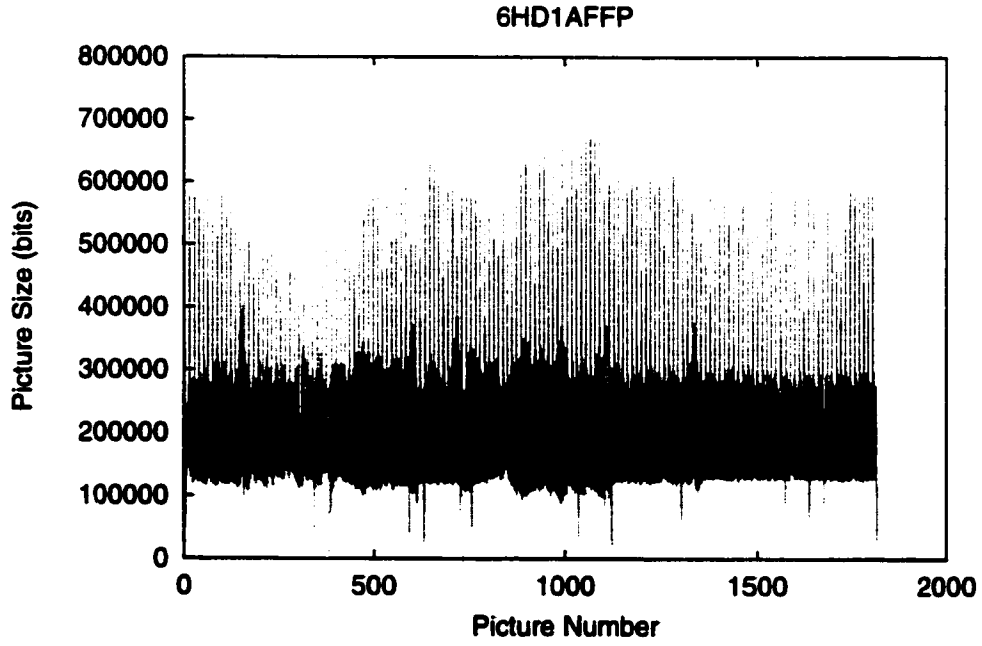

Figure 13. Picture Sizes: 6HD1AFFP.MPG, CBR, 6Mbps, 30fps

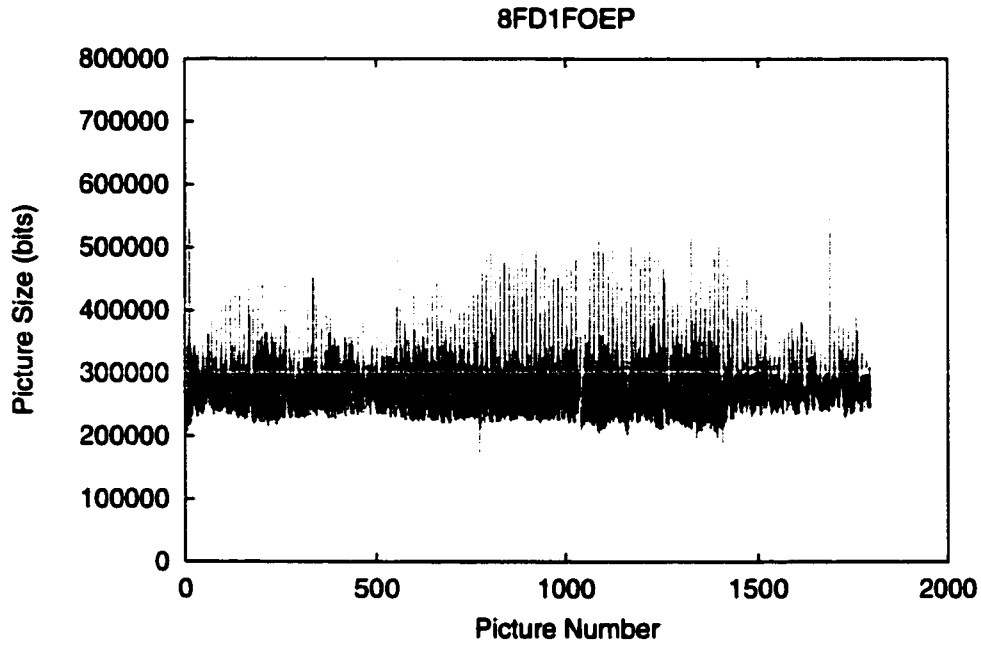

Figure 14. Picture Sizes: 8FD1FOEP.MPG, CBR, 8Mbps, 30fps

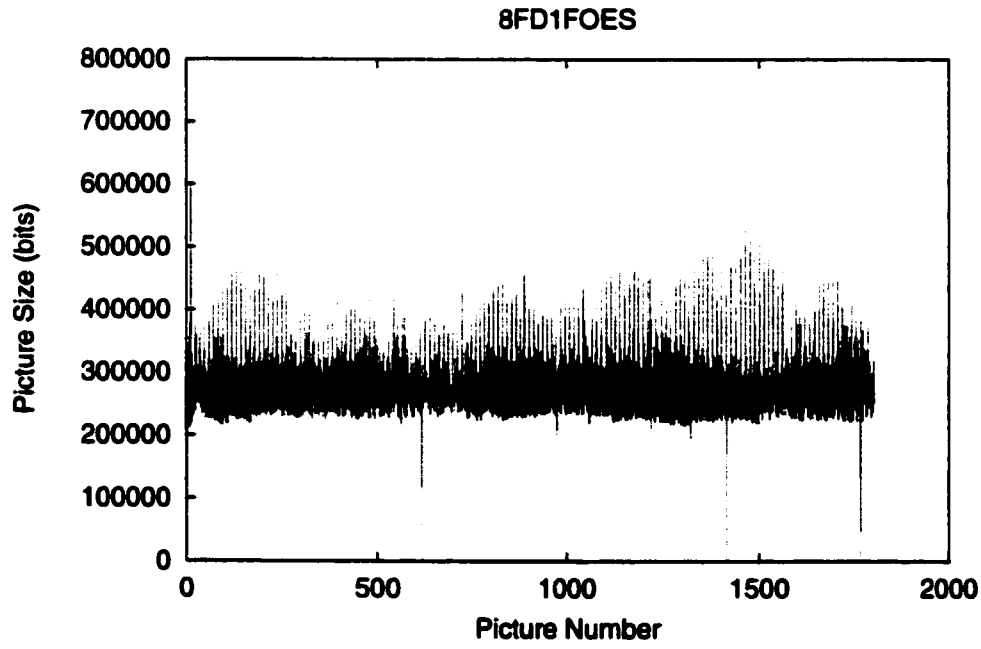

Figure 15. Picture Sizes: 8FD1FOES.MPG, CBR, 8Mbps, 30fps

III.4 Protocol Architecture

The VOD (Video on Demand) application consists of a server and a client. The server is single threaded, i.e. it can have only one client but it can be upgraded to a multi-client server. MPEG-2 video files are stored on the server and are transmitted to the client upon its request. Figure 16 shows the protocol stack implemented at the server and the client. UDP (User Datagram Protocol) was chosen as the transport protocol for two reasons:

1. Video applications are very demanding in terms of bandwidth. The overhead imposed by UDP is only 8 bytes per datagram.
2. Video applications are delay sensitive. Therefore, most of the times they can not tolerate flow control mechanisms such as retransmissions used in TCP. Also, small overhead of UDP requires little processing time.

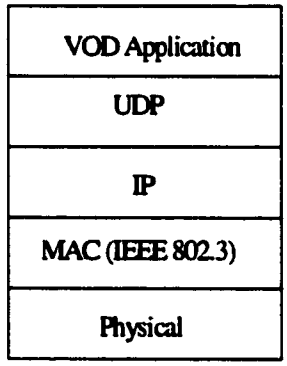

Figure 16. Protocol stack at the server and the client

III.5 Protocol Data Units

Every time an application-frame is fragmented into UDP-datagrams an IP-header of length 20 bytes is added to each UDP-datagram to form an IP-datagram. Thereafter, the IP-datagrams are passed to the physical layer, i.e., Ethernet. Since the MAC-frames (Ethernet-frames) can convey at most a payload of 1500 bytes, an IP-datagram may need to be segmented into two or more Ethernet-frames. In this case, an IP-datagram is broken

into a few 1500-byte IP-packets and perhaps one IP-packet shorter than 1500 bytes, each of which carries the same IP-header but it also has a segmentation offset that is needed for reassembly at the destination. Then, an 18-byte MAC-header is attached to each IP-packet to form a MAC-frame. Figure 17 depicts the protocol data units at the various levels of the protocol architecture.[7]

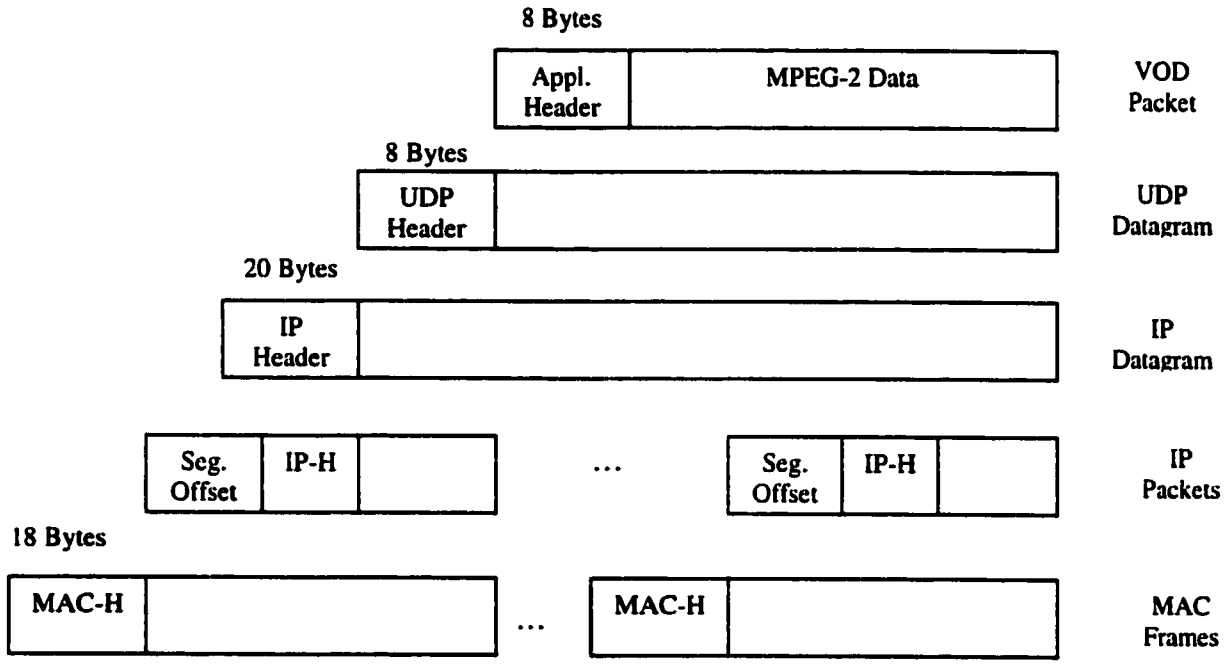

Figure 17. Protocol Data Units

III.6 Scheduling of Video Traffic

As was mentioned before, we have considered future enhancement of this application to a multicast system. This will require the design of a scalable communication system. For instance, controlling the flow of video stream making use of feedback from the clients does not make sense in such an environment. In the absence of such flow control

[7] For details on segmentation procedures and communication protocols refer to any data communication book such as [56],[57] and [58].

mechanisms, we are required to design proper mechanisms to avoid overflowing or underflowing the client's buffers when delivering the video stream. The Real Time Clock (RTC) of Linux has been employed at the server to properly schedule the transmission of video streams. Using this facility we make sure that the rate of picture transmission does not exceed the rate at which the decoder can process the video stream. This rate is actually the encoding rate of the video sequence typically 24 or 30 frames per second.

To perform this task successfully, the video file is processed using "mpegstat" application where the picture sizes are obtained in terms of bits per picture. This way the video delivery process knows the size of the video frames, in advance. Before a picture is sent to the client, a portion of the file as big as the picture size is read from the hard disk into the memory. In order to reduce the burstiness of the video traffic, pictures within each second are not sent back to back. Instead, each picture is sent in an appropriate fraction of a second (referred to as a *picture-interval* in our analysis in the previous chapter) so that we will eventually keep the desired picture rate. This fraction depends on the encoding rate.

To reduce the burstiness even further, each picture can be divided into PPF (Partitions Per Frame) partitions and sent gradually but in a way that the picture rate is kept constant. Equivalently, the server can control the maximum amount of data that is carried by a packet (TR_BLK_SIZE). These two methods can be combined.

Every time the server's UDP socket is invoked, a UDP datagram is sent to the client's UDP socket. If a partition of the current picture (if PPF=1 the whole picture is considered one partition) is smaller than TR_BLK_SIZE, it will be sent. Otherwise, it will be further divided into sub-partitions as shown in Figure 18.

The VOD application introduces an overhead of 8 bytes per packet that is merely used for measurement purposes at the client and has no role in the video transmission process. Therefore, "TR_BLK_SIZE + 8" is an artificial maximum for the PDU (Protocol Data Unit) length of a UDP datagram. This 8-byte header contains two fields,

the first of which carries the number of picture to which the packet belongs (pic-no). The second filed holds the number of packet within that particular picture (id).

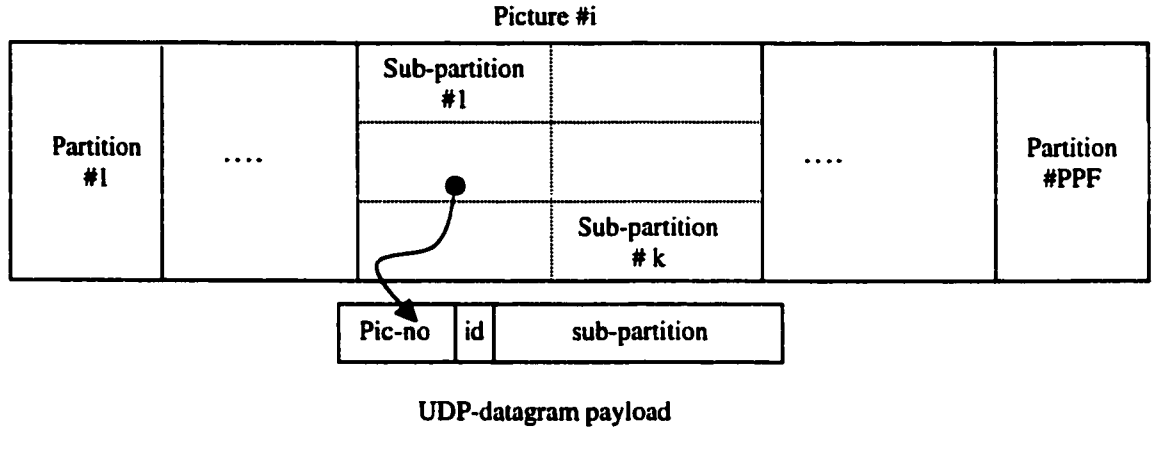

Figure 18. Partitioning a picture

As noted above, depending on the encoding rate of the sequence, a fraction of a second is allocated to each picture. If the transmission of a picture takes less than the allocated time interval, the server waits until the end of picture-interval before it starts delivering the next picture to its UDP socket. On the other hand, if a picture is so large that the transmission time exceeds the picture-interval, the server starts sending the next picture immediately. However, this will reduce the overall picture transmission rate slightly and the server will not try to compensate for that. For instance, in video sequence "6HD1AFFP.MPG" whose trace is shown in figure 13 many frames are of size 500,000 bits or larger. With a transmission rate of 10Mbps each of these frames need at least 1/20 seconds to be sent. Thus, each one of them adds (1/20 – 1/30) seconds to the accumulated delay. Clearly, if such large pictures are numerous, after a while the decoder buffer will run out of data and the viewer will experience a picture freeze in the video. As possible solutions for this problem, the following procedures are suggested:

Procedure 1.Collectively Reduced Picture-interval Scheduling:

For each sequence, adjust the length of the picture-interval based on empirical statistics of the video so that overall transmission rate of 30 fps (24 fps) is achieved. In other

words, for some video sequences that have many large pictures we need to allocate less than 1/30 (1/24) sec to a picture. At first, the contrary may seem right. However, if small pictures occupy time slots that are slightly shorter than 1/30sec, they can compensate for those pictures that need more time to be transmitted. Figure 19 helps explain this idea.

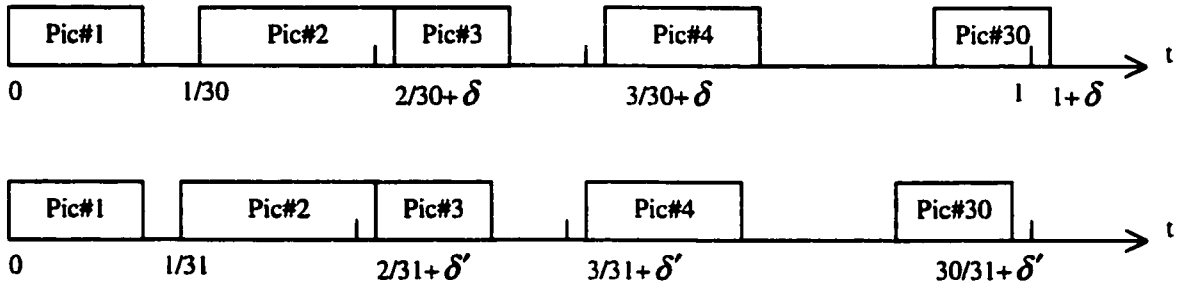

Figure 19. Collectively Reduced Picture-interval Scheduling

Consider a 1-sec interval and suppose that one picture (pic#2) among 30 pictures takes $(1/30+\delta)$ sec to be transmitted. Using the first scheme (depicted in the upper part of Figure 19), we need $(1+\delta)$sec to send all 30 pictures. Now, consider the second scheme (depicted in the lower part of Figure 9) in which 1/31sec is allocated to each picture. The second frame needs $(1/31+\delta')$ sec to be sent ($\delta' > \delta$). However, the total time needed to transmit all the pictures is $(30/31+\delta')$ sec which is less than 1 sec as long as $\delta' < 1/31$ sec.

Procedure 2.Delay Compensation Scheduling:

It was previously mentioned that the current VOD System does not redeem for those pictures whose transmission times surpass the designated picture-interval. Though not very efficient, this method of video scheduling is easy to implement. The *Delay Compensation Scheduling* that is introduced in this section is more elaborate but it could solve the delay problem. In this scheme, every small picture can participate in eliminating the total lag (L) that is produced by large pictures. Any picture #j whose estimated transmission time (e_j) is shorter than the target picture-interval (T) can donate its spare time to reduce the total lag (L). In this system, the scheduler should be able to anticipate the transmission time of each frame. If F_j is the size of picture #j and R is the

transmission rate, then simply $e_j = F_j / R$. The effect of errors in this estimation can be masked by buffering video at the receiver. In this mechanism, picture-interval dynamically changes to maintain the required frame rate but it can never exceed T.

The amount of time that a picture can contribute to reduce the backlog depends not only on its own estimated transmission time but also on L. In other words, the VOD system sends one picture every T seconds unless there is a backlog, in which case it reduces future picture-intervals until L becomes zero again. The flow diagram in figure 20 presents this algorithm.

Due to the time-dependent nature of video, pictures have to meet their deadlines. Therefore, if L exceeds some threshold, delivering delayed pictures, especially in conversational applications, is pointless and they may have to be dropped. However, this problem can be overcome by buffering at the receiver-end and starting video display a little later than the first picture arrives.

Other Techniques

The problem of scheduling a bursty video traffic[8] has been addressed by many researches [32-36]. The techniques that are developed in this area are called *bandwidth smoothing methods*. Procedures 1 and 2, though somewhat crude, can be placed among these methods. More sophisticated traffic smoothing schemes include [32]:

- Average Bandwidth Allocation algorithm (ABA)
- Critical Bandwidth Allocation algorithm (CBA)
- Minimum Changes Bandwidth Allocation algorithm (MCBA)
- Minimum Variability Bandwidth Allocation algorithm (MVBA)
- Piecewise constant Rate Transmission and Transport (PCRTT)

In the case of prerecorded video, these algorithms take advantage of prior knowledge of the picture sizes. The main idea here is to transfer some of the load of large frames to

[8] Compressed video is intrinsically bursty due to its frame structure and scene changes.

smaller frames. This way, large frames can be sent at a slower rate than they require and still meet their deadlines.

Based on the frame sizes and the size of the buffer at the receiver, these methods generate a *transmission plan* that consists of some fixed-rate *runs*. The goal is to reduce the peak bandwidth requirement with the minimum number of runs. Note that, every rate change at the beginning of a run requires bandwidth negotiations with the network.

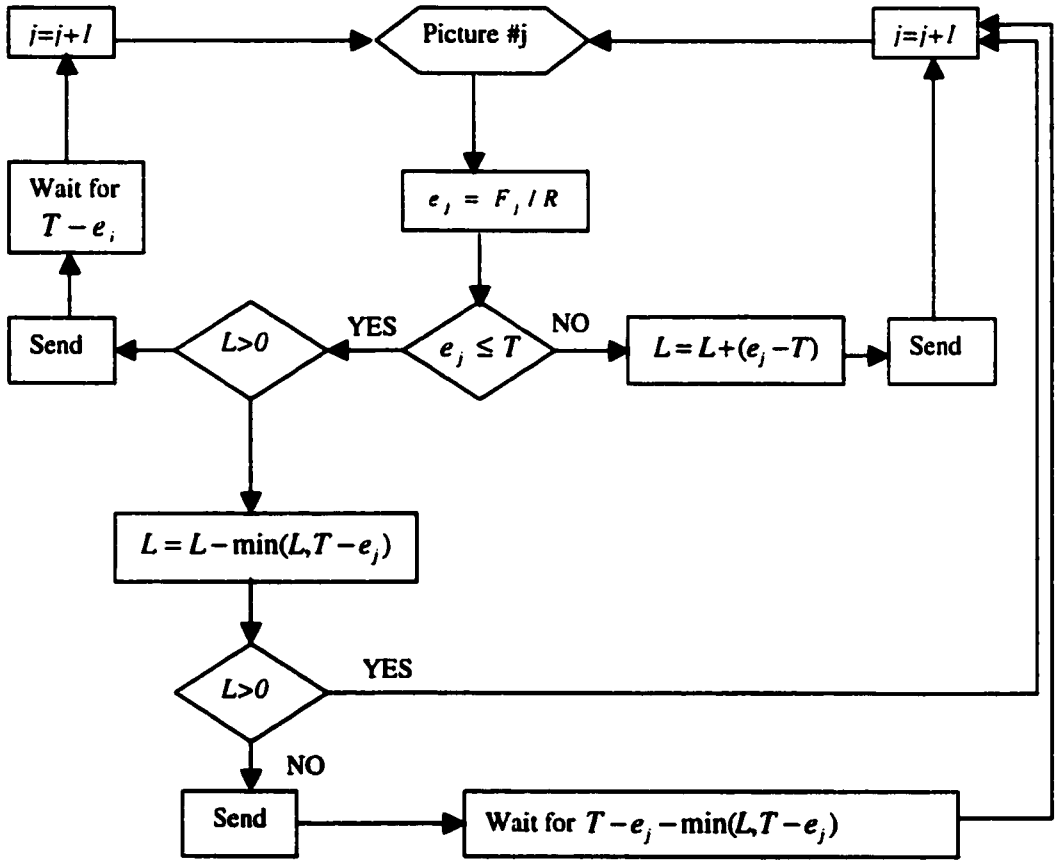

Figure 20. Delay Compensation Scheduling

III.7 System Parameters

TR_BLK_SIZE: Since the maximum allowable length of a UDP-datagram is 64 KB, TR_BLK_SIZE can be at most 64 KB less 16 bytes to account for 8 bytes UDP-header

and 8 bytes application overhead per packet. However, for all the experiments, this parameter was set at 32 KB. The reason behind this selection was that after a few trials, the loss ratio due to collisions seemed to be higher in presence of load when very large UDP packets were used. This observation can be justified noting that a complete UDP packet is lost if any of its constituent IP packets is lost. Therefore, longer UDP packets represent larger steps in data loss accumulation. On the other hand, very small UDP packets introduce more overhead and also more I/O operations specially at the client that can cause delays and perhaps data loss due to receive-socket buffer overflow even with no-load condition.

The true impact of TR_BLK_SIZE on the video transmission has yet to be explored. The effect of I/O overhead on the performance of VOD application was proven in practice when the original 128 MHz CPU was upgraded to a 300 MHz CPU. While 8Mbps video sequences could not be successfully received and viewed with the slower CPU, after replacing it, not only could we receive the bit-streams without any loss in the socket buffer but we could also activate the monitoring features of our application which also involved a great deal of I/O overhead and kernel space procedures. The VOD system was also proven to be very demanding in terms of CPU-time consumption at the server. Any process running simultaneously on the server could slow down the rate of transmission and cause pauses in the picture.

Another reason for choosing 32 KB for TR_BLK_SIZE was the fact that for some of our video clips that contain pictures larger than 32 KB, we would have the opportunity to study fragmentation of MPEG-frames into more than one packet and distinguish between application and UDP layers.

PPF (packet per frame): It was set to 1, i.e., pictures were not partitioned before being passed to the transport layer for reasons similar to those mentioned above.

Socket Buffers: The default value for the socket buffer size in Linux is 64 KB. However, it has been observed that this parameter plays an important role in the process of video transmission and its effect has to be closely investigated.

In the absence of load, send-buffer does not seem to originate any problem even when it is much smaller than the default value. As a matter of fact, for the video sequence '6HD1AFFS.MPG' we found a buffer of size, oddly enough, 3400 bytes to be sufficient but necessary. Receive-buffer on the other hand, is more stringent. For the same sequence, under no-load conditions the minimum required buffer-size at the client was measured to be 55000 bytes. Although, one should note that 1-min video clips may not be long enough to cause an overflow situation.

In the presence of load, the default value may not be large enough for an MPEG-2 application especially for the send-socket buffer. As was explained before, the server sends each picture in a picture-interval with no attention to the conditions of the network. So, if the network is congested the data received from the application has to be stored in the socket buffer before gaining access to the channel. Under heavy-load conditions the required buffer space could be much more than 64KB.

On the other end of the channel, the client's socket has to also be larger as the network load increases. This is due to the fact that as soon as the server gains access to the channel, it will attempt to transmit all the data accumulated in its buffer. This may result in an overflow condition at the client's receive-buffer. Obviously, if the client replenishes its socket buffer at a slower rate than it receives data from the network (maximum 10 Mbps) overflow is inevitable.

Similar behaviors have been reported for Ethernet in [45] and [49]. Authors of these papers have noticed the effect of background traffic and large pictures in creating long delay variations (jitter) and consequently overflow/underflow situations. They classify the jitter of Ethernet frames as follows:

1. *Pre-capture Jitter:* When the channel is busy, the first MAC frame in the sender buffer has to wait to gain access to the network (*Deferment Jitter*). In this case, all the packets behind this one experience *head of queue jitter.* When a collision occurs, one of the other nodes may capture the channel. This makes the packets in the sender buffer to experience *collision jitter.* Collision jitter can be as high as $\sum_{i=1}^{16}(2^{\min(i,10)}-1)$ time slots or 263 msec for a 10 Mbps Ethernet where time slots are $51.2\mu s$ each. Deferment jitter is less significant. The maximum length of a background traffic packet is 1518 bytes. So, the maximum amount of time a video packet has to wait to attempt transmission is:

$$\frac{1518\times 8}{10Mbps}+9.6\mu s = 1.2ms$$

Pre-capture effect results in *packet clumping* where many packets are stored in the sender's buffer.

2. *Mid-capture Jitter:* when the sender finally gets the control of the channel after a period of pre-capture, it is likely to transmit many packets that have been clumped in its buffer. Capture effect contributes to this process. This will result in small inter-arrival times at the receiver.

Another factor that may contribute to overflow problem at the client is the fact that the decoder is only able to accept 30 pictures per second and it has only 1.92 MB to store video data. So, the extra amount of data has to be kept temporarily in the receive-socket buffer.

In order to avoid the possibility of overflowing the socket buffer, the socket buffer space was increased to 1.5 MB on both sides. To make sure that this value is large enough, a modified version of VOD software was used where 30 pictures were sent in a row within 1-sec intervals instead of sending one picture at a time. Under no-load

conditions, no overflow was experienced. Indeed, for the sequence "4HD1AFFS.MPG" a 410,000-byte receive-buffer and a 3400-byte send-buffer were big enough. Therefore, with the original VOD system these buffers can accommodate periods of no-access to the channel up to at least 30*1/30=1sec.

At the time these experiments were conducted, the analyses in chapter II had not been completed. If this was not the case, condition (II.5-6) could have been used to estimate the overall required buffer sizes at the server and the client for each sequence. Despite the fact that large buffers introduce large delays, we decided that in a VOD system, as a presentational application, preventing data loss is more critical.

Under these conditions, the main source of data loss will be the random binary exponential back-off algorithm when the VOD application attempts to gain access to the channel.

In order to increase the accuracy of our results, for each sequence and for each load condition, tests were performed five times and an average of the five trials was used in the analysis stage. In each trial, along with some other things that will be mentioned in future sections, inter-arrival times for MPEG-frames, UDP-datagrams and Ethernet-frames (from hereon the term *data-unit* may be used equivalently for all these data structures) were measured. Then first, second and third moments of them were evaluated.

III.8 VOD System Operation

When the server is launched, it opens a socket and waits for a call from the client. If it receives a request it sends an acknowledgment back to the client and opens the video file. Then client sends a message indicating that it is ready to receive the video. The server starts by sending the pictures reading one picture from the hard disk at a time until the end of the file is reached at which point the server stops transmitting and exits. The last

packet sent to the client is always an empty packet that the client interprets as the end of transmission and stops reading from its socket. The video decoder has thirty buffers of size 64KB each (1.92MB in total) to store the data before decoding. The playback delay depends on the video trace. For example, for a 4Mbps encoded video clip under no-load condition, if 20 buffers have to be pre-filled this delay can be found to be:

$$\frac{20 \times 64\ KB}{4\ Mbps} = 2.56\ \text{sec}$$

One of the decoder's requirements is that it only accepts data in blocks of 64 KB. Thus, the client application makes sure not to pass the data read from the socket to the decoder before at least 64KB of data is available each time. Hence, the data is read from the socket into a temporary buffer of size 128 KB in the main memory first. Since UDP-datagrams have different lengths (always less than 64KB) the accumulated data in this buffer may be more than 64 KB before being transferred to the decoder. The portion of the data that is not sent to the decoder is copied to the beginning of the 128KB buffer and future data are stored in this buffer in a way that the order of received data is preserved.

CHAPTER IV

MEASUREMENTS AND RESULTS

IV.1 Performance Metrics

In this chapter, first we explain how the VOD application and IW95000 test-tool were used to perform measurements on the traffic. A list of statistics that were obtained for inter-arrival times of IP-packets and MPEG frames is provided in section IV.2. Finally, in section IV.3 we analyze the measurements of interest.

IV.1.1 Application Level

Data Loss Ratio: The server application reports the size (in terms of bytes) and the number of each picture being transmitted as well as constituent packets. Also, at the end of transmission it reports the total number of pictures, packets (UDP-datagrams equivalently), video-bytes and data-bytes that were transmitted. In case pictures are being partitioned before transmission, the size of each partition is also reported. This data can be saved in a file and be reviewed later.

On the other side, the client application records information that pertain to the received data such as pic-no, packet-id, size of pictures and size of packets. By comparison, one can identify the lost pictures or packets. A picture is considered lost if none of its constructing packets are received. At the end of transmission, the client reports the total number of pictures, datagrams, video-bytes, data-bytes and the duration of transmission. Using this information and the corresponding items reported by the server, the amount of lost data can be found in terms of bytes, datagrams and pictures.

The client also records the offset time of arrival of each UDP-datagram and each picture in separate files along with corresponding pic-no and packet-id. Using these files,

one can know the approximate time that a certain lost part of the video was supposed to arrive.

Inter-arrival Times: Using the above information, the delay between arrivals of two consecutive pictures or datagrams can be calculated. By processing this data, the frequency distribution of picture inter-arrival times and datagram inter-arrival times as well as their running averages can be found.

Decoder Buffer Occupancy: Every time the client attempts to write a block of 64KB to the decoder it also inquires the number of decoder buffers that are full at the time and records that in a file (buf_occ) as well as the current time. This way we can study buffer requirements of different video sequences.

Socket Input/Output Rates: When the server finishes writing one picture to its socket, it computes the time passed since the end of last picture (last_time_passed) and then registers the ratio "pic_size/last_time_passed" in a file as well as the current time. Note that this value is in fact the source rate, which is input to the socket not to the network. At the end of transmission it also reports the mean rate which is the total number of bytes transmitted divided by duration of transmission.

Similarly, every time the client reaches the beginning of a new picture, it computes the time passed since the beginning of the current picture (last_time_passed) and records the ratio "pic_size/last_time_passed" along with the current time in a file. Note that this value is not the socket throughput rather it is the output of the socket, which depends on how often the client reads from the socket.

IV.1.2 Network Level

Network Utilization: IW95000 plots the level of network utilization versus time by MPEG-2 traffic generated from the server as well as total utilization in case there is a background traffic on the network. An average of both quantities is also available.

Network Throughput: IW95000 plots the bandwidth consumed in terms of bytes/sec versus time for the MPEG-2 traffic as well as the total occupied bandwidth if there is a background traffic. A separate graph is provided by the test-tool that shows throughput in terms of MAC-frames per second. An average of these quantities is also available.

MAC-frame Inter-arrival Times: IW95000 can decode Ethernet frames according to a specified protocol filter. The arrival time of each frame is given which can be used to calculate the inter-arrival times (with 1 μsec precision).

Collision Counts: IW95000 can monitor the number of collisions between stations connected to the network as well as collisions between stations and its own traffic generator.

IV.2 Statistics

For each sequence and for each load condition the following statistics were obtained for data-units inter-arrival times (denoted by X from hereon).

- *Sample Mean ($\bar{x}$):* Sum of all inter-arrival times in a trial divided by sample size (N), as a measure of central tendency.
- *Sample Standard Deviation (STD):* A measure of variation and dispersion

$$STD = \sqrt{\frac{\sum_{i=1}^{N}(x_i - \bar{x})^2}{N-1}}$$

- *Median:* A value that 50% of the total number of X's lie below it.
- *First Quartile (Q1):* A value that 25% of the total number of X's lie below it.
- *Third Quartile (Q3):* A value that 75% of the total number of X's lie below it.

- *Quartile Deviation (QD):* $QD = \frac{Q3 - Q1}{2}$

- *Mean Absolute Deviation (MAD):*

$$MAD = \sum_{i=1}^{N} \frac{|x_i - median|}{N}$$

- *Coefficient of Variation (COV):* A measure of relative variation

$$COV = STD / Mean$$

- *Kurtosis:* A measure of concentration about the mean compared to a Normal distribution with the same standard deviation.[59]

- *Skewness (sk):* A measure of symmetry of a distribution around the mean. [59]

- *Index of Dispersion for Inter-arrivals (IDI)* [60]: A measure of burstiness.
  Let S_k be the process that represents the sum of k consecutive inter-arrival times $\{X_1,...,X_k\}$. Then IDI denoted here by J_k is defined as $Var[S_k]$ standardized by its value for a Poisson process;

$$J_k = \frac{Var[S_k]}{kE^2[X]}$$

  Since $E[S_k] = kE[X]$ it follows:

$$J_k = \frac{kVar[S_k]}{E^2[S_k]}$$

  According to the definition, for a Poisson process $J_k = 1$ for any k. In general however, for a renewal process[9], J_k is equal to the squared coefficient of variation because $Var[S_k] = kVar[X]$.

[9] A renewal process is a series of events in which the times between events are independently and identically distributed (iid). Poisson is an example of a renewal process.[60]

Essentially, IDI compares a process against a renewal process. Obviously, our arrival process at none of the three layers is a renewal process because arrivals are not iid. For example, IP-packet arrivals at the beginning of each picture-interval are more likely than at the end. Also, in a period of access to the channel immediately after a period of no access, arrivals at all three layers are more probable. In order to calculate IDI for our arrival process we used the sample mean and sample variance of S_k as estimators for $E[S_k]$ and $Var[S_k]$. In each trial, k was chosen to be 1/1000 of the sample size.

$$IDI = \frac{\sigma^2_{S_k}}{m_{S_k}}$$

- *Index of Dispersion for Counts* (IDC) [60]: Another measure of burstiness that also describes how a process deviates from a renewal process. Let N_t be the process that counts the number of arrivals in the interval (0,t]. IDC denoted here by I(t) is defined as:

$$I(t) = \frac{Var[N_t]}{E[N_t]}$$

For all three layers we chose t=100msec. Number of arrivals in each one of these time intervals forms a sample set. Again, we used sample mean and sample variance of N_t as estimators of $E[N_t]$ and $Var[N_t]$.

$$IDC = \frac{\sigma^2_{N_t}}{m_{N_t}}$$

IV.3 Results

IV.3.1. Decoder's Buffer Occupancy

Figures 21 through 24 show the decoder's buffer length with 0% and 97% background load for "4HD1AFFP.MPG" (CBR) and "6hook10q.mpg" (VBR).

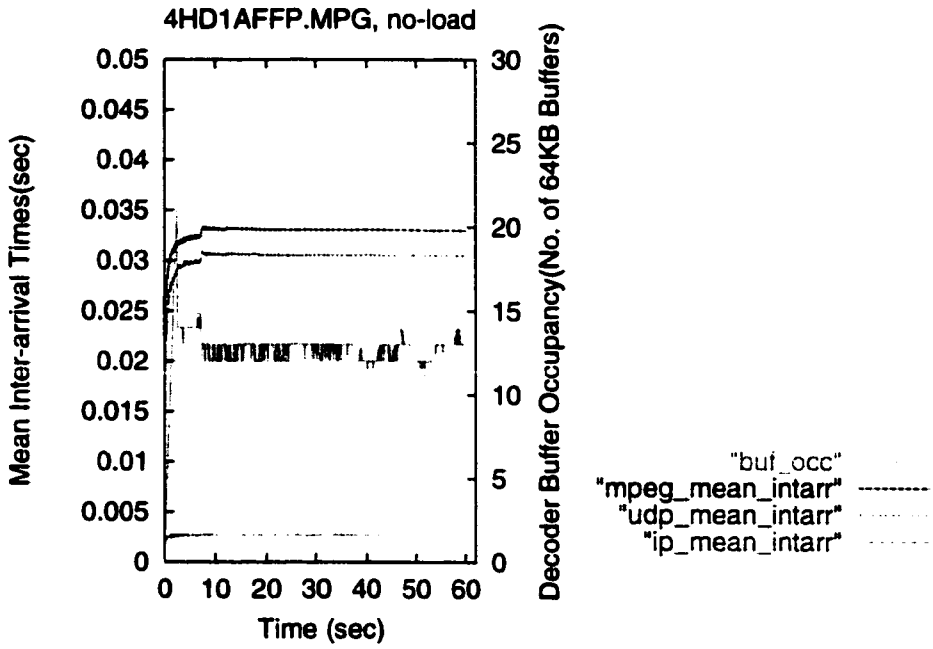

Figure 21. Decoder's Buffer Length: 4HD1AFFP.MPG, CBR, 4Mbps, 30fps No-load

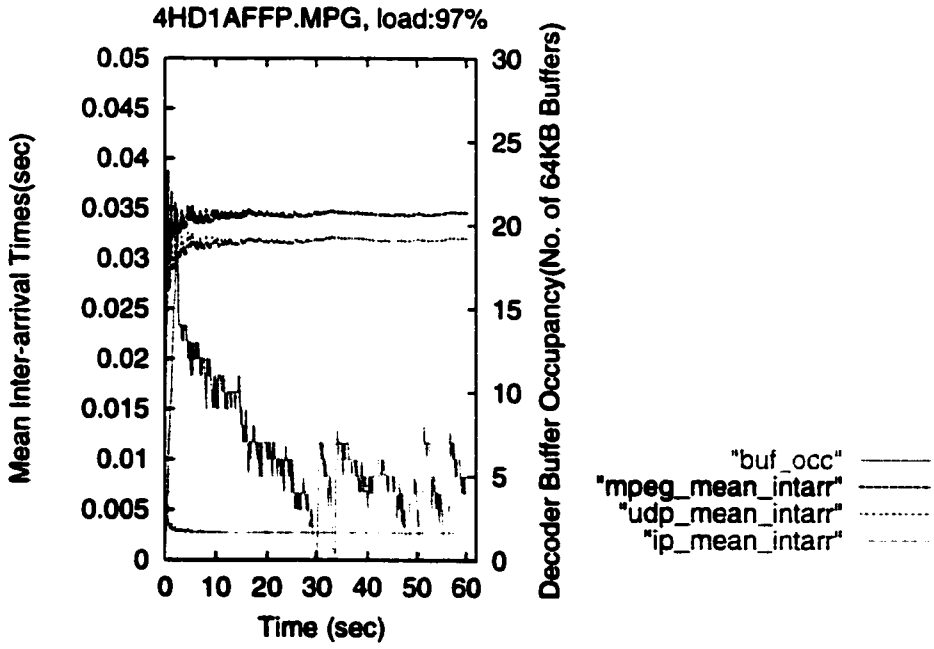

Figure 22. Decoder's Buffer Length: 4HD1AFFP.MPG, CBR, 4Mbps, 30fps Load=97%

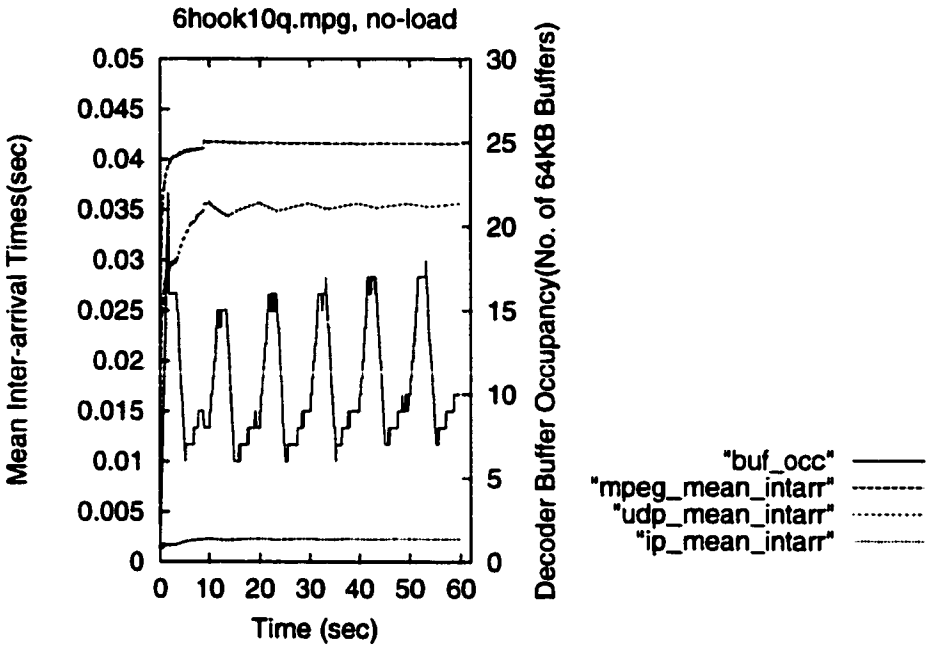

Figure 23. Decoder's Buffer Length: 6hook10q.mpg, VBR, Q=10, 24fps No-load

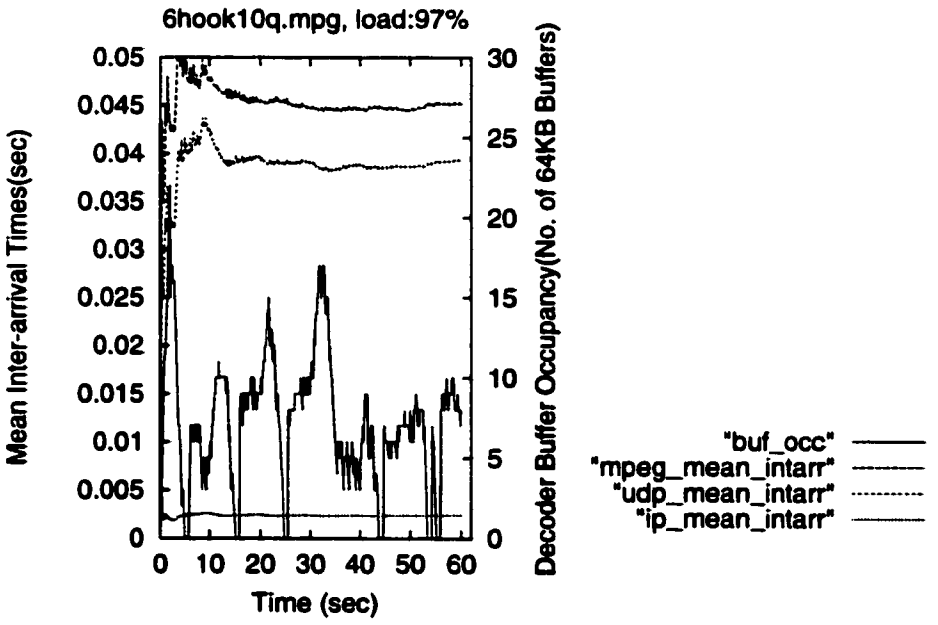

Figure 24. Decoder's Buffer Length: 6hook10q.mpg, VBR, Q=10, 24fps Load=97%

As previously mentioned, the video decoder requires at least 16 full buffers (each 64 KB) before launching the playback process. During all our experimental tests we have preloaded 20 out of 30 buffers. As the network load increases, transmission delay increases that results in pauses in the video represented by drainage points where the graph meets the horizontal axis. Each time the decoder runs out of data it waits for at least eight buffers to be filled before restarting the video.

From our experiments it has become evident that the buffer requirements for the VBR sequences are much more stringent than those for the CBR sequences. Note that for the VBR sequence "6hook10q.mpg" at some points the decoder has required to retrieve almost 8 of its buffers (512 KB) in a short period of time in order to satisfy the desired display rate of 24 frames per second. This means that we need a buffer size in the order of 512 KB to compensate for a slow or temporarily congested channel. On the other hand, the buffer occupancy for CBR sequences is almost constant which means the rate at which the decoder retrieves data from its socket is almost equal to the rate at which data are received. In the case of CBR video, the need to a buffer is not significant.

Our experimental results also indicate that VBR sequences are more sensitive to background load. Figures 25 and 26 compare the decoder's buffer occupancy for 6hook10q.mpg and 4HD1AFFP.MPG for 95%load. Even though the mean bit-rates of both sequences are in the same order of magnitude, the VBR sequence is much more affected by the load than the CBR sequence.

Figures 21 through 26 depict also the running average of the inter-arrival times at the 3 layers under study. When the level of background traffic is low, these curves quickly converge to their mean values but at higher levels of load this convergence is much slower.

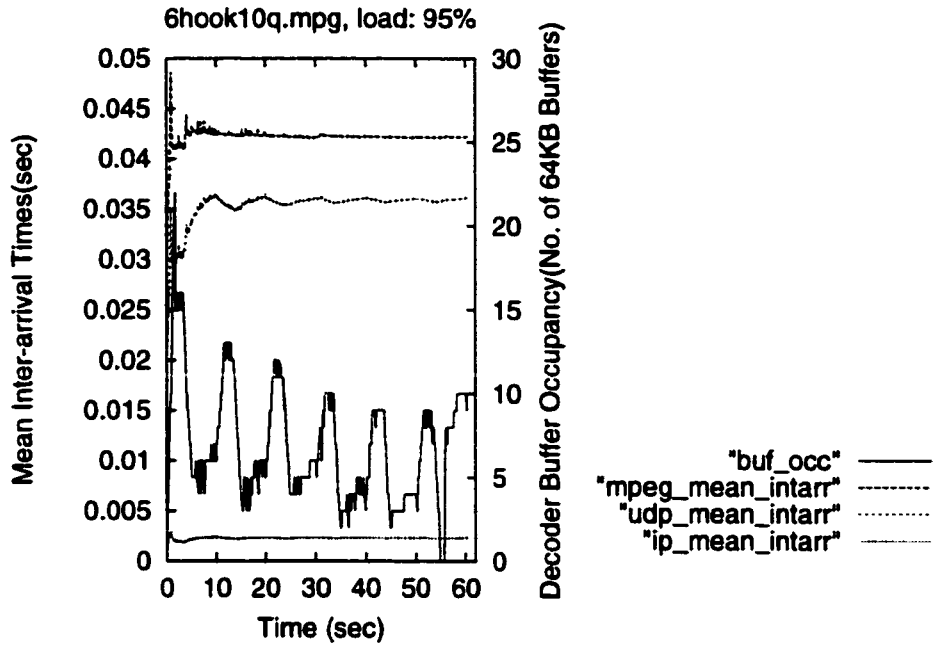

Figure 25. Decoder's Buffer Length: 6hook10q.mpg, VBR, Q=10, 24fps Load=95%

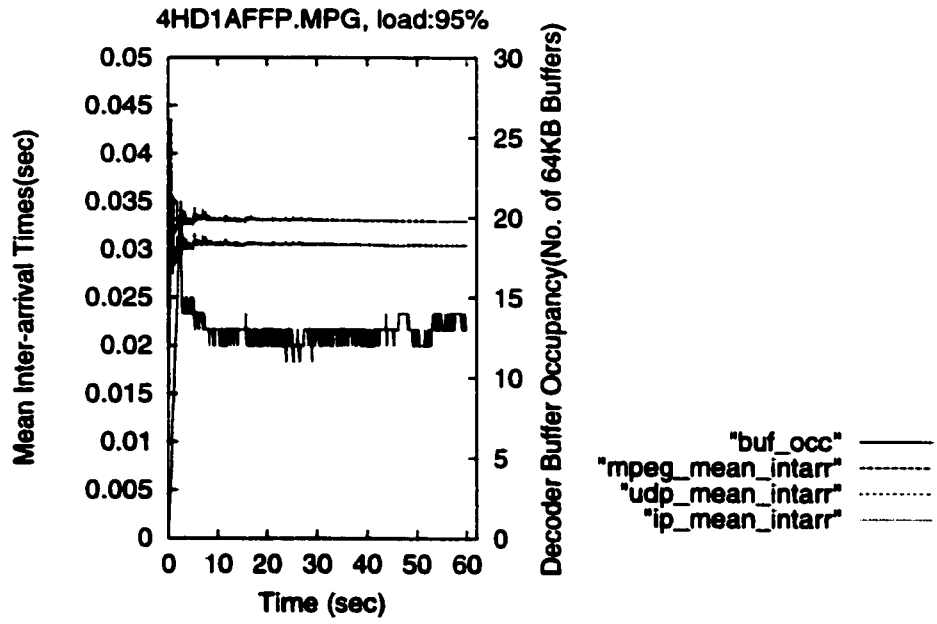

Figure 26. Decoder's Buffer Length: 4HD1AFFP.MPG, CBR, 4Mbps, 30fps Load=95%

IV.3.2. Inter-arrival Times

Figures 27 through 32 display examples of graphs that were obtained for inter-arrival times vs. time for all layers. In these figures, variations of inter-arrival times of data-units over time with 0% and 97% background load have been shown for the sequence "6hook10q.mpg".

As can be seen, in the no-load condition, inter-arrival times at all layers are slightly and regularly spread out around a mean value whereas in the presence of background traffic they demonstrate a volatile behavior evident from the large spikes in the graphs. These long delays are due to collisions in the channel that result in long waiting times for video to be transmitted. On the other hand, we can easily notice some down-spikes in the graphs which represent very small inter-arrival times. These ones are due to collisions too. In fact, in periods of no access to the channel, application data accumulate in the server's socket buffer and once the access to the network is granted a burst of data is generated where delay between consecutive data-units is very small. This phenomenon was explained earlier as pre-capture and mid-capture effects.

Because each picture is sent in one picture-interval, one may wonder why there are some variations, however small, around the mean inter-arrival time even in the absence of background traffic. A small number of inter-arrival times above the mean are of course due to some large pictures that need more time than a picture-interval to be transmitted. However, the rest of them can be explained as follows:

In section "IV.1 Performance Metrics" we described how picture inter-arrival times are calculated after the receive-socket and in the application space. Some UDP-datagrams that arrive at the client may not be read from the socket immediately. Indeed, they may pile up and then be read at once. So, the measured inter-arrival times between them and perhaps pictures that they belong to are minimal. On the other hand, the delay between this "read" operation and the last one may be so large that the overall measured picture inter-arrival time becomes larger than its corresponding value at the server.

However, this delay is not visually experienced because enough data exists in the decoder's buffers. Remember that 20 of the decoder's buffers are filled before playback of the video starts.

IV.3.3. Frequency Distribution of Inter-arrival Times

For the first trial, among five similar trials performed under the same load conditions, frequency distribution of inter-arrival times of all data-units were obtained. Figures 33 through 38 demonstrate those graphs for sequence "6hook10q.mpg" With 0% and 97% background load. It has to be noted that class widths for MPEG and UDP layers are 1msec while at the IP-layer they are 10 μ sec.

For all bit-streams, it was observed that the distribution of inter-arrival times at all layers approaches a Poisson-like PDF as the background load increases from a Gaussian-like PDF that exists for no-load condition. In other words, when no background traffic is present data-units arrive regularly therefore inter-arrival times are concentrated about some mean value. However, when background load is injected in the channel, some data-units are delayed longer than the mean value but for each such data-unit and right after that, a group of data-units will have inter-arrival times that are smaller than the mean value. This mechanism corresponds to capture effect previously explained. Because of intense concentration of inter-arrival times about 2msec at the IP-layer, one can appreciate this concept more easily using figures 39 and 40 which depict the same distributions as in figures 33 and 34 but in a logarithmic scale. Nevertheless, in all cases the mean value of inter-arrival times slightly increases with background load as can be seen from figures 41 and 42 which will be explained next.

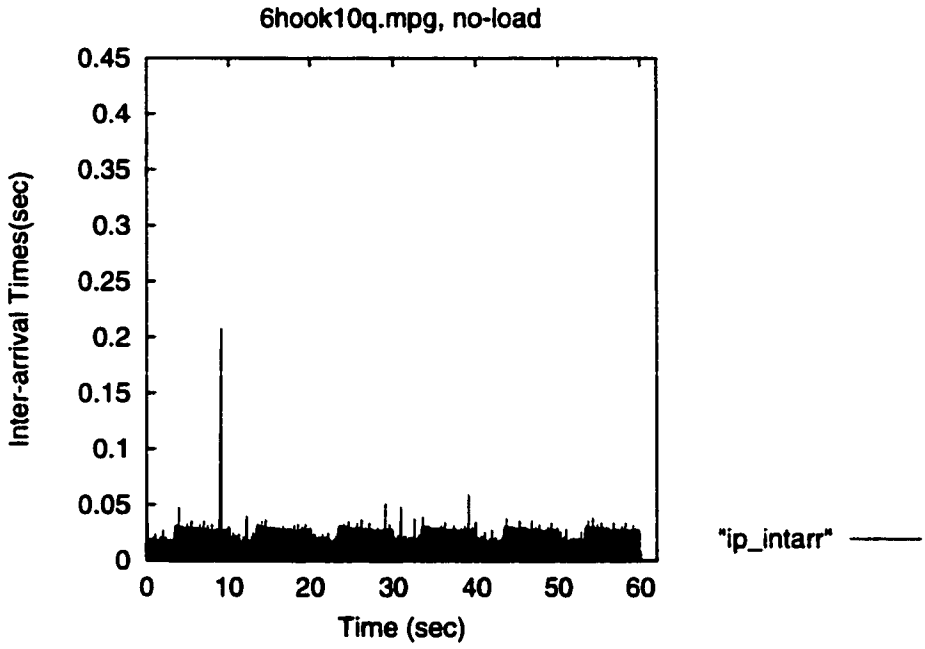

Figure 27. IP Inter-arrival Times: 6hook10q.mpg, VBR, Q=10, 24fps No-load

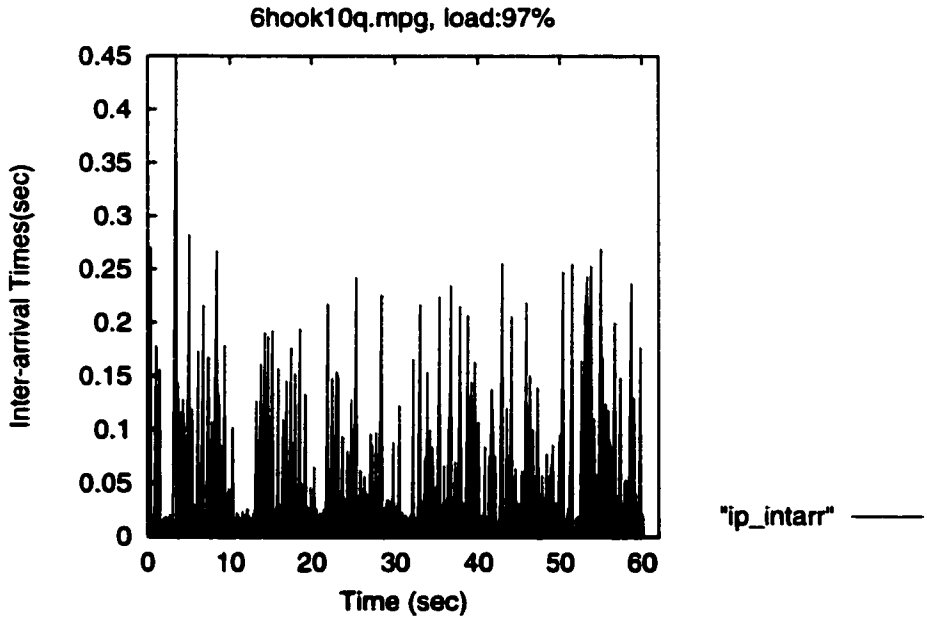

Figure 28. IP Inter-arrival Times: 6hook10q.mpg, VBR, Q=10, 24fps Load=97%

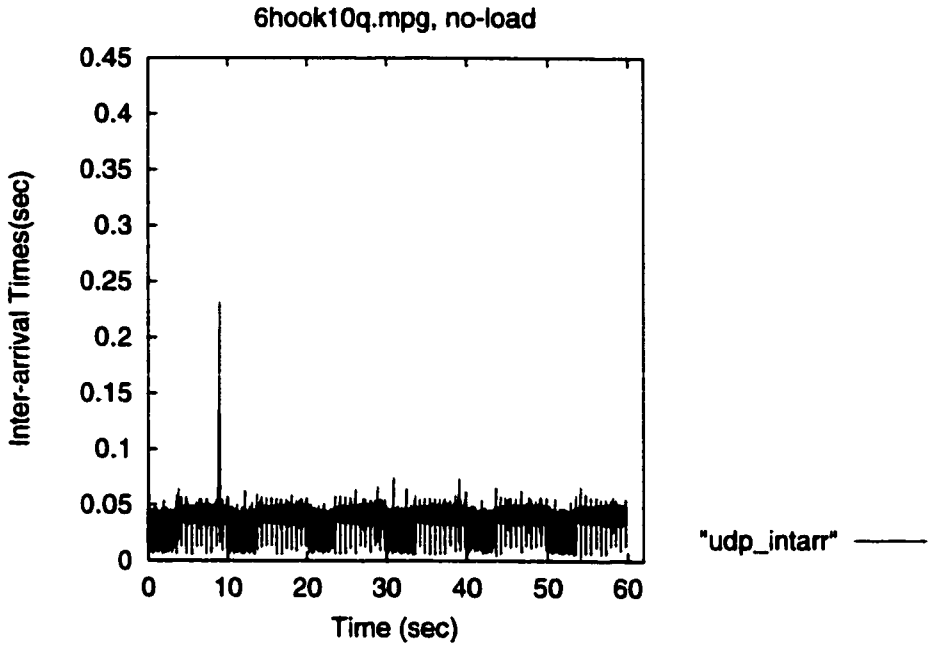

Figure 29. UDP Inter-arrival Times: 6hook10q.mpg, VBR, Q=10, 24fps No-load

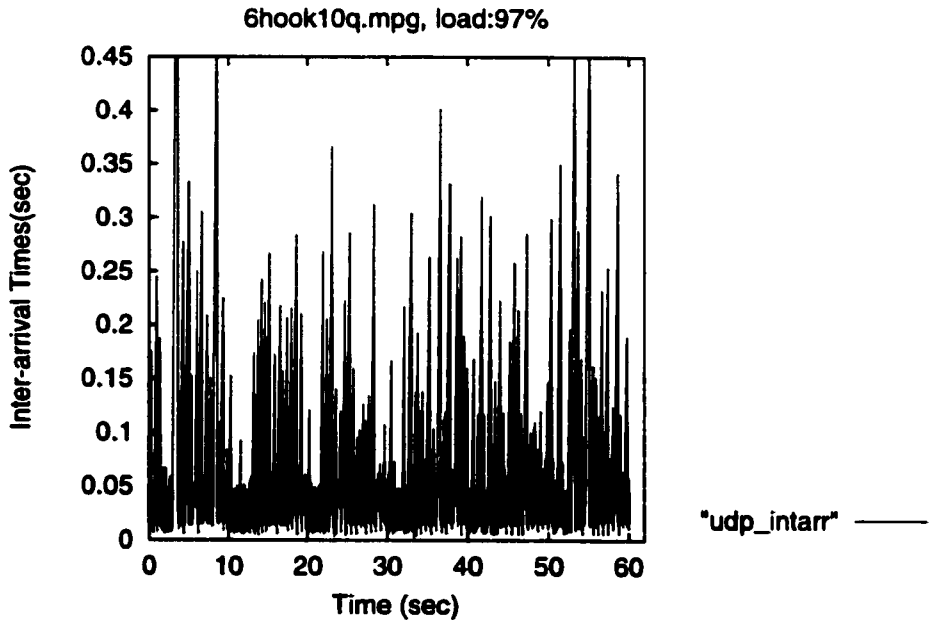

Figure 30. UDP Inter-arrival Times: 6hook10q.mpg, VBR, Q=10, 24fps Load=97%

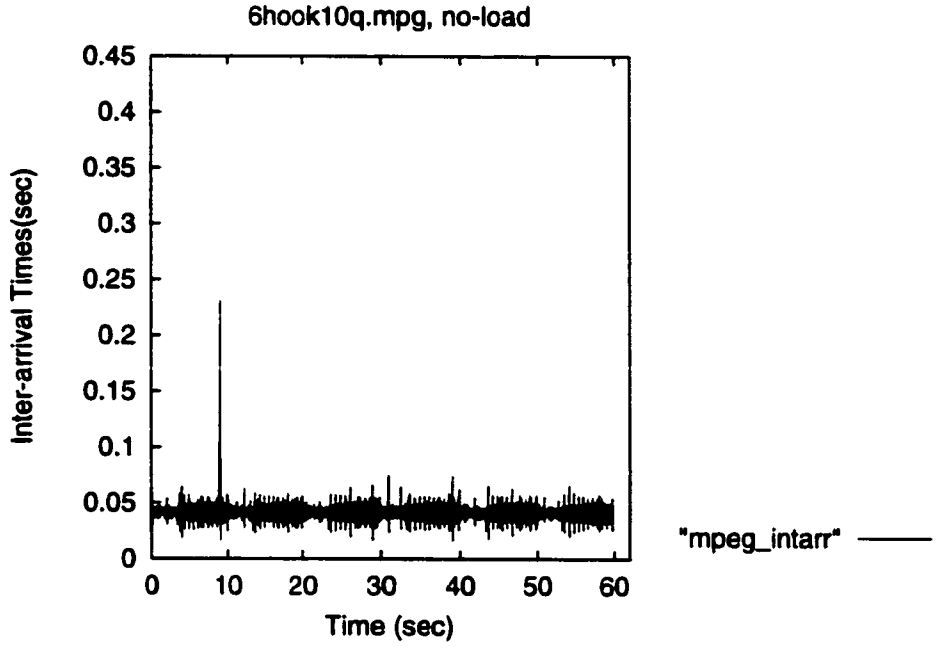

Figure 31. MPEG Inter-arrival Times: 6hook10q.mpg, VBR, Q=10, 24fps No-load

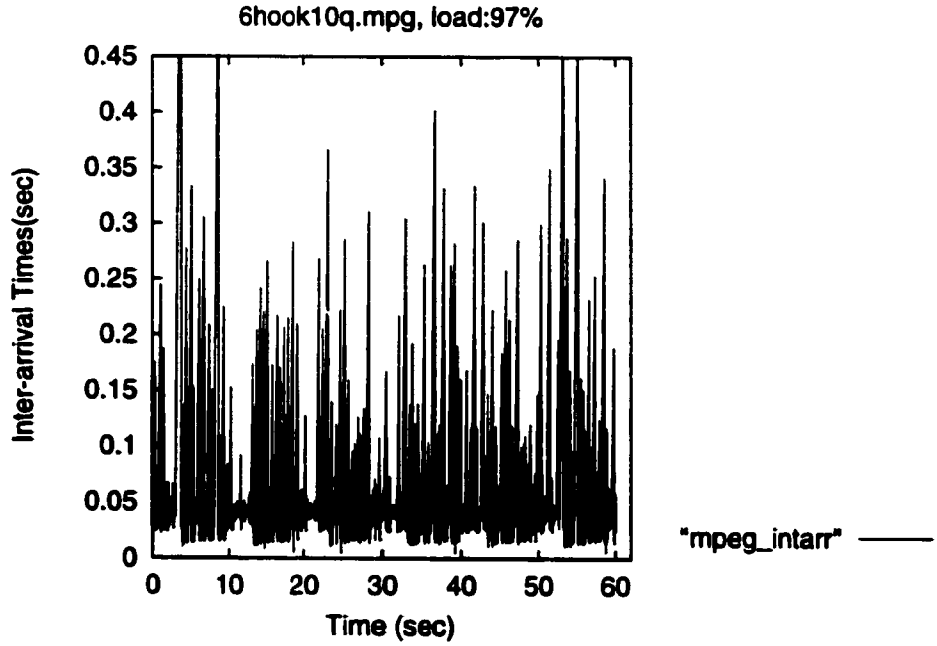

Figure 32. MPEG Inter-arrival Times: 6hook10q.mpg, VBR, Q=10, 24fps Load=97%

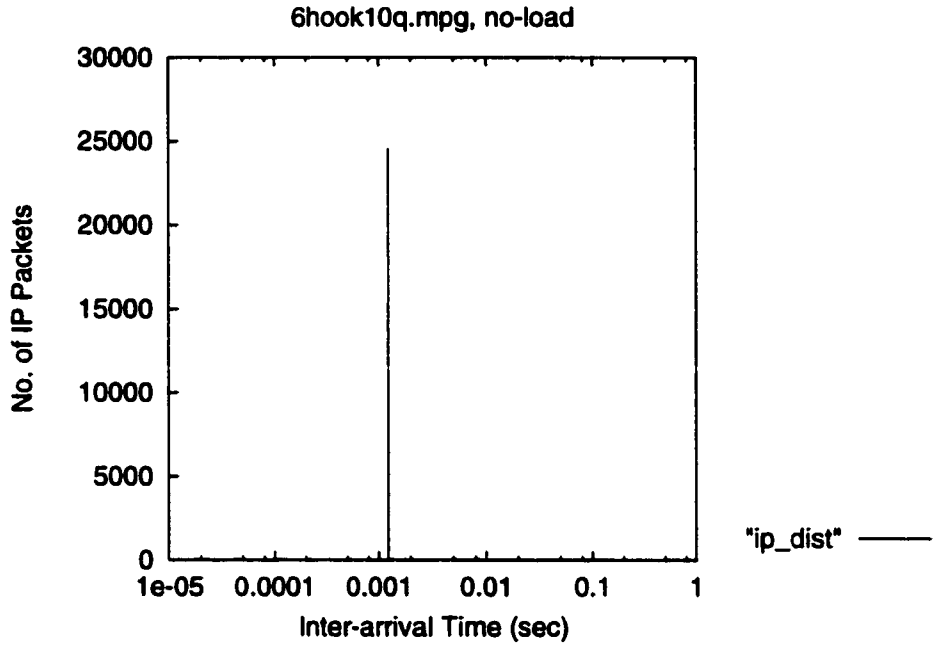

Figure 33. Distribution of IP Inter-arrival Times: 6hook10q.mpg , VBR No-load

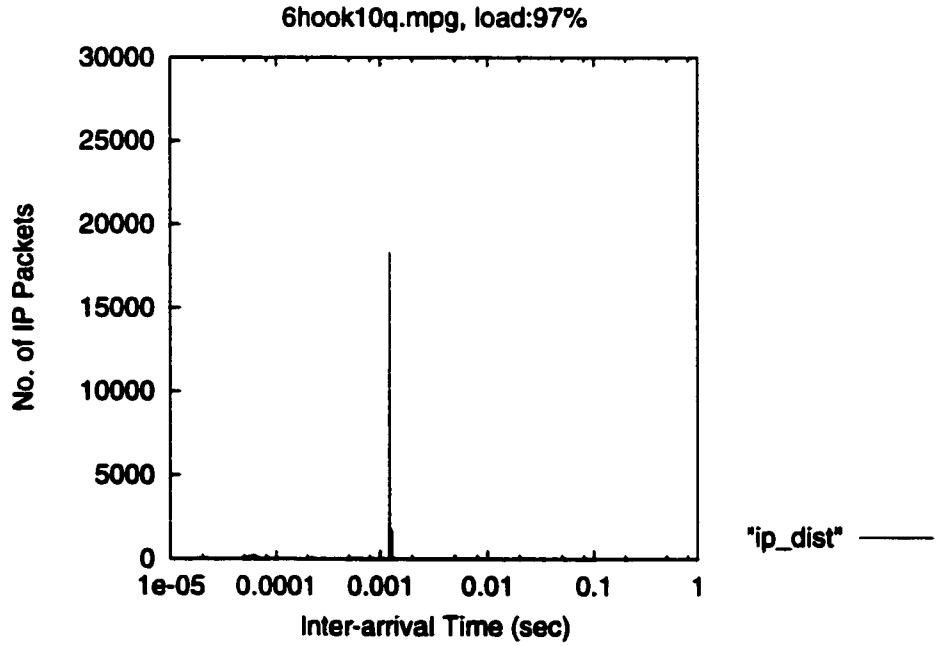

Figure 34. Distribution of IP Inter-arrival Times: 6hook10q.mpg, VBR Load=97%

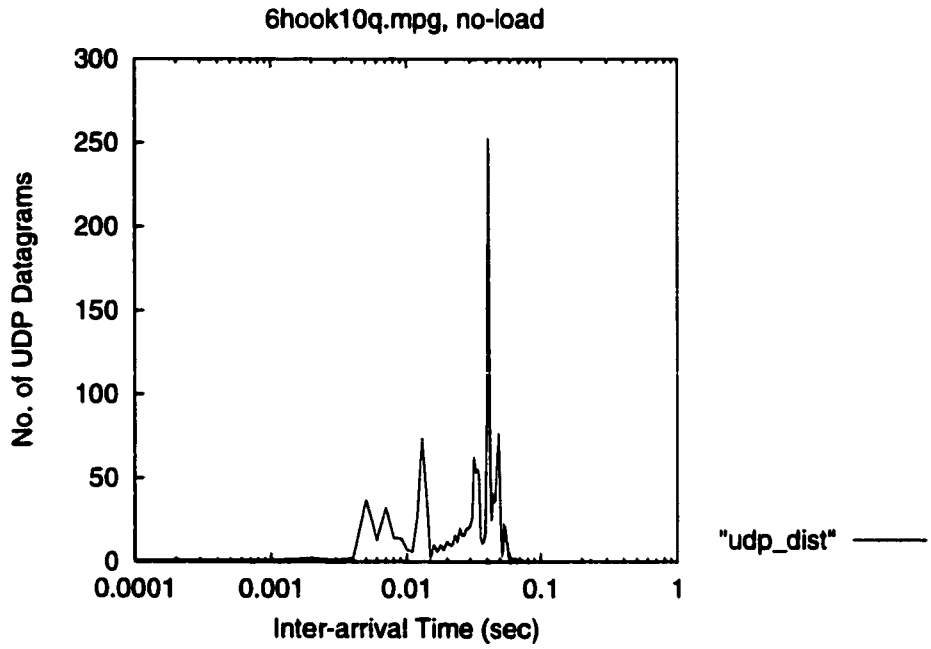

Figure 35. Distribution of UDP Inter-arrival Times: 6hook10q.mpg, VBR No-load

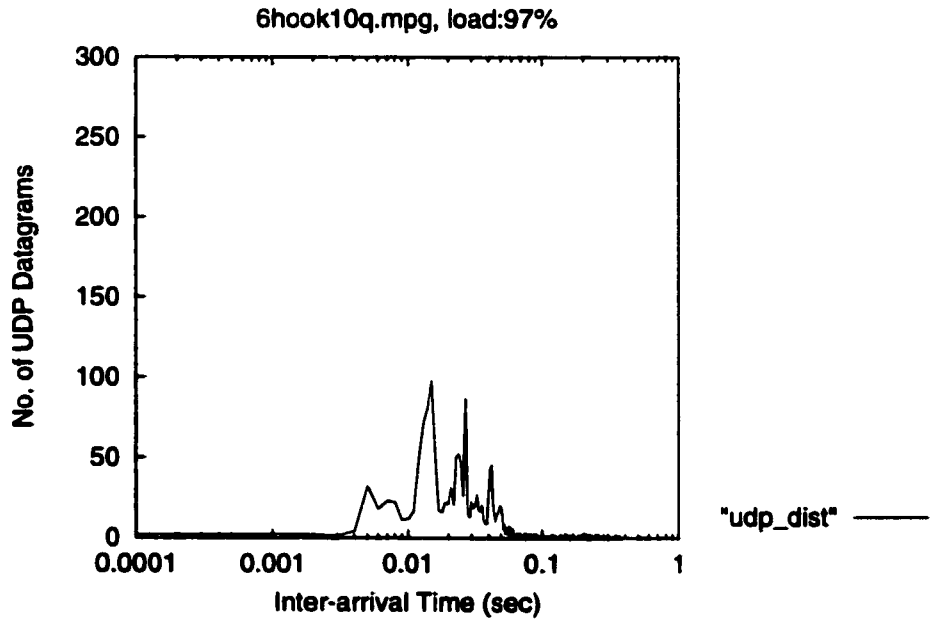

Figure 36. Distribution of UDP Inter-arrival Times: 6hook10q.mpg, VBR Load=97%

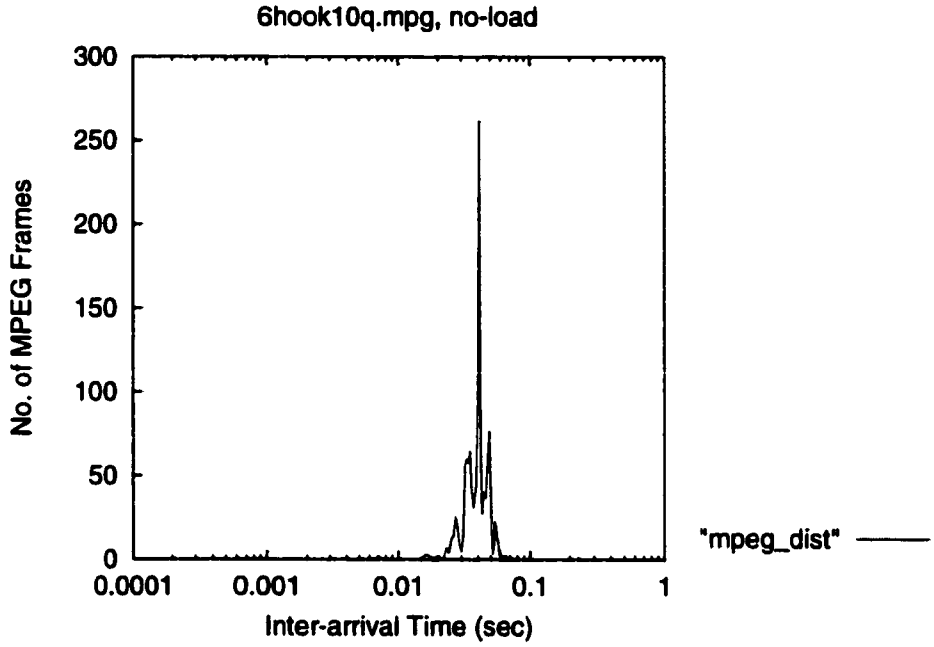

Figure 37. Distribution of MPEG Inter-arrival Times: 6hook10q.mpg, VBR No-load

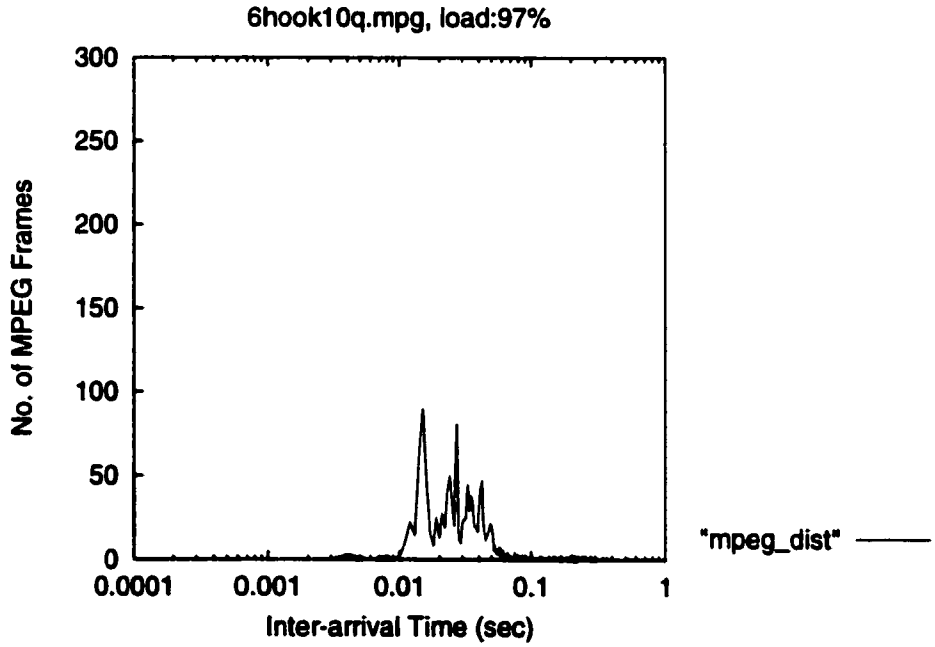

Figure 38. Distribution of MPEG Inter-arrival Times: 6hook10q.mpg, VBR Load=97%

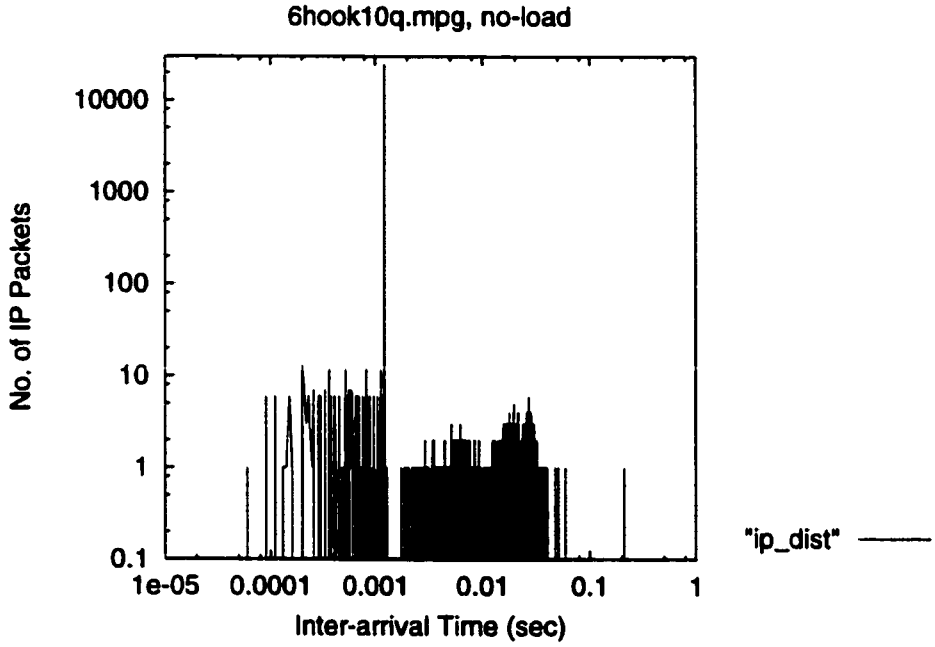

Figure 39. Distribution of IP Inter-arrival Times: 6hook10q.mpg, VBR No-load

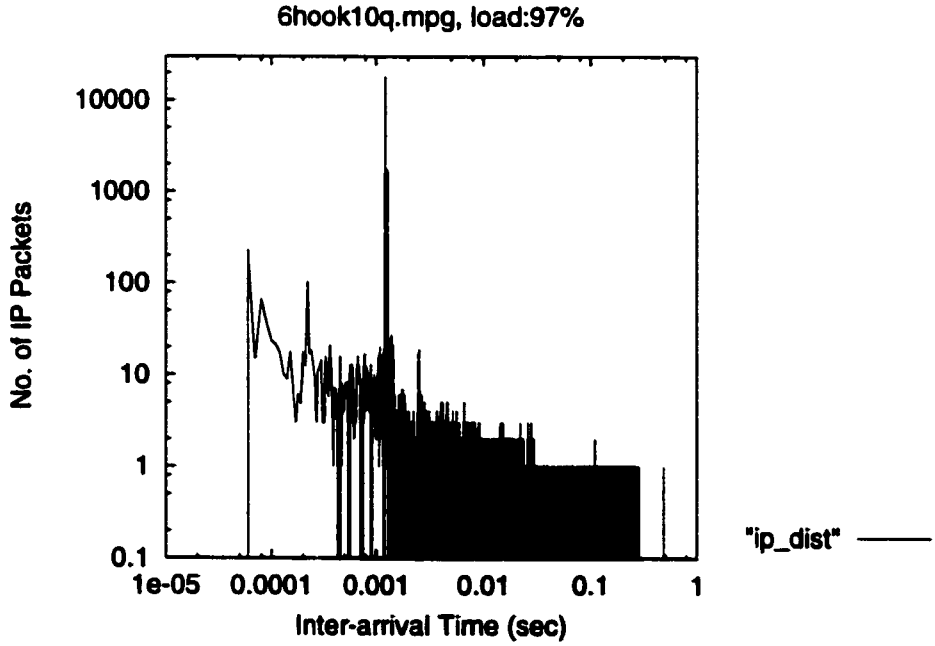

Figure 40. Distribution of IP Inter-arrival Times: 6hook10q.mpg, VBR Load=97%

IV.3.4. QoS Metrics

Among all the statistics and measurements that were obtained for inter-arrival times, in each set of tests, it seems that COV, STD, IDI and IDC could better serve our purpose, i.e. their variations follow the network conditions and quality of the video.

Figures 41 and 42 show COV for sequence "6hook10q.mpg" for MPEG and IP layers. Similar graphs were obtained for all other video clips. In these figures, probability of loss for video bytes can be read on the right hand axis and the horizontal axis gives the background load in terms of available bandwidth. As can be seen, those parts of the graphs below 70% load, have been cut off because the variations are negligible. On the other hand, for load levels of 95% or higher, the graphs demonstrate a fast rate of increase.

This behavior can be attributed to the contention-based nature of Ethernet. In fact, as load increases Ethernet shows a snowball effect where each collision results in even more collisions. As a result, most of the time on the channel is spent resolving collisions rather than data transmission [58][10]. Many researches such as [45] and [49] have reported this kind of behavior for Ethernet. They have noticed that the performance of Ethernet in transmission of compressed video is adequate up to almost 60% utilization of the network. At this point, network becomes saturated and performance reduction as a function of utilization shows a sharp slope. Our network starts to saturate at higher utilizations because we have only two transmitting nodes, the server and the traffic generator. This fact has also been noted in [44] where video transmission is successful with network utilization of 90% because only two traffic streams share the channel.

It has to be mentioned that the points on the horizontal axis do not actually represent the utilization of the network by the traffic generator in terms of bandwidth. Indeed, they are utilization in terms of time. For instance, a utilization of 95% corresponds to a throughput that is less than 9.5Mbps. The reason for this discrepancy lies in the fact that we have used very short frames of length 64 bytes for the background

traffic. Since there is at least 9.6 μs delay between any two consecutive Ethernet frames (IEEE 802.3 requirement), a considerable portion of the time on the channel is actually idle. We decided to use short frames because of their more severe effect as they result in more collisions due to a larger number of transmission attempts in order to maintain a certain level of utilization.

Figures 43 through 48 depict similar graphs for STD, IDI and IDC which demonstrate the same kind of increase with load as COV. As can be seen from all these graphs, as the load increases and naturally the quality of video decreases, these statistics of IP-packet inter-arrival times increase. Moreover, variations of each of them follow the variations of their MPEG layer counterpart. So, if the desirable level of any of these metrics at the application (MPEG) level is known we can decide what its IP-level counterpart should be or vice versa.

[10] Page 371

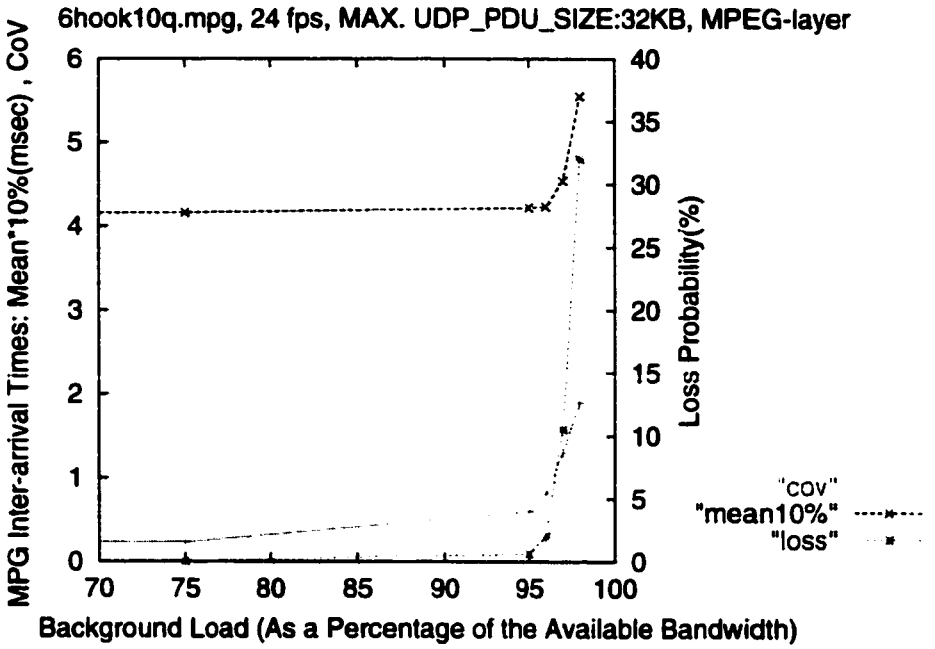

Figure 41. Coefficient of Variation of MPEG-Frame Inter-arrival Times 6hook10q.mpg

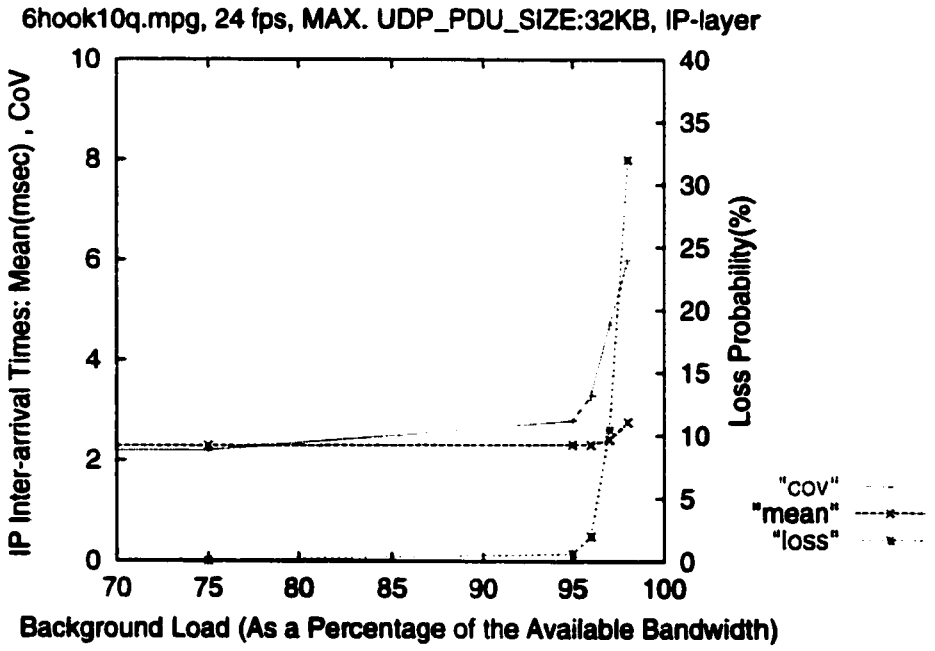

Figure 42. Coefficient of Variation of IP Inter-arrival Times 6hook10q.mpg

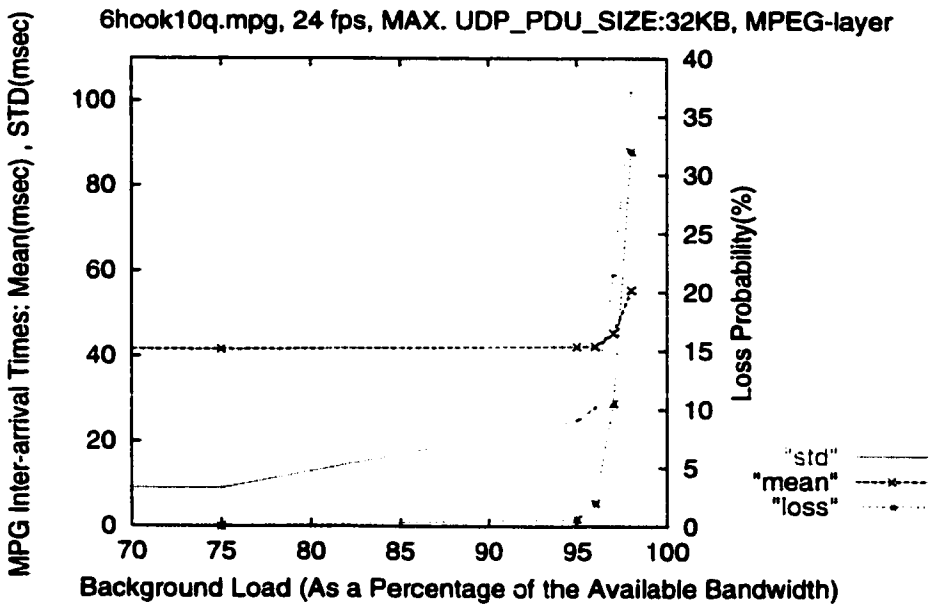

Figure 43. Standard Deviation of MPEG Inter-arrival Times 6hook10q.mpg

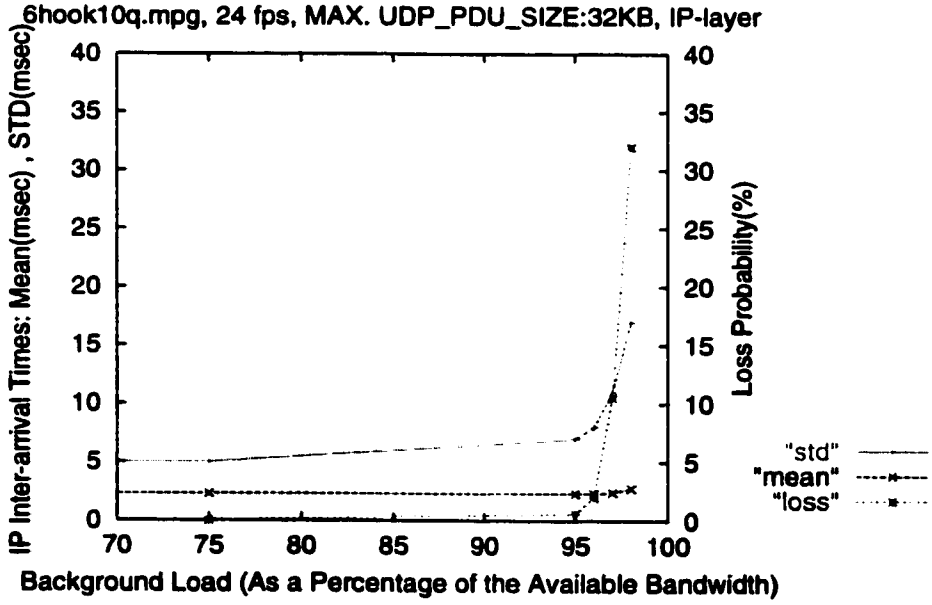

Figure 44. Standard Deviation of IP Inter-arrival Times 6hook10q.mpg

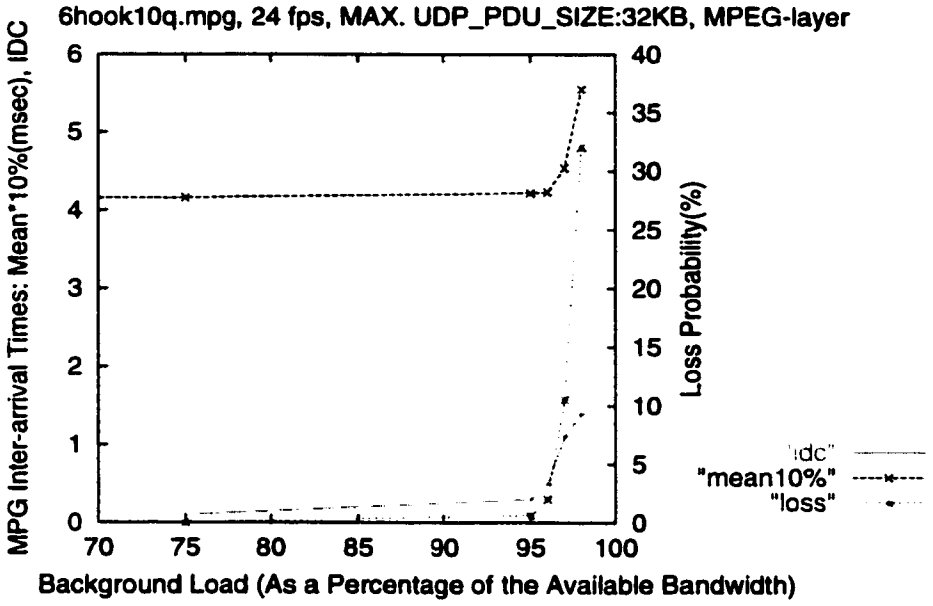

Figure 45. Index of Dispersion for Counts of MPEG frames 6hook10.mpg

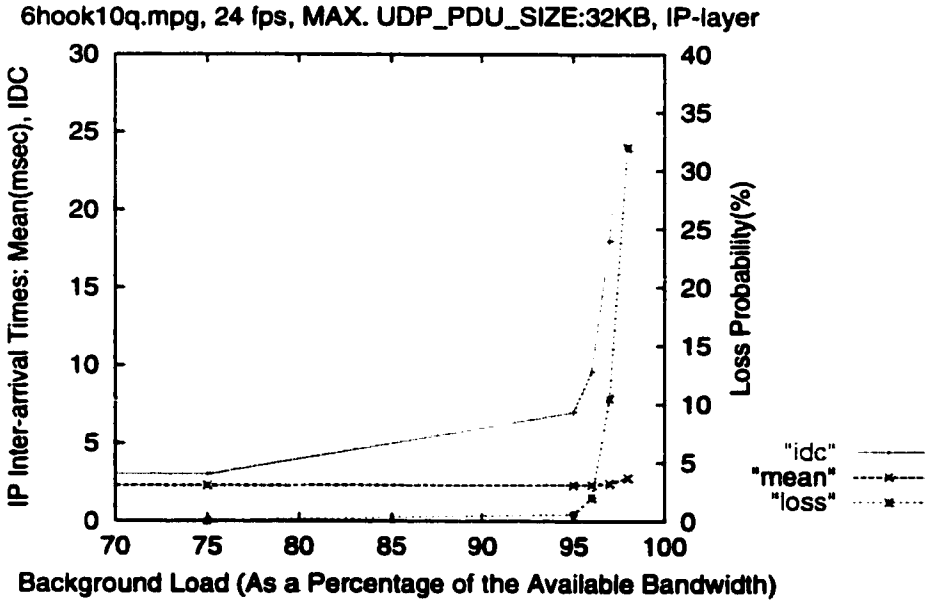

Figure 46. Index of Dispersion for Counts of IP Packets 6hook10q.mpg

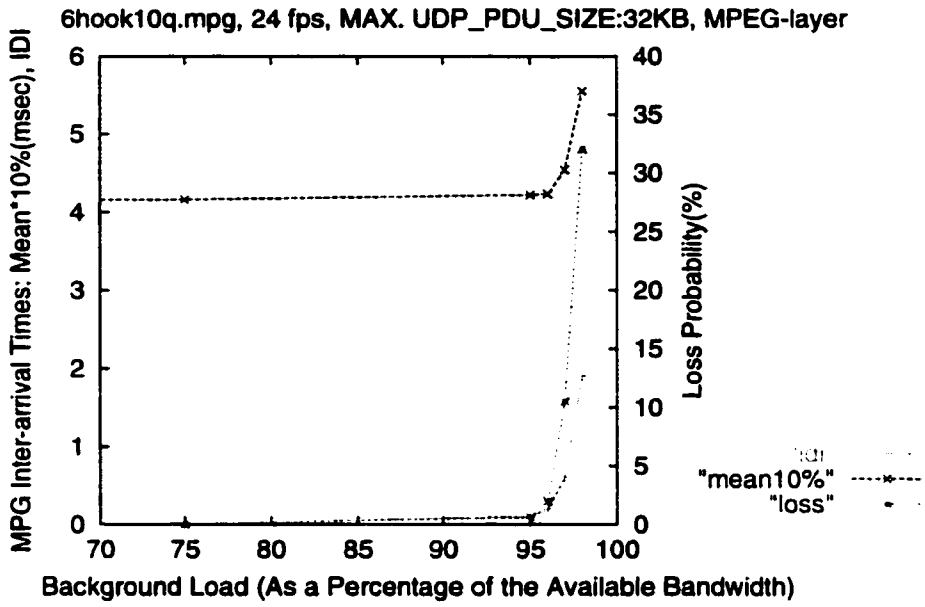

Figure 47. Index of Dispersion for Inter-arrival Times of MPEG Frames 6hook10q.mpg

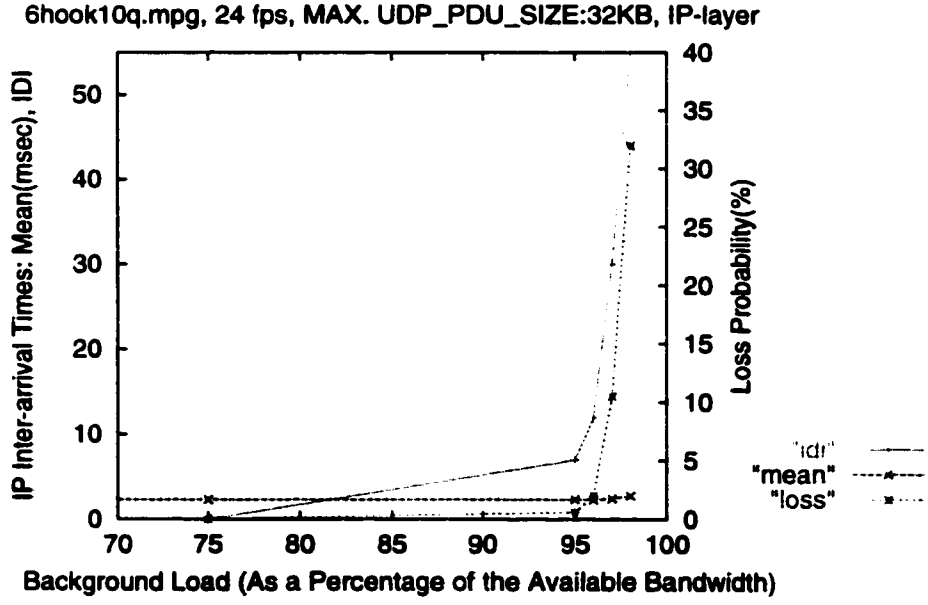

Figure 48. Index of Dispersion for Inter-arrival Times of IP Packets 6hook10q.mpg

CHAPTER VI

CONCLUSIONS AND RECOMMENDATIONS

By realizing the importance of buffer overflow/ underflow conditions at the sender and the receiver in a VOD application, through an analytical study, we found some constraints (boundaries) on the network throughput in each picture interval based on quantities known prior to that picture interval. These quantities include picture sizes, physical sizes of the buffers at the sender and the receiver, buffer length at the sender in the preceding interval and the playback delay. These constraints should be satisfied at the sender side. Another result of this analysis was an overall minimum buffer size at the sender and the receiver based on the size of pictures in the sequence.

We also performed an analysis on the impact of burstiness of the video traffic delivered to the decoder, on the jitter perceived by the user. This burstiness is the result of network inability to follow the variability in the picture sizes. To avoid jitter, we need to prevent starvation at the decoder's buffer. We found a relationship between the maximum acceptable jitter and the minimum sustained frame rate in the network. This relationship can be viewed as a mapping rule between user's QoS parameters (jitter) and network QoS parameters (throughput). At the same time, to avoid overflow we need a buffer that can accommodate the longest delay possibly experienced by a frame in the play-out buffer. This resulted in a formula for the physical size of the play-out buffer based on the maximum rate at which frames are fed to the receiver. Having found this value, knowing the minimum overall buffer requirement found previously, a minimum can be found for the sender buffer.

In our experiments, we concentrated mainly on QoS mapping. We believe that graphs such as those ones shown in figures 42 to 48 are providing us with the QoS mapping rule that we are looking for. In other words, any of STD, COV, IDI or IDC of

IP-packet inter-arrival times may be used to find out about the user-level quality of service. On one hand, variations of each of these quantities with network conditions (background load) at the network level, closely follow their counterparts at the application level. On the other hand, at both of these levels, they consistently increase as the background load increases (quality of video deteriorates). Especially, COV seems the most appropriate as it demonstrates the least degree of dependency on the type of video bit-stream. For the sake of brevity, similar graphs for other video clips have not been included.

During these experiments, we considered a subjective rating scale that reflected the visual quality of video as perceived by the user in which Q=1 and Q=10 represented the perfect and the poorest quality, respectively. In all our graphs for different metrics, the breaking point corresponds to Q=2 or Q=3 which always happens at 95% load. COV at this point, in our experiments, depending on the bit-stream, lies in the range [2.3,2.8]. Noticing that COV can have values of up to 6.2, this range of variation seems very small. Hence, it can be claimed that for any type of video (VBR with different Q-factors and CBR with different bit-rates) the user will find the quality of video acceptable as long as COV of IP-packet inter-arrival times is below 2.8.

Obviously, many more tests with different setups, have to be conducted to confirm this statement and to find the precise values or perhaps closed form formulas. As a potential continuation of this project, it would be interesting to experiment with video clips that are available at various levels of quality. For example, different encoding rates for the same CBR sequence or different quantization factor for the same VBR sequence. Also, more sophisticated subjective and objective methods of rating can be used to evaluate video quality for as was mentioned before mere data loss ratio is not an adequate indicative of video quality. Because, MPEG video is compressed in a way that different pieces of bit-stream vary in terms of significance. For example, loss of an I-frame is more destructive than a P-frame or a B-frame, in two ways. Firstly, because they carry more information and secondly because their loss propagates to future frames.

Looking back at our original motivation in figure 1, our newly found rule can be implemented in "QoS Evaluation" module. However, it is worth mentioning that the dynamics of COV or any other metric need to be explored further. To clarify this subject, remember that our experiments were conducted under certain conditions including sample sizes that correspond to 1-minute long video clips. Suppose that this is the minimum acceptable duration of sampling period in order to have a required accuracy and suppose that "QoS Evaluation" module performs a test similar to our experiments every 1 second and every time it uses a set of data gathered over past 1-minute interval. Therefore, in case of a change in the network conditions, 59/60 of the data used in the first test under new conditions will not be valid. If no other changes happen in the network, this module will need 1 minute to completely recover from the last change and produce reliable results. So, the rate of update and the amount of old data that can be used have to be optimized. This may also help verify which one of our statistics is the most robust in terms of response to the changes in network conditions.

Besides, we noticed that VBR sequences are more sensitive to the background load in the network and also have more stringent buffer requirements at the receiver.

REFERENCES

1. Ronnie T. Apteker, James A. Fisher, Valentin S. Kisimov, Hanoch Neishlos, "Video Acceptability and Frame Rate", IEEE Multimedia, Vol.2, No.3, Fall 1995, pp. 32-40.

2. CCIR Recommendation 500-3, "Method for the Subjective Assessment of the Quality of Television Picture" , 1986

3. Cristina Aurrecoechea, Andrew campbell, Linda Hauw Center for telecommunication research, Columbia University "A Survey of Quality of Service Architectures", ACM/ Springer Verlag Multimedia Systems Journal, Special Issue on QoS Architectures, May 1998, Vol. 6, No. 3, pp. 138- 151

4. Vogel, B. Kerherve, G.V. Bochmann and Jan Gecsei, "Distributed Multimedia and QoS: A Survey", IEEE Multimedia, Vol. 2, No. 2, Summer 95, pp. 10-18

5. Jean-Francois Huard and Aurel A.Lazar, "On QoS mapping in Multimedia Networks", http://comet.ctr.columbia.edu/~{jfhuard, aurel}

6. Stephan Voran and Stephan Wolf, "An Objective Technique for Assessing Video Impairments", IEEE Pacific Rim Conference on Communications, Computers and Signal Processing, 1993, Vol.1, pp. 161- 165

7. S. Banerjee, "Translating Application Requirements to ATM Cell Level Requirements", Proc. IEEE ICC 1997, Montreal, pp. 443- 447.

8. Klara Nahrstedt and Jonathan M.Smith, "The QoS Broker", IEEE Multimedia, Spring 1995, pp. 53- 67

9. Marco Alfano, "Design and Implementation of a Cooperative Multimedia Environment with QoS Control", Computer Communications, Vol. 21, 1998, pp. 350-361

10. Joao Bom, Paulo Marques, Miguel Correia and Paulo Pinto, "QoS Control: an Application Integrated Framework" , ICATM'98, First IEEE International Conference on ATM, pp. 283- 291

11. Lek Heng Ngoh, Aurel Lazar and Huanxu Pan, "A Multimedia-on-Demand System with End-to-End Quality of Service Guarantee", Proceedings (12th International Conference on Computer Communications) Information Highway for a Smaller World and Better Living, Seoul, Korea, August 1995, Vol.1, pp.35- 40

12. A. Seneviratne, M. Fry, V. Withana, V. Saparamdu, A. Richards and E.Horlait, "Quality of Service Management for Distributed Multimedia Applications", 1994

IEEE 13th Annual International Phoenix Conference on Computers and Communications, pp. 434- 439

13. W. Tawbi, L. Fedaoui, E. Horlait, "Dynamic QoS Issues in Distributed Multimedia Systems", Broadband Island, Second International Conference, Bridging the Services Gap (1993), pp. 69-75

14. B. Kerherve, A. Vogal, G. V. Bochman, R. Dssouli, J. Gecsei, A. Hafid, "On Distributed Multimedia Presentational Applications: Functional and Computational Architecture and QoS Negotiation", PFHSN'94, The 4th International IFIP Workshop on Protocols for high speed Networks, pp. 1-17

15. W. Tawbi, L. Fedaoui, E. Horlait, "Management of Multimedia Applications QoS on ATM Networks", IFIP International Conference on Computer Networks, Architectures and Applications, India, October 1992, pp. 26- 28

16. A.Campbell, G. Coulson, F. Garcia, D. Hutchinson, H. Leopold, "Integrated Quality of Service for Multimedia Communications", IEEE INFOCOM'93, San Francisco, USA, pp. 732- 739

17. H. Leopold, A. Campbell, D. Hitchinson and N.Singer, "Towards an Integrated Quality of Service Architecture for Distributed Multimedia Communications", IFIP Conference on High Performance Networking, Liege, Belgium, Dec. 1992, pp. 46- 55

18. S. Fischer. A. Hafid, G. V. Bochmann, H. de Meer, "Cooperative QoS Management for Multimedia Applications", ICMCS'97, pp. 303- 310

19. Stephan Bocking, "TIP's Performance Quality of Service", IEEE Communications Magazine, August 1995, pp. 74-81

20. N. Venkatasubramanian and K. Nahrstadt, "An Integrated Metric for Video QoS", ACM Multimedia Conference Proceedings' 97, pp. 371- 380

21. Z. Wang and J. Crowcroft, "Analysis of Burstiness and Jitter in Multimedia Communications", Proc. IEEE Globecom'93, Houston, Texas, USA, pp. 1496-1500.

22. R. L. Cruz, "A Calculus for Network Delay, Part I: Network Elements in Isolation", IEEE Trans. on Information Theory 37: 114-31, 1991, p. 90

23. P. Francis-Cobley and N. Davies, "Performance Implications of QoS Mapping in Heterogeneous Networks Involving ATM", Proc. IEEE ICATM'98, Colmar, Farnce, pp. 529-535.

24. U. Horn and B. Girod, "Scalable video transmission for the Internet", Computer Networks and ISDN Systems, Vol. 29, 1997, pp. 1833-1842.

25. C. Yuan Hsu, A. Ortega and A. R. Reibman, "Joint Selection of Source and Channel Rate for VBR Video Transmission Under ATM Policing Constraints", IEEE Journal on Selected Areas in Communications, Vol. 15, No. 6, August 1997, pp. 1016- 1028

26. L. Zhang, S. Deering, D. Estrin, S. Shenker and D. Zappala, "RSVP: A New Resource Reservation Protocol", IEEE Network Magazine, Vol.7, pp. 8- 18, September 1993

27. D. K. Y. Yau, S. S. Lam, "Adaptive Rate-Controlled Scheduling for Multimedia Applications", IEEE/ACM Transactions on Networking, Vol.5, No.4, August 1997, pp. 475- 488

28. P.Goyal, H.M. Vin, C. Shen, P.J. Shenoy, "A Reliable, Adaptive Network Protocol for Video Transport", Infocom'96, Vol.3, pp. 1080-1090

29. C. Rosado-Sosa, I. Rubin, "Jitter Compensation Scheduling Schemes for the Support of Real-Time Communications", ICC'98, IEEE International Conference on Communications, Atlanta, Georgia, USA, pp. 885- 890

30. H. Schulzrinne, S. Casner, R. frederick and V. Jacobson, "RTP: A Transport Protocol for Real-Time Applications", RFC-1889, 1996

31. Stefan Fischer and Stefan Leue, "Formal Methods for Broadband and Multimedia Systems", Computer Networks and ISDN Systems 30 (1998), pp. 865- 899

32. W. Feng, J. Rexford, "A Comparison of Bandwidth Smoothing Techniques for the Transmission of Prerecorded Compressed Video", INFOCOM'97, Vol.1, pp.58- 68

33. W. Feng, S. Sechrest, "Critical Bandwidth Allocation for the Delivery of Compressed Video", Computer Communications, Vol.18, No.10, October 1995, pp. 709- 717

34. W. Feng, S. Sechrest, "Smoothing and Buffering for Delivery of Prerecorded Compressed Video", Proc. Of the IS&T/SPIE Symposium on Multimedia Communications and Networking, February 1995, pp. 234- 242

35. W. Feng, F. Jahanian, S. Sechrest, "Optimal Buffering for the Delivery of Compressed Prerecorded Video", Proc. Of the IASTED/ISMM International Conference on Networks, January 1995, pp. 35- 46

36. J. Salehi, Z. Zhang, J. Kurose, D. Towsley, "Supporting Stored Video: Reducing Rate Variability and end-to-end Resource Requirements through Optimal Smoothing", Proc. Of ACM SIGMETRICS, May 1996, pp. 222- 231

37. Inwhee Joe, "Packet Loss and Jitter Control for Real-Time MPEG Video Communications", Computer Communications, Vol.19, 1996, pp. 901- 914

38. P. Cuenca, A. Garrido, F. Quiles and L. Orozco-Barbosa, "Performance Evaluation of Cell Discarding Mechanisms for the Distribution of VBR MPEG-2 Video over ATM Networks", IEEE Transactions on Broadcasting, Vol. 44, No. 2, June 1998, pp. 206-215

39. E. Mellaney, L.Orozco Barbosa and G. Gagnon, "Study of MPEG-2-Based Video Communications over an ATM Metropolitan Network", IEEE Transactions on Circuits and Systems for Video Technology, Vol. 7, No. 4, pp. 663-674, August 1997.

40. S. Gupta and C. L. Williamson, "An experimental Study of Video Traffic on an Ethernet Local Area Network", Proc. IEEE Globecom'94, pp. 558-562.

41. J. Dunlop, "Techniques for the Integration of Packet Voice and Data on IEEE 802.3 LANs", Computer Communications, Vol.12, No.5, October 1989, pp.273-280.

42. T. Apostolopoulos, "Model for the Analysis of a CSMA/CD Channel for Data and Voice Applications", *Computer* Communications, Vol. 14, No.2, March 1991, pp. 71-79

43. I. Dalgiç, W. Chien and G.A. Tobagi, "Evaluation of 10Base-T and 100Base-T Ethernets Carrying Video Audio and Data Traffic," Porc. IEEE INFOCOM'94, Toronto, pp. 1094-1102.

44. F. Tobagi, I. Dalgic, "Performance Evaluation of 10BASE-T and 100BASE-T Ethernets Carrying Multimedia Traffic", IEEE Journal on Selected Areas in Communications, Vol. 14, No.7, September 1996, pp. 1436- 1454

45. S. Deng, A. Bugos and P. Hill, "Design and Evaluation of an Ethernet-Based Residential Network", IEEE Journal on Selected Areas in Communications, Vol.14, No.6, August 1996, pp. 1138- 1150

46. Kathleen M. Nicholas, "Network Performance of Packet Video on a Local Area Network", 1992 IEEE 11th Annual International Phoenix Conference on Computers and Communications, pp. 659- 666

47. Francis Edwards, Mark Sculz, "Performance of VBR Packet Video Communications on an Ethernet LAN: A Trace-Driven Simulation Study", 1994 IEEE 13th Annual International Phoenix Conference on Computers and Communications, pp. 427- 433

48. P.D. Amer et al, "Local Area Broadcast Network Measurement: Traffic Characterization", Proc. Of the IEEE Spring Compcon'87, San Francisco, CA, USA, February 23- 27th, pp. 64- 70

49. Shuang Deng, "Capture Effect in Residential Ethernet LAN", IEEE Globecom'95, Vol.3, pp. 1678- 1682

50. J. Zdepski, K. Joseph, D. Raychaudhuri, "Packet Transport of VBR Interframe DCT compressed Digital Video on a CSMA/CD LAN", IEEE Globecom'89, Vol.2, pp. 886- 892

51. K. K. Ramakrishnan, H. Yang, "The Ethernet Capture Effect: Analysis and Solution", Proc. Of 19th Conference on Local Computer Networks, 1994, pp. 228-240

52. M. L. Molle, "A New Binary Logarithmic Arbitration Method for Ethernet", Computer Systems Research Institute, University of Toronto Technical Report, CSRI 298, April 1994

53. Csaba Szabo, "An Ethernet Compatible Protocol to Support Real-Time Traffic and Multimedia Applications", Computer networks and ISDN Systems 29 (1997), pp. 335- 342

54. ISO/IEC 13818-2: 1995(E) MPEG-2 Specifications"

55. B. G. Haskell, A. Puri, A. N. Netravali, "Digital Video: An Introduction to MPEG-2", Chapman & Hall, c1997

56. W. Richards Stevens, "TCP/IP Illustrated, Volume1: The Protocols", Addison Wesley, 1st Edition, c1997

57. D. E. Comer, "Internetworking with TCP/IP Volume1: Principles, Protocols and Architecture", 3rd Edition, Prentice Hall Inc., NJ 07458

58. W. Stallings, "Data and Computer Communications", 4th Edition, 1994 Prentice Hall Inc., NJ07458

59. F. C. Mills, "Statistical Methods ...", Holt, Reinhart and Winston Inc , 3rd edition

60. D. R. Cox, P. A. Lewis, "The Statistical Analysis of series of Events", London: New York: Chapman & Hall Press, 1978, c1966

61. "Vela 2000-0206 SCSI-2 MPEG-2 Decoder Installation and User Manual", Vela Research Inc.

62. "IW95000 Protocol Analyzer User Guide", GN-Nettest Inc.

APPENDICES

A. MPEG Standards[11]

Digital video is among the most demanding applications in terms of storage space and transmission bandwidth. Luckily, video data contains a great degree of redundancy, both within each frame (spatial redundancy) and between consecutive frames (temporal redundancy). Compression techniques that are used to reduce the amount of essential data for storage or transmission take advantage of these redundancies. Some compression standards such as JPEG encode each frame by itself and therefore do not provide inter-frame reduction. In 1992, the Moving Picture Experts Group (MPEG) developed the MPEG-1 standard (ISO/IEC 11172) that supports Inter-frame coding as well.

MPEG-1 was designed to produce VHS quality compressed audio/video at a bit-rate of approximately 1.5 Mbps for storing on digital media. However, bit-rates as high as 5 Mbps are achievable.

The increasing need for higher bit-rates and error resilience in transmission applications motivated the MPEG committee to introduce MPEG-2 (ISO/IEC 13818)[54] in 1994. MPEG-2 is an extension on MPEG-1 that can be used by a larger range of applications at bit-rates of 1.5-10 Mbps and at different resolutions. Later, a project to define the MPEG-3 standard was proposed for HDTV. However, this project was cancelled when MPEG-2 proved that with a few enhancements it could also support HDTV services. Nowadays, MPEG-2 is the most widely accepted standard in digital broadcast TV and HDTV. For applications such as video conferencing, which require lower bit-rates and delay bounds, an MPEG-4 standard is being developed.

The MPEG specifications include standards for compressing, multiplexing and reconstructing of video streams and consist of three parts: video, audio and system. The

[11] For a complete source on video compression and MPEG standards refer to [55].

video and audio parts deal with encoding processes while the system part of these standards guarantees the integration of the audio and video streams with proper time stamping to allow synchronization of the two. The MPEG standards do not specify any particular encoding algorithm but they indicate the format (*syntax*) in which data should be presented to a decoder and also the rules of decoding process (*semantics*). Therefore, an encoder can use any encoding process to produce a bit-stream that complies with the syntax and can be decoded according to the decoding semantics.

The system portion of the standard specifies how one or more elementary audio and video bit-streams are combined along with timing information to produce a single bit-stream suitable for storage and transmission (a *system stream*). MPEG-2 defines two types of system streams: *Program* and *Transport* streams. The MPEG-2 Program Stream (PS) is compatible with MPEG-1 and is used for relatively error-free environments. MPEG-2 Transport Stream (TS) is designed for use in error-prone environments. TS packets have a fixed length of 188 bytes while PS streams use long variable length packets.

The MPEG compression algorithms are lossy, i.e. decoded bit-streams can not completely reconstruct the original data. These algorithms exploit the human inability in distinguishing small changes in pixel values and perceiving small quantization noise in audio signals. High levels of compression are achievable through removing redundant information. The MPEG video compression scheme is based on two techniques: *Discrete Cosine Transform (DCT)* for reducing spatial redundancy and *Motion Compensation* for reducing temporal redundancy. DCT converts pixel intensities to a frequency-based equivalent, which is a series of numbers that represent every detail in a pixel block. Motion compensation is used to find the closest macro-blocks from one picture to another. A macro-block is a portion of the screen consisting of a few pixels. For instance, in the format 4:2:0 a macro-block is 16x16 pixels in the luminance space and 8x8 in the chrominance space. In temporal compression, the two pictures are subtracted from one another and the difference is then encoded as a still image.

In a video sequence, contents of close pictures are usually similar and total scene changes are rare. Temporal compression takes advantage of this fact to remove those parts of information that are common between consecutive pictures. An MPEG video stream contains three types of pictures:

- *Intra-frames (I-frames)*: Self-contained pictures that are coded as still images. These frames are encoded without reference to any other frame and thus can be used as decoding starting points. These frames are normally the largest among the 3 types because they can only be spatially compressed.

- *Predictive frames (P*-frames): Pictures that are coded with reference to a previous I-frame or P-frame and can be used as a reference point for future P-frames or B-frames. They are usually smaller than I-frames because of temporal reduction.

- *Bi-directional frames (B-frames)*: Pictures that are coded with reference to the closest two I-frames or P-frames, one encoded in the past and one encoded in the future. They can not be used as a reference point for other pictures. B-frames have the highest level of compression.

MPEG video bit-streams are organized in groups of pictures. A *Group of Pictures* (GOP) normally contains one I-frame, a few P-frames and possibly some B-frames. An MPEG sequence may look like: IBBPBBPBBPBBIBBP.... The structure of a GOP is specified by two parameters: the number of pictures in it (N) and the number of B-frames between two successive P-frames (M). Obviously, such a sequence of data results in a very bursty traffic due to the difference between picture sizes. Also, scene changes cause variation of size within a certain type of frames. Such traffic is called Variable Bit Rate (VBR). Because bursty traffics have undesirable impacts on networks, MPEG encoders can also produce Constant Bit Rate (CBR) video streams where frame sizes do not vary a lot but the picture quality could be poor. The reason for this is that, in order to maintain a fixed bit rate, pictures with different sizes need to be encoded at different *Quantization Parameters* (Q-factors).

B. Scalable Video Encoding

Another challenge for real-time applications especially video on the Internet is heterogeneity. Users of the Internet are very different in terms of decoding capabilities and connection speed. Suppose the video bit-streams are encoded at a fixed bitrate. If this bitrate is low enough, everyone can receive and decode the video sequences at a low quality in which case those with high-speed connections will not be able to take advantage of their fast connections. On the other hand, if the bit-streams are encoded at a high bitrate, people with slower connections can not see the video in real-time. Scalable video encoding is a technique that is proposed [24] to deal with this problem without having to store video bit-streams at different bitrates in order to accommodate needs of heterogeneous clients.

A scalable video encoder produces a bit-stream that can be decoded at different bitrates. Bit-streams are encoded only once but the video server decides the number of bits from the sequence to be transmitted depending on the available bitrate.

In addition to heterogeneous networks, digital TV broadcasting can also benefit from scalable codecs to achieve graceful degradation of picture quality depending on available bandwidth.

This method can very well go hand in hand with resource reservation protocols such as RSVP[26] because bit-streams can be pruned at routers in order to adapt to available bandwidth at different links.

C. Optimal Channel Rate Allocation

In this sequel, we follow the analysis in [25] but applied to a VOD system comprising scalable video facilities. The notation adopted here is consistent with chapter II.5. One of the requirements of a seamless video transmission is a constant delay for all the pictures in the sequence. This condition guarantees a jitter-free reception. Let D_{ete} denote the end-to-end delay of each frame. We can write:

$$D_{ete} = \delta t_s + \delta t_c + \delta t_r$$

where δt_s *and* δt_r are the delays in the sender and receiver buffers and δt_c is the frame transmission time. Given that the picture interval is T, the total number of video frames stored in the server and client buffers at any time is given by:

$$\Delta N = \frac{D_{ete} - \delta t_c}{T}$$

Similar to the analysis in [25] we assume that the server and the client are shifted by an amount equal to the transmission delay, δt_c. Note that this assumption is particularly true if the server is able to retrieve all of a video frame from the hard disk on time to be delivered over the channel.

The size of the buffer at the client plays a major role in the performance of the system.

$$B_r(j) = \begin{cases} \sum_{i=1}^{j} R(i) - \sum_{i=1}^{j-\Delta N} S(i) & \text{when } j \geq \Delta N \\ \sum_{i=1}^{j} R(i) & \text{when } j \leq \Delta N \end{cases} \qquad (1)$$

The client buffer occupancy can be expressed by:

In order to find the buffer size needed to prevent a buffer underflow, we must get the client buffer's occupancy in time interval (j + ΔN).

From (1), we obtain:

$$B_r(j+\Delta N) = \sum_{i=1}^{j+\Delta N} R(i) - \sum_{i=1}^{j} S(i)$$
$$= \sum_{i=j+1}^{j+\Delta N} R(i) - \left(\sum_{i=1}^{j} S(i) - \sum_{i=1}^{j} R(i) \right) \qquad (2)$$

where the term in parenthesis represents the amount of data ready at the output buffer of the server at the end of the j^{th} interval. A buffer underflow at the client side will not occur as far as the right-hand side of (2) remains greater than zero.
The conditions to prevent a buffer underflow-overflow at the client can be expressed by:

$$0 \le \sum_{i=1}^{j+\Delta N} R(i) - \sum_{i=1}^{j} S\ (i) \le B_r^{phys} \qquad (3)$$

Assume the use of scalable video encoded video streams with M different levels. Assume further P possible channel rates for each frame interval. Define the sequence of possible resolution scales by $\mathbf{x} = \{x(1), x(2), ..., x(N)\}$, where $x(i)$ is the scale index for frame i. The number of bits generated is given by $S_{x(i)}(i)$, and the associated distortion is $D_{x(i)}(i)$. Let $\mathbf{y} = \{y(1), y(2), ... y(N + \Delta N)\}$ denote the sequence of transmission rate choices where $y(i) \in \{1,...,P\}$ and the associated channel rate is $R_{y(i)}(i)$. Obviously, the latter will be constrained as well by the scheduling algorithm. In this way the sequence of $S_{x(i)}(i)$ and $D_{x(i)}(i)$ represent the rate and distortion of each frame for a given choice of $\mathbf{x}$, and $R_{y(i)}(i)$ represents the channel rates for each frame interval for a given $\mathbf{y}$.

Due to the end-to-end delay, ΔN, the client processes the $(i - \Delta N)^{th}$ frame at time i (with respect to the server). In this case, we have to properly set $R_{y(i)}(i)$ and $S_{x(i-\Delta N)}(i-\Delta N)$ to prevent the decoder buffer underflow. This is expressed by:

$$B_r(i) = B_r\ (i-1) + R_{y(i)}(i) - S_{x(i-\Delta N)}(i-\Delta N) \ge 0$$

Channel rate is also constrained by the scheduling policy and in the general case by the policing function. These two policies are implemented by monitoring and enforcing the bit rate of the (video) sources. Let *L(i)* denote the state of the monitor function and assume that the monitor function depends only on the previous step *L(i-1)* and the channel rate $R_{y(i)}(i)$, i.e.,

$$L(i) = F(R_{y(i)}(i), L(i-1))$$

The policing function decides whether to admit the data with bit rate $R_{y(i)}(i)$ into the network according to a criterion:

if $L(i) = F(R_{y(i)}(i), L(i-1)) \in L$

admit $R_{y(i)}(i)$; otherwise

reject $R_{y(i)}(i)$

In the general case, the last two expressions constrain the client buffer states as well as the monitor function transitions. In this way, we can formulate the traffic policy and data generation rate as follows [25].

Find mappings $\mathbf{x} : (1,...,N) \rightarrow (1,...M)$ and $\mathbf{y} : (1,...,N+\Delta N) \rightarrow (1,...P)$ that solve

$$min \sum_{i=1}^{N} D_{x(i)}(i)$$

subject to the constraints

$$B_r(i) \geq 0$$
$$L(i) \in L \qquad \forall i = 1,...N+\Delta N$$

In this way, the channel rates and the encoder bit rates are allocated jointly to avoid decoder buffer underflow and meet the last two constraints.