SYMBIOTIC AUDIO COMMUNICATION ON INTERACTIVE TRANSPORT

A thesis submitted
to Kent State University in partial
fulfillment of the requirements for the
degree of Masters of Science

by

Olufunke Ibitesho Olaleye

May 2007

Thesis written by
Olufunke Ibitesho Olaleye
B.Sc., Yaba College of Technology, 1995
M.S., Kent State University, 2007

Approved by

_______________________________________, Advisor
Dr. Javed I. Khan

_______________________________________, Chair, Department of Computer Science
Dr. Robert A. Walker

_______________________________________, Dean, College of Arts and Sciences
Dean Jerry Feezel

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGEMENTS

I want to express a heart of gratuide to my lovely husband, Anthony Olaleye for all his love, encouragement and support in my studies.

Thanks to my advisor, Dr Javed Khan for all his ideas. Thanks to Dr. Hassan Peyravi and Dr. Paul Farrell for agreeing to review my thesis.

I am very grateful to almighty God for all wisdom, strength and journey mercy during my studies.

Chapter 1

Introduction

1.1 Network Overview

Over the years, the Internet has grown and is still growing. Technology has moved from the simple sharing of files on diskette to sharing files over the Internet around the world. Files transferred over the Internet do not only include data but audio. Audio traffic is sensitive and it shares bandwidth with non-sensitive traffic as shown in figure 1-1.

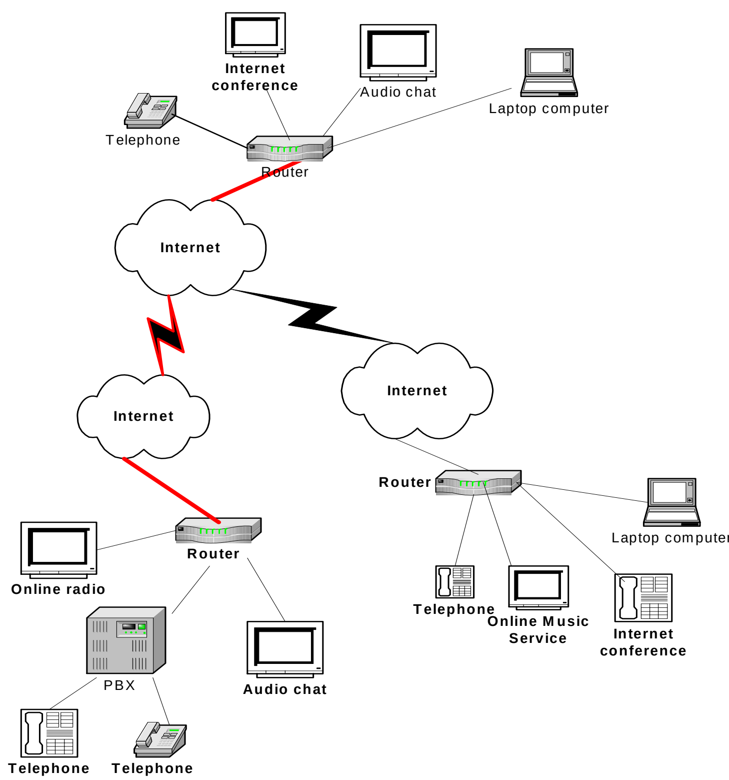


Figure 1-1 Bandwidth Limited Network

Voice traffic includes VoIP, audio chat, online radio, real time Internet lecture, real time Internet conference, online music service and Internet news.

While technology has moved many businesses online, the use of audio traffic over the Internet has growth exponentially. Recent research shows the global growth of VoIP will reach about one hundred and thirty three million subscribers in the year 2009 as shown in figure 1-2 and table 1-1.

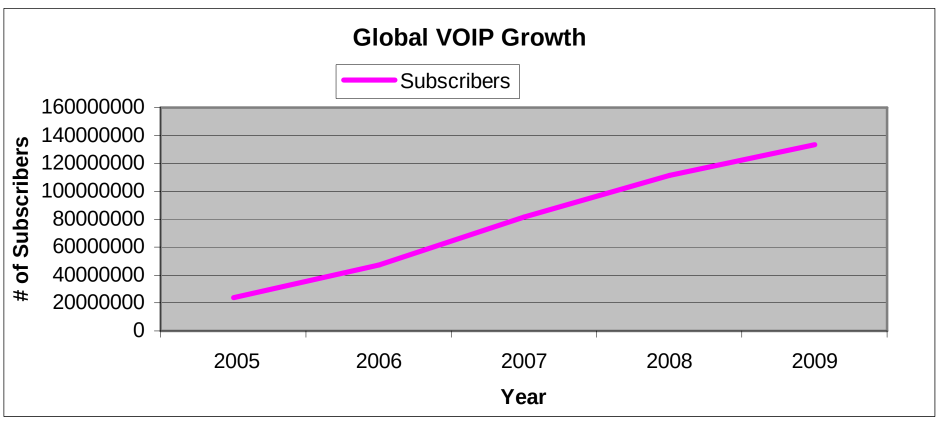


Figure 1-2 Graphical Global VoIP Growth

Global VoIP Growth

	2005	2006	2007	2008	2009
Total Subscribers	24,043,303	47,346,874	81,618,331	111,209,271	133,633,938
Growth %	67	97	72	36	20
Net New Subscribers	9,682,349	23,303,571	34,271,457	29,590,940	22,424,668

Table 1-1 Analysis of Global VoIP Growth[1]

[1] Source from Infonetics Research, February 2006

Audio including voice and music has become a major contributor to Internet traffic. Besides the growth of VOIP, many household applications such as IM®, Skype®, today regularly provide voice service. Audio perception is highly susceptible to disturbance in temporal quality. However, the audio quality degrades when the packets encounter congestion.

A number of factors that affect the quality of service are packet loss, delay and jitter (delay variation). Jitter is often resolved by adaptive playout algorithm at the receiver's end. Packets that experience delay arrive late at the receiver end, the player pause and waits for such packets.

Recently, several algorithms have been proposed to minimize the impact of current packet delays and losses. These algorithms adapt the size of the playout buffer [3], the coding rate [12] and packet path diversity [25] to maximize the audio/voice quality.

However, a major challenge of these algorithms is receiving feedback from the network about congestions.

Many solutions avoid TCP and use UDP to circumvent the buffer clogging caused by late audio packets. However, one lost packet with critical field may mean the loss of a long sequence of UDP data. Network generally do not discriminate the source of a UDP or TCP datagram. Thus, making it a challenge to build network friendly applications that is interactive.

1.2 Research Goal of this thesis

The research goal of this thesis is to reduce the delay and jitter faced by audio traffic in the network during congestion. As already discussed in the introduction, audio traffic is delay-sensitive; a well-engineered, end-to-end network is necessary to transmit audio over the Internet. Although today's networks are not perfect and can be unpredictable, an efficient solution to reduce jitter and delay in audio traffic will be a the sender's (encoder) ability to sense the state of the network and react accordingly. In this thesis, the author studied the MP3 encoder and developed an adaptive encoder that is capable of detecting the underlying bandwidth of the network and modifying the target bit rate to suite the network condition. Thus, the design of this adaptive encoder is interactive and TCP friendly.

Some recent attempts recreate new transport service [1, 2, 21], however this would invariably generate new functionality at the system or network level.

There are some TCP-friendly control mechanisms and many control methods have been researched including:

1. **Retransmission:** based on Automatic Repeat Request (ARQ) has been successful in TCP (Transmission Control Protocol). However, due to the sensitivity of audio over the Internet, retransmission is not suitable for real-time traffic such as audio. This is because it increases the end-to-end delay (RTT) between sender and

receiver thereby exceeding the allowable delay of 150ms for interactive voice application. Alternative to ARQ is FEC.

2. **Channel Coding: FEC**, Forward Error Correction (FEC) is another recovery technique. It transmits redundant information of each packet in subsequent packets. A lost packet can be recovered from the redundant copies piggy backed to subsequent packets. Once a packet is lost, the receiver uses the redundant information containing the same segment to reconstruct the lost information. It however adds delay to the network. [4], [24] indicates this type of technique is not viable because of increase in delay at the expense of bandwidth.
3. **Loss concealment** algorithm: is used to correct losses that FEC fails to recover. It uses repair mechanism like insertion, interpolation or regeneration. Loss concealment algorithm [19] is a method for exploiting inter-channel redundancy in order to conceal losses in a channel with the substitution from other channels by appropriate packetizing of the stream. It interpolates among different channels to derive a waveform substitution for a lost session, instead of using sample from same channels.
4. **Source-coding**: a source and channel rate adaptation algorithm; it adapts its bit rate to the available bandwidth [9] to maximize subjective quality. The algorithm used does not consist of predefined combinations but rather determines the optimal solution given specific constraints on the maximum delay, packet loss, and allowed bandwidth. It exploits the multi-rate capability of adaptive multi-rate codec along with media-independent FEC.

5. **Multiple Path Scheme**: Path diversity [25], sending packets simultaneously over multiple paths to overcome the unpredictability and congestion of the Internet. It sends multiple redundant [4, 19 and 25] descriptions of the audio stream over different independent paths, instead of restricting transmission to one network. It takes advantage of their largely uncorrelated loss and delay characteristics; however, this will eventually increase the burden on the network.
6. **Double Sided Pitch Waveform Replications.** DSPWR [23] is a sophisticated recovery method that can tolerate a much higher packet loss rate with less computation. It resolves the delay due to processing.
7. **Priority Queuing:** (PQ) in [4] is used to improve QoS by giving higher priority to voice packets to alleviate delay of VoIP in the network.

Recent research work on adaptive audio schemes include the following:

A joint rate/error/playout delay control algorithm in [3] optimizes the perceived audio quality and is TCP-Friendly. It uses a channel model for both loss and delay. It relies on the Real-Time Transport Protocol (RTP) [6] and RTCP control report.

The source performs equation based congestion control based on feedback information contained in RTCP reports and adjusts its sending rate by changing the packet size, the time interval (jitter) between packets remaining fixed. It avoids an increase of the playout delay when not needed to maximize the utility function of the audio source.

The parameters of optimization are playout delay and FEC. It jointly chooses the playout delay and FEC method.

A joint source and channel coding adaptation algorithm for adaptive multi-rate (AMR) in [12] optimizes the perceived audio quality. It adapts the codec bit rate to the changing network conditions in order to preserve acceptable levels of quality.

It exploits the multi-rate capability of adaptive multi-rate codec along with media-independent FEC with respect to packet loss within specific constraints. The algorithm finds a compromise between packet loss recovery and end-to-end delay to maximize perceived audio quality.

The parameters of optimization are source and channel bit rate.

A multi-path [25] transmission of real-time voice using multiple redundant descriptions of voice streams is sent over independent network paths. Scheduling the playout of the received voice packets is based on multi-stream adaptive playout scheduling that uses a Larangian cost function to trade delay for loss.

It takes advantage of their largely uncorrelated loss and delay characteristics.

A promising research is in the new TCP friendly paradigm [3, 5, 7, 9, 12, 18 and 20]. However, the inability to receive feedback from the network layer make these paradigms rely completely on application-layer scheme to face the network problem, except for [3] that relies on the Real-Time Transport Protocol (RTP).

An intelligent method called Symbiotic Rate Control (SRC) developed in [14,15,16] adjusts the sender bit rate to imitate the network condition. This method is effective to mitigate the effect of packet loss due to network congestion. Based on this symbiotic rate control approach earlier used on video streaming [13,14], a similar approach for audio streaming is designed. The adaptive encoder reduces the encoding target bit rate when congestion is detected. The listening ear can tolerate a slight reduction in audio quality better than audio delay or complete loss of the sound. Various experiments were performed and conclusions are drawn that this is an effective scheme to reduce jitter and delay faced by sensitive traffic (such as audio: voice and music) in the network.

1.3 Organization of the thesis

Chapter 2 describes the MP3 standards, encoder and human auditory perception. Chapter 3 discusses TCP's congestion control mechanism and internal events. Chapter 4 discusses iTCP, its event model and an overview of its implementation architecture. A complete discussion of iTCP can be found in our technical report [16]. The explanation of the mathematical model for the symbiosis-throttling model is described. Chapter 5 provides the exponential backoff scheme for audio encoding rate and the audio rate control design implementation. Chapter 6 shows how the T -ware modules are implemented based on the symbiosis throttling model and iTCP feedback. Chapter 7 details the experimental setup and results and chapter 8 concludes the thesis.

Chapter 2

MP3 Standard/Encoder

MPEG (Motion Picture Experts Group) is a group of researchers involved in coding audio and video in a new format. MPEG is a compression algorithm standard that reduces the audio and video signal to produce a compressed bit stream for storage.

The standard consist of three layers, layer I, II and III. The layers differ in coding and psychoacoustic models. Layer III is referred to as MP3 taking its name from MPEG layer III.

The ISO/IEC 11172-3 [8] documentation standard describes in detail how audio streams are sampled at audio sample rates of 32 KHz, 44.1 KHz or 48 KHz and encoded at bit rate ranges from 32 kbps to 448kbps depending on the layer selection.

The MP3 encoding takes advantage of the fact that the human auditory system has limitations in perceiving high quality audio. Much of the compression results from the removal of perceptually irrelevant parts of the audio signal.

2.1 The Human Auditory Perception

The human ear perceives signals of frequencies within the range 20 and 20000 Hz. The sensitivity of the human auditory system is high for frequencies between 2.5 and 5 kHz and decreases beyond and below this frequency band.

The perception of the human ear is based on two principles, the threshold in quiet and masking threshold. Threshold in Quiet is also known as the absolute threshold of hearing (ATH) is measured in dB. As illustrated in figure 2.1, the sound pressure level represents signal strength and any signal strengths below the threshold in quiet are inaudible to the human ear.

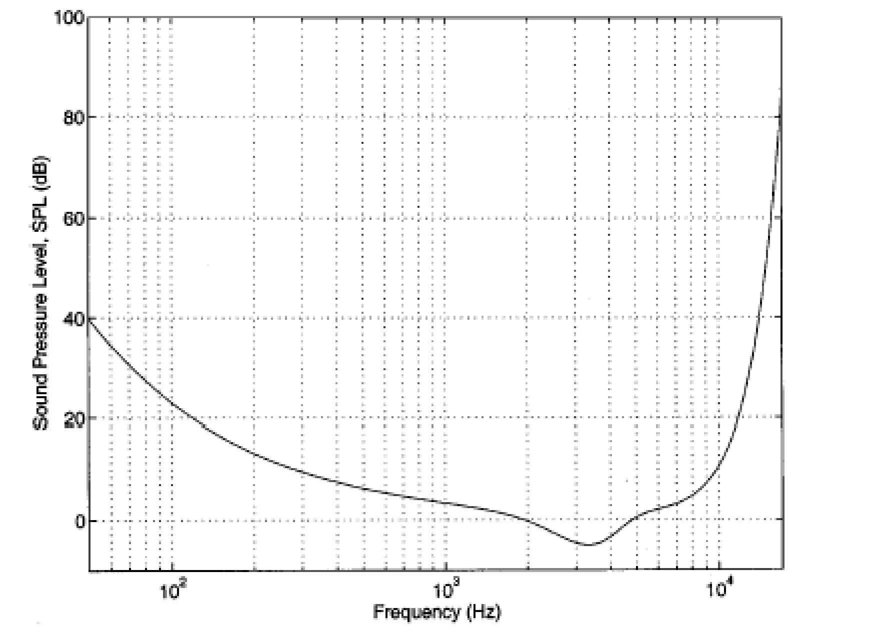


Figure 2.1 Threshold in Quiet

Masking threshold is the threshold at which the human auditory system cannot perceive signal in the presence of another audio signal and it is a function of frequency and time. Figure 2.2 illustrates the threshold in quiet and the masking threshold. Without the presence of a masker, an audio signal with its signal strength below the threshold of quiet is inaudible.

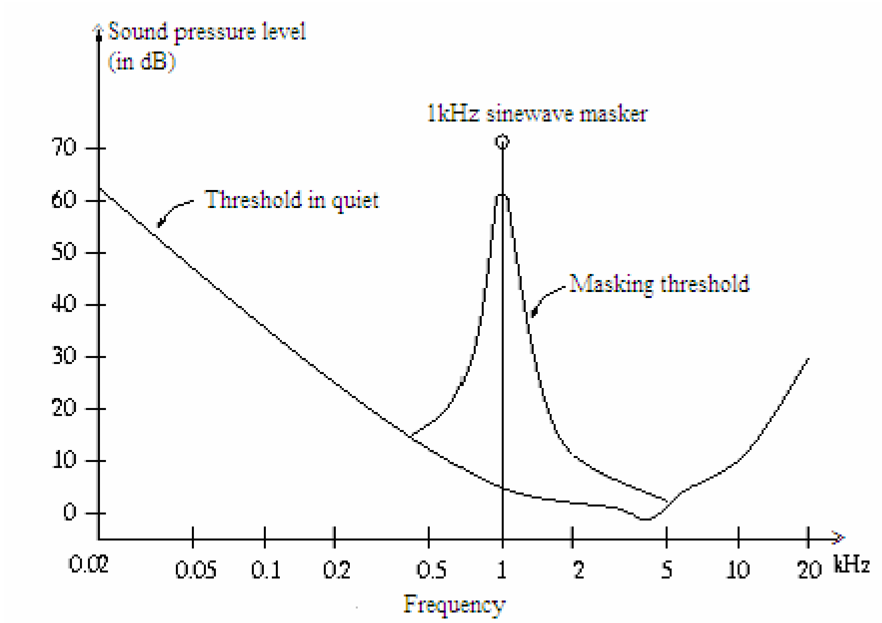


Figure 2.2 Threshold in quiet and Masking Threshold

Masking occurs in two ways, temporal or simultaneous masking. Simultaneous Masking occurs when a lower audio signal (the maskee) becomes inaudible due to the presence of a stronger audio signal (masker) or if two audio signals are close enough to each other in frequency.

The masking ability is dependent on signal strength and critical band frequency of the masker. The human auditory system perceives signal in a non-equal width sub-band called critical bands. It analyses signal in each band independently. Critical bandwidth corresponds to width of one Bark, a unit of measurement as shown in figure 2.3

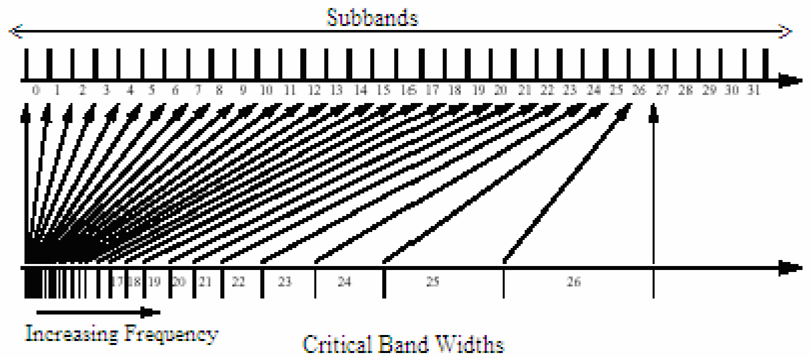


Figure 2. 3 Critical Bandwidth

The critical bandwidth is constant for about 100Hz for centre frequencies of up to 500 Hz but above 500 Hz, the critical bandwidth is 20% larger than those less than 500 Hz.

However the temporal masking occurs when a stronger audio signal (masker) begin to conceal a lower audio signal (maskee) before the loud signal is heard, the effect is known as pre-masking and it continues for a moment after the loud signal has vanished, known as post-masking. Figure 2.4 illustrates temporal masking where the effect of the post masking takes about 200ms and the pre-masking last for 20ms.

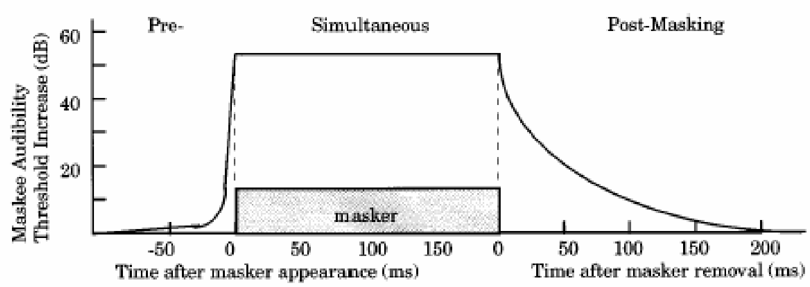


Figure 2.4 Temporal Masking Effect

The impacts of the threshold in quiet and masking threshold are illustrated below in figure 2.5a – 2.5g. Signal strengths below the threshold in quiet in figure 2.5b are not audible. Applying the masking threshold in figure 2.5c, the sinusoidal wave of 1KHz

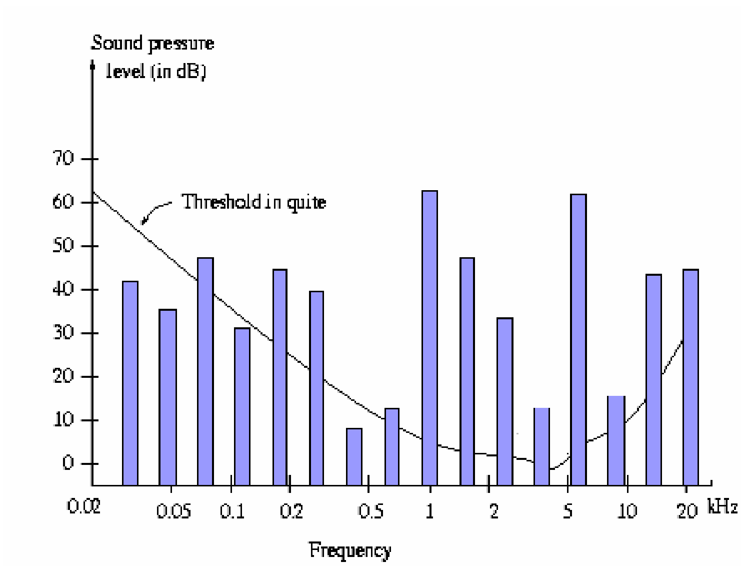


Figure 2. 5 a Quantized Values Without Adaptation

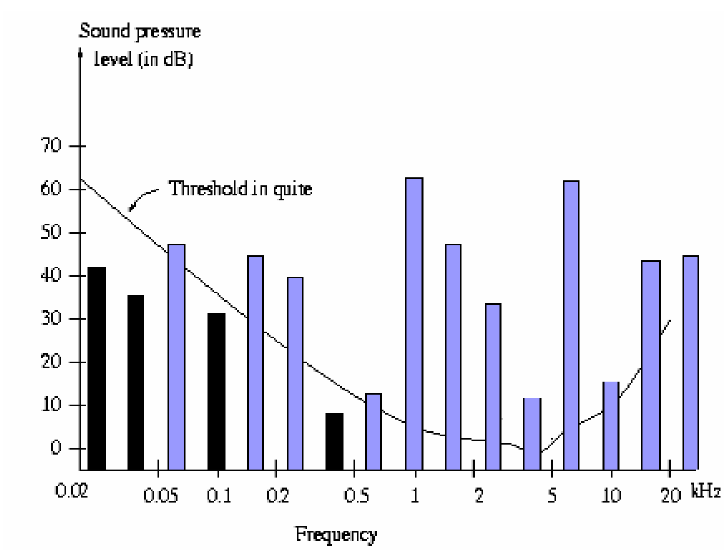


Figure 2.5 b Quantized Values with 25% Adaptation

and signal strength of 60dB masks the neighboring signals of lesser signal strength of about 20 – 30 dB. With another signal strength of 60dB at 6KHz, more signals are masked. With another signal strength of 60dB at 6KHz, more signals are masked.

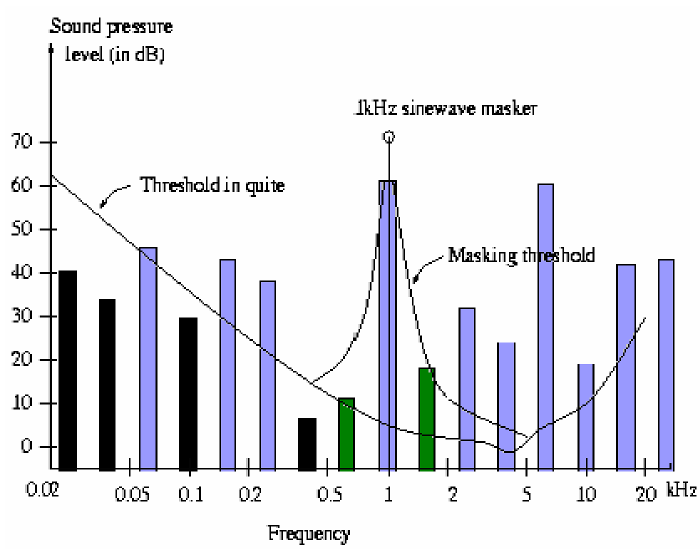


Figure 2.5c Quantized Values with 37.5% Adaptation.

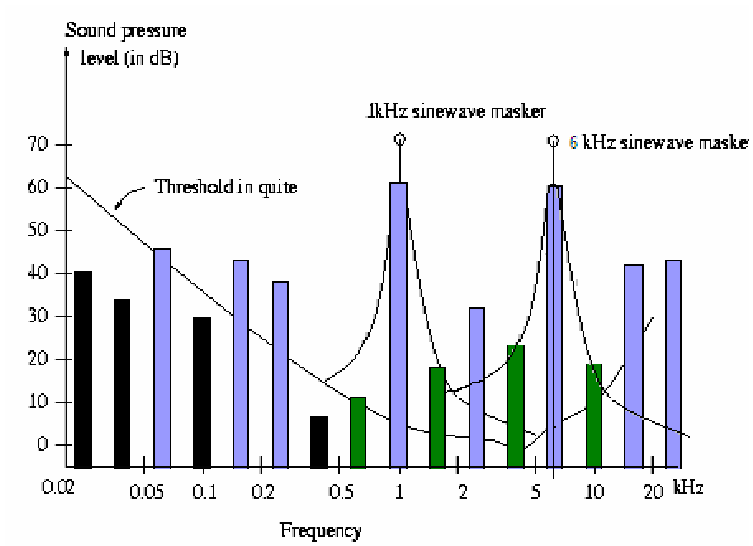


Figure 2.5d Quantized Values with 50% Adaptation.

Figure 2.5e shows the final output of signals that are encoded,

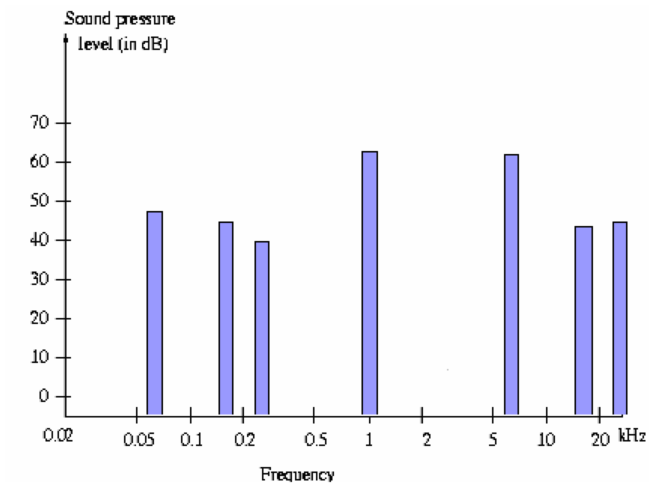


Figure 2.5e Impacts of Threshold in Quiet and Masking Threshold.

Figure 2.5f illustrates multiple signals in 3-dimension display.

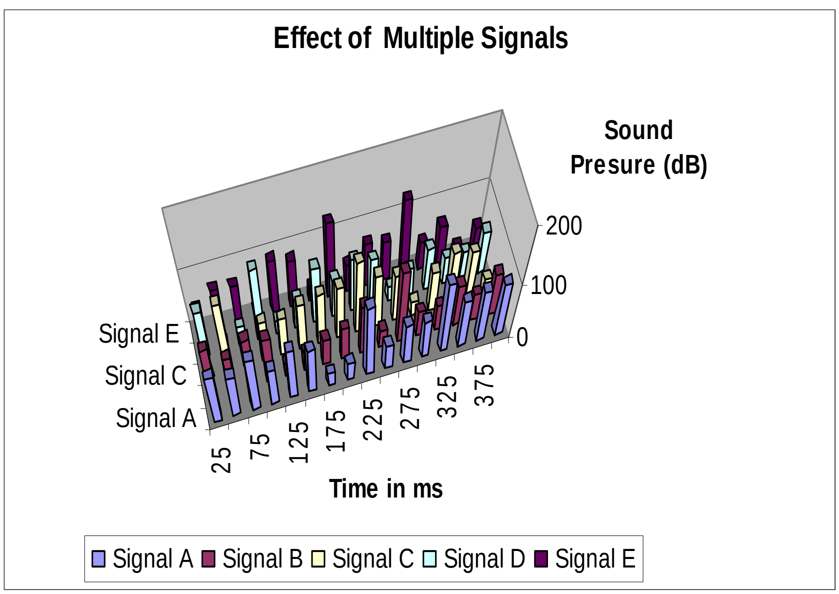


Figure 2.5f Multiple Signals Without Adaptation.

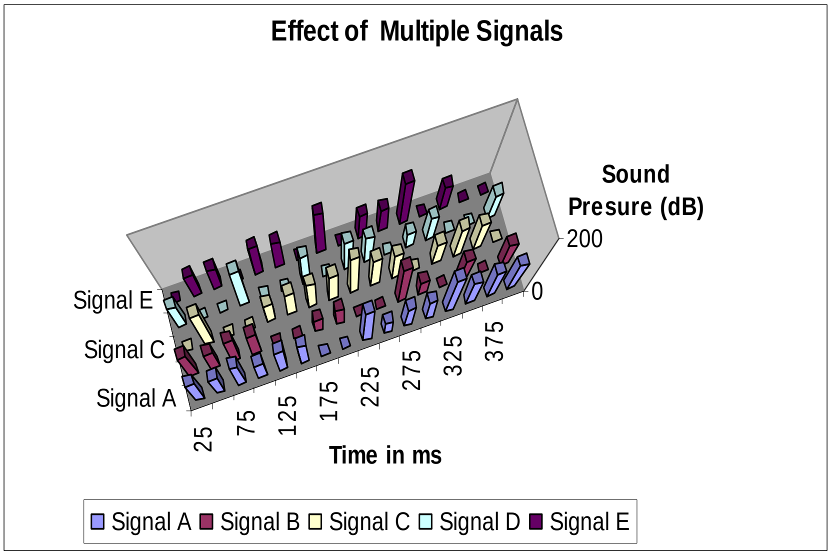


Figure 2. 5g Effect of Threshold in Quiet and Masking Threshold on Multiple Signals.

2.2 Audio Adaptation Scheme

The encoder comprises of the following blocks; the hybrid filter block, the psychoacoustic model block, noise allocation loop block and bitstream formatting block as illustrated in figure 2.6

The input audio signal simultaneously passes though the hybrid filter bank and psychoacoustic model. The hybrid filter bank block contains the Polyphase filter bank and MDCT (Modulated Discrete Cosine Transform) filter bank. The hybrid filter bank divides the input signal into chunks of 576 samples called granules and 32 sub bands of frequencies. The hybrid filter bank block provides a specified mapping in time and frequency, which allows a better time resolution for transients. However, due to the partitioning of the signal into granules, a granule that is originally silent could contains a sudden loud signal (attack) termed pre-echo.

If a granule contain pre-echo, three short time windows are used with six samples per subband. Three MDCT are applied to the window values, giving three time windows of 192 frequency samples.

If there is no pre-echo in a granule, one long time window is used and one MDCT is applied to the window, giving 576 frequency samples output.

The combined output of the entire subband forms either one long window of 576 frequency samples or three short time windows of 192 frequency samples.

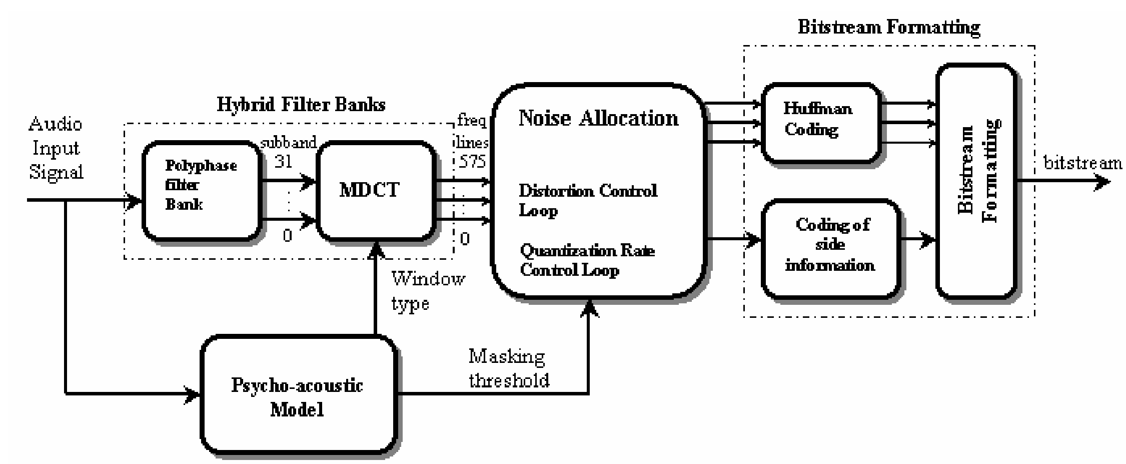


Figure 2.6 Block Diagram of the MP3 encoder

The psychoacoustic model is a mathematical model of the masking behavior of the human auditory system that generates a set of data to control the quantizers and coding.

It computes a just noticeable noise level in each subband in the filter bank and determines the ratio of signal energy to the masking threshold for each subband. The human auditory system is frequency dependent, such that the masking threshold at any given frequency is dependent on the signal energy within a limited bandwidth neighborhood of that particular frequency. Therefore the masking ability is dependent on its frequency position and loudness.

The psychoacoustic model passes the audio signal through a high pass filter FFT (Fast Fourier Transform) that helps to detect a sudden transient in the signal and achieve a finer frequency resolution that accurately calculates the masking threshold.

The Psychoacoustic model determines the block (window) type, long or short.

The Psychoacoustic model runs two computations in parallel, one for short window and the other for long window. It computes the energy in each partition band (threshold calculation partitions band) using table 2.1 and convolves the partitioned energy by applying the spreading function to partitioned band energy. The parameter bval is the bark value, qthr is the threshold in quiet, norm is the normalizing constant for each sub band and the strength of masking is limited by minval.

However, for the long window, in addition, it also computes the tonality of the signal. The signal to noise (SNR) ratio is calculated for the long window while the short window is read from a table. It then computes the threshold for each partition. For the signals that contain maskers, it adds up the simultaneous masking thresholds to form a combined/global masking threshold.

no.	FFT-lines	minval	qthr	norm	bval
0	1	24.5	4.532	0.951	0
1	1	24.5	4.532	0.7	0.431
2	1	24.5	4.532	0.681	0.861
-	-	-	-	-	-
-	-	-	-	-	-
5	1	20	0.09	0.665	2.153
6	1	20	0.09	0.664	2.584
7	1	20	0.029	0.664	3.015
-	-	-	-	-	-
12	1	18	0.009	0.578	5.057
-	-	-	-	-	-
15	2	12	0.018	0.856	6.422
16	2	6	0.018	0.846	7.026
-	-	-	-	-	-
60	36	0	32.554	0.483	23.897

(a) --- (44.1 KHz sampling rate – long)

no.	FFT-lines	qthr	norm	SNR (db)	bval
0	1	4.532	0.952	-8.24	0
1	1	0.904	0.7	-8.24	1.723
2	1	0.029	0.681	-8.24	3.445
-	-	-	-	-	-
-	-	-	-	-	-
5	1	0.009	0.665	-8.24	7.609
6	1	0.009	0.664	-8.24	8.71
-	-	-	-	-	-
-	-	-	-	-	-
12	1	0.009	0.578	-7.447	13.21
13	1	0.009	0.541	-7.447	13.748
14	1	0.009	0.575	-7.447	14.241
-	-	-	-	-	-
-	-	-	-	-	-
37	7	6.33	0.57	-5.229	23.828

(b) --- (44.1 KHz sampling rate – short)

Table 2.1 Threshold calculation partitions

Some constants (2, 16) are applied to control pre-echo. The threshold is compared with the last threshold and the threshold in quite and the maximum is taken. The threshold calculation partition is converted directly to the scalefactor bands as shown in table 2.2

no. sb	**cbw**	**bu**	**bo**	**w1**	**w2**
0	3	0	4	1	0.056
1	3	4	7	0.944	0.611
2	4	7	11	0.389	0.167
-	-	-	-	-	-
-	-	-	-	-	-
5	1	17	18	0.861	0.917
6	3	18	21	0.083	0.583
-	-	-	-	-	-
-	-	-	-	-	-
12	4	36	40	0.18	0.1
13	3	40	43	0.9	0.468
14	3	43	46	0.532	0.623
-	-	-	-	-	-
-	-	-	-	-	-
20	2	59	61	0.278	0.96

(a) --- (44.1 KHz sampling rate – long)

no. sb	**cbw**	**bu**	**bo**	**w1**	**w2**
0	2	0	3	1	0.167
1	2	3	5	0.833	0.833
2	3	5	8	0.167	0.5
-	-	-	-	-	-
-	-	-	-	-	-
5	5	15	20	0.833	0.25
6	3	20	23	0.75	0.583
-	-	-	-	-	-
-	-	-	-	-	-
9	3	30	33	0.625	0.3
10	3	33	36	0.7	0.167
11	2	36	38	0.833	1

(b) --- (44.1 KHz sampling rate – short)
Table 2.2 Calculation partitions to Scalefactor band

There are 21 sub bands for long window and 12 sub bands for short window, (cbw) is the number of partitions converted to one scalefactor band (excluding the first and last partition). The first partition added to the scalefactor band is weighted with w1 and the last partition by w2, “bo” and “bu” are the first and last index values of cbw.

Then, the signal to mask ratio is calculated.

The noise allocation block uses the output of the psychoacoustic model, “signal to mask” ratio to determine how to allocate the number of available codes to quantize the signals, which minimizes the audibly of the quantization noise.

Within a given block, bit-rate and output from the psycho-acoustic model, the noise allocation block uses two nested iteration loops, distortion and quantization control loop to find the best gain and scalefactors in an analysis-by-synthesis way. It iteratively varies the quantizer in an orderly way, counts the number of Huffman code bits required to code a frame and calculates the resulting noise.

It compares the distortion in each scalefactor band with the allowed distortion computed by the psychoacoustic model, if there are scalefactor band that are more than the allowed distortion, the noise allocation block amplifies the values in those scalefactor bands by decreasing the quantizer step size for those bands.

Once the best quantization setting and the bit rate are achieved, the quantized values are coded by Huffman coding to further reduce their space requirement.

The bitstream formatting block assembles and formats the frames. It formats the quantized subband signal using Huffman code and other side information into bitstream. This is then passed into the network for transmission.

Chapter 3

TCP Congestion Control

Congestion is caused as a result of excessive traffic in the network. The receiving buffer of a router stores packets in the queue for transmission. Provided there is still available storage in the buffer, the router continues to store data until it exceeds its capacity and the subsequent packets are discarded. As a result of the packet loss, a need for retransmission emerges which would reduce the throughput and eventually may lead to a congestion collapse. A practical resolution is the congestion control by TCP.

TCP provides a connection-oriented, reliable delivery of data streams between two applications or hosts. TCP maintains the advertised window flow control and congestion window for congestion control. The advertised window also called the receiver window; uses acknowledgment (ACK) packet to determine the byte size the receiver can receive. Congestion window reflects the dynamic state of the network; it is the maximum bytes the sender sends without introducing congestion into the network. TCP uses timeout and duplicate ACKs to detect packet loss in the network; and if it occurs, it reduces the size of congestion window.

3.1 Congestion Control Algorithms

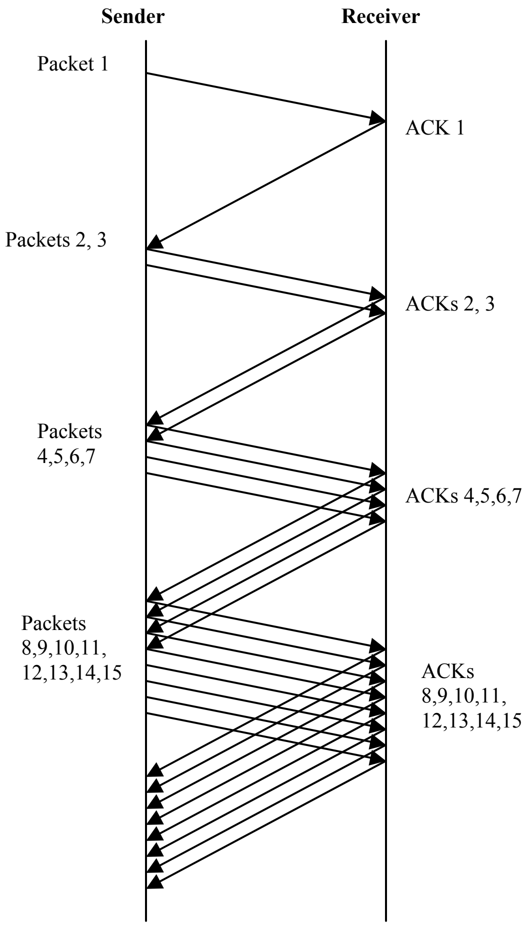
Essentially, there are four congestion control algorithms, however they are interconnected. They are the slow start, congestion avoidance, fast retransmit and fast recovery.

Slow Start and Congestion Avoidance Algorithm

The slow start algorithm helps the network to quickly utilize the available network bandwidth without introducing congestion collapse. It uses slow start when starting a TCP connection and when connection goes down waiting for a timeout. It also uses congestion window and slow start threshold. It starts with a congestion window of one maximum size segment (1 MSS) and doubles the congestion window at each roundtrip time (RTT) as the acknowledgments are received. It increases the congestion window exponentially (i.e. 1,2,4,8 etc).

As illustrated in figure 3.1, it sends packet 0, waits for acknowledgement 0 and increases the congestion window by 1. It sends packet 1 and 2, wait for acknowledgement 1 and 2 to arrive, increase congestion window, then it sends packet 3, 4, 5 & 6 and it continues.

As the congestion window approaches its threshold, the slow start algorithm stops and congestion avoidance, an additive increase multiplicative decrease (AIMD) scheme kicks in. It uses one segment per RTT. When the sender does not receive an acknowledgment (indication of loss of a segment) and there is timeout, TCP retransmits the packet. The congestion window is reduced to half and the slow start kicks in again. Figure 3.1 illustrates the significant of the slow start/congestion avoidance (SSCA) mechanism.



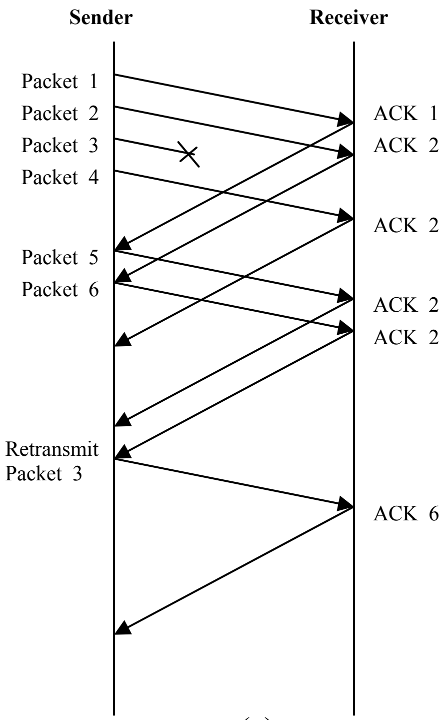

```
initially, cwnd = segmentSize (1 segment)
ssthresh =65535 bytes
win_size = min(cwnd, snd_wnd)
When congestion occurs, do:
      ssthresh = max(win_size/2, 2);
      if congestion was due to timeout
         cwnd = segmentSize;
      for every ACK received;
            if (cwnd <= ssthresh);
               cwnd -= 1;
            else
               cwnd -= 1/cwnd;
```

(a) (b)

Figure 3.1 Slow Start/Congestion Avoidance (SSCA) mechanism.

Fast Retransmit and Fast Recovery Algorithm

Fast Retransmit is another algorithm that is introduced by TCP to quickly reduce the idle period of waiting for time-out to occur when a segment is lost. The fast retransmit uses duplicate acknowledgement to trigger retransmission without waiting for time-out event to occur. If a receiver receives a data segment that is out of order, it sends duplicate acknowledgment to the sender. If the sender gets three duplicate acknowledgments, it assumes that the data segment is lost and it immediately retransmits the lost segment without waiting for a time-out event to occur. This speeds up the recovery of a lost segment. As illustrated in figure 3.2a, packet 3 is lost but the receiver sends ACK 2 to the sender. The sender sends packet 5 and 6; again, ACK 2 is send to the sender.



(a)

```
When a 3rd duplicate ACK is received
   ssthresh = max (2, min(cwnd, snd_wnd) /2)
   Retransmit missing segment
   cwnd = ssthresh +3;
 Each time another duplicate ACK arrives, do:
   cwnd = cwnd + segment_size;
   transmit a new segment;
 When a new ACK arrives, do:
   cwnd  = ssthresh:
```

(b)

Figure 3.2 Fast Retransmit/Fast Recovery (FRFR) mechanism

On receiving the 3rd acknowledgement, fast retransmit algorithm retransmits the packet 3 without waiting for time-out event to occur and the receiver sends ACK 6 to the sender thereafter.

In fast retransmit algorithm, TCP eliminate the slow start algorithm phase and moves directly to the congestion avoidance phase. Figure 3.2b shows the fast retransmit and fast recovery (FRFR) mechanism. When a 3rd duplicate ACK is received, the slow start threshold (ssthresh) is set to half the size of the current congestion window (cwnd) but not less than 2 segments. It then retransmits the missing segments. It sets the congestion window to (ssthresh + 3 segment). This reduced the TCP widow size to half and helps to compensate for the TCP segments that have already been buffered by the receiver. Each time another duplicate ACK is received, which indicates that the receiver has buffered one more segment, TCP increases the congestion window to compensate for the delivered segment. Eventually, the lost segment is retransmitted, the receiver sends a new ACK to the sender. The retransmitted segment and all the buffered segments are passed to the application layer. TCP acknowledges all the segments that it had buffered.

The congestion window is then set to slow start threshold.

3.2 Congestion Control Events

Some of the interesting congestion control events that occur when TCP session invokes a congestion control algorithm are shown in Table 3.1, a tabular and graphic view of explanation of session 3.1. The column sub indicates the events that are subscribable and are invoked. Event 1,2 and 3 in table 1 shows the event in a slow start/congestion avoidance algorithm. Event 4, 5 and 6 occur in a Fast Retransmit/Fast Recovery algorithm. Interactive transport control protocol (iTCP) that is further explained in chapter 4 subscribes to event 1, 4 and 6. Also figure 3.3a and 3.3b illustrate the effective bandwidth changes that occur due to TCP congestion control event.

Event	Denotation	Explanation	SSCA	FRFR	Sub
1	Retransmission time out	Congested network / Lost segment.	✓		✓
2	New ACK received	Increment snd_cwnd exponential or linearly.	✓		
3	snd_cwnd reached the slow start threshold ssthresh	Switch snd_cwnd increment from exponential to linear.	✓		
4	Third duplicate ACK received	Lost segment, execute fast retransmit.		✓	✓
5	Fourth (or more) duplicate ACK received	A segment left the network; transmit a new segment.		✓	
6	New ACK received	Retransmitted segment arrived at the receiver and all out of order segments buffered at the receiver are acknowledged		✓	✓

Table 3.1. TCP Congestion Control Internal Events

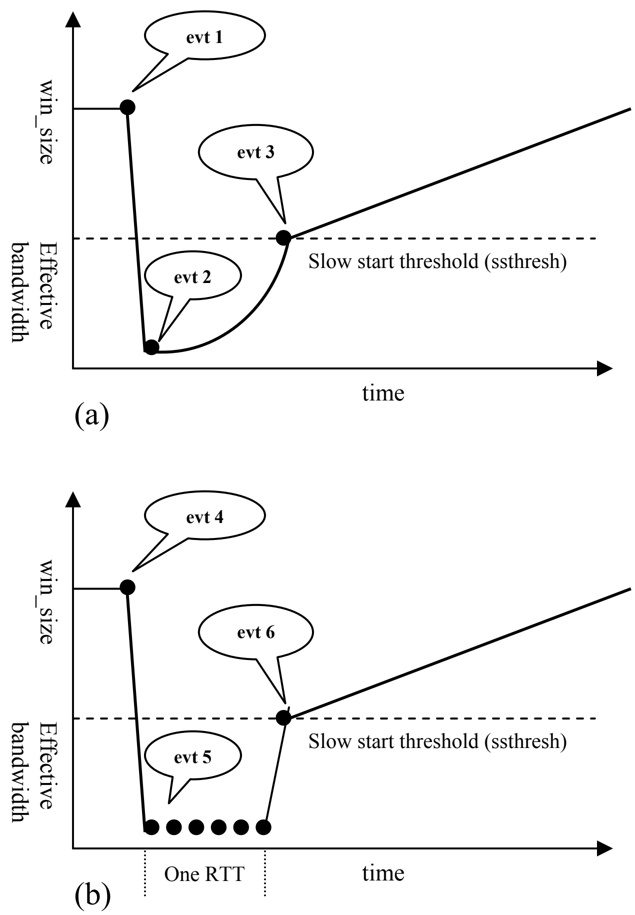


Figure 3.3 Effective bandwidth changes due to TCP congestion control internal events

Chapter 4

iTCP: interactive TCP

4.1 Model Architecture

The transport service protocol TCP does not provide a global view of the network or system. However, the introduction of interactivity between the higher (application) layer and the lower (transport) layer can provide a means for the application to receive notification about the state information and status by subscribing to a range of available transport service level events.

In this chapter, a concise description of the framework of previous work on interactivity is outlined; more detailed explanations can be found in [14, 15. 16]

ITCP is modeled similarly to the conventional TCP extended finite state model (EFSM) aside from the added option of subscribing to a set of accessible events.

To subscribe to an event means an application connects itself to the network protocol to equally receive information from the lower layer when an event or state changes in that level. The information received is in the form of signals, which can be acted upon by Transient modules called the T-ware of the application layer. T-wares are executable programs for managing events. The architectural diagram of the iTCP protocol is illustrated in figure 4, which shows the iTCP extension and API. It consists of three sections, the user space, the system and the kernel.

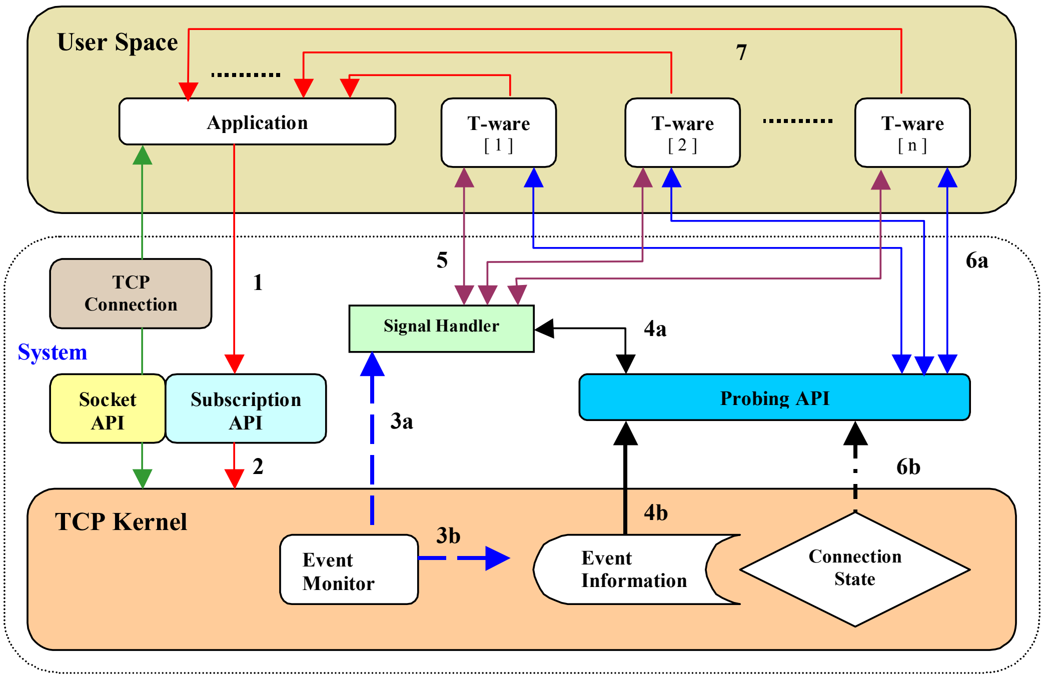


Figure 4. The iTCP extension and API.

Further to the explanation, the option to subscribe lies with the application, if that is chosen, the system acts in the interactive mode, however, if the application chooses not to subscribe, the system acts identical to the classic TCP mode.

After a connection has been established on both side of the sender and the receiver, each side binds to a specified port of the socket. An application that has subscribed to the kernel also binds a T-ware to a selected TCP event as shown by line 1 and 2 of figure 4. The application connects to the TCP kernel through the subscription API. When the TCP state changes as a result of congestion in the network, it also causes an event to occur in the TCP kernel. The event monitor is aware of the changes that occurred; thus, it responds instantly by sending a signal in (3a) to the signal handler and also stores the event information in (3b). The signal handler catches the signal from the kernel and requests the event type (4a, 4b) from the kernel via the probing API. The appropriate *Transientware Modules* (or *T-ware)* (5) is triggered to serve the particular event. Additional information about the TCP connection state (6a, 6b) is available through the probing API. Amongst the information from the TCP control block includes:

(i) *congestion-control window size*.

(ii) *send window size* (snd_wnd).

(iii) *retransmit value* (t_rxtcur).

(iv) *round-trip time* (t_rtttime).

(v) *slow-start* (snd_ssthresh).

These information are used by the T-ware to decide the appropriate action for the application such as reduce the target bitrate during congestion, which helps to continue transmitting the data with tradeoff of quality for delay.

4.2 Compatibility & Interoperability

The design of the Interactive TCP protocol is identical to the functionalities in TCP classic, keeping the important protocol principles of

(i) *network functional compliancy*

(ii) *state-transition compliancy, and*

(iii) *default-to-classical extension* interface model.

It is fully compatible, operational and interoperable with improved extended model. The state transition performance does not change other network embedded transport elements. Thus, interactivity provides a new approach of implementing advanced optimization in the network that would resolve various network problems by controlling the data transmission to guarantee minimum delay.

Chapter 5

Symbiosis Throttling Model

In an effort to reduce the delay and jitter problem of audio traffic in the networks by regulating the rate of bits flowing in the networks, a symbiosis throttling model is developed. The model dynamically controls the flow of data at the instance of an event notification. It uses a feedback mechanism from the underlying interactive transport layer to form corresponding bit rate information or data that is use for regulating bits formation in the encoder before transmission. The condition of the network is continually evaluated through the feedback received for any network changes.

The symbiosis throttling model functions like a valve that dynamically regulates the data transmission at an instance of an alert and the controlling function is the target bit rate. During congestion, which is detected by the time-out event ($\zeta = 1$), the model reduces the target bit rate to minimum rate for continuous data transmission with a precise audio quality/ delay trade off.

5.1 Analysis of Symbiotic Throttling Model

Let the target bitrate at normal network condition be b_{max}, when a time-out event ($\zeta = 1$) occurs due to congestion, the model retract to a non-zero minimum acceptable target bit rate b_{min}.

Under normal condition, the application generates data at maximum target bit rate while the condition of the network is continuously evaluated through the received feedback for any possible network changes. However, when congestion (loss event) occurs in the network as a result of heavy traffic in the network, packet loss or retransmission, it is essential for the target bit rate to change.

As explained in chapter 3, the limited receiving buffer of the application or a receiver soon gets filled up and the trailing packets are discarded. The discarded packets are not received at the application layer, which therefore calls for re-transmission that would probably add another level of delay to the network. To react to the loss event, the models change the initial target bit rate to a smaller one. Thus allowing the data traffic to continue to flow with trade off of quality for reduce delay.

Figure 5.1 and 5.2 explain the input and output target rate of the model.

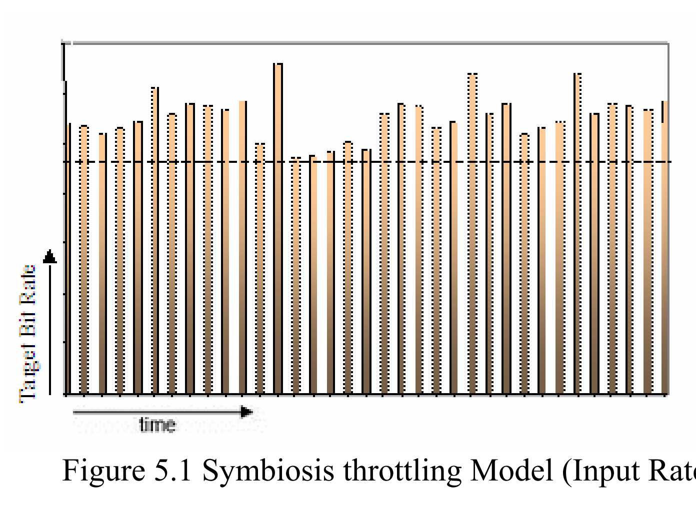


Figure 5.1 Symbiosis throttling Model (Input Rate)

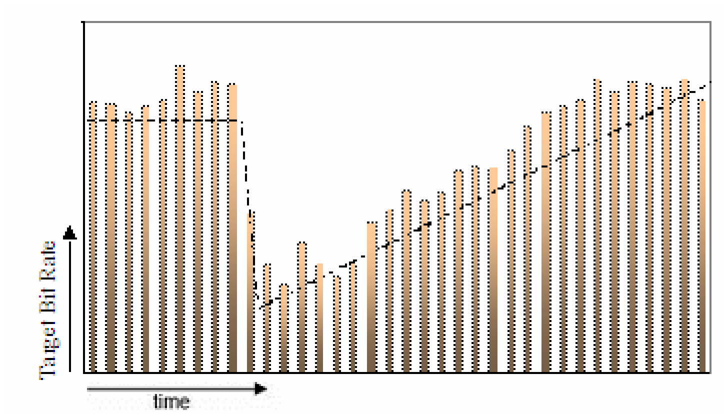


Figure 5.2 Symbiosis throttling Model (Output Rate)

Further explanation to this, the models works on the idea closely related to the TCP sliding window protocol, when congestion is detected the receiver sends a message to the sender to reduce its sending rate by half and by half again until there is continuous data flow. The iTCP transport control is similar to TCP (binary-exponential-back-off and additive – increase step of *b(t)*).

A reduction ratio called rate retraction ratio ρ is defined based on the acceptable level of quality.

$$\rho \;=\; \frac{b_{mim}}{b_{\max}} \tag{5.1}$$

Again, a function that links the underlying TCP with the model is defined as the "*running generation threshold*" g_T function.

$$\begin{aligned} g_T(t) &= \frac{1}{2} g_T\,(t-1) \quad when \quad \xi = 1 \\ &= g_T\,(t-1) \quad otherwise \end{aligned} \tag{5.2}$$

In the above equation, it illustrates that when there is congestion in the network detected by time-out event, "*running generation threshold*" retracts to half its current size, however normal condition, the *running generation threshold*" of the underlying TCP remain the same.

$$
\begin{aligned}
b(t) &= \rho \,.\, b_{\max} \quad when \quad \xi = 1 \\
&= 2 \,.\, b(t-1) \; when \quad \xi \neq 1, \;\; b(t) \leq \frac{1}{2} g_T(t-1) \\
&= \min[\, b_{\max} \;, b(t-1) + 1] \; when \quad \xi \neq 1, \;\; b(t-1) \leq g_T \;(t-1)
\end{aligned} \tag{5.3}
$$

As seen above, the "*running control generation function*" *b(t)* perform binary-exponential-backoff and additive increase within the limits given by the generation parameter ρ and the normal network condition target bit rate b_{max}. When there is a time-out event occurs, the symbiosis continues to reduce *b(t)* until it gets to the minimum target bit rate and then additively increase *b(t)* until the control target bitrate recovers to it initial normal target rate.

5.2 MPEG 1 Audio Rate Controller

At the beginning of encoding a frame, the target bitrate plays a key role in the output of bits that is delivered by the encoder. The encoder uses target bitrate to allocate the target bits per frame. Ideally, the target bitrate is set at the beginning of encoding the frames and does not change during encoding; as a result the output audio stream remains the same even when the state of the network changes.

However, if the bitrate were modifiable, the number of bits encoded would reflect the changes in the network status. In this thesis, the design of the adaptive encoder enables the output audio streams to reflect the network status or changes.

In audio compression, the quality of the audio stream is controlled by quantization with an additional support of the bit reservoir. These two key features play important role in achieving quality in audio.

The bit reservoir is a control feature that allocates bits during localized peak period of higher bit demand associated with transient or sudden attack. However, it can only borrow from the last frame that donated bits and not from a future frame, thus the bits are limited.

Quantization, on the other hand, is essential to lossy compression. It is a process of approximating the continuous set of values in the audio data to discrete set of values. Quantization depends on quantization step size, a difference between quantization levels. Quantization noise is the noise introduced to digital signal as a result of approximation of samples between quantization levels. Generally, a smaller step size means less quantization noise, and more quantization levels means smaller step size. The more steps in the quantization process, the more closely the digital information resembles the original audio signal.

In effect, the control function in (5.3) works closely with quantization to achieve the desirable audio output bitstream during the frugal state with acceptable quality.

5.3 Audio Rate Control Design

We explain the rate control design in figure 5.3. Adaptive output audio stream in the encoder is achieved through the rate retraction ratio ρ applied to the target bitrate.

At the start point, the target bits/granule is determined. Each frame contains two granules. The target bits/granule *T(t)* is derived from bitrate, sampling frequency and side information and header.

$$
\begin{aligned}
T(t) &= ((b_{\max}\ .\ 1152/\ freq) - Z)/2 \quad when \quad \xi \neq 1 \\
&= ((\rho\ .\ b_{\max}\ .\ 1152/\ freq) - Z)/2 \quad when \quad \xi = 1
\end{aligned}
\tag{5.4}
$$

where *Z* is the conveying information about the elementary stream data contained in the packet data. *freq* is the frequency at which the audio is sampled. *T(t)* is the target bits that the allocated bits determined for a granule before encoding. When a time-out event is detected and *T(t)* is recalculated to reflect the change in network state.

max-bit is the needed bits based on the perceptual energy, which is the summation of a fraction of the target bits per granule and the current reservoir size. *max-bit* is referred to as *estimated target buffer fullness (ETBF)* in this thesis.

$$
ETBF = 0.9 * T(t) + R_s \tag{5.5}
$$

As the target number of bits decreases, the *estimated target buffer fullness (ETBF)* also decreases.

The reservoir size, R_s is updated after each frame is encoded. A is the actual number of bits encoded per granule.

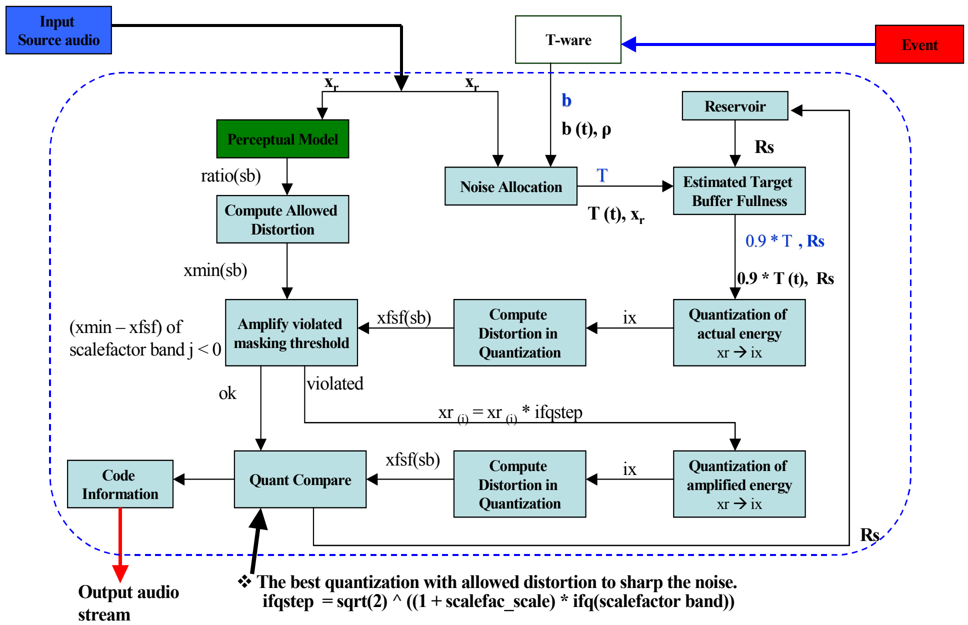


Figure 5.3 Rate Control of Encoder

$$R_s = T - A \tag{5.6}$$

The noise allocation block in figure 5.3 uses the output of the psychoacoustic model, “signal to mask” ratio (ratio of threshold to energy) to determine how to allocate the available number of codes to quantize the subband signals, thus minimizing the audibility of the quantization noise.

As explained in Chapter 2, it uses two nested iteration loops, distortion and rate control loop to find the optimum gain and scalefactors for a given bit-rate and output from the psycho-acoustic model in an analysis-by-synthesis way. It iteratively varies the quantizer in an orderly way, counts the number of Huffman code bit required to code a frame and calculates the resulting noise.

The outer iteration loop controls the masking conditions of all scalefactor bands. It computes the best scalefactor and global gain through the Quant-compare in figure 5.3. The first step of the outer loop is the inner iteration loop (rate control loop), which is responsible for quantizing input vector of spectral data (audio signal energy $xr_{(i)}$) to $ix_{(i)}$ as shown in equation 5.7. It increases the quantizer step size. The outer loop then checks the distortion of each scalefactor band, if the allowed distortion is violated, it amplifies the scalefactor band and invoke the inner loop again until the overall bit sum is less than the available bits to encode a frame. It outputs the number of bits that the Huffman code length table uses to code information.

$$ix_{(i)} = \left(\frac{(\ |xr_{(i)}|\ )^{0.75}}{\sqrt[4]{2^{global\,gain}}} - 0.0946\right) \tag{5.7}$$

The *global gain* is summation of *qquant* (counter for the quantizer step size) and *quantanf,* system-constant for all the scaling operations of the encoder and the offset to achieve the correct output with decoding process.

The distortion *xfsf(sb)* within each scalefactor band in this quantization is then computed

$$xfsf(sb) = \sum_{lbl(sb)}^{i=ibl(sb)+bw(sb)-1} \frac{(|xr_{(i)}| - ix_{(i)}^{4/3} * \sqrt[4]{2^{qquant+quan\tan f}})^2}{number\ of\ (sb)} \tag{5.8}$$

The *lbl(sb)* is the number of the component representing the lowest frequency in a scalefactor and *bw(sb)* is the number of component within this band.

The computed distortion in the quantization *xfsf(sb)* of a particular scalefactor band j is compared with the allowed distortion *xmin(sb)*.

$$xmin(sb) - xfsf(sb) \text{ of scalefactor band } j < 0 \tag{5.9}$$

$$\text{where } xmin(sb) = ratio(sb) * en(sb)/bw(sb) \tag{5.10}$$

The *ratio(sb)* is ratio of threshold to energy in each scalefactor band and *en(sb)* is the energy in each scalefactor band.

If the values computed distortion *xfsf(sb)* is greater than the allowed distortion *xmin(sb)*, the scalefactor band is amplified by a factor *ifqstep* and the quantizer step size is increased.

$$xr_{(i)} = xr_{(i)} * ifqstep \tag{5.11}$$

$$\text{where } ifqstep = 2^{(0.5 * (1 + scalefac - scale))} \tag{5.12}$$

The *scalefac* and *scale* in (5.12) are reference from a table.

When a scalefactor band is amplified, it makes the next quantization to use more bits for that band.

It quantizes the new scalefactor, global gain and computes the distortion. It continues with different scalefactor, global gain until the best quantization with allowed distortion to sharp the noise is achieved determined by the quant-compare.

The best quantized values are coded using Huffman code and other side information into bit streams. With the influence of the rate retraction ratio ρ and quantization, the final audio output stream is adaptive.

Chapter 6

Symbiosis Mechanism: The *T-ware*

The control mechanism used to reduce delay plays a key role during congestion in the network. The mechanism used in this thesis is called *T-ware*. It gives the transport layer the ability to communicate with the application layer (encoder). The key element is the loss event handler that generates a signal and the signal information is used to probe iTCP service. Couple with the retraction ratio ρ, the rate is reduced to achieve the objective of the thesis.

6.1 *T-ware* Implementation

As mention, the backbone of the loss event notification is the signal handler, which is triggered by the "retransmit timer out" event.

The purpose of the signal handler is to catch a signal from the kernel. As explained earlier, the signal handler plays an important role of the loss event notification for the symbiosis throttling model in chapter 5. The signal handler notifies a process of a particular event that has occurred in the kernel and which causes an interrupt. However, in this thesis, the signal handler catches the signal and invokes the T-ware. Particularly, the "retransmit timer out" event that indicates a time-out in the transport control protocol is the focus of this thesis and as for the encoder, it only subscribes to this event.

Figure 6-1 illustrates the pseudo code of the signal handler. When the signal handler catches the signal for “retransmit timer out” event and the event control flag is enabled, the socket ID is retrieved in (line 4). (line 5) probes the socket for the event number and the event handler name. Also, in (line 6), a timer is set for restoring the encoder target rate to normal after a reduction due to the retraction ratio in (line 8).

```
1.   SignalHandler (signum) {
2.      struct evtSubInfo * handInfo;
3.      if  ((signum == SIGIO) && (EVENT) {
4.          s = GetSockid ( );
5.          ProbeEvtInfo (s, handInfo);
6.          StartRecover ( );
7.          if ( ! ( child = fork (  ) ) ) {
8.              execl (handInfo → handler, retraction-ratio);
9.              exit ( 0 );
10.          }
11.     }
12.  }
```

(a)

Figure 6-1. Pseudo code of the Signal Handler

When the event number and event handler name are retrieved, it forks a new child (line 7). (Line 8 –10) executes the T-ware, which couples the retraction ratio to calculate a new reduced target bitrate. The new reduced rate is stored in “rate.par” file, the symbiosis encoder senses any change in the “rate.par” immediately. Fig 6-2 also shows the pseudo code of a recovery handler, whose purpose is to signal an alarm to restore the encoder bitrate in an addictive increment scheme until it reaches the original rate before congestion occurred. When the signal alarm is notified (line 2). When the waitTime is equal to the predefined value (line 4), the recovery *T -ware* kicks-in. The rate retraction rate (ρ) is multiplied by a factor of 1.2. The new rate is computed in (line 6) using the current rate. (Line 7 –10) determines the new current rate, which written to the rate file “rate.par”.

```
1.   Recovery –T-ware (signum ) {
2.      if ( signum ==  SIGALRM) {
3.          waitTimecount ++;
4.          if  ((waitTimecount == 400) || ((waitTimecount > 400) &&
                 ((waitTimecount % 300) == 0))) && (rateOK == 0) {
5.                  if (waitTimecount > 400)
                       A =  ρ  * 1.2;
6.                 newRate = currentRate /A;
7.                 if (newRate > = originalRate) {
8.                    newcurrentRate = originalRate;
                      rateOK = 1;
                     }
9.                    else
10.                   newcurrentRate = newRate;
11.                ratefile = fopen ( “rate.par”, “w”);
12.                fwrite ( newcurrentRate, ratefile);
13.                }
14.         }
15.     }
```
(b)

Figure 6-2. Pseudo code of the Recovery handler

Chapter 7

Experiment and Performance Analysis

The experiment performed in this chapter measures the effect of delay and jitter on audio samples in three modes of operation of TCP. The results further prove the effective performance of the Symbiosis throttling model implemented on the MPEG-1 layer III encoder to minimize congestion by reducing the target bit rate.

Delay is the time it takes a signal to be transmitted from one end to another. Jitter is the variation of time between packets.

7.1 Experiment Setup

The experiment layout includes an interactive TCP transport on FreeBSD system, the symbiotic encoder runs on the FreeBSD system and player on a remote host. The player runs on Linux/Windows platform. Figure 7.1 illustrates the experiment setup.

The audio samples are encoded with base frame rate of 128kbps. The player initiates communication with encoder and in return, the encoder sends audio stream to the player. Artificial interruption is induced into the network between the encoder and the player by the congestion injector to demonstrate congestion in the network for 3 seconds. The interrupt creates congestion bursts, which triggers 1 to 2 retransmit time out events according to the location of the player. The experiment is repeated with three types of audio sound quality to a listening ear: high quality music (HighQmusic), low or poor quality music (LowQmusic) and speech mixed with music (SpeechMusic). The experiment data (frame arrival, packet size and file size) are retrieved from the player and encoder and are processed. The three running modes of operation considered in this thesis are the iEXP, iOFF and Classic mode as described in Table 7.1.

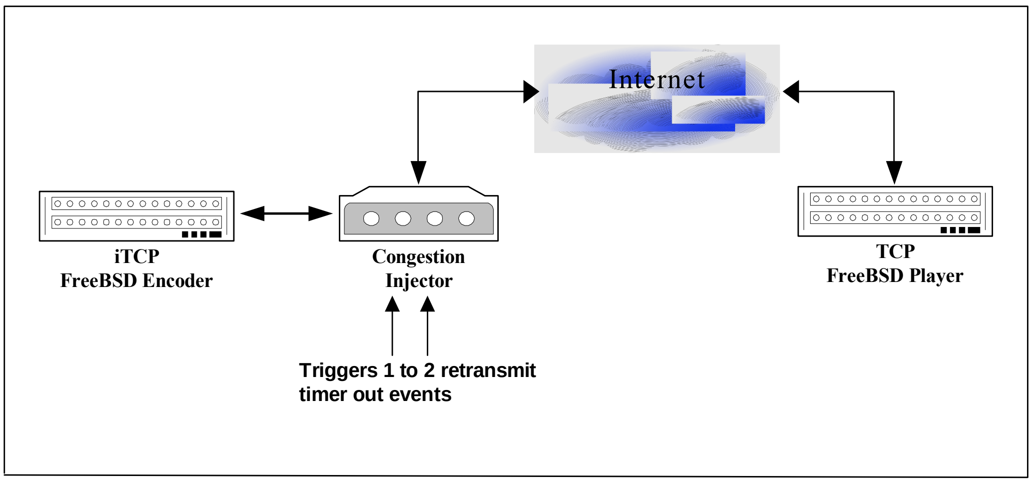


Figure 7.1 Experiment setup

In the iEXP (exponential backoff) mode, the interactivity service, the event subscription notification service and symbiosis rate adaptive feature are enabled to maximize the full interactivity of the symbiosis throttling. In the iOFF mode, the interactivity service, the event subscription notification service is enabled but symbiosis rate adaptive feature is disabled (no target bit rate reduction) to illustrate the overhead of the event processing. The audio encoder subscribes with iTCP to get retransmission time out event. Whereas, in the Classic mode, the added services of interactivity service, the event subscription notification service and symbiosis rate adaptive feature are all disabled to demonstrate the normal TCP mode.

Running mode	Control Flag			Notes
	iTCP	**EVENT**	**SYMB**	
	Enable / Disable Interactivity Service	Enable / Disable Event Notification Service	Enable / Disable Symbiosis Feature of Encoder	
iEXP	**ON**	**ON**	**ON**	Maximum interactivity
iOFF	**ON**	**ON**	**OFF**	Interactivity without bit rate modification. To measure overhead
Classic	**OFF**	**X**	**X**	Disable all interactivity features.

Table 7.1 Experiment control flags and running modes

7.2 Effect of Frame Delay on Audio.

As explained above, the parameters used in the experiments are the:

(i) *rate retraction ratio* (ρ = 0.50) and

(ii) bit rate.

The experiment is carried out in the iEXP mode and repeated with the same congestion pattern in the classic TCP mode to demonstrate the normal TCP mode. The experiment is also repeated in the iOFF mode for overhead estimation.

As the experiment runs, frames information are collected for the first 800 up to 1200 audio frames of the encoder and the player. At the player end, frames that are successfully received and not received are monitored.

As explained earlier, delay is the time it takes a frame to be transmitted from one end to another; the ideal expected time for the frame to arrive is called "*Expected*". The experiment is preformed several times of the day; however snapshots of each audio quality are taken as shown in figure 7.2 to illustrates the delay experienced by the audio frame in the iEXP, iOFF, classic and "Expected" modes.

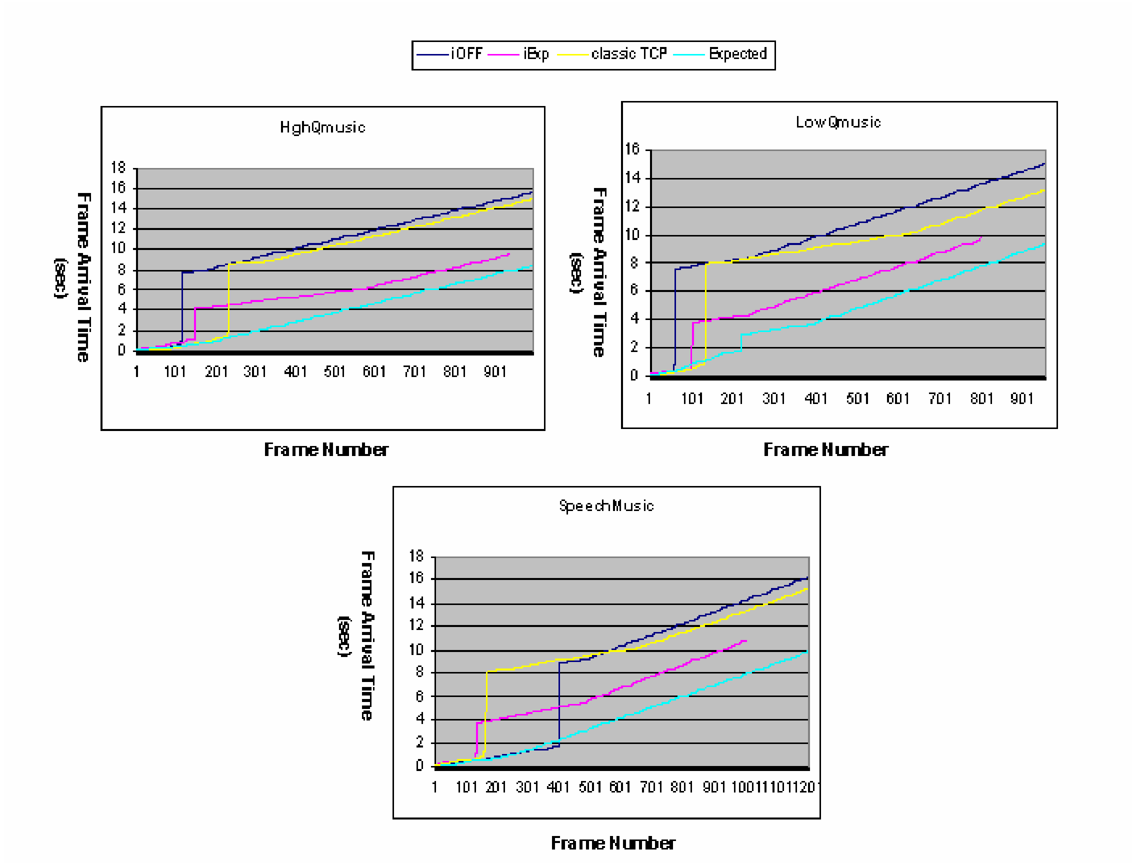


Figure 7.2 Frame arrival delay on the three audio qualities

Congestion interrupts are artificially induced into the network as closely as the same time for each run of experiment. The "Expected" mode is the ideal network state but, the slight jump in the "Expected" mode of the LowQmusic could be attributed to the state of the network at the time the experiment was performed. From the plot, the performance of iEXP is better than the classic TCP.

The delay buildup in Classic and iOFF are much higher than that experienced by the iEXP (iTCP) as indicated by the step jump. Looking closely, the step jump of iEXP is much smaller as a result of the rate retraction ρ and it was able to recover from delay buildup in few seconds compare to the Classic and iOFF.

7.3 Effect of Jitter on Audio

Jitter is the variation of time between packet arriving due to congestion or change of route in the network.

Assuming the packet $j = 1, 2, 3 \ldots n$ of the same size are transmitted over a network. During congestion, packet arrives at the receiver end at time *tj*. If the expected arrival time for the same packets j under normal condition of the network is *ej*. Referential jitter *refJitter(j)*, for packet *j*, *j* is calculated as the difference between *tj* and *ej*.

$$refjitter(j) = tj - ej \tag{7.1}$$

The *refjitter(j)* is negative if packet *j* arrives early at the receiver end and *refjitter(j)* is positive, if packet *j* has arrived late at the receiver end. In as such, packets that arrive early can be buffer and then played by the player at the actual time while the player will pause and wait for packet that arrive late. Figure 7.3 shows the corresponding reference jitter experienced by the three types of audio sound: high quality music (HighQmusic), low quality music (LowQmusic) and speech mixed with music (SpeechMusic).

The difference between the expected ideal arrival time and the actual arrival time for each frame is recorded.

The higher the delay, the higher the jitter experience by the frames. As in the case of delay, it is also observed that the step jumps in the iEXP for the three audio samples were much smaller than those in TCP-classic and iOFF for the same reason that delay in iEXP were smaller. Furthermore, this indicates that the interactive TCP reduces jitter.

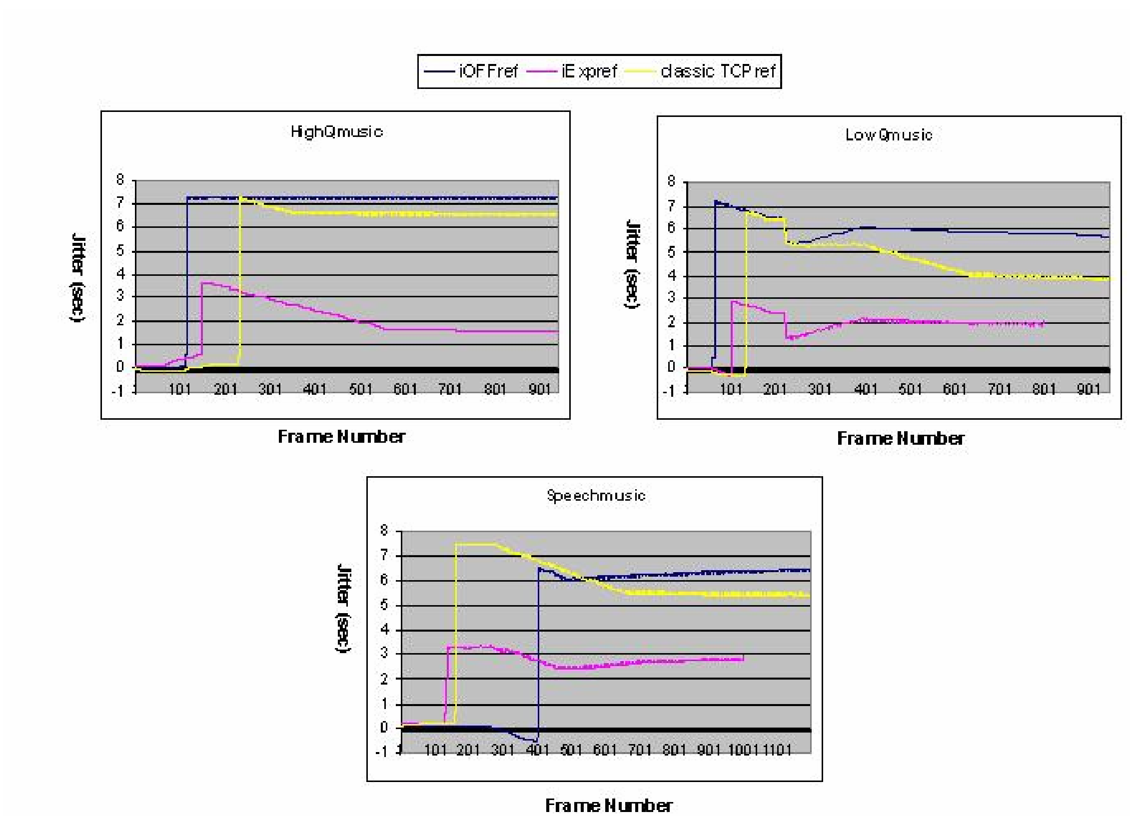


Figure 7.3. Referential Jitter on the three audio qualities

7.4 Symbiotic Rate Control

Figure 7.4 shows the plot of symbiotic rate reduction that occurred as a result of the rate modification between the rate controller of the encoder and the symbiosis unit in three types of audio sound: high quality music (HighQmusic), low quality music (LowQmusic) and speech mixed with music (SpeechMusic). This figure displays the target bits and the actual bits generated by the encoder for each frame. When a time out event ($\zeta = 1$) is triggered, the rate retraction ratio of the symbiosis kicks in and the target bit rate is reduced to the minimum target bit rate. The effect is observed on the plots as number of bits drop in accordance with the rate retraction ratio.

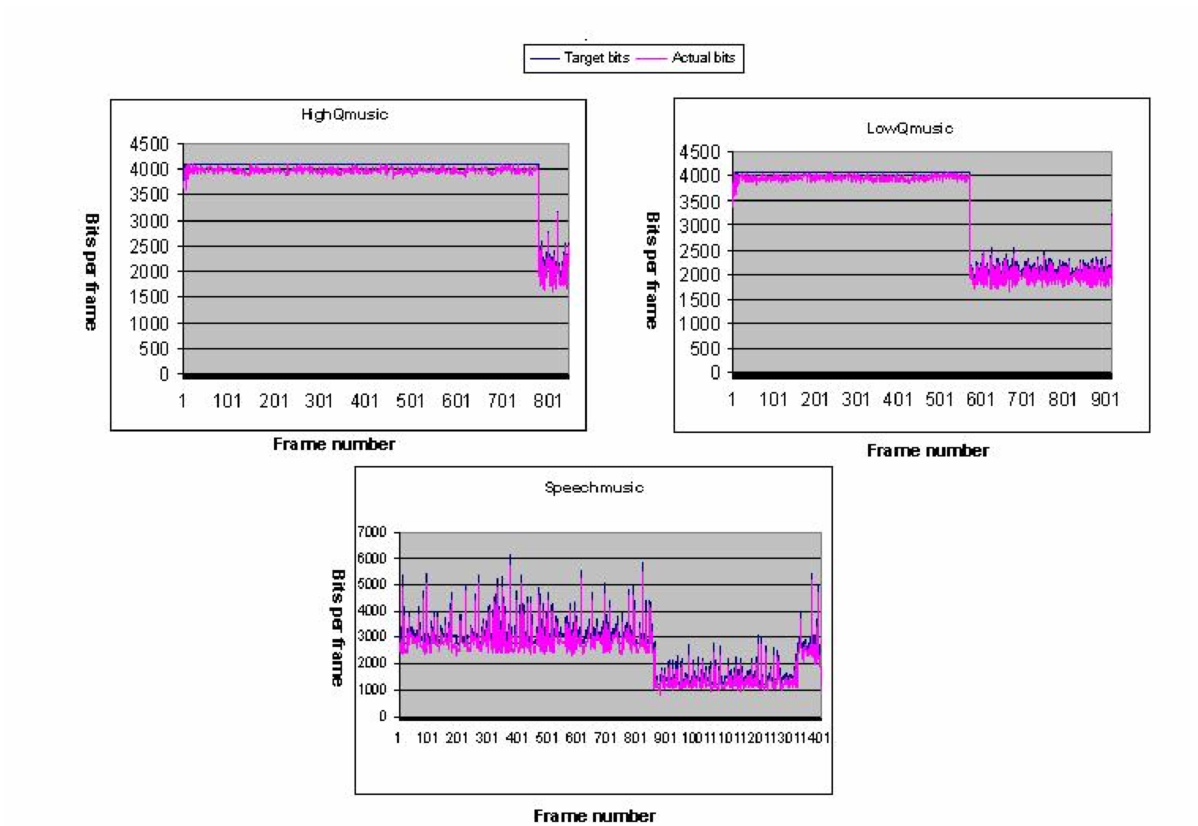


Figure 7.4. Symbiotic Rate Reduction on the three audio qualities

Table 7.2 provides a comparison of frame delay and acceptance ratio for the three types of audio sound: high quality music (HighQmusic), low quality music (LowQmusic) and speech mixed with music (SpeechMusic) at a player node. A delay tolerance of d = 2, 4, and 6 seconds are introduced to measure the average frame delay and acceptance ratio. Assuming the network has a delay of "*d*" seconds, we find the average delay experience by each frame to arrive at the player. The acceptance ratio is the ratio of frames that are delivered on time at the player. iEXP mode experience a low delay and high acceptance ratio while Classic mode experience the opposite, a high delay and low acceptance ratio. iOFF mode also experience a high delay and low acceptance ratio compare to the classic mode due to the overhead of event processing. Without a doubt, the iTCP's T-ware mechanism allowed the application to use sophisticated techniques to control the temporal qualities of its traffic.

	mode	**HighQmusic**		**LowQmusic**		**SpeechMusic**	
		Average delay	Accept ratio	Average delay	Accept ratio	Average delay	Accept ratio
***d* =2**	iEXP	-1.091	0.869	-2.192	1.00	-1.554	0.922
	iOFF	4.451	0.484	3.613	0.548	2.211	0.598
	Classic	3.167	0.539	2.048	0.627	3.274	0.662
***d* =4**	iEXP	-3.091	0.899	-4.19	1.00	-3.553	0.929
	iOFF	2.451	0.417	1.615	0.611	0.211	0.656
	Classic	1.167	0.609	0.05	0.685	1.274	0.606
***d* =6**	iEXP	-5.091	0.917	-6.187	1.00	-5.55	0.933
	iOFF	0.451	0.609	-0.383	0.658	-1.789	0.698
	Classic	-0.833	0.661	-1.947	0.727	-0.726	0.652

Table 7.2 Average Frame Delay and acceptance ratio

Table 7.3 gives the overall stream compression of the entire audio stream for the iEXP, iOFF and Classic mode. The overall delivery bits in the iEXP mode reduce to 80 – 90%

of the original bits whereas as in the case of iOFF and Classic cases, there is no adaptation.

Mode	HighQmusic		LowQmusic		SpeechMusic	
	Target Bits	Actual Bits	Target Bits	Actual Bits	Target Bits	Actual Bits
iEXP	0.900	0.900	0.811	0.810	0.812	0.805
iOFF	1.000	1.000	1.000	1.000	1.000	1.000
Classic	1.000	1.000	1.000	1.000	1.000	1.000

Table 7.3 Percentage of total bits delivered for each mode

7.5 Observation at Application Level:

Frames are encapsulated as they leave the application layer to the physical layer. The transport layer and network provides transport services to the frames. The encapsulated frames are stripped off as they arrive at the receiver end. The audio quality between the sending end and the receiving end is achieved perceptually by the temporal and spectral resolution. However, iTCP provides a means of tradeoff of severe frame delay for a satisfactory reduction of quality.

The original files are in uncompressed wav format, as the encoder encodes the frames, they are converted to compressed mp3 format and are sent from the encoder to the player. It is interesting to note that the sizes of audio files generated (mp3 format) by iEXP are smaller compared to those of the iOFF and Classical TCP in table 7.4. This also proves that the total traffic throughput in the network during congestion is smaller in iEXP mode.

The iEXP mode achieved the tradeoff, though to the listening ear shows no difference in quality due to the masking effect. Interested readers can listen to the test audio files [26]. The original audio file samples (wav) are downloaded from the Internet.

Audio Qualities	Original (wav)	iOFF (mp3)	Classic (mp3)	iEXP (mp3)
HighQmusic	544KB	497KB	497KB	447KB
LowQmusic	515KB	470KB	470KB	453KB
Speech&Music	3193KB	581KB	581KB	577KB

Table 7.4 Sizes of the audio files

7.6 Interactivity Overhead

The main purpose of the iOFF mode for the experiment performed is to study the overhead introduced by the event notification service of the symbiosis throttling model on the application. In effect, the total data transmission time for the three mode of operation were measured. Figure 7.5 shows the plots of overhead of interactivity service. As earlier explained, in the iEXP mode, interactivity, event notification and symbiosis rate adaptive feature services were all enable while in Classic TCP, the services were disable, indicating no overhead was introduced. As in the case of iOFF, the interactivity and event notification were enable but the symbiosis rate adaptive feature services was disable to measure overhead. As observed from the figure below, there was increase in the total transmission time for all modes, however the iEXP mode is much smaller than the other modes, such indicating the application level performance outweighs the overhead.

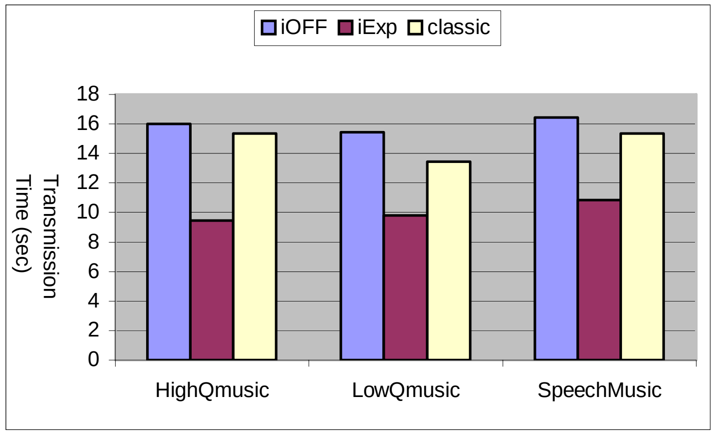


Figure 7.5 Overhead of the interactivity service

Chapter 8

Conclusions

Audio (voice and music) traffic suffers delay and jitter as it travels through a congested network, which invariably causes distortion and affects its quality. The main objective of this thesis is to reduce the delay and jitter faced by audio traffic in the network during congestion. A solution considered is a scheme that can receive state feedback from the network and reduce the target bit rate to respond to the network congestion. In this thesis, we developed a symbiosis encoder that can adapt to congestion. This adaptive encoder based on the symbiosis rate control mechanism is capable of detecting the underlying bandwidth of the network and modifying the target bit rate to suit the network condition. The feature of this scheme is simple, interactive and TCP friendly.

The functionality and effectiveness of the adaptive encoder is demonstrated by evaluating its performance in a classical TCP mode and interactive TCP mode.

Several set of experiments were performed on real audio session over the Internet and the conclusions were drawn from the result of the experiments.

Amongst the key achievement and benefits of this thesis are:

- Dynamic reduction of jitter and delay of time sensitive traffic during congestion.
- Congestion reduction in the network by reducing the traffic at the source. Data sent at a reduced target rate reduce the throughput in the network.

- Trade off quality for delay and jitter. More importantly, due to the perceptual nature of the human ear, a listening ear barely perceive the slight transient quality change as some data are pushed out of the network with reduced bit rate.
- This approach is simple and does not alter any network dynamic to be optimal when compared with other congestion control schemes like fair queuing. Its effect is entirely on the application layer.
- It also further validates interactive transport control protocol (iTCP) usefulness and the efficiency through the idea of the event notification. This is an efficient solution to the network problem that cannot be attained through the current Internet protocol suite or TCP.

In conclusion, the scheme in effect will change the way data is transferred particularly during congestion in the network and this calls for interactivity in the original TCP protocol suite (RFC007, RF793).

Technically, this scheme is not applicable to non-elastic traffic such as simple file transfer.

The parameters in the scheme are user defined but in the future can be optimized with the symbiotic throttling model in [13]. The experiment can be repeated using the third duplicate ACK.

BIBLOGRPHY

[1] Andersen D., Bansal D., Curtis D., Seshan S., and Balakrishnan H., "System Support for Bandwidth Management and Content Adaptation in Internet Applications," Proc. of OSDI'OO, Oct. 2000, San Diego, CA

[2] Balakrishnan H., Rahul H., and Seshan S., "An Integrated Congestion Management Architecture for Internet Hosts," Proc. of ACM SIGCOMM, Cambridge, MA, Sep 1999. pp.115-187.

[3] Catherine Boutremans, Jean-Yve L Boudec, “Adaptive Joint Playout Buffer and FEC adjustment for the Internet Telephony,” 2003.

[4] Eitan Altman, Chadi Barakat, Victor Ramos, “Queuing Analysis of Simple FEC schemes for IP Telephony,” 2001.

[5] Floyd S., Handley M., Padhye J., and Widmer J.,“Equation-Based Congestion Control for Unicast Applications,” August 2000. SIGCOMM 2000

[6] H. Schulzrinne, S. Casner, R. Frederick, and V. Jacobson, “RTP: A transport protocol for real-time applications,” RFC 1889, 1996

[7] Handley M., Floyd S., Pahdye J., and Widmer J., “TCP Friendly Rate Control” (TFRC): Protocol

[8] ISO/IEC 11172-3. Information technology -- Coding of moving pictures and associated audio for digital storage media at up to about 1,5 Mbit/s -- Part 3

[9] J. Mahdavi and S. Floyd, “TCP-Friendly unicast rate-based flow control,” in Draft posted on end2end mailing list, January 1997, http://www.psc.edu/networking/papers/tcp%20friendly.html.

[10] Jacobson V. and Michael J. Karels, "Congestion Avoidance and Control," Computer Communication Review, vol. 18, no. 4, pp. 314-329, Aug. 1988.

[11] Jacobson V., "Modified TCP Congestion Avoidance Algorithm," end2end-interest mailing list, April 1990

[12] Johnny Matta, Christine Pepin, Khosrow Lashkari, Ravi Jain, “A source and channel rate adaptation algorithm for AMR in VoIP using E-model,” June 2003.

[13] Khan J. and Zaghal R., "Symbotic Rate adaptation for Time sensitive Elastic Traffic with Interactive Transport," Journal of Computer Networks, Elsevier Science, March. 2006.

[14] Khan J., Zaghal R., and Gu Q., "Rate Control in an MPEG-2 Video Rate Transcoder for Transport Feedback based Quality-Rate Tradeoff;" PV2002, Pittsburgh, P A, April 2002.

[15] Khan J., Zaghal R., and Gu Q., "Dynamic QoS Adaptation for Time Sensitive Traffic with Transientware," IASTED WOC'03, Banff; Canada, July 2003.

[16] Khan J. and Zaghal R., "Event Model and Application Programming Interface of TCP Interactive," Technical Report 'TR2003-02-02', Feb. 2003.

[17] N. Shacham and P.M Kenney, “Packet recovery in high-speed networks using coding and buffer management,” in Proc. IEEE Infocom 1999

[18] Pradhan P., Chiueh T. and Neogi A., "Aggregate TCP Congestion Control Using Multiple Network Probing," Proc. Of the 20th International Conference on Distributing Systems, ICDCS 2000.

[19] Rishi Sinha, Christos Paradopoulos, Chris Kyriakakis, "Loss Concealment for Multi-channel streaming audio," June 2003.

[20] Sisalem D. and Wolisz A., "Towards TCP-Friendly Adaptive Multimedia Applications Based on RTP," Proc. of the 4th IEEE Symposium on Computers and Communications, 1998.

[21] Stevens W. R., "TCP/IP Illustrated, Volume 1: The Protocols," Addison-Wesley, 1994.

[22] Wang R., Yamada K., Sanadidi M. Y., and Gerla M.,"TCP with sender-side intelligence to handle dynamic, large, leaky pipes," IEEE Journal on Selected Areas in Communications, 23(2):235-248, 2005.

[23] Wen-Tsai Liao, Janet J.C.Chen, Ming-Syan Chen Chen, "Adaptive Recovery Techniques for Real Time Audio Stream"

[24] Wenyu Jiang, Hening Schulzrinne, "VoIP: Comparision and optimatization of packet loss repair methods in VoIP perceived quality under bursty loss," May 2002

[25] Yi J. Liang, Eckehard G. Steinbach, Bernd Girod, "Voice over IP: Real-time voice communication over the Internet using packet path diversity," October 2001.

[26] Olaleye Olufunke and Khan Javed, “Symbiosis Audio Communication On Interactive Transport”, Technical Report TR2007-12-01 February 2007. http://www.medianet.kent.edu/techreports/TR2007-12-01-audioset-OK/TR2007-12-01-audioset-OK.html