M.Tech. Dissertation

Video Streaming in Wireless Environments

Submitted in partial fulfillment of requirements
for the degree of
Master of Technology

By

C. Manoj Kumar

Roll No. : 00329012

Under the guidance of

Prof. Sridhar Iyer

Kanwal Rekhi School of Information Technology
Indian Institute of Technology, Bombay
Mumbai, 400 076

January 24, 2002

Dissertation Approval Sheet

This is to certify that the dissertation titled

Video Streaming in Wireless Environments

By

C. Manoj Kumar

(00329012)

is approved for the degree of **Master of Technology**.

Prof. Sridhar Iyer

(Guide)

Internal Examiner

External Examiner

Chairman

Date :

Acknowledgments

January 24, 2002

I am greatly indebted to my guide **Prof. Sridhar Iyer** for his invaluable guidance during the course of the project. He gave me a lot of freedom in choosing my work. He helped many a time when the work was not moving ahead. He was a true guide to me not only in technical but also personal matters.

I thank the **Mobile Computing Research Group, KReSIT, IIT Bombay** for their support and encouragement. They provided some really useful comments and the discussions with them were indeed very useful and helped me in gaining a better understanding of the problem. It was a great experience working with them.

I would also like to thank my classmates **Hrishikesh Karmarkar**, **Anand Janakiraman**, **Balsree** for their valuable ideas and suggestions.

Finally, I would like to thank the entire **KReSIT** family for making it such a wonderful place to work at.

Manoj Kumar
IIT Bombay.

Abstract

Video Streaming refers to the real-time transmission of *stored video*. In order to control network congestion, *rate control schemes* adjust the output rate of the video stream to the estimated available bandwidth in the network. Loss based rate control schemes like AIMD, TFRC employ the packet loss rate reported by the receiver as the principal feedback parameter to estimate the state of the network.

In a heterogeneous network, where the streaming server is located on the wired network and the client/receiver is located on the wireless network, due to the high-bursty error rates of the wireless channels, the loss rate reported by the receiver may not be a correct indicator of congestion in the network. Especially during bad wireless channel conditions, when the channel error rate is high, the loss rate reported will be high. Hence, loss based rate control schemes may inaccurately estimate the state of the network and respond by decreasing the output rate which will affect the quality of video received by the clients.

Two schemes have been proposed to alleviate the above problem. In *Report only congestion losses* scheme, the receiver is made to report only fraction of the packets lost due to congestion. Due to this, the errors in the wireless channel cannot affect the rate control and quality of video as these errors are not reported to the sender. In *Report correlation of loss and delay*, besides loss rate, the correlation of the packet loss and delay during a feedback interval is reported. This parameter is used by the sender in addition to the loss rate to respond to only congestion.

Contents

List of Figures

Chapter 1

Introduction

Video Streaming refers to the real-time transmission of *Stored Video*. There are two modes for transmission of stored video over the Internet, namely the *download mode* and the *streaming mode*. In the download mode, a user downloads the entire video file and then plays back the video file. However, full file transfer in the download mode usually suffers long and perhaps unacceptable transfer time due to the large size of the video files. In contrast, in streaming mode, the video content need not be downloaded in full, but will be played immediately while parts of the content are being received and decoded.

1.1 Streaming in the Internet

Due to the real-time nature, video streaming typically has stringent bandwidth, delay and loss requirements. However, the current Internet which offers best-effort service doesn't offer any Quality of Service(QoS) guarantees to streaming video. Two distinct approaches have emerged to tackle this problem. One approach is to design new protocols and router scheduling disciplines to provide the desired performance guarantees. But, these mechanisms are not expected to be available in the very near future. Another approach is to make the applications adapt the packet rate according to the state of the network, the objective being to limit the packet rate to the capacity of the network. This is achieved by adjusting the output rate of the video coder through the adjustment of parameters inside the video coder. The above mechanism is known as *"Rate Control Mechanism"*[3]. The objective of "Rate Control" is to avoid congestion and to maximize quality in the presence of packet loss. Feedback regarding the state of the network can be obtained explicitly from the routers and receivers in the form of notifications in the packets and QoS feedback reports. The feedback can also be obtained implicitly from the packet loss, delay data during data transfer.

Real-Time Protocol(RTP) is an Internet standard protocol designed to provide end-to-end transport functions for supporting real-time applications. Real-Time Control Protocol(RTCP) is a companion protocol with RTP and is designed to provide QoS feedback to the participants of an RTP session. RTP is a data transfer protocol and provides functions in support of media streaming like timestamping, sequence numbering, payload type identification, source identification. RTCP is a control protocol and provides services like QoS feedback, participant identification [9].

In unicast streaming, the streaming application encapsulates the data in RTP packets and sends it to the receiver. It periodically receives QoS feedback from the receiver in the form of RTCP receiver reports. The feedback includes statistics regarding the quality of reception of packets such as fraction of packets lost during a feedback interval, packet inter-arrival jitter, cumulative number of packets lost etc. Based on this feedback, the application calculates the output rate of the video coder using the *"Rate Control Algorithm"*. For example, in the case of Additive Increase Multiplicative Decrease (AIMD) rate control algorithm, the algorithm probes for additional bandwidth and if the probe experiment succeeds, it additively increases the rate otherwise it decreases the rate multiplicatively. Similarly, there is another class of rate control algorithms which adjust the rate of the coder based on a model (e.g.,like throughput model of a TCP connection) using the QoS feedback provided by the receivers.

1.2 Streaming in Wireless Environments

Video streaming in wireless networks is considered to be a hard problem because wireless networks are characterized by low bandwidth and higher error rates than that of wired networks due to multi-path fading, mobility of devices, and interference with other entities such as microwaves etc. The problem becomes even harder when the streaming server is located on the wired network and the client/receiver is located on the wireless network as they both have different packet loss characteristics. A packet loss in a wired network is primarily due to congestion whereas a packet loss in a wireless channel is primarily due to bad channel conditions. In addition, wireless channel errors are usually bursty when compared to the random packet errors in the wired networks. The frequent and back-back losses can seriously degrade the QoS received by the wireless host.

The next section presents an overview of the problem and outlines the solution schemes.

1.3 Overview of the Problem

In a heterogeneous wired/wireless network scenario where the streaming server is located on the wired network and the client/receiver is located on the wireless network, due to the different packet loss characteristics of the two networks, the loss rate reported by the receiver may not be a correct indicator of congestion in the network. Hence, rate control schemes which employ loss rate as their principal feedback parameter inaccurately estimate the state of the network and respond by decreasing their output rate assuming congestion. This affects the quality of the video delivered to the client. Especially during bad wireless channel conditions, when the error rate is high due to bursty errors, the loss rate reported will be high and will drastically affect the quality of video. The above problem occurs due to two prime reasons

- The inability of the receiver to distinguish between packet losses due to congestion in the network and wireless channel errors. As a result of this, the receiver reports the total loss rate experienced which may include both congestion and wireless losses.
- The sender side rate control relies mainly on the loss rate reported by the receivers which may not be accurate in a heterogeneous network environment.

The solution schemes proposed in the project try to address the above issues. The next section presents a brief outline of the schemes.

1.4 Solution Outline

The two solution schemes which are proposed are

- Report Only Congestion Losses
- Report Correlation of Loss and Delay

In *Report Only Congestion Losses*, the receiver is enabled to report only the fraction of packets lost due to congestion during a particular feedback interval.

In *Report Correlation of Loss and Delay*, the receiver reports the correlation of packet loss and delay besides the usual QoS feedback parameters. The sender uses this parameter to intelligently adapt only to congestion in the network.

1.5 Organization of the Report

Chapter 2 explains the various architectures for streaming video in the Internet. It also introduces the various protocols used in media streaming like RTP, RTCP, RTSP etc. Chapter 3 briefly explains the error characteristics of wireless channels and it's impact on rate control algorithms. It presents a detailed view of the problem. Finally, it discusses some related work and the focus of the current project. Chapter 4 explains the system model which was assumed while solving the problem and the main assumptions which were made about the system. It also explains the two solution schemes in detail. Chapter 5 describes the simulation experiments which were conducted to test the proposed schemes and presents their results. Finally, Chapter 6 concludes the report with pointers to future work.

Chapter 2

Streaming in the Internet

Before the advent of *streaming media*, audio and video clips were digitized , encoded and presented as files on the computer's file system. Special multimedia programs were used to decompress and render such files to view the information content.

The first and most natural extension of this paradigm on the Internet was the concept of *downloadable media*. Compressed media files from the web were expected to be downloaded on local machines, where they could be played using the standard multimedia software. However, this was not a satisfactory solution for users with limited amounts of disk space, slow connection speeds and/or limited patience. This essentially created the need for *streaming media*, a technology that enabled the user to experience a multimedia presentation on-the-fly, while it was being downloaded from the Internet.

This chapter discusses some architectures for streaming video over the Internet. It specifically describes application-layer QoS control mechanisms for streaming video. Finally, it presents an overview of the key protocols used for streaming video.

2.1 Architecture for Video Streaming

There are two different architectures for delivering streaming service in the Internet. They are

- HTTP based Streaming
- Streaming Server based Streaming

This section examines each of the above architectures in more detail

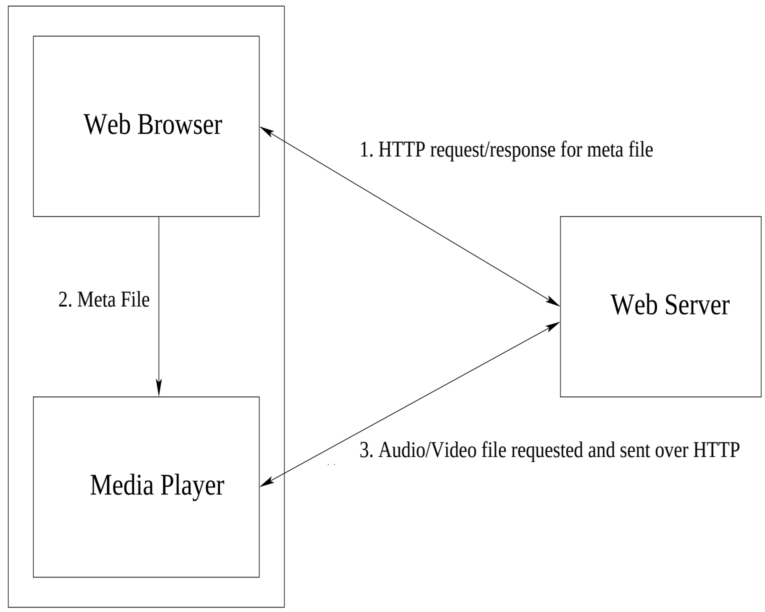


Figure 2.1: HTTP based Streaming

2.1.1 HTTP based Streaming

This architecture is based on using standard Web servers to deliver encoded media content [10]. In this case, the video file is an ordinary object in the server's file system, just as are HTML and JPEG files. whenever a user clicks on a hyperlink for an video file then

- The hyperlink does not point directly to the video file, but instead to a meta file. The meta file contains the the URL of the actual video file. The HTTP response message that encapsulates the meta file includes a content-type: header line that indicates the specific video application.

- The client browser examines the content-type header line of the response message, launches the associated media player, and passes the meta file to the media player.

- The media player sets up a TCP connection directly with the HTTP server. The media player sends an HTTP request message for the video file into the TCP connection.

- As shown in Fig. 2.1, the video file is sent within an HTTP response message to the media player. The media player streams out the video file.

However, being originally designed for serving static documents, HTTP protocol was not particularly suited for real-time streaming. For example, the lack of control over the rate at which the Web server pushes data through the network, as well as the use of the guaranteed-delivery protocol(TCP), caused substantial fluctuation in the delivery times for the fragments of the encoded data. Some media players used a quite large(5-20s) *preroll buffer* that was meant to compensate for the burstiness of such a delivery process. But, if for some reason the delivery of the next fragment of data was delayed by more than the available *preroll time*, the player had to suspend rendering until the buffer was refilled. This so-called *re-buffering* process was a frequent cause of diminished user experience.

Some other challenges of using Web servers are streaming of live presentations and implementing VCR-style navigation features such as seek, fast-forward, and rewind for on-demand streaming.

2.1.2 Streaming Server based Streaming

In order to get around HTTP and/or TCP, the video can be stored on and sent from a streaming server to the media player. With a streaming server, the video can be sent over UDP(rather then TCP) using application-layer protocols that may be tailored to video streaming than is HTTP. The architecture is as shown in Fig. 2.2 [14].

Streaming server is one of the key components of a streaming architecture. To offer quality streaming services, streaming servers are required to process multimedia data under timing constraints. They need to retrieve media components in a synchronous fashion.

In Fig 2.2, raw video and audio data are pre-compressed by *video compression* and audio compression algorithms and then saved in storage devices. Upon the client's request, a *streaming server* retrieves compressed audio/video data from storage devices and then the *application-layer QoS control* module adapts the video/audio bit-streams according to the network status and QoS requirements. After the adaptation, the transport *protocols* packetize the compressed bit-streams and send the video/audio packets to the Internet. Packets may be dropped or experience excessive delay inside the Internet due to congestion. To improve the quality of video/audio transmission, *continuous media distribution services* are deployed in the Internet. For packets that are successfully delivered to the receiver, they first pass through the transport layers and are then processed by the application layer before being decoded at video/audio decoder. To achieve *synchronization* between video and audio streams during presentation, *media synchronization mechanisms* is required.

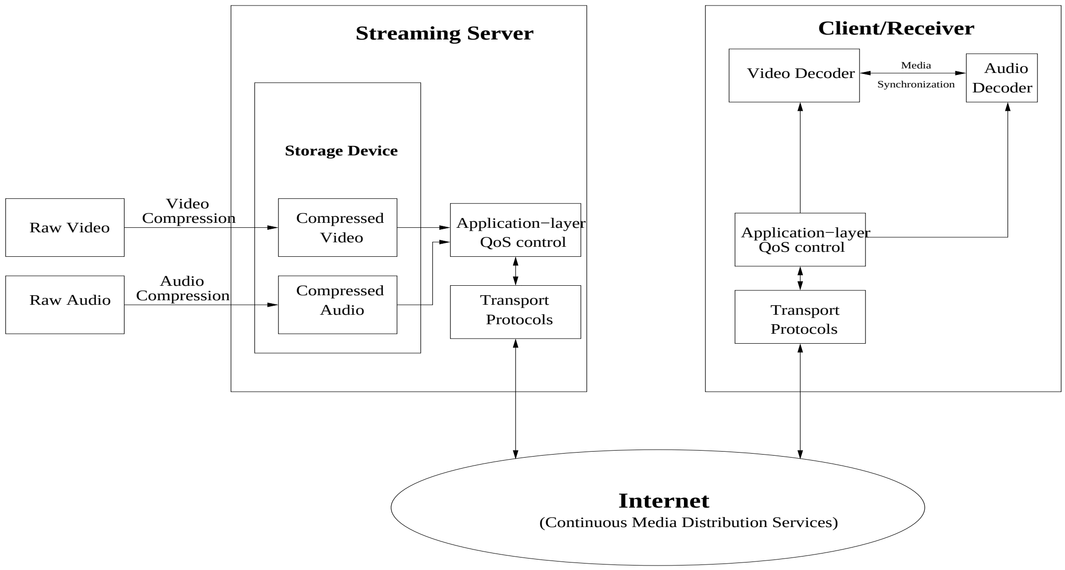


Figure 2.2: Streaming Server based Streaming

- *Video compression*

  Raw video must be compressed before transmission to achieve efficiency. Video compression schemes can be classified into two categories : scalable and non-scalable video coding.

  A non-scalable video encoder generates one compressed bit-stream. In contrast, a scalable video encoder compresses a raw video sequence into multiple sub-streams. One of the compressed sub-streams is the base sub-stream, which can be independently decoded and provides coarse visual quality. Other compressed sub-streams are enhancement sub-streams, which can only be decoded together with the base sub-stream and can provide better visual quality. The complete bit-stream provides the highest quality. Specifically, compared with decoding the complete bit-stream, decoding the base sub-stream or multiple sub-streams produces pictures with degraded quality, or a smaller image size or a lower frame rate. Compared with non-scalable video, scalable video is more adaptable to the varying available bandwidth in the network

- *Application-layer QoS Control*

  The main function of application-layer QoS control is to maintain the presentation quality amidst changing network conditions. For this, it performs congestion control and error control. Congestion control is employed to prevent packet loss and reduce delay. Error control is to improve video presentation quality in the presence of packet loss. Error control mechanisms include forward error correction (FEC), re-transmission, error-resilient encoding and error concealment. Section 2.2 explains the various techniques in more detail.

- *Continuous Media Distribution Services*
  The Internet offers best-effort service and hence quality multimedia presentations need network support to reduce the transport delay and packet loss ratio. Continuous media distribution services are built on the top of Internet Protocol (IP), to achieve QoS and efficiency. They refer to techniques like network filtering, application-level multicast, and content replication.

- *Media Synchronization Mechanisms*
  Media synchronization refers to maintaining the temporal relationships within one data stream and between various media streams. It is a major feature that distinguishes multimedia applications from other traditional data applications. With media synchronization mechanisms, the application at the receiver side can present various media streams in the same way as they were originally captured.

- *Transport Protocols*
  Protocols are designed and standardized for communication between clients and streaming servers. Protocols for streaming media provide such services as network addressing, transport and session control. Section 2.3 briefly explains some standard protocols which are used in video streaming.

2.2 Application-Layer QoS Control : Rate Control

The objective of application-layer QoS control is to avoid congestion and maximize video quality in the presence of packetloss. The application-layer QoS control techniques include congestion control and error control. These techniques are employed by the end-systems and do not require any QoS support from the network [14]. This section presents the various approaches for congestion control and briefly describes some mechanisms for error control.

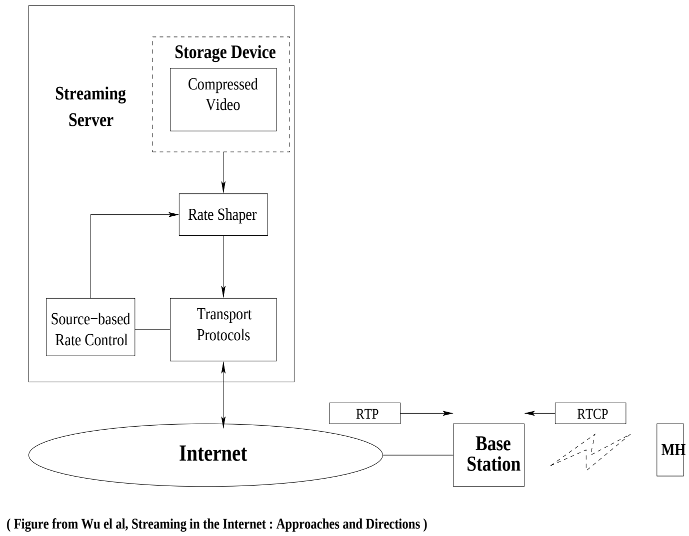


Figure 2.3: Architecture for Source-Based Rate Control

2.2.1 Congestion Control

Network congestion causes excessive packet losses which usually has a devastating effect on video presentation quality. Thus, congestion control mechanisms at end systems are necessary to help reducing packet loss and delay. Typically, for streaming video, congestion control takes the form of rate control. Rate control attempts to minimize the possibility of network congestion by matching the rate of the video stream to the available network bandwidth [3].

Rate Control

Rate control is employed to adapt the output rate of the video coder based on the estimated available bandwidth in the network. Existing rate-control schemes can be classified into two categories : source-based and receiver-based. These schemes are briefly descried below.

1. Source-Based Rate Control

   In the source-based rate control scheme, the source performs the task of adapting to the state of network by adjusting the output rate based on the feedback about the network. The architecture of source-based rate control is as shown in Figure 2.3 [14]. For unicast video, the existing source-based rate control mechanisms follow two approaches : probe-based and model-based approach.

   In the probe based approach, the source probes for the available bandwidth and maintains the packet loss ratio above some threshold P_{th}. There are two ways of adjusting the sending rate :

(a) additive increase and multiplicative decrease (AIMD)

(b) multiplicative increase and multiplicative decrease (MIMD)

.

The model-based approach adjusts the sending rate based on some model of the network (like the throughput model of a transmission control protocol (TCP) connection). Specifically, the throughput of a TCP connection can be characterized by the following formula [6]

$$\lambda = \frac{1.22 \times MTU}{RTT \times \sqrt{p}} \tag{2.1}$$

where

λ	throughput of a TCP connection;
MTU	(maximum transit unit) is the packet size used by the connection;
RTT	round-trip time for the connection;
p	packet loss event ratio experienced by the connection.

The sender determines the output rate of the video stream based on Equation 2.1. Thus, the video connection can avoid congestion in a similar way to that of TCP and it can compete fairly with TCP flows. For this reason, the model-based rate control is also called "TCP-Friendly" rate control [6].

2. Receiver-Based Rate Control

In receiver-based rate control, the receivers adapt themselves to the state of the network while the sender doesn't participate in rate control. This is done by adding/dropping channels of the video stream. Typically, receiver-based rate control is used in multicasting scalable video, where there are several layers in the scalable video and each layer corresponds to one channel in the multicast tree.

Similar to the source-based rate control, the receiver-based rate control mechanisms follow two approaches : probe-based and model-based approach. The basic probe-based rate control consists of two parts

(a) When no congestion is detected, a receiver probes for the available bandwidth by joining a layer/channel, resulting in a increase of its receiving rate. If no congestion is detected after joining, the join-experiment is successful. Otherwise, the receiver drops the newly added layer.

(b) When congestion is detected, a receiver drops a layer(i.e.,leaves a channel), resulting in a reduction of its receiving rate.

Unlike the probe-based approach, which implicitly estimates the available bandwidth through probing experiments, the model based approach uses explicit estimation for the available bandwidth and is based on Equation 2.1.

In some cases, receivers can regulate the receiving rate of video streams by adding/dropping channels, while the sender can also adjust the transmission rate of each channel based on feedback from the receivers. The above scheme which combines both the source-based and receiver-based rate control approaches is called "Hybrid Rate Control".

Rate Shaping

The objective of rate shaping is to match the rate of a pre-compressed video bit-stream to the target rate constraint. A rate shaper (or filter) which performs rate shaping, is required for the source-based rate control [14]. This is because the stored video may be pre-compressed at a certain rate, which may not match the available bandwidth in the network.

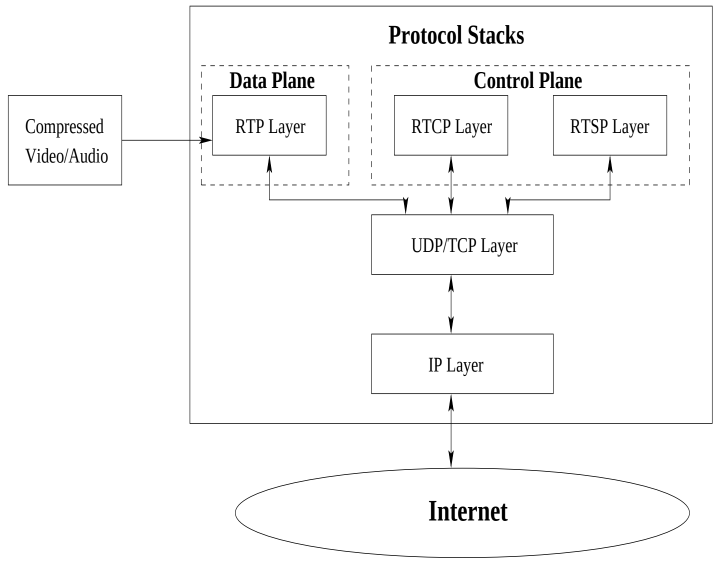
There are many types of filters, such as codec filter, frame-dropping filter, layer-dropping filter, frequency filter, and re-quantization filter, which are described as follows. These filters try to adapt the output rate of the video coder by transcoding to a different compression scheme, dropping frames, dropping layers of video, changing color frames to monochrome etc.,.

2.2.2 Error Control

Packet loss is inevitable in the Internet and may have significant impact on the perceptual quality. Error control maximizes the video presentation quality in the presence of packet loss. Error control mechanisms include Forward Error Correction(FEC), re-transmission, error-resilient encoding and error concealment [10].

- *FEC*

  The principle of FEC is to add redundant information so that original message can be reconstructed in the presence of packet loss. Based on



(Figure from Wu et al, Streaming in the Internet : Approaches and Directions)

Figure 2.4: Protocol stacks for Media Streaming

the kind of redundant information to be added, FEC schemes can be classified into three categories : channel coding, source coding and joint source/channel coding. In channel coding, a video stream is first chopped into segments, each of which is packetized into k packets; then, for each segment, a block code is applied to the k packets to generate an n-packet block, where $n > k$. To recover a segment, a user only needs to receive any k packets in the n-packet block. In source coding, the n^{th} packet contains the n^{th} group-of-blocks (GOB) and redundant information about the $(n-1)^{th}$ GOB, which is a compressed version of the $(n-1)^{th}$ GOB with larger quantizer. Joint source/channel coding is an approach to optimal rate allocation between source coding and channel coding.

2.3 Protocols for Video Streaming

The protocols related to Internet streaming can be classified into the following categories.

1. *Network-layer protocol*

   It provides basic network service support such as network addressing. The IP serves as the network-layer protocol for Internet video streaming.

2. *Transport protocol*

It provides end-end network transport functions for streaming applications. The transport protocol family for media streaming includes UDP, TCP, Real-Time Protocol (RTP), and Real-Time Control Protocol (RTCP) protocols.

UDP and TCP provide basic transport functions like multiplexing, error control while RTP/RTCP run on top of UDP/TCP. Different from UDP, TCP employs congestion control to avoid sending too much traffic, which may cause network congestion. It also employs flow control to prevent the receiver buffer from overflowing while UDP doesn't have any flow control mechanism. Finally, TCP offers a reliable transport service and uses re-transmission to recover lost packets. Since TCP re-transmission introduces delays that are not acceptable for streaming applications with stringent delay requirements, UDP is typically employed as the transport protocol for video streams. In addition, since UDP doesn't guarantee packet delivery, the receiver needs to rely on upper layer (i.e., RTP) to detect packet loss.

RTP is an Internet standard protocol designed to provide end-to-end transport functions for supporting real-time applications [9]. RTCP is the control protocol designed to work in conjunction with RTP. RTP doesn't guarantee QoS or reliable delivery, but rather, provides the following functions to support media streaming

(a) *Time-stamping :* RTP provides time-stamping to synchronize different media streams. RTP is not responsible for synchronization but it is left to the applications.

(b) *Sequence numbering :* RTP employs sequence numbering to place the RTP packets in the correct order. The sequence number is also used for packet loss detection.

(c) *Payload type identification :* The type of the payload contained in an RTP packet is indicated by an RTP-header field called payload type identifier. The receiver interprets the content of the packet based on this field.

(d) *Source identification :* The source of each RTP packet is identified by an RTP-header field called Synchronization Source identifier (SSRC) which provides a means for the receiver to distinguish different sources.

The primary function of RTCP is to provide feedback to an application regarding the quality of data distribution. The feedback is in the form of sender reports (sent by the source) and receiver reports (sent by the receiver). The reports contain information on the quality of reception such as

- fraction of the lost RTP packets since the last report;
- cumulative number of lost packets, since the beginning of reception;
- packet inter-arrival jitter;
- delay since last report;

Based on the feedback, the sender can adjust its transmission rate. RTCP SDES (Source Description) packets contain textual information called canonical names as globally unique identifiers of the session participants which provides a human-friendly mechanism for source identification.

To scale the RTCP control packet transmission with the number of participants, a control mechanism has been designed. The control mechanism keeps the total control packets to 5% of the total session bandwidth. Among the control packets, 25% are allocated to the sender reports and 75% to the receiver reports. RTCP sender reports contain an indication of real-time and the corresponding RTP timestamp which aids in inter-media synchronization at the receiver.

3. *Session control protocol*

   Real-Time Streaming Protocol (RTSP) is a session control protocol for streaming media over the Internet [8]. One of the main functions of RTSP is to support VCR-like control operations such as stop, pause/resume, fast forward, and fast backward. Other main function is to establish and control streams of continuous audio and video media between the media servers and the clients. It also provides means for choosing delivery channels (e.g.,UDP, TCP), and delivery mechanisms based on RTP.

   In RTSP, each presentation and media stream is identified by an RTSP universal resource locator (URL). The overall presentation and the properties of the media are defined in a presentation descriptor file, which may include encoding, language, RTSP URLs, destination address, port, and other parameters. The presentation descriptor file can be obtained by the client using HTTP, email or other means.

This chapter provided an overview of the various streaming architectures in the Internet. The next chapter explains the challenges involved in delivering streaming service to wireless networks and discusses the related work done.

Chapter 3

Streaming in Wireless Environments

The challenge of being able to support wireless streaming is significant since video is generally recognized as being bandwidth hungry, error sensitive, and sometimes delay intolerant; On the other hand, wireless channels are characterized as having limited capacity and high bit error rates, and being time-varying. As the streaming applications are adaptive, the high bit error rates may cause serious problems to rate control schemes which employ loss based feedback.

This chapter briefly describes the error characteristics of wireless channels. It explains the effect of the loss rate reported by the receiver on rate control. It defines the problem and describes some related work done to alleviate the problem. Finally, it explains the focus of the current project.

3.1 Error Characteristics of Wireless Channels

Wireless channels are highly unpredictable. The unreliability comes from errors that arise due to distinct propagation phenomena such as multi-path fading, shadowing, path loss, noise, and interference from other users, all of which have a multiplicative effect on the transmitted signal, causing it to deteriorate.

Multi-path propagation, caused by super-position of radio waves reflected from surrounding objects, gives rise to frequency-selective fading, resulting in rapid fluctuations of the phase and amplitude of the signal. This usually happens when the receiver moves over a distance on the order of a wavelength or more. Shadowing, caused by the presence of large physical objects(buildings, walls, etc.) which preclude a direct line of sight between the radio transmitter and receiver is a medium-scale effect and results in strong signal power attenuation. Path loss causes the received power to vary gradually due to signal attenuation determined by the geometry of the path profile in its entirety. All this is in addition to the local propagation mechanisms, which are determined by terrain features in the immediate vicinity of the antennas. The combined effect of these phenomena is that the receiver has to deal with a bit-stream that is corrupted by both random bit errors and bursty errors. The duration of these bursts in the wireless channel is a function of the receiver velocity and the nature of the time-varying environment.

3.2 Effect of Loss Rate on Rate Control

As described in Section 2.2.1, source-based rate control mechanisms employ feedback to adjust the output rate of the video coder according to the state of the network. The Internet infrastructure typically doesn't provide sources of traffic with explicit feedback information. The only easily available information is implicit information such as measures of losses and/or round-trip delays. Experiments and simulations have shown that control schemes which use packet delays as feedback information can't compete with TCP like traffic which use loss-based feedback [5]. Hence, most rate control algorithms (like AIMD, TFRC etc.,) choose packet losses measured at the receivers as feedback information.

In probe-based rate control schemes like AIMD, if the fraction of packets during a feedback interval p is above some *tolerable limit*, the maximum output rate of the coder is decreased by a constant factor β otherwise the maximum output rate is increased by adding a constant α. Thus, the control algorithm is as follows :

if ($p > tolerable\_loss$)

$maxrate = maxrate - maxrate/\beta;$

else

$maxrate = maxrate + \alpha;$

As described earlier in Section 2.2.1, in model-based rate control schemes such as TFRC [7], the maximum output rate of the coder is adjusted according to Equation 2.1. Assuming there is no congestion in the network the RTT can be assumed to be nearly constant. As stated in [7], If the packet size is also assumed to be constant then the maximum output rate of the sender if it uses TFRC is inversely proportional to the square root of the loss event rate p reported by the client/receiver.

$$\lambda \propto \frac{1}{\sqrt{p}} \tag{3.1}$$

From the above discussion it is evident that the loss rate reported by the receiver significantly affects the rate control algorithm. Hence, it influences the maximum output rate of the video coder which in turn influences the quality of the video received by the client/receiver.

3.3 Problem Definition

Consider a scenario in which the streaming server is located on a wired network and the client/receiver is located in a wireless network and is connected to the wired-domain through a base-station node. The streaming server sends video data in the form of RTP packets to the wireless client. The receiver periodically sends RTCP receiver reports which contain QoS feedback parameters which include fraction of packets lost during that interval (loss rate p), average delay jitter etc.,.

The packets lost from the server to the client may be due to congestion in the wired domain or transmission errors in the wireless domain. Whenever the channel is in a bad state due to fading or interference, the probability of transmission errors will be high and hence a number of packets may be lost in a bursty fashion. Hence, the RTCP reports sent by the receiver during the period when the quality of channel is bad will indicate high loss rate. The adaptation at the sender's end is based on the loss-rate assuming network congestion. As the loss rate is high, the sender decreases the output rate of the coder and hence the quality of video decreases. So, the quality of video whenever the channel is in a bad state is drastically affected.

One of the prime reasons for the above problem is the inability of the receiver to distinguish between the congestion losses and wireless channel losses. Due to this, the receiver reports the total loss rate which it experiences which may also include packets due to wireless errors. On the other hand, the sender's hypothesis that packet loss is an indication of congestion is not valid in a heterogeneous network environment. Hence, the sender side rate control scheme should not solely depend on the loss rate for its adaptation.

3.4 Related Work

To alleviate the above problem arising primarily due to the heterogeneity of the network connecting the streaming server and the client, the application-level video gateway [1] was proposed. Figure 3.1 illustrates the role of a video gateway in a topology containing mobile wireless hosts. The video gateway is

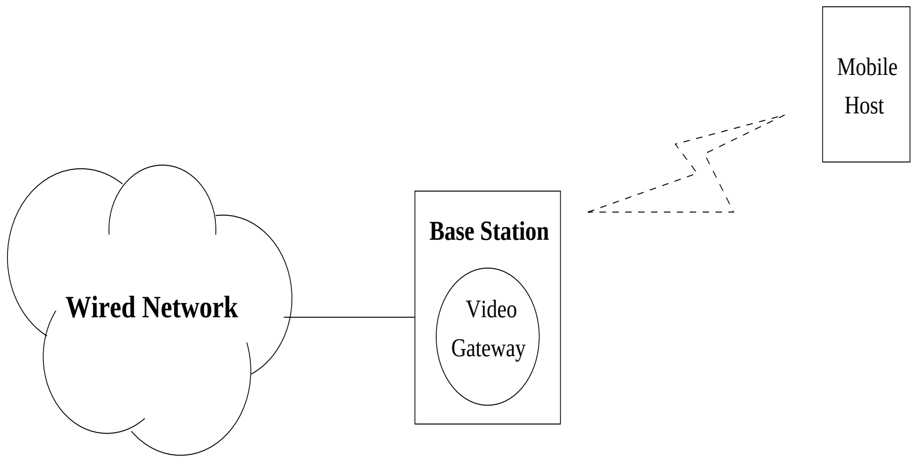


Figure 3.1: Application Level Video Gateway

present at the Base Station (BS) and employs a split-connection approach to solve the problem. It receives the RTP packets containing the video data from the streaming server and transcodes it into a lower bandwidth video stream which is suitable for the wireless host. It acts as a receiver for the streaming server and as a sender to the wireless host. So, it sends the RTCP receiver reports to the streaming server and RTCP sender reports to the wireless host. Whenever the channel is in a bad state, the video gateway controls the output transmission rate over the wireless link. So, even if the loss rate on the wireless link is high, the streaming server will not adapt it's rate as it streams video to the video gateway.

But, the above split-connection approach has some disadvantages like

- Increase in end-end delay due to transcoding of video stream
- Problems in transcoding when the packets in a network are encrypted

3.5 Focus of the Project

The focus of the current project is at

- Modifying the sender side rate control algorithm such that it does not solely depend on the loss rate reported by the receiver for its adaptation.
- Modifying the receiver side algorithm used for reporting the loss rate such that it can avoid reporting wireless transmission errors.

- Investigating new end-end statistics based on packet loss and delay which when reported by the receiver in addition to packet loss rate will aid the sender in better rate control.

The current project does not focus on any kind of video coding techniques to alleviate the effect of burst-errors on quality of video in wireless channels. Also, it does not focus on the effect of user mobility on the QoS received by the receiver or on rate control algorithms.

This chapter explained the characteristics of wireless networks which distinguish them from wired networks. It also explained the problem involved in performing *rate control* in heterogeneous network environments. The next chapter explains the system model which was assumed while solving the problem and presents the solution schemes proposed in detail.

Chapter 4

System Model and Solution Scheme(s)

This chapter briefly explains the system model which was assumed while solving the problem and states some assumptions which were made about the system. Then, it proceeds to explain the two schemes proposed for solving the problem.

4.1 System Model

Figure 4.1 shows the system model which was assumed while defining the problem.

The streaming server is located in the wired network and the wireless client is connected to the streaming server using a base-station node. The streaming server retrieves compressed video from the storage device and then the rate shaper adapts the video bit-stream based on the maximum output rate specified by the source-based rate control module. The source-based rate control computes the maximum output rate based on the RTCP QoS feedback reports sent periodically by the receivers. The source adapts itself only at discrete points of time and the interval between consecutive adaptations is approximately equal to the RTCP feedback interval of the receiver. After the adaptation, the resultant bit-stream is encapsulated into RTP packets and sent to the receiver using TCP or UDP. The sender also periodically sends RTP sender reports containing information about the data transfer such as the cumulative number of packets transferred, maximum sequence number etc.,.

The project assumes that the sender uses some loss based rate control scheme such as AIMD, TFRC or RAP [11] etc.,. It is assumed that only the last link of the network from the streaming server to the receiver is wireless and

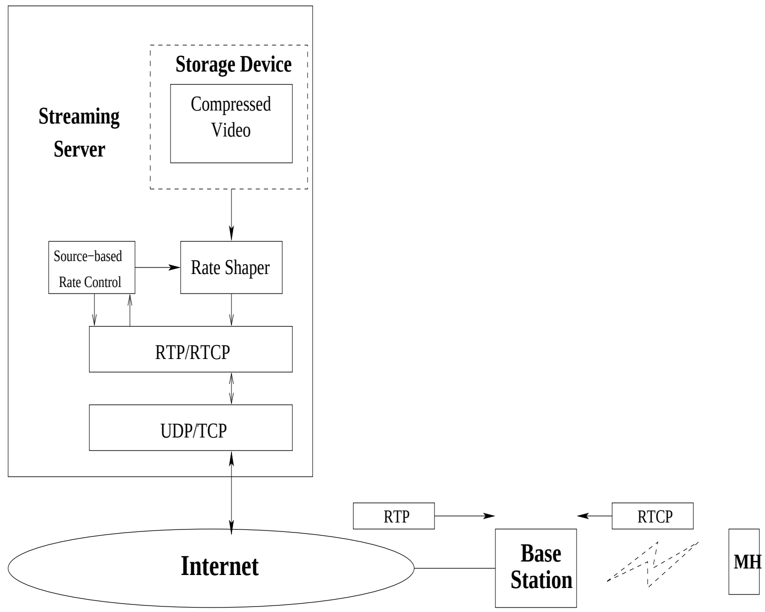


Figure 4.1: System Model

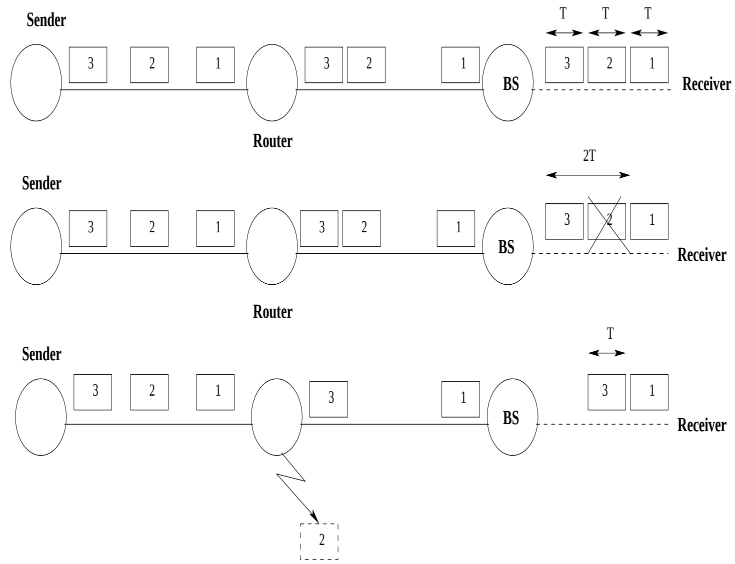
also the bottleneck for the connection.

4.2 The Solution Scheme(s)

The main objective of rate control as explained in Section 2.2 is congestion control. In order to do this, a class of rate control schemes employ the packet loss rate reported by the receiver as an indicator of congestion in the network. In the problem defined in Section 3.3, the packet loss experienced by the receiver may be due to congestion in the network and/or bad channel conditions. Since the receiver cannot distinguish between them, the loss rate reported may not be an exact indicator of network congestion. This is the primary reason which leads to the problem defined in Section 3.3. Two schemes which were proposed during this project to alleviate this are explained in the next sections.

4.2.1 Reporting Only Congestion Losses

The central idea of the following scheme is to allow the receiver to report packet losses which are caused only due to congestion and to ignore the packet loss caused by the wireless channel. As a result of this, the high-bursty error rate of the wireless channel will not affect the rate control and the quality of video as these errors are not reported to the streaming server. Hence, rate control algorithm at the sender will now start responding only to congestion in the



(From Nitin Vaidya et al, Discriminating Congestion Losses and Wireless Losses Using Inter−arrival times at the Receiver)

Figure 4.2: Discriminating Congestion and Wireless losses at the Receiver

network which is desirable. In this scheme, the receiver side algorithm used for detecting packet losses is altered to enable it to distinguish the packets lost due to congestion and wireless transmission errors. Saad Biaz and Nitin H. Vaidya have proposed a heuristic in [2] to discriminate congestion losses from wireless losses at the receiver. This heuristic is used by the current scheme at the receiver end to distinguish losses due to congestion and wireless errors. The heuristic uses the inter-arrival times of packets at the receiver to distinguish congestion losses and wireless channel losses.

Consider a scenario as shown in Figure 4.2 in which packets 1, 2 and 3 are transmitted from a source and only packets 1 and 3 arrive at the receiver whereas 2 is lost due to wireless channel errors. The receiver computes the inter-arrival time between 1 and 3 and computes the average inter-packet separation for each of the lost packets. If this lies within the range of $[averagepacketsize - 2 \times meandeviation, averagepacketsize + 2 \times meandeviation]$ then the receiver assumes that all the packets were lost due to wireless channel errors otherwise it assumes that the they were lost due to congestion. In this case, as the inter-arrival time between 1 and 3 is $2T$, where T is the average packet size, the receiver concludes that it is lost due to wireless channel errors.

```
For each received packet pkt
{     highest_seq = pkt→seqno;
      recv_lastpkttime = current_time();
      interarrival_time = recv_lastpkttime - prev_lastpkttime;
      prev_lastpkttime = recv_lastpkttime;
      loss = highest_seq - expected;
      if(loss > 0)
      {
            if(interarrival_time ≥ avg_packetsize - 2 × mean_deviation &&
            interarrival_time ≤ avg_packetsize + 2 × mean_deviation)
            {
                  packet_loss+= loss;
            }
      }
      expected = highest_seq + 1;
      interval_received++;
}

For each feedback interval expire event
{     pkt→fraction_lost = packet_loss / interval_received;
      interval_received = 0;
      send(pkt);
}
```

The receiver uses the above heuristic and computes the fraction of the total packets lost due to network congestion during that feedback interval and reports it to the streaming server. Two metrics are used to evaluate the discriminating ability of this heuristic : Accuracy of congestion loss discrimination A_w and Accuracy of wireless loss discrimination A_c. These are defined as the ratio of

A_w : Number of wireless losses detected to the total number of wireless losses

A_c : Number of congestion losses detected to the total number of congestion losses

Assuming 100% accuracy of the above hueristic, this scheme will cause the sender to adapt to only congestion in the network completely independent of the wireless channel losses. Hence, under such conditions, the performance of this scheme can serve as an upper-bound for evaluting other schemes.

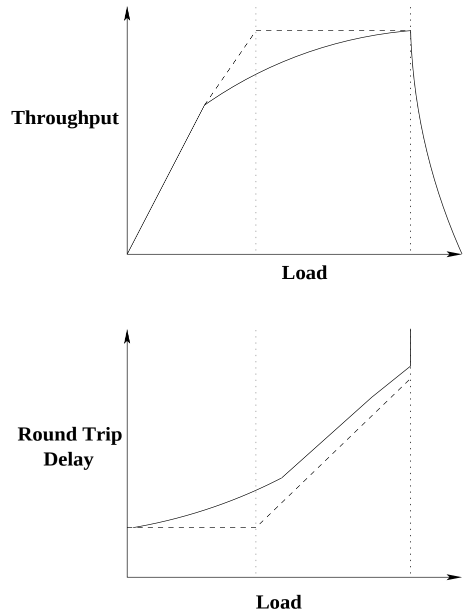


Figure 4.3: Network performance as a function of load

4.2.2 Reporting Correlation of Delay and Packet Loss

The aim of current scheme is to provide another end-end statistical parameter to the sender so that it does not solely depend on the loss rate reported by the receiver for rate control.

Figure 4.3 shows the general patterns of response time and throughput of a computer network as a function of its load. As the load increases, throughput increases and after it reaches the network capacity, it stops increasing. If the load is increased any further, the queues at the intermediate routers start building, resulting in packets being dropped. Similarly, the response time increases little with the load and once the queues start building up, the response time increase linearly until finally, as the queue starts overflowing, the response time increases drastically. The current scheme is based on the observation that in the zone to the right of the cliff in Figure 4.3, as the throughput decreases, the response time increases. So, during congestion, there is a positive correlation between the delay curve and the packet loss curve [4]. The above result is used by the sender during rate control in addition to the packet loss rate reported by the receiver.

Whenever packets are lost due to wireless channel errors and the network is not congested, though the fraction of packet loss increases, the packet delay will remain nearly constant. Hence, the correlation of packet loss and packet delay will not be positive. This prevents the sender from decreasing the output rate due to wireless channel errors when the network is not congested.

Receiver Side Algorithm

For each received packet *pkt*

```
{   highest_seq = pkt→seqno;
    recv_lastpkttime = current_time();
    packet_delay = recv_lastpkttime - pkt→timestamp();
    delay_array[count] = packet_delay;
    loss = highest_seq - expected;
    if(loss > 0)
    {
            interval_loss+= loss;
            cumul_loss+= loss;
    }
    loss_array[count] = cumul_loss;
    count++;
    expected = highest_seq + 1;
    interval_received++;
}
```

For each feedback interval expire event

```
{   pkt→fraction_lost = interval_loss / interval_received;
    pkt→correl_coeff = compute_correlation(loss_array, delay_array);
    interval_received = 0;
    count = 0;
    send(pkt);
}
```

Sender Side Algorithm for AIMD

```
if ( p > tolerable_loss )
    if(correl_coeff > 0)
        maxrate = maxrate − maxrate/β;
else
    maxrate = maxrate + α;
```

During each feedback interval, the receiver computes the correlation of the packet delay with the fraction of packets lost. It also reports the correlation coefficient besides the usual QoS parameters like packet loss rate, average delay-jitter etc.,. The sender decreases the output rate only when the loss rate is greater than the tolerable loss limit and the correlation is positive. If the loss rate is greater than the tolerable loss limit and the correlation is negative then the sender concludes that the loss is due to wireless channel errors and keeps the output rate constant. If the loss rate is less than the tolerable limit, the rate control algorithm is allowed to increase the output rate.

This chapter explained the system model which was assumed while solving the problem and presented the two schemes which were proposed in detail. The next chapter explains the simulation environment used to evaluate the above schemes.

Chapter 5

Simulation Experiments and Results

This chapter describes the simulation experiments that were conducted to evaluate the performance of the schemes proposed in Section 4.2. It also presents the results of these simulations.

5.1 The Network Simulator (NS)

The schemes were evaluated using a network simulation tool called *ns-2* (version 2.1b8a) from UC Berkeley [13]. *ns* is an event driven object oriented simulator written in C++, with an OTcl interpreter as a front end. It implements network protocols such as TCP and UDP, traffic source behavior such as FTP, Telnet, Web etc.,. In simple words, *ns* is an object-oriented Tcl interpreter that has a simulation event scheduler and network component object libraries, network setup module libraries.

To setup and run a simulation network, a user should write an OTcl script that sets up the network topology using the network objects and initiates the event scheduler. The script also tells the traffic sources when to start and stop transmitting packets through the event scheduler. *ns* produces one or more text-based output files that contain detailed simulation data. The data can be used for simulation analysis or as an input to a graphical simulation display tool called Network Animator (NAM).

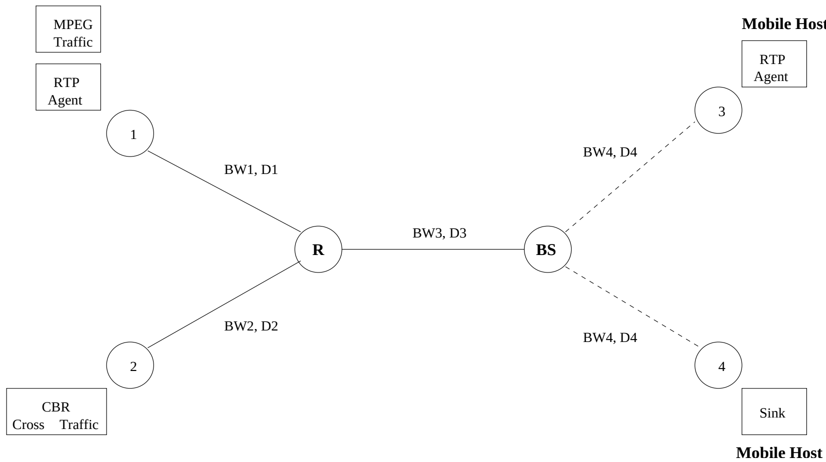


Figure 5.1: Simulation Model

5.2 Simulation Model

Figure 5.1 depicts the topology used for simulation. There are 2 fixed hosts and 2 mobile hosts which are connected through a Router **R** and Base Station node (**BS**). The link between Node 1 and R has a bandwidth of BW_1 and a propagation delay of D_1. Similarly, all the links are labeled with their bandwidth and propagation delays in the figure. There is an MPEG video traffic source at Node 1. This was simulated using MPEG traffic trace files generated from movies. An RTP connection is made between the RTP Agent at Node 1 and the RTP Agent at Node 3. Using this, the MPEG data is sent to Node 3. For ease of implementation, the schemes were tested with AIMD rate control algorithm at the sender with $\alpha = 20kbps$ and $\beta = 4$. But, the proposed schemes generalize to any loss based rate control algorithm with minor modifications. The RTP sender and receiver agents periodically send RTCP reports. The RTP connection shares the link with the cross traffic which is generated by the *Traffic/Expo* agent. The traffic flows from CBR at Node 2 to Sink at Node 4. CBR is a constant-bit rate source with idle time and busy time exponentially distributed. UDP Agent is used to carry this traffic.

The link between the Base Station and the Nodes 3 and 4 is wireless. The wireless channel was modeled using a two state Markov model to capture the bursty nature of the packet errors. The model as shown in Figure 5.2 has two states, a good state (S_0) and a bad state (S_1). A packet is transmitted correctly when the channel is in the good state and errors occur when the channel is in the bad state. P_{00}, P_{01}, P_{10}, P_{11} are the state transition probabilities. The transition between these states occur at each packet instant. The packet error

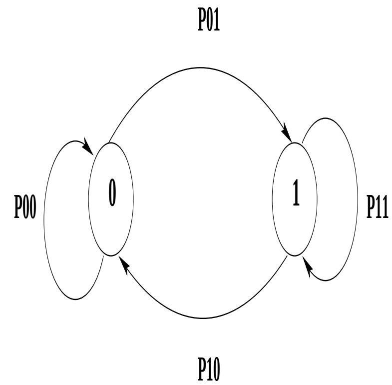


Figure 5.2: Two-state Markov channel model

statistics will vary according to the values of the transition probabilities. For this simulation, the following values were assumed which correspond to a slow fading channel which may be encountered with a walking speed. As stated in [12], this corresponds to an average burst length of 19 packets and a packet error rate of 0.1479.

$$\mathbf{P} = \begin{bmatrix} P_{00} & P_{01} \\ P_{10} & P_{11} \end{bmatrix} = \begin{bmatrix} 0.9909 & 0.0091 \\ 0.0526 & 0.9474 \end{bmatrix}$$

5.3 Modifications to NS

This section explains the modifications made to the *ns* class library to implement the required components for the simulation.

ns already has an implementation of RTP and RTCP Agent. The RTP Agent just implements the basic fields in the RTP packet structure like sequence numbers and synchronization source identifier. The RTCP agent implements the feedback mechanism which is restricted to minimal features. So, a new RTP agent called *RTPMM Agent* was implemented. The *RTPMM Agent* has the following features

- Periodic transmission of RTCP sender and receiver reports with the typical feedback parameters like fraction of packets lost, delay since last sender report etc., which are required for the current simulation.

- Application level output rate control at the sender using an application-level packet buffer(Implemented using queues).

- Integration of rate control algorithm into the RTP Agent in a such a way that it provides feedback to the application traffic for adaptation.

- A more detailed RTP packet structure with fields like timestamp, packet type etc., which are useful in simulating multimedia applications.

In order to implement the two solution schemes, another two versions of RTPMM Agent were created.

The main files which were modified in the *ns* library are the following *rtp.cc*, *rtp.h* and *rtcp.cc* which contain the implementation of RTP and RTCP Agents.

5.4 Simulation Results

This section describes the simulation methodology and presents the results.

In the first experiment, the network is set in an uncongested state so that the only losses which occur are due to wireless channel errors. Accordingly, the bandwidth and propagation delay of the links have been initialized to the following values.

$$\begin{aligned} BW_1 = BW_2 = 1Mbps \quad &, \quad D_1 = D_2 = 2ms \\ BW_3 = 256kbps \quad &, \quad D_3 = 10ms \\ BW_4 = 64kbps \quad &, \quad D_4 = 1ms \end{aligned}$$

Figure 5.3 shows the results of the above experiment with no modification in either the sender or the receiver. The graph shows the evolution of maximum output rate of the sender to the loss rate reported by the receiver. The figure clearly shows the rate fluctuations caused by the wireless channel errors. Figure 5.4 shows the condition after the proposed *Report Only Congestion Losses (ROCL)* algorithm was implemented at the receiver end. The sender is seen as adapting only to congestion losses. The heuristic described in Section 4.2.1 was able to discriminate congestion and wireless losses with the following accuracy

$$\begin{aligned} A_w &= 98.42\% \\ A_c &= 86.80\% \end{aligned}$$

Figure 5.5 shows the results of the experiment when the *Report Correlation between Loss and Delay (RCLD)* scheme was implemented at the receiver and sender ends. It can observed that during bad channel conditions, the sender does not decrease it's rate but keeps the rate constant. This prevents the decrease in the quality of video delivered to the user during bad channel conditions.

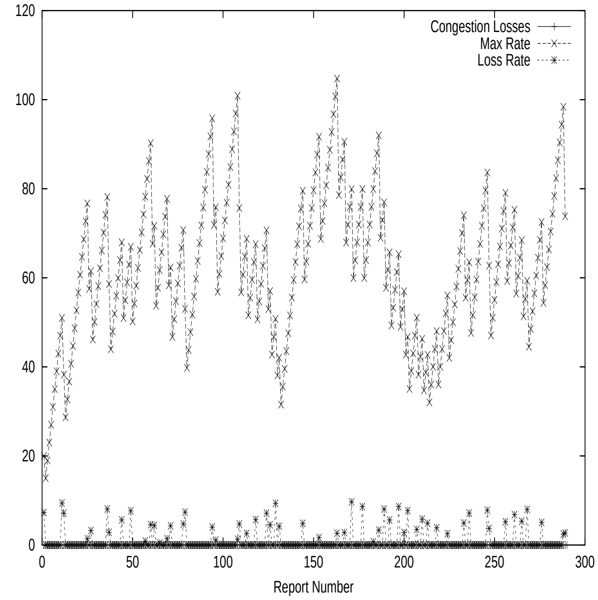


Figure 5.3: Original Scheme Without Modification

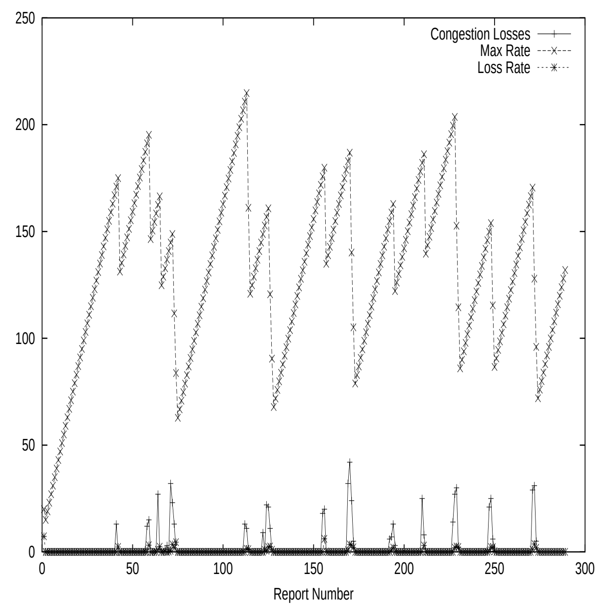


Figure 5.4: With ROCL Modification

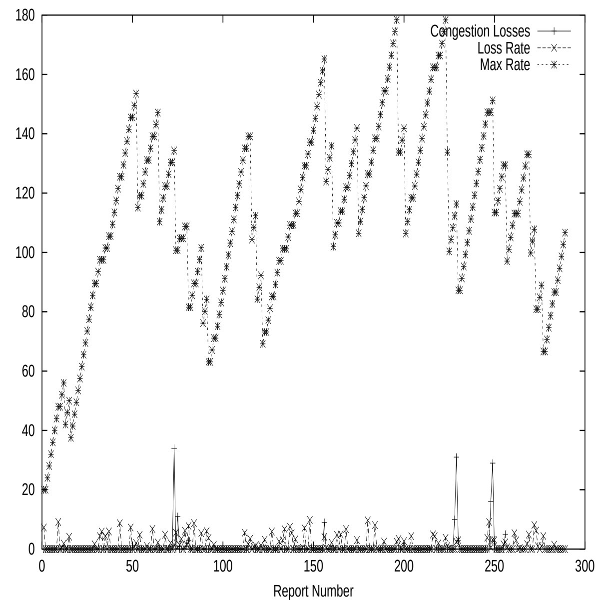


Figure 5.5: With RCLD Modification

In the second experiment, the network is congested by introducing cross traffic from another *Traffic/Expo* source which sends bursts of data once in a fixed interval. The *Traffic/Expo* source sends data at the rate of $32kbps$. The simulation parameters are as follows

$$BW_1 = BW_2 = 128kbps \quad , \quad D_1 = D_2 = 2ms$$
$$BW_3 = 80kbps \quad , \quad D_3 = 10ms$$
$$BW_4 = 64kbps \quad , \quad D_4 = 1ms$$

Figure 5.6 shows the results of the above experiment with no modification in either the sender or the receiver.

Figure 5.7 shows the performance after the $ROCL$ modification. The results clearly show that the modified scheme only responds to congestion where as the original scheme tries to respond to all the losses. Due to this, the sender output rate decreases drastically. The accuracy of the heuristic in this experiment was

$$A_w = 68.85\%$$
$$A_c = 98.04\%$$

The decrease in the accuracy A_w was due to the decrease in the ratio of wired to wireless bandwidth at the base-station node. This results in less buffering of packets at the base-station node which causes the heuristic to fail. Figure 5.8 shows the performance after the $RCLD$ implementation.

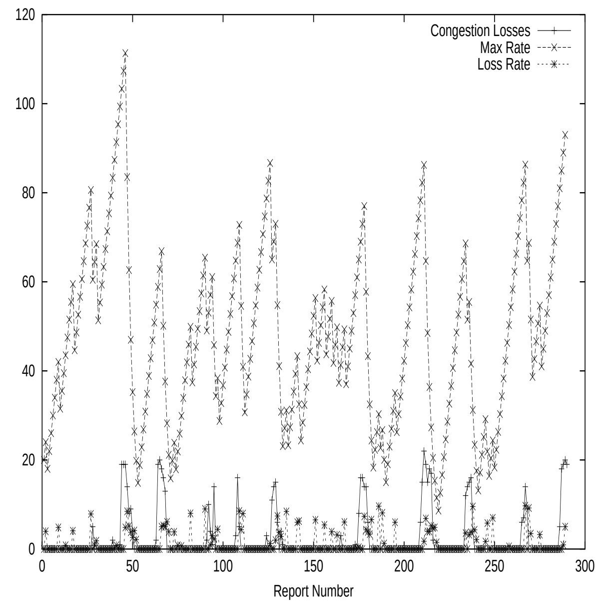


Figure 5.6: Original Scheme Without Modification

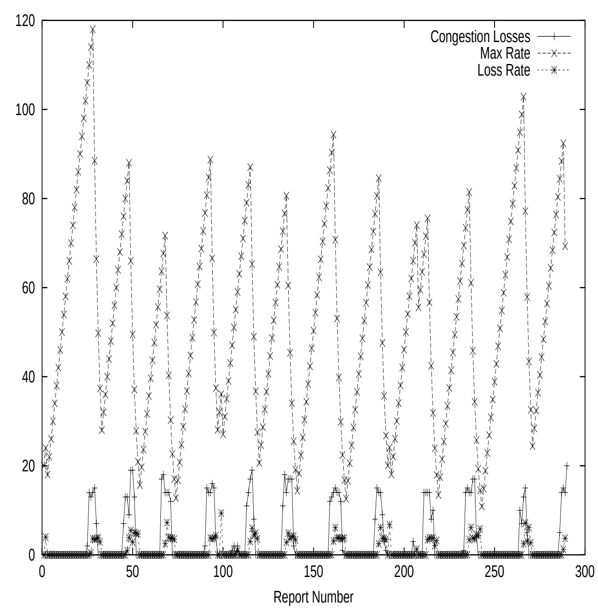


Figure 5.7: With ROCL Modification

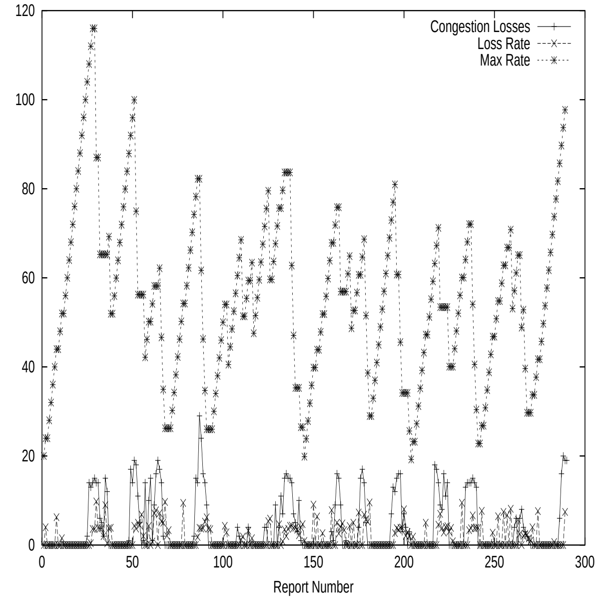


Figure 5.8: With RCLD Modification

This chapter provided a detailed overview of the simulation environment which was used to evaluate the two proposed schemes. It also presented the results obtained. The next chapter concludes the report with pointers to future work.

Chapter 6

Conclusion

The proliferation of wireless hand-held devices with multimedia capability indicates the trend towards heterogeneity in the network connecting the streaming server and the client. Robust rate control schemes which take into account the high error rate due to the presence of wireless channel are required to deliver good QoS to the clients.

In this context, we have proposed two schemes for adapting the loss based rate control schemes to heterogeneous wired/wireless environments. The results show a significant performance improvement when compared to the original schemes. While the original scheme suffers whenever the channel is in a bad state, the above schemes try to respond only to congestion.

6.1 Future Work

Model based schemes like TFRC use Equation 2.1 to adjust their output rate. In a wired/wireless heterogeneous network, the loss event rate p may not be a good parameter to use for the reasons cited in Section 3.3. Hence, one possible direction of future work is to investigate appropriate functions to replace loss event rate p. The sender makes independent adaptation decisions at discrete intervals of time. It will be helpful during adaptation if the sender maintains *state*. If the network was not congested in the previous interval, then the chances for the loss rate to become very high in the immediate interval is very less. Such intelligent adaptation decisions may be possible if we maintain *state* at the sender.

Bibliography

[1] Elan Amir, Steve McCanne, and Hui Zhang. An application level video gateway. *In Proc. ACM Multimedia '95, San Francisco, CA*, 1995.

[2] S. Biaz and N. Vaidya. Discriminating congestion losses from wireless losses using inter-arrival times at the receiver. *IEEE Symposium ASSET'99, Richardson, TX, USA*, 1999.

[3] Jean-Chrysostome Bolot and Thierry Turletti. A rate control mechanism for packet video in the internet. In *INFOCOM (3)*, pages 1216–1223, 1994.

[4] D. Chiu and R. Jain. Analysis of the increase and decrease algorithms for congestion avoidance in computer networks. *Journal of Computer Networks and ISDN Systems*, June, 1989.

[5] W. Dabbous. Analysis of a delay-based congestion avoidance algorithm. *In Proc. 4th IFIP Conf. on High Performance Networking*, 1992.

[6] S. Floyd and K. Fall. Promoting the use of end-end congestion control in the internet. *IEEE Transactions of Networking*, Vol. 7:458–472, Aug 1999.

[7] Sally Floyd, Mark Handley, Jitendra Padhye, and Jorg Widmer. Equation-based congestion control for unicast applications. In *SIGCOMM 2000*, pages 43–56, Stockholm, Sweden, Auguest 2000.

[8] R. Lanphier H. Schulzrinne, A. Rao. *Real-Time Streaming Protocol (RTSP)*. Internet Engineering Task Force, Apr 1998.

[9] V. Jacobson H. Schulzrinne, R. Frederick. *RTP: A Transport Protocol for Real-Time Applications*. Internet Engineering Task Force, Jan 1996.

[10] Keith W. Ross James F. Kurose. *Computer Networking: A Top-Down Approach Featuring the Internet*. Addison Wesley, 2000.

[11] Reza Rejaie, Mark Handley, and Deborah Estrin. RAP: An end-to-end rate-based congestion control mechanism for realtime streams in the internet. In *INFOCOM (3)*, pages 1337–1345, 1999.

[12] Ming-Ting Sun Supavade Aramvith, I-Ming Pao. A rate control scheme for video transport over wireless chaneels. *IEEE Transactions on Circuits and Systems for Video Technology*, Vol. 11, May 2001.

[13] UC Berkeley, LBL, USC/ISI, and Xerox PARC. *ns Notes and Documentation*, Feb 2000.

[14] Dapeng Wu and et al. Streaming video over the internet: Approaches and directions. *IEEE Transactions on Circuits and Systems for Video Technology*, 11:282, 2001.