,

To those whose wisdom I wish to carry with me forever:
my parents, my sister, and my husband

ACKNOWLEDGEMENTS

I would like to take this opportunity to thank my parents, Qinglan Xu and Bingheng Wang, although no amount of thanks could sufficiently repay their unfailing love, support, and patience. They instilled in me a desire to know, and undertook great sacrifices to allow me to do so.

Professor Benjamin Wah has taught me more than anyone what diligent work means. I would like to thank him for his guidance, encouragement and valuable advice in this work. I have learned much from working with him, and hope to continue to do so in the future.

I would like to thank professor David Goldberg for his insight, intelligence, and knowledge of the Zone. His support and encouragement made this thesis possible. Our weekly meetings have always been a continuing source of guidance.

In my time in the research group, I have met many outstanding people. I thank them for their assistance, humor, and endless arguments (you know who you are ...).

Special thanks go to my husband for his love and care. I thank my sister, Lixiao Wang, for showing me what is truly important.

Support for this work was provided by NSF grant 96-32316 and NASA contract NAG 1-613.

Contents

Chapter 1

Introduction

1.1 Introduction to the Internet and Video on Demand

The Internet was begun about 20 years ago as a U.S. Defense Department network called ARPAnet. It was originally an experimental network designed to support military research—in particular, research about how to build networks that could withstand partial outages and still function. The network itself is assumed to be unreliable, as any portion of it could fail at any moment. It was designed to require a minimal amount of information from the computer clients. To send a message in the network, a computer only had to put its data into an envelope, called an Internet Protocol (IP) packet, and "address" the packet correctly. The communicating computers—not the network itself—were also given the responsibility to ensure that the communication was accomplished. The philosophy was that every computer on the network could talk, as a peer, with any other computer.

Early stages of Internet applications involved mostly reliable file transfer, like HTTP, FTP, SMTP, etc., and the TCP/IP suite was designed and adopted to tailor these needs. With the fast growth of multimedia applications and increasing bandwidth, Multimedia on Demand has become an attractive network application. VCR-like Video on Demand is one such application. Video on Demand can be defined as the generic case of an end-user-initiated reception of remotely stored video for viewing within an acceptable short time scale, where the available choice of video material is very large. Many protocols and commercial systems have been implemented. Early systems for the delivery of real-time video and audio on the Internet, such as the CM Player required high-end workstations and high-speed Ethernet or T1 data rates. Tenet [1] requires ISDN service that supports admission control and resource reservation. Commercial products also demand extra hardware to support playback speed. These systems concentrate on transmitting high-quality (and consequently high-bandwidth) video and audio over the Internet. RTP is the first standard real-time protocol which transmits directly over the Internet, without putting additional requirements on the underlying network. Then comes VDP in Vosaic (Video Vosaic), which is completely dedicated to transmitting real-time multimedia over the Internet.

In the next section, we identify the major problems involved in transferring real-time multimedia data over the Internet and propose our solution.

1.2 Problem Definition

Multimedia information systems introduce new challenges, due to their new and unique characteristics [2]. In traditional data communications, layered protocol structures are often employed to deliver data correctly from a source to an intended destination. Error detection and recovery, flow control, packetization, multiplexing, connection control, internetworking, and other mechanisms are used to ensure the proper delivery of information. Time, however, is not considered to be an important issue in regular data communication protocols.

Time represents a major concern in multimedia delivery systems. Unlike data, multimedia information is usually composed of multiple streams, each representing a specific type of media: data, video, audio, etc. These streams have to be synchronized when they are presented to the final user application that displays (or renders) them to users. Consequently, the notions of time and synchronization have to be included somehow in the protocol suites used for multimedia applications.

Another characteristic of multimedia communication systems is their asymmetry. Typically, short requests and control commands flow from a user application to a digital-storage-media server while a very large amount of information flows downstream to the user. For example, an MPEG-2 video file is typically transmitted at 3 M bits per second if it is to be displayed in real time, whereas control information seldom exceeds a few hundred bits per second.

The current Internet transport layer protocol, TCP, is believed to be unsuitable for real-time multimedia data transfer for several reasons. First of all, TCP imposes its own flow control and windowing schemes on the data stream. These mechanisms effectively destroy the temporal relations shared between video frames and audio packets. Secondly, reliable message delivery is not required for video and audio streams which can tolerate frame losses. Losses are seldom fatal, although they are detrimental to picture and sound quality. TCP retransmissions cause further jitter and skew internally between frames and externally between associated video and audio streams. Moreover, resource reservation, an effective approach to guarantee bandwidth, jitter, or delay on ISDN/ATM, is not possible on the Internet. The delivery of IP packets across the Internet is typically best-effort and subject to network variability outside the control of any video server or client. Because of the high heterogeneity and large geographic areas covered by the Internet, interframe jitter can be high, which adversely affects the playback quality. In addition, network congestion may cause frame delays or losses.

These claims have been made by many in the field. The purpose of this thesis is to examine experimentally the feasibility of using TCP for real-time multimedia traffic. We first analyze the characteristics of an MPEG stream and its bandwidth requirement. Then we study the quality of service over TCP in terms of delay, jitter, and throughput. By comparing real-time multimedia traffic requirement and the QoS provided by TCP, we believe that TCP is unable to support real-time multimedia traffic. Experimental setup are presented and results recorded in Chapter 3.

1.3 Roadmap of this Thesis

This chapter presents an introduction to the Internet and the problem of Video on Demand. This exposition leads to the problem definition, followed by our proposed solution.

Chapter 2 investigates the characteristics of real-time multimedia traffic, Then prior claims that TCP is unsuitable for real-time transmission are presented. The chapter concludes with an introduction to Video on Demand and existing VoD prototypes and implementations.

Chapter 3 presents the experimental environment and results. MPEG streams are examined in terms of bandwidth requirements. Experimental results on TCP Quality of Service are presented. By comparing real-time multimedia traffic requirement and the QoS provided by TCP, we believe that TCP is able to support real-time multimedia traffic.

Chapter 4 presents a summary of this thesis.

1.4 Summary

This chapter began with an exploration of the context within which the work in this thesis is done. We define the problem we are trying to solve and briefly cover the reasons why the work in this thesis is important. We then present the methodology we use in this thesis. Finally, we conclude with a road map for the rest of this thesis.

Chapter 2

TCP and Multimedia on Demand

There has recently been a flood of interest in real-time multimedia distribution over the Internet from both industry and academia.

The purpose of this chapter is to examine previous work done in real-time Multimedia on Demand over the Internet and proposed solutions. This review is divided into three sections:

- An investigation of the characteristics of delay-sensitive VBR traffic. TCP is analyzed in the context of real-time multimedia transfer.
- A review of previous efforts to incorporate real-time traffic into Internet framework, namely RTP, Tenet, and VDP.
- An introduction to Video on Demand and existing VoD prototypes and implementations.

The following section begins by exploring the issues associated with transferring real-time multimedia traffic over the Internet.

2.1 Delay-Sensitive VBR Traffic and TCP

This section identifies the characteristics of delay-sensitive VBR traffic. The purpose of this section is to identify the problems and motivate actions for this thesis.

A multimedia information system introduces new challenges due to its new and unique characteristics [2]. In traditional data communications, layered protocol structures are often employed to deliver data correctly from a source to an intended destination. Error detection and recovery, flow control, packetization, multiplexing, connection control, internetworking, and other mechanisms are used to ensure the proper delivery of information. Time, however, is not considered to be an important issue in regular data communication protocols.

Time represents a major concern in multimedia delivery systems. Unlike data, multimedia information is usually composed of multiple streams, each representing a specific type of media: data, video, audio, and so forth. These streams have to be synchronized when they are presented to the final user application that displays (or renders) them to the user. Consequently, the notions of time and synchronization have to be included somehow in the protocol suites used for multimedia applications. Moreover, the different media streams are clearly very different in terms of their sensitivity to errors, delay and jitter, bandwidth needs, etc. This suggests that one protocol suite may not satisfy all types of streams.

Another characteristic of multimedia communication systems is their asymmetry. Typically, short requests and control commands flow from the user application to the digital storage media server, while a very large amount of information flows downstream to the user. For example, an MPEG-2 video file is typically transmitted at several hundred of Megabytes per second if it is to be displayed in real time, whereas control information seldom exceeds a few hundreds of bits per second. In addition to asymmetry in bandwidth, upstream and downstream data are often made of different types of data multiplexed together. For example, in an MPEG-2 transport stream flowing from a server to a client, commands and responses should normally be transported using reliable data protocols, whereas timeliness is far more important than error recovery in the video stream delivery.

TCP (Transport Control Protocol) is the most commonly used transport-layer protocol for reliable transmission over the Internet. TCP is the underlying protocol for many popular Internet applications, such as session-layer protocols like FTP, HTTP, and SMTP. TCP is believed to be unsuitable for real-time multimedia data transfer for several reasons. First of all, TCP imposes its own flow control and windowing schemes on the data stream. These mechanisms effectively destroy the temporal relations shared between video frames and audio packets. Multimedia application was just over the horizon when TCP was designed and became the Internet standard. TCP was tailored to traditional data transfer, like file transfer, e-mail, and HTTP. Reliability is the overwhelming issue for these applications, embodied in every aspect of TCP algorithms. Second, reliable message delivery is not required for video and audio transmission. Video and audio streams tolerate frame losses. Losses are seldom fatal, although they are detrimental to picture and sound quality. TCP retransmission causes further jitter and skew internally between frames and externally between associated video and audio streams. For example, TCP employs slow start, fast retransmit, fast recovery, and congestion control to ensure reliability. All of these are good flow-control mechanisms for traditional file transfers, but operate at the price of sacrificing the real-time constraint, which is of utmost importance to Multimedia on Demand applications. Reliability and timeliness are two conflicting objectives in data transfers, and every transport layer protocol needs to strike a balance by choosing an appropriate trade-off. TCP chooses reliability over timeliness, rendering itself unsuitable for real-time traffic.

Despite the above observations, the purpose of this thesis is to study experimentally the suitability of TCP for VBR traffic.

2.2 Real-time Transport Layer Protocols

Because some features of TCP are conflicting with real-time multimedia applications over the Internet, new transport layer protocols were designed to cater to the special needs of real-time applications. This section briefly reviews a couple of influential studies that have shaped the way real-time traffic is currently handled, and also its relevance to this thesis. We start from RTP, real-time TCP, an Internet standard. Then we introduce the Tenet suite, which provides a statistical and deterministic guarantee of quality of service. Lastly, we study VDP, the datagram protocol in Vosaic (Video Mosaic).

RTP is the Internet-standard protocol for the transport of real-time data, including audio and video. It can be used for Media on Demand as well as interactive services such as Internet telephony. On November 22, 1995, RTP was approved by IESG as an Internet proposed standard.

RTP consists of a data part and a control part. The latter is called RTCP. The data part of RTP is a thin protocol providing support for applications with real-time properties, such as continuous media (e.g., audio and video), including timing reconstruction, loss detection, security, and content identification.

Named "real-time" protocol, RTP does not ensure real-time delivery, and the authors claimed that in-time delivery requires the support of lower layers that actually have control over resources in switches and routers. RTP provides functionality suited for carrying real-time content, e.g., a timestamp and control mechanisms for synchronizing different streams with timing properties. RTP does not define any mechanisms for recovering from packet loss.

RTP is receiving more attention from industries. Intel, Microsoft, and a consortium of over 100 technology vendors vowed to build an open platform based on existing standards "to make video, voice, and data communications over the Internet as commonplace as a simple telephone call." The International Multimedia Teleconferencing Consortium (IMTC), the group pushing for an open Internet communications platform, said that its implementation will be based on International Telecommunications Union (ITU) standards and Internet Engineering Task Force (IETF) specifications, including T.120 (for data conferencing), H.323 (for audio and videoconferencing), and the RTP/RTCP and RSVP specifications. Microsoft said that it will include these communication capabilities as part of its ActiveX Technologies in future releases of the Windows operating system and associated developer kits.

RTP was designed to be a multicast protocol, and many RTP algorithms are multicast-specific. RTP is proven to be effective for simple multicast audio conference, audio and video conference, and mixers and translators. RTP does not provide adaptation mechanisms to guarantee statistic or deterministic quality of service, which is the focus of this thesis.

Tenet real-time protocol suite is a set of network protocols providing guaranteed performance communication in heterogenous packet-switching internetworks [3].

The Tenet protocol suite is the first complete set of communication protocols that can transfer real-time streams with guaranteed quality in packet-switching internetworks. It runs on a packet-switching internetwork, and can coexist with the popular Internet Suite, relying on the use of connection-oriented communication, admission control, and channel rate control.

The Tenet protocol suite consists of various protocols designated for different purposes [4]. Real-time Channel Administration Protocol (RCAP) sets up and tears down real-time channels, supports status inquiries about established channels, and administers the channels. RCAP is designed to control real-time channels in a heterogenous internetwork. In such an environment, different networks may have different scheduling policies or different models, which will require different admission control tests. This heterogeneity requires different parameters to be passed between nodes during channel establishment. RCAP takes a hierarchical view of channel establishment in which the links and nodes in a subnetwork are abstracted into a single “logical link” in the internetwork. Such an approach allows RCAP to utilize the characteristics peculiar to an individual network in order to provide guarantees, yet hide the underlying details of that network whenever possible [5].

Real-Time Internet Protocol (RTIP) is the network layer of the Tenet suite [6, 7]. Its main function is to deliver packets in such a way as to meet the real-time requirements of the corresponding channels. RMTP provides a message-based abstraction on top of RTIP. An instance of RMTP is needed at both endpoints of a real-time channel. RMTP was designed as a light-weight transport protocol, and resides at the same layer in the protocol stack as TCP.

The Tenet protocol suite contains many good algorithms for admission control, channel control, and resource renegotiation. All of those, though, are based on the presumption that the underlying network supports bandwidth reservation, which is obviously not true over the Internet.

Vosaic is an ongoing effort to integrate video and audio into the World Wide Web. The authors claimed that full file transfer and TCP are unsuitable for continuous media, such as real-time audio and video. In order for the WWW to support continuous media, the authors extended the architecture of the WWW to encompass the dynamic, real-time information space of video and audio. Vosaic, shorthand for Video Mosaic, incorporates real-time video and audio into standard hypertext pages that are displayed in place. Video and audio transfers occur in real time, with no file retrieval latency. Real-time video and audio data can be effectively served over the present-day Internet with the proper transmission protocol. A real-time protocol VDP was developed to handle real-time video over WWW. VDP minimizes interframe jitter and dynamically adapts to the client CPU's load and network congestion. The authors claim their work enables a *video-enhanced Web*.

2.3 Video on Demand

There has recently been a flood of interest in Video on Demand over the Internet from both industry and academia. In this thesis we implement a Video on Demand project as a testbed for a real-time, self-adaptive algorithm we have designed. This section reviews the background of the developing technologies and services associated with Video on Demand. We begin by fine-tuning the VoD concept, followed by an introduction to existing VoD prototypes and implementations.

There are three related concepts in the literature: Video on Demand (VOD), Pure Video on Demand (PVOD), and Near Video on Demand (NVOD). These terms are often interchangeable in the literature, and there seems to be little consensus on their definitions. In this thesis, we adopt Cleary's definition [8]. We define Video on Demand to be the generic case of the end-user-initiated reception of remotely stored video for viewing within an acceptable short time scale, where the available choice of video material is very large. The ambiguity involved in the notion of an "acceptably short" delay in receiving the film material and the meaning of a "very large" selection of video material gives rise to the further categories of PVOD and NVOD. Obviously, in an ideal case, there would be an infinite number of videos to choose from, and the reception of a desired film would be instantaneous. Pure Video on Demand is defined in such a way that it pertains to that family of Video on Demand technologies and services that approach the ideal case. Conversely, Near Video on Demand is defined such that it pertains to that family of Video on Demand technologies and services that do not tend to the ideal case, but rather involve an inevitable and predictable delay between the act of choosing from a limited selection of videos and the ability to view one's choice. In the sense given above, PVoD is a more personalized service than NVoD and aims to empower the user to view "what he wants, when he wants to." Moreover, there are further hierarchies within the category of PVoD, with VCR-like VOD sitting on the top. In this thesis we use video on demand to refer to the most generic term: VOD.

Video on Demand services present new requirements on the underlying network. One significant driver for the consumer use of high bandwidth in the near future will be interactive video on demand. A range of service types can be deployed, based on different amounts of sophistication, that must be traded against the network costs and component costs. The potential aggregate bandwidth required is huge, hence it is essential to properly engineer the network to reduce the bandwidth required. A cost function was introduced to capture the combined bandwidth and storage requirement of the network [9]. This cost function can be used by network designers to determine optimal topology, sharing and caching strategies for the desired bandwidth versus memory costs in a particular network deployment. The overall contribution of that work is that it evaluated the effects of various server, cache, and sharing strategies on the bandwidth and storage requirements of the network and their proper placement within the network. In this thesis, we measure network requirements, available bandwidth, and storage requirements by a simplified version of that cost function, which is presented in Chapter 3.

A number of protocols have been proposed, tailored to Video on Demand, over the past few years. Many industrial companies also turned out several prototypes and implementations to provide Multimedia on Demand service. This section briefly reviews several relevant Video on Demand protocols and commercial implementations.

Timed Token Protocols were studied and modified with respect to real-time packet traffic in local area networks [10], such as FDDI and token bus, employed in distributed control systems. A "selected access scheme" was proposed for renegotiating continuous media delivery from a Video on Demand server. Caching was introduced into the video server to better satisfy stringent real-time constraints.

A number of industrial giants have turned out their VoD products. Oracle's solution is to deliver its own media server software via massively parallel computers built by n-cube. IBM has developed two video servers, currently undergoing trial. One is based on an IBM RS/6000 UNIX mid-range computer, and is capable of delivering 40 MPEG streams of compressed video. The other, based around an ES/9000 mainframe, has a somewhat higher capacity. Microsoft is developing video server software called Tiger, which runs on a Microsoft NT server and consists of a filing system that stores video, graphics, and audio in digital form across an array of disk drives. Tiger is due to be trialed by Microsoft and TCI, a US cable company. Compaq, which manufactures PC network servers, will develop media server products for the Tiger system.

The group that comes closest to achieving commercial VoD is Tele-Communication Inc., US West and AT&T who formed a consortium in 1991 to test VoD in a realistic market environment using a field test with the name of Viewer Controlled Cable Television (VCTV). Their take-one service makes a catalogue of at least 2000 movies available for immediate viewing, any of which can be ordered using the VCTV remote control device. A subscriber can interrupt the movie being viewed up to five times, for a total of ten minutes. In addition, a subscriber can reserve a movie up to one week in advance.

2.4 Summary

We began this chapter by identifying the characteristics of real-time multimedia traffic. TCP quality of service is examined in the context of delay-sensitive VBR traffic. Finally, we reviewed previous works that attempt to incorporate real-time multimedia applications into the Internet, either as a new transport layer protocol or as an industrial implementation effort.

Chapter 3

Experimental Results

This chapter studies experimentally the compatibility of MPEG streams and TCP Quality of Service. MPEG streams are examined in terms of bandwidth requirement and burstiness. The TCP QoS is investigated in terms of throughput, RTT and jitter. For each set of experiment, setup details are presented, followed by results and analysis. Based on experimental results in this chapter, a conclusion is reached at the end of the chapter that TCP is compatible with MPEG stream.

3.1 MPEG Bandwidth Requirement

MPEG is a family of compression algorithms. The most significant feature of MPEG is that it uses inter-frame dependent encoding. For example, a sequence of MPEG video frames has I, P, and B frames, where I frames have intraframe data coded with JPEG compression, P frames are predictively coded with respect to a past picture, and B frames are bidirectionally predictively coded. MPEG frames are arranged into groups with a sequence that corresponds to the pattern I B B P B B P B B. The I frame is needed by all P and B frames in order to be decoded. The P frames are needed by all B frames. This encoding method makes some frames more important than others. The display quality is strongly dependent on the receipt of important frames. Since data transmission is unreliable over UDP, there exists a possibility of frame loss. If an I frame is lost in a sequence group of MPEG video frames, I B B P B B P B B recorded at 9 frames/second, the entire sequence becomes undecodable. This results in a one-second gap in the video stream.

Another feature brought about by temporal prediction is the fluctuation and periodicy of an MPEG stream. The I frame is the largest frame in an MPEG stream, since it is coded independently using the JPEG scheme. P and B frames are much smaller, due to the fact that they are coded based on their difference from the last I frame. Hence, the MPEG-stream-bandwidth requirement is at the rate of P/B frames most of the time, with periodical bursts caused by I frames.

In this thesis, we measure MPEG bandwidth requirement using two different approaches. RealMagic MPEG decompression card is utilized in the first method. Related features of RealMagic card is introduced, then measuring method and algorithms are presented, followed by results on six movies. Then dozens of MPEG movie clips from the WWW are examined in terms of frame rate, frame size, stream size and playback time. By doing so, we hope to capture the bandwidth requirement of MPEG of different specifications.

3.1.1 RealMagic MPEG decompression card

The RealMagic Maxima Pro card is used for MPEG bandwidth measurement in this thesis. Several related features of the RealMagic card are presented in this section. These features greatly effect implementation, performance, and evaluation.

The RealMagic Maxima Pro card is an MPEG-1 stream decompression card. The basic stream data can be provided in two ways in the card: either from a file, where the driver directly processes the file and the buffer; or from buffers, where the calling application provides the stream data. The buffer play mode greatly facilitate the measurement of MPEG bandwidth requirement. In

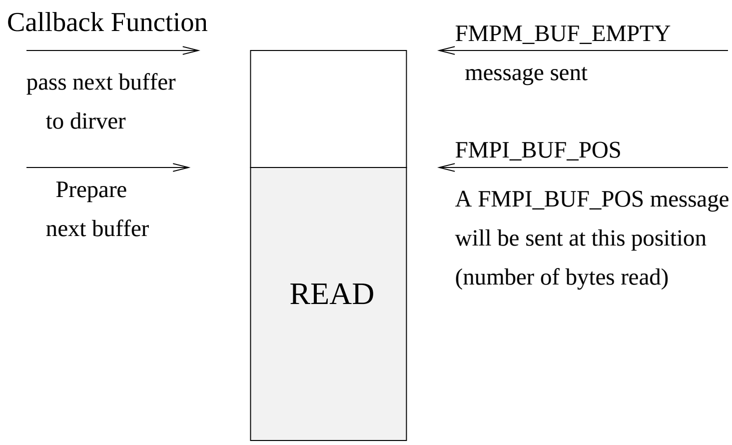


Figure 3.1: RealMagic buffer configuration and buffer-related messages.

buffered streams, applications are responsible for providing valid data. A buffered stream will use a buffer in an application working space, and no copy will be made. Hence, an overwrite will be disastrous. The RealMagic buffer and related messages are shown in Figure 3.1. A user-defined callback function is associated with the buffer. When the driver reads past a specific position in the buffer, a corresponding message will be sent to that callback function. With the messages defined, buffers can be managed inside an application without external initialization, destruction, or prefill. When the callback function receives FMPI_BUF_POS, it prepares the next buffer. When it receives FMPM_BUF_EMPTY, it passes the address and the size of the next buffer to the driver for playback.

One advantage of the RealMagic API is that it provides the application with accurate rate control information. By counting on the number of FMPI_BUF_POS messages, we keep close track of the number of packets that have been consumed. This facilitates bandwidth measurement. Moreover, with the RealMagic card, MPEG decompression does not take up CPU time, which spares the CPU for measurement-related data-collecting activities.

Table 3.1: Specifications of the six MPEG streams measured in this section

Stream name	content	stream size(KB)	playback time(seconds)	average rate(KB/S)
Bounty hunter	Action	20003	91	220
RM logo	Logo	1351	6.08	222
Defensive Driving	training	22475	101	222
Horde	legend	33351	197	169
Elvis	music (MTV)	8285	45.24	183
Razor ad	commercials	5054	29.87	170

3.1.2 Experimental setup and results

To measure MPEG bandwidth, we timestamp every FMPM_BUF_EMPTY message. Since FMPM_BUF_EMPTY is sent by the RealMagic driver when the driver runs out of data, the time elapsed between two consecutive FMPM_BUF_EMPTY messages, denoted as δt, is a rough estimation of the time during which the last buffer is played. Buffer size, denoted as S_b, is initialized as a constant before playback starts, so the bandwidth requirement in the past δtperiod can be estimated to be $\frac{S_b}{\delta t}$. The psuedocode for the algorithms is presented in Figure 3.2. The output from the algorithms is a list of 2-tuples, each corresponding to one point in the resulting graph. The experimental result of six MPEG-1 streams are presented in Figure 3.3. Figure 3.2 also shows the procedure to generate the graph in Figure 3.3. The features of these six streams are listed in Table 3.1.

All these six clips are $600 * 800$ frame size and plays 24 frames every second. The playback rates range from 169 KBytes/Sec. to 222 KBytes/Sec. Bounty hunter is an action movie clip, which features quickly-changing scenes and fast-moving objects, thus the high bandwidth requirement, much higher than the rest five. RealMagic Logo is a commercial for RealMagic company. It does not involve much action. Each scene are relatively simple. Horde is a PG movie, depicting a dinner scene in a castle. Defensive driving is training course, moving back and forth between instructor

```
//MPEG bandwidth measurement algorithms

void MPEG-bandwidth-measurement()
{
  timestamp old_stamp, new_stamp;

  switch(call-back-message)

     CASE FMPM_BUF_INIT:  // initialize buffer
     CASE FMPM_BUF_POS:   // prepare next buffer
     CASE FMPM_BUF_CLOSE: // clear up
       .
       .
       .

     CASE FMPM_BUF_EMPTY:

       new_stamp = gettimeofday();
       bandwidth = buffer_size / (new_stamp - old_stamp);
       writetofile(new_stamp - old_stamp, bandwidth);
       old_stamp = new_stamp;

       // pass new buffer to the driver
           .
           .
           .
}

void graph-generation()
{
  int time_elapsed = 0;
  while(!End_Of_File)
  {
    readfile(time_stamp,bandwidth);
    time_elapsed += time_stamp;
    plot_a_point_at(time_elapsed,bandwidth);
  }
}

}
```

Figure 3.2: The code for MPEG bandwidth measurement using RealMagic card. In our experiment, buffer_size is initialized to be 32KBytes, and stays constant throughout the experiment.

Table 3.2: Specifications of the MPEG streams collected from the Web.

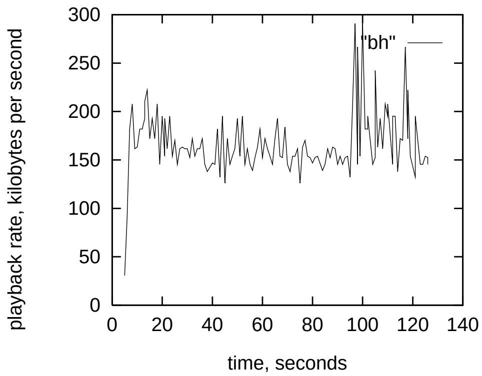
Stream Name	Frame size (pixel*pixel)	Frame rate (frame/second)	Stream size (KBytes)	Playback time (seconds)	Average rate (KBytes/second)
Organized Sheep	100*150	23	411	18	31
Opening Sequence	100 * 250	23.1	1062	24	44.25
Flying ...	100 * 250	21.3	1219	23.5	51.8
First Death	200 * 300	10.5	3140	50	62
AI smashes ..	100 * 150	21.2	43.7	8.5	52
Raw Movie	600 * 800	5.3	244	9.2	26.5
Morphid	600 * 800	4.7	668	29.1	22.8
Comparison	600 * 800	5.3	668	25.8	25.8
Charlie George	100 * 150	13.4	155	3	52
Alan Smith	100 * 150	12.7	422	6.5	58
Alan Sutherland	100 * 150	12.5	324	6	55
Jellyfish	100 * 150	21.2	402	8	50
Trigger fish	100 * 150	21.6	265	9	29

and board. Elvis shows four violinist play “The Juliet Letter” in a well-lit room. Razor ad contains 18 different scenes in which men from different walks of life and different ages chat about the razor.

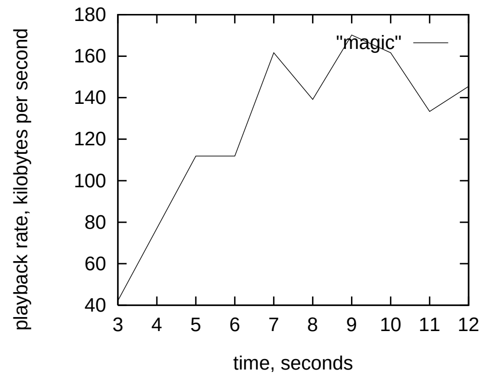
3.1.3 More MPEG streams from the Web

This section presents the specification of 13 MPEG movie clips collected from the Web and played by Berkeley MPEG-1 software decoder. Table 3.2 lists the specifications of those MPEG streams.

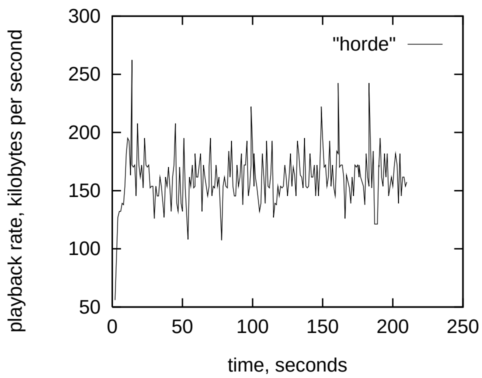
The first five are various movie clips taken from “http: //wwwzenger. informatik. tu-muenchen. de/ persons/ paula/ mpeg/ index.html”, where “Organized sheep” from “A Close Shave”, “Opening Sequence” and “Flying ...” from “Blade Runner”, “First Death” from “Highlander”, and “AI smashes ..” from “Married with Children”. Raw movie, Morphid movie and Comparison movie are located at “http: //rsd.gsfc.nasa.gov/ movies/”, they are animations of explosive convection in Hurricane Guillermo. “Charlie George”, “Alan Smith” and “Alan Sutherland” are snapshots of football games that can be found at “http: //www.netlink.co.uk/ users/ arseweb/ multim/



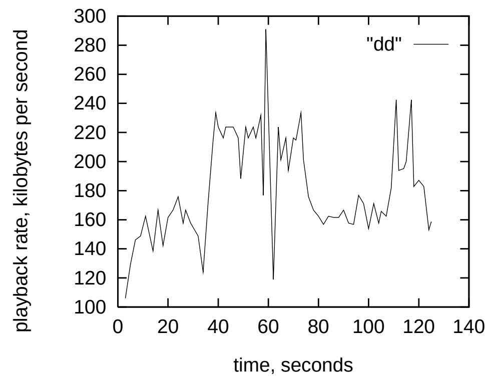
(a) Bounty Hunter



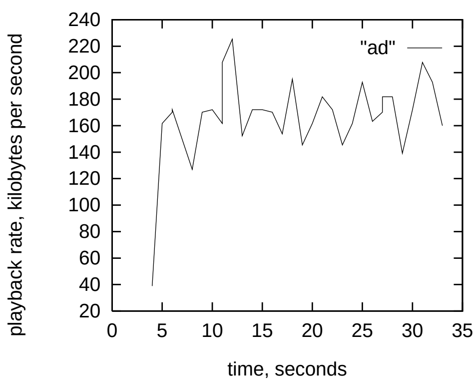
(b) RealMagic Logo



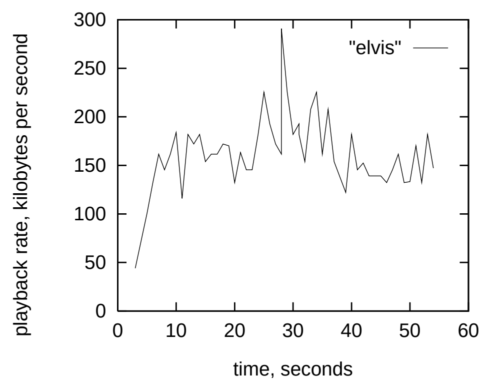
(c) Horde



(d) Defensive Driving



(e) Razor advertisement



(f) Elvis music

Figure 3.3: MPEG streams of six movies. Results were obtained by RealMagic card in buffer play mode. Buffer size were fixed at 32KBytes throughput the experiment to ensure smooth playback.

mpegs.html". "Jellyfish" and "Trigger fish" depicts the fish swimming in a tank at "http: //uwalpha.uwinnipeg.ca: 8001/ mov.html".

These 13 movies have bandwidth requirement ranging from 22.8 KBytes/Sec. to 62 KBytes/Sec.. Compare with the six MPEG streams in last section, movies in this section have a much smaller playback rate. This is achieved by smaller frame size, slower frame rate and lower accuracy. Even lower bandwidth requiment can be achieved if picture quality can be further sacrificed.

3.2 TCP QoS

Now that we have examined the bandwidth requirement for MPEG streams with different specifications, TCP Quality of Service is investigated in this section to find out the compatibility between MPEG and TCP.

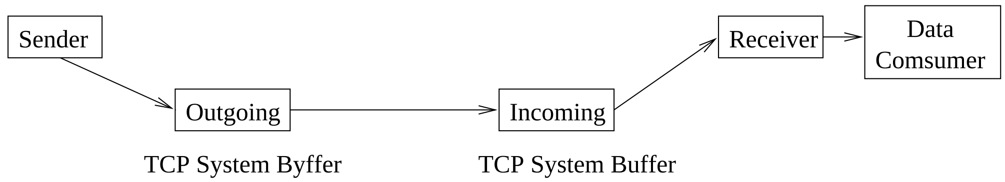
3.2.1 Experimental setup

We investigate TCP's quality of service in terms of its throughput, delay, and jitter. Five remote hosts were chosen in an effort to cover a large geographical area. Packets of fixed size at 1 KBytes were sent to the echo ports of these computers at various rates. 1 KBytes was chosen due to the fact that most echo ports we encountered echo only the first 1KBytes of the packages, regardless of the package size that was sent to the echo ports.

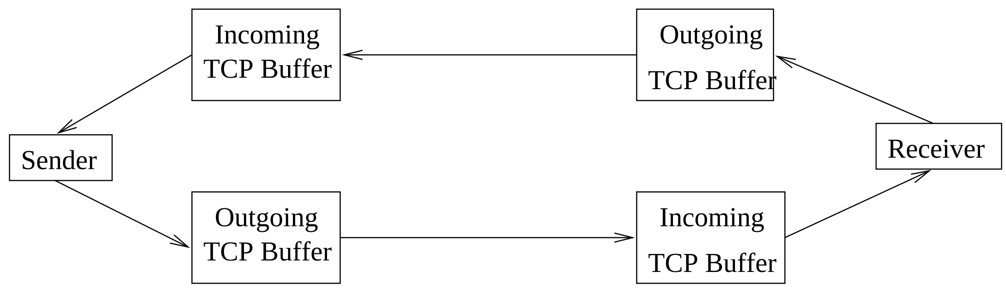
To measure throughput, packages are sent out as quickly as possible. The next package is sent out as soon as the last "send" function call returns. Two timestamps are recorded for every package: a sending time recording the time when the "send" function returns, and a receive time recording the time when the last byte of the package is received at the receiver. Whne computing RTT, corresponding timestamp from the two files are taken out and a difference computed. Since TCP deliver package in sequence, this time difference is the take elapsed after the package is copied into

```
// measure TCP Quality of Service, namely, throughput RTT and jitter

void measure_TCP_QoS()
{
  sockfd = open_echo_port(remote_host_name);
  if (fork() == 0)
  {
    while (!done)
       receive_package(sockfd, p)
       receiving_time = gettimeofday()
       write_to_recv_file(receiving_time)
  }
  else
  {
    while(!done)
       send_package(sockfd, p)
       sending_time = gettimeofday()
       write_to_send_file(sending_time)
  }
}

void obtain_RTT_throughput_jitter
{
  fpread = open(send_time_file)
  fpwrite = open(recv_time_file)
  for (each timestamp in fpread, R, and fpwrite, W)
     write_to_rtt_file(R - W)
  RTT = average(all records in RTT file);
  throughput = (all the packets sent)/
               (last timestamp in fpwrite - first record in fpwrite)
}
```

Figure 3.4: The code for measuring TCP Quality of Service over the Internet. Throughout the experiment *packet_size* is 512 Bytes. Throughput,RTT and jitter are estimated in this algorithms.



(a) Data transfer in a VoD session.



(b) Data transfer to and from echo port of a remote host.

Figure 3.5: Difference between Vod session data transfer and echo port of a remote host. This difference may result in lower estimated throughput than actual TCP throughput.

system buffer at the sender side and before the last byte of the package is handed off to the receiver application. This is considered to be one estimation of RTT. An average RTT are computed over all the packages sent and received. Throughput are obtained by dividing the sum of all the packages sent by the time elapsed between the first "send" function and the last "send" function, i.e., the time elapsed until the last byte of the last package is in system buffer. Jitter is calculated as the standard deviation of RTT. The psuedocode for the measuring method is illustrated in Figure 3.4. This process goes on until estimation stabilizes and converges. Experiment was carried out every hour in a 24-hour day and 24 estimations at different time of day are collected.

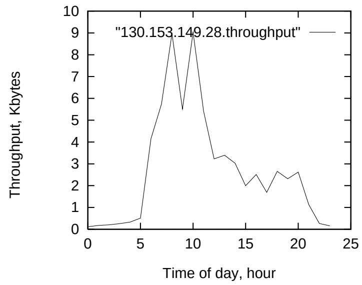
This method should accurately capture the RTT, but the estimated throughput might be slightly lower than actual TCP throughput due to reasons listed below. Figure 3.5 illustrated the difference between an actual VoD data transfer session and the estimation algorithms. In a real-world VoD data transfer, receiver reads data from TCP system buffer, hands the data off to the player, then comes right back to read the next package. whereas in most echo port server, after the receiver reads a package from TCP system buffer, it doesn't come back until this package is sent out and echoed back to the sender. This "send" will have to wait for the TCP outgoing system buffer, which might not be immediately available. On average, the TCP incoming system buffer might not clear up as quickly as in a VoD session. Therefore, the throughout measure using the algorithms above might be lower than the actual TCP throughput. Another limitation of this method is the fixed 1KB package size. Results might be slightly different if larger package size is feasible when sending to echo ports of remote hosts.

3.2.2 Experimental results and analysis

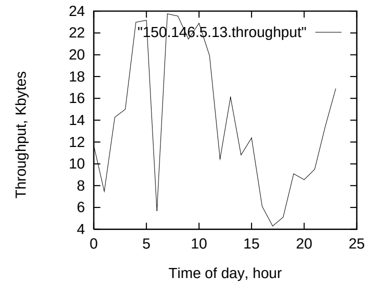
Experimental results on five remote hosts are presented in Figure 3.6, 3.7, and 3.8. In this set of experiment, 2000 packets were sent to collect data. For domstic sites, sender took around 5 minutes to complete sending, whereas internaltional sites require 10 to 20 minutes to finish. The two domestic sites can achieve peak throughput at around 120 KBytes per second, which can accommodate most MPEG streams presented in the last section. Two Asian sites suffered lowest throughput at only several KBytes per second, which cannot possibally sustain many MPEG streams. But peak throughput at these two sites are large enough to sustain low picture quality MPEG stream with bandwidth requirement lower than 10 KBytes/Sec.. On average, TCP throughput measured at those five sites are large enough to sustain small MPEG streams presented in the last section.

3.3 Summary

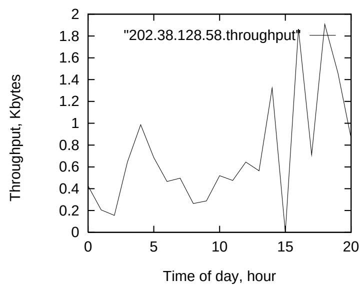
This chapter studies experimentally the compatibility of MPEG streams and TCP Quality of Service. MPEG streams are examined in terms of bandwidth requirement and burstiness. The



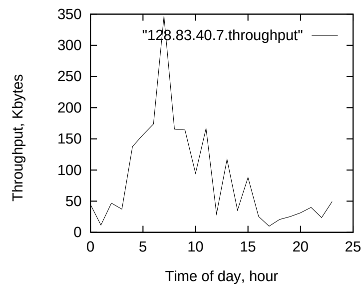
(a) *faraday.ee.uec.ac.jp*



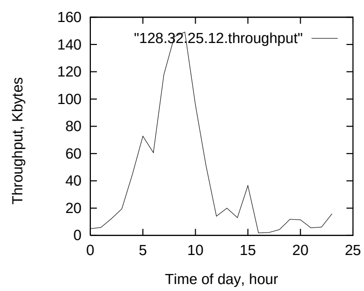
(b) *iasi.rm.cnr.it*



(c) *pub.zjptb.net.cn*

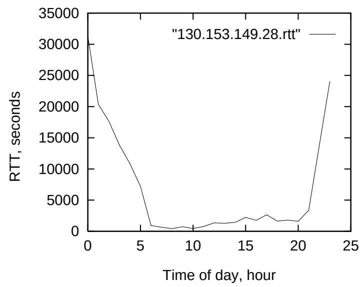


(d) *www.utexas.edu*

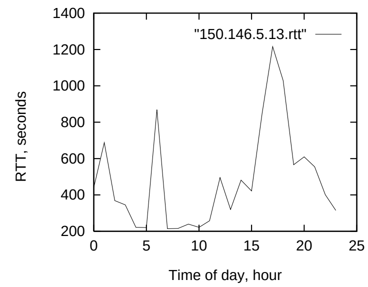


(e) *www.berkeley.edu*

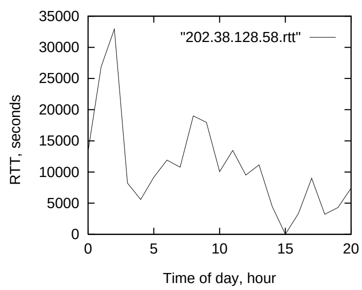
Figure 3.6: TCP throughputs for five remote hosts. These results were obtained by the algorithm presented in Figure 3.4.



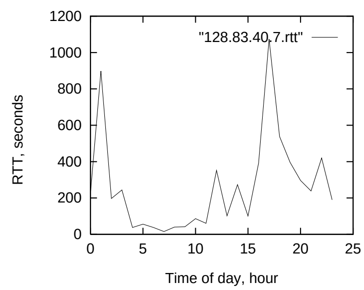
(a) *faraday.ee.uec.ac.jp*



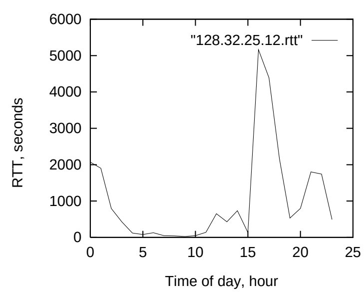
(b) *iasi.rm.cnr.it*



(c) *pub.zjptb.net.cn*

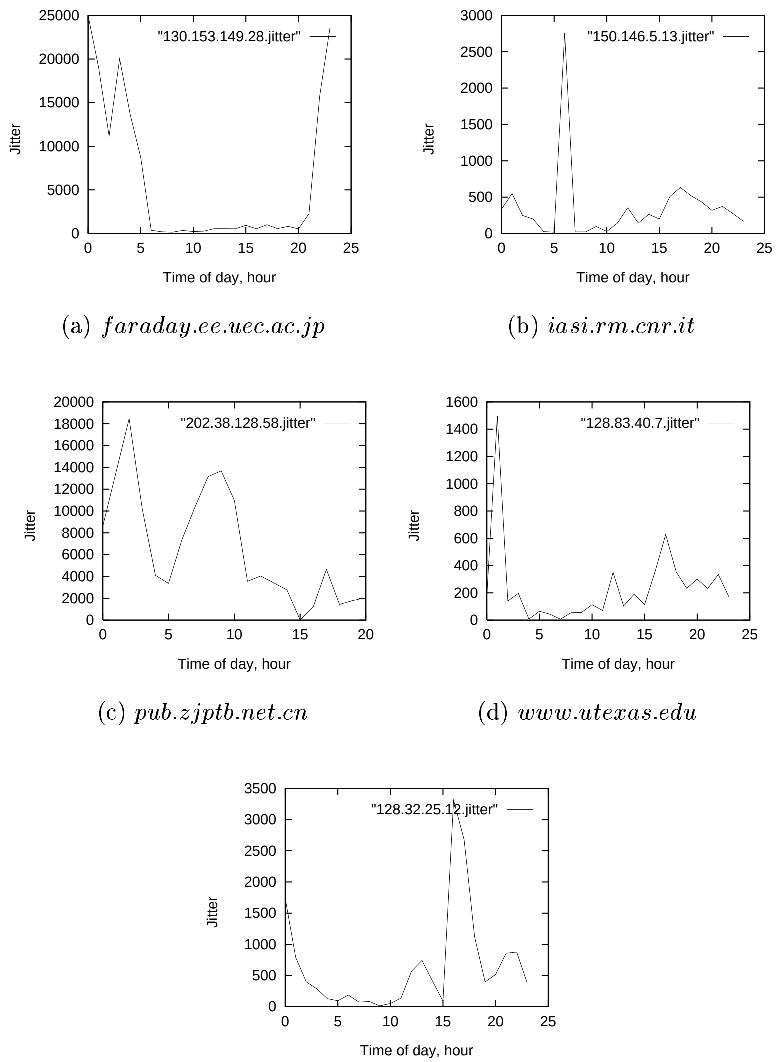


(d) *www.utexas.edu*



(e) *www.berkeley.edu*

Figure 3.7: RTT over TCP from five remote hosts. Results were obtained by the algorithm presented in Figure 3.4. Packet size is 512 bytes.



(a) *faraday.ee.uec.ac.jp*

(b) *iasi.rm.cnr.it*

(c) *pub.zjptb.net.cn*

(d) *www.utexas.edu*

(e) *www.berkeley.edu*

Figure 3.8: TCP jitters for five remote hosts. These results were obtained by the algorithm presented in Figure 3.4. Packet size is 512 bytes.

TCP QoS is investigated in terms of throughput, RTT and jitter. For each set of experiment, setup details are presented, followed by results and analysis. Based on experimental results in this chapter, a conclusion is reached at the end of the chapter that TCP is compatible with MPEG stream.

Chapter 4

Conclusions

There has recently been a flood of interest in real-time multimedia over the Internet, from both industry and academia. Multimedia information systems introduce new challenges because of their new and unique characteristics. This thesis examines experimentally the compatibility between TCP and real-time multimedia traffic.

Chapter 1 explores the context within which the work in this thesis is done. We define the problem we try to solve and briefly cover the reasons why the work in this thesis is important. Chapter 2 reviews past efforts in real-time Multimedia on Demand over the Internet. Chapter 3 presents our experimental setup and the results for MPEG bandwidth requirement and TCP QoS. An analysis of the experimental results concludes the chapter.

We show experimentally in this thesis that TCP bandwidth is not compatible with real-time multimedia transfers over the Internet, therefore makes it difficult to use TCP for multimedia on demand application.

Bibliography

[1] A. Banerjea, D. Ferrari, and B. Mah, "The tenet real-time protocol suite: design, implementation, and experiences," *IEEE/ACM Transactions on Networking,* vol. 4, pp. 1 – 10, Feb. 1996.

[2] A. Basso, G. Cash, and M. Civanlar, "Transmission of MPEG-2 streams over non-guaranteed quality of service networks," in *Proc. of Picture Coding Symposium, Berlin, Germany,* 1997, pp. 19 – 23.

[3] V. Frost, and B. Melamed, "Traffic modeling for telecommunication networks," *IEEE Communication Magazine, 32(4),* April 1994, pp. 70 – 81.

[4] E. Knightly, and H. Zhang,. "Traffic characterization and switch utilization using deterministic bounding interval dependent traffic models," *Proc. IEEE INFOCOM'95, Boston, MA,* 1995, pp. 1137 – 1145.

[5] D. Ferrari, "Real-time communication in an internetwork," *Journal of High Speed Networks, Vol 1,* 1992, pp. 79 – 103.

[6] A. Elwalid, and D. Mitra, "Effective bandwidth of general Markovian traffic sources and admission control of high speed networks," *IEEE/ACM Transactions on Networking, 1(3),* June 1993, pp. 329 – 343.

[7] J. Kurose, "On computing per-session performance bounds in high-speed multi-hop computer networks," in *Proceedings of ACM SIGMETRCS'92, Newport, Rhode Island,* 1992, pp. 128 – 139.

[8] K. Cleary, "Video on demand – competing technology and services," in *IEEE International Broadcasting convention.*, 1995, pp. 432 – 437.

[9] J. Nussbaumer, B. Patel, F. Schaffa, and J. Sterbenz, "Networking requirement for interactive video in demand," *IEEE journal on selected areas in communications, Vol. 13, No. 5,* June 1995, pp. 779 – 787.

[10] S. Jacek, "Approximate performance analysis of real-time traffic over heavily loaded networks with time token protocols," *IEEE/ACM Transaction on networking. Vol. 4, No. 3,* June 1996, pp. 470 – 478.