View metadata, citation and similar papers at core.ac.uk

brought to you by CORE

provided by ScholarBank@NUS

PERFORMANCE EVALUATION OF SPEECH QUALITY FOR VOIP ON THE INTERNET

LIU XIAOMIN

A THESIS SUBMITTED FOR THE DEGREE OF

MASTER OF SCIENCE

SCHOOL OF COMPUTING

NATIONAL UNIVERSITY OF SINGAPORE

2011

Acknowledgement

First, I would like to express my most profound gratitude to my advisor, Professor Roger Zimmermann, for his guidance and support. Working with him has been an invaluable experience in my life. Professor Zimmermann is a brilliant computer scientist and a great man with a gentle heart. It has been a huge honor for me to be his student.

Second, I would like to extend my appreciation to my peers who gave me great support during my life at NUS.

Abstract

This thesis reports on a measurement study to evaluate the speech quality for two widely used VoIP codecs, Speex and SILK, on the Internet using the *Perceptual Evaluation of Speech Quality* (PESQ). To obtain realistic results, we developed our testbed on PlanetLab[1], so that all the experiments were conducted on a shared network. We chose different sets of parameters for each experiment for the two codecs to evaluate the speech quality under different conditions. Overall, we found that the SILK codec performs slightly better than the Speex codec.

[1] http://www.planet-lab.org/

Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 Motivation

In recent years, the growth of the Internet has not only made our work lives much easier – for example, we can conduct a video or audio conference at home instead of physically meeting each other – but it has also affected and redefined many areas of entertainment, in particular how we consume video and audio. Now, we can conveniently enjoy real-time Internet music streaming services like Jango (jango.com). Furthermore, the population of people who enjoy online video and audio is growing very quickly. Streaming technology has become a very hot topic. Since many Internet services are commercial entities, there has been a growing interest in the quality of online media delivery including video and audio streaming. This thesis reports on a detailed study on speech quality evaluation measurements for two modern, important Voice-over-IP (VoIP) codecs, Speex[1]

[1] http://www.speex.org/

and SILK[2].

Speex is a free and open source codec which is often used for free IP audio communication applications. Speex is quite flexible in that it has been designed for packet networks and VoIP applications, as well as file-based compression. Due to these important characteristics, it has drawn significant attention from researchers. We chose this codec as one of the candidates to study the achievable VoIP speech quality on a real network. SILK has been developed by a company called Skype for their VoIP application. The Skype software is very popular and it has users world-wide. At the end of 2009, there were already 500 million registered Skype users. Recently the company was acquired by Microsoft and the number of users is expected to further increase. Our goal is to explore the performances of these two codecs to see how well they work in realistic environments. In the literature survey section we will introduce these two codecs in detail.

As there are many configuration parameters for both Speex and SILK, our objective was to explore the differences between these two codecs over a range of different encoding and decoding parameters. To achieve realistic results we have built a testbed software for the PlanetLab environment and conducted extensive experiments.

[2] http://developer.skype.com/silk

1.2 Thesis Objectives

The objective in this thesis is to study the speech quality for two modern, widely used VoIP codecs, Speex and SILK. As there are many configuration parameters for both Speex and SILK, our methodology is to select common parameters which will be changed over a wide range of settings during our experiments to investigate how they affect the speech quality when running in a common environment.

As our testbed we selected PlanetLab, which is a global research network that provides resources to researchers at academic institutions and industrial research labs to develop new network services. With the PlanetLab resources, we designed a point-to-point testing software to simulate real-time speech streaming and the resulting output was then evaluated for its speech quality using the PESQ standard. Our testbed is built on an public, shared network, which means that packet loss is unpredictable. Due to this reason, we also conducted some local lab experiments, where the packet loss rate can be controlled. We will use the Gilbert Model to simulate packet losses for our local tests.

1.3 Thesis Contributions

The main contributions of my thesis are summarized as follows:

- First, we implemented an n-way point-to-point testing system on PlanetLab and conducted a set of experiments for speech quality evaluations.

- Second, we studied the characteristics of the Speex and SILK codecs. We compared the performance of two codecs under different sets of parameters using the PESQ metric.

1.4 Organization of the Thesis

To better describe my work, I have organized my thesis into four chapters.

Chapter 1 **Introduction** explains the motivation, objective and the contributions for my thesis.

Chapter 2 **Background and Literature Survey** first introduces the transmission protocols which will be used for our testbed. Next this section discusses audio coding algorithms and error control mechanisms. Lastly, prior work in the field of VoIP measurements are studied.

Chapter 3 **Experimental Setup and Results** describes our testbed architecture as well as our test results.

Chapter 4 **Conclusions** concludes the work performed in the thesis.

Chapter 2

Background and Literature Survey

Presently, digital media are well established as an integral part of many applications. A considerable amount of research has focused on the audio streaming over the Internet. In this chapter, we will introduce the techniques commonly used in audio streaming, including transmission protocols for audio streaming, audio coding algorithms, and packet loss recovery mechanisms. We will also study the previous work in the area of Voice-over-IP (VoIP) measurements.

2.1 Transmission Protocols

End-system applications often do not implement all the detailed communication features; instead, they make use of existing communication protocols. There exist a number of protocols which can be used for audio streaming. For example, a network protocol can be used to forward datagrams across a physical channel, and a transport protocol can be used for end-to-end services. The combination of protocols is called a protocol stack. The typical protocol stack used for data transmission over the Internet is TCP or UDP (the *user datagram protocol*) [28] on top of IP (the *Internet Protocol*) [27]. Figure 2.1 shows an example of a protocol stack.

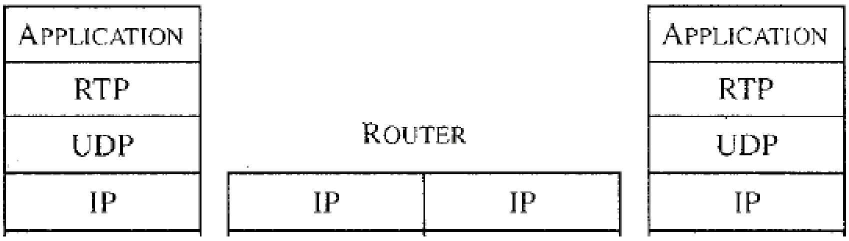


Figure 2.1: Example of a protocol stack: application data is transported via RTP, UDP, IP and Ethernet/ATM [18].

TCP is a connection-oriented, reliable and full-duplex protocol. It uses an acknowledgement and retransmission scheme to make sure that every packet is received by the receiver. Moreover, TCP ensures an ordered reception of packets by delivering packet p_m only when all the previous packets, $p_j, j < m$, have been received. Because of these mechanisms provided by TCP, it is very suitable for applications such as *ftp, telnet* and *web servers*, etc., while it is less suitable for real-time media delivery, because of its potentially long delay of packets. Thus, real-time applications tend to use UDP, which is connectionless, best-effort and without flow control mechanism. However, it can provide low latency service for real-time audio streaming.

The Real-time Transport Protocol (RTP) [35] is a transport layer protocol framework which was developed by the Internet Engineering Task Force (IETF)

Audio/Video Transport working group in order to deliver streamed media over the Internet. Each packet contains time information, packet sequence numbers and optional parameters. Different payload formats have been developed according to different audio and video compression standards. It has been proposed to combine RTP with the receiver-initiated retransmission scheme mentioned in S-RM [25]. RTP can also cooperate with RTCP (the Real-Time Transport Control Protocol) [15] which allows the collection of feedback from receivers. It provides end-to-end network transport functions for real-time audio streaming [42]. Its specification states that "*RTP is intended to be malleable to provide the information required by a particular application and will often be integrated into the application processing rather than being implemented as a separate layer.*" In practice, RTP usually runs on top of UDP [22].

RTCP is an accompanying protocol of RTP designed to exchange control information related to real time data transmissions. Either UDP or TCP can be used as the underlying transmission protocol, depending on the requirements of the application. Since RTCP was designed with large-scale multimedia applications in mind, the protocol can offer considerable control information.

For our testing system, we employed the RTP-over-UDP protocol stack.

2.2 Audio Coding Introduction

In this section, we will introduce a few audio formats which are used in conjunction with different codecs and audio coding algorithms. The details are described in the following paragraphs.

2.2.1 Audio File Formats

An audio file format is a file format for storing audio data on a storage media. The general method for storing digital audio is to sample the audio waveform (i.e., voltage) which, on playback, corresponds to a certain level of signal in an individual channel with a certain resolution. The data can be stored uncompressed or compressed to reduce the file size. There are three groups of audio file formats:

- Uncompressed audio formats such as WAV, AIFF, AU or raw header-less PCM.
- Audio formats with lossless compression such as FLAC, Monkey's Audio (filename extension APE), TTA, Apple Lossless, MPEG-4 SLS, MPEG-4 ALS, MPEG-4 DST and Windows Media Audio Lossless (WMA Lossless).
- Audio formats with lossy compression such as MP3, Vorbis, AAC, ATRAC and lossy Windows Media Audio (WMA).

There is one major uncompressed audio format, Pulse-Code Modulation (PCM), which is usually stored as .WAV files on Windows or as .AIFF on Mac OS X. WAV and AIFF files are suitable for storing and archiving original recordings due to their flexible file formats to store more or less any combination of sampling rates and sample resolutions. In our system, we use WAV files as high-quality inputs for the audio streaming codecs.

2.2.2 Audio Coding Algorithms

Data compression can convert an input data stream into another data stream which is of smaller size compared to the original. This is very useful for transmissions when the network bandwidth is limited, especially for real-time audio or video streaming, which may require considerable bandwidth. There exist several types of data compression algorithms such as methods for text compression, image compression, simple dictionary compression, video compression and audio compression. Here, we will only introduce a few algorithms for audio compression. Many types of codecs have been developed for audio encoding such as μ-Law and A-Law companding, ADPCM, MLP, speech compression, FLAC, Monkey's audio, etc. Here, we only introduce a few of commonly used ones.

Two important, distinguishing characteristics for audio compression algorithms are:

- Whether the compression is lossy or lossless; and
- Whether the encoding and decoding complexities are symmetric or not, i.e., how fast the decompression is.

There exist both lossy and lossless algorithms for audio compression. Audio is often stored in compressed form which is then decompressed in real-time and played back to listeners. Thus, most audio compression methods are asymmetric. The encoder can be slow, but the decoder must be fast. As there are many kinds of compression algorithms, we only introduce a few of them in the following sections.

Speech Compression

Some audio codecs are specifically designed for speech signals. As this kind of audio is human speech, it has many properties which can be exploited for efficient compression. There exist considerable research on this topic such as the codecs introduced in the book by Jayant *et al.* [17].

There are three main types of speech codecs. Waveform speech codecs produce good to excellent quality of speech after compression and decompression, but generate bit rates of 10 to 64 kbps. Source codecs (vocoders) generally produce poor to fair quality of speech, but can compress the bit rate to a very low level, for example 2 kbps. Hybrid codecs combine these two methods and can generate fair to good quality speech with bit rates between 2 and 16 kbps. Figure 2.2 shows the qualitative speech quality versus the bit rate of these three codec types.

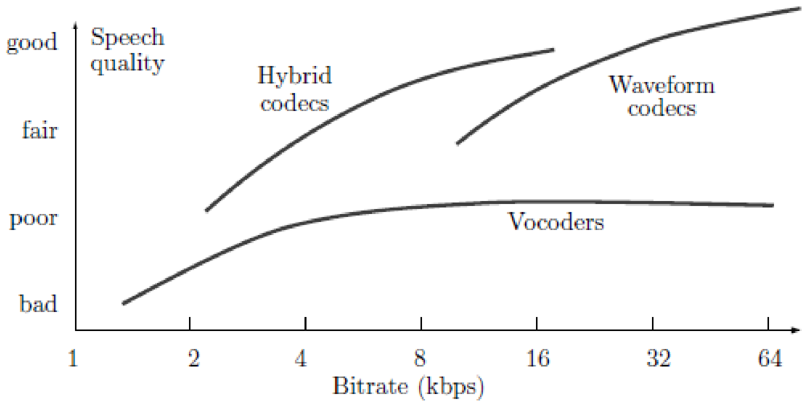


Figure 2.2: Speech quality versus the bit rate for speech codec types [33].

Waveform codecs. This codec type is not specifically concerned about how the original sound was generated, but tries to produce the decompressed audio signal as closely-matching as possible to the original signals. It is not designed for speech specifically and can be used for other kinds of audio data. The simplest waveform encoder is pulse code modulation (PCM). Enhanced versions are the differential PCM and ADPCM encoders. Waveform coders may also operate in the frequency domain.

Source codecs. In general, a source encoder uses a mathematical model of the source of the data. The model depends on certain parameters, which are obtained through the input data. After the parameters are computed, they are written into the compressed stream. The decoder uses the parameters and the mathematical model to rebuild the original data. If the original data is audio, the source coder is also called a vocoder.

Hybrid codecs. This kind of speech codec combines both of the previously described codecs. The most popular hybrid codecs are Analysis-by-Synthesis (AbS) time-domain algorithms. An AbS encoder starts with a set of speech samples (a frame), encodes the samples in a similar way to a LPC (Linear Predictive Coder) [29], decodes them, and subtracts the decoded samples from the original ones. The differences are sent through an error minimization process that outputs improved encoding samples. These samples are again decoded, subtracted from the original samples, and new differences computed. This process is repeated until the differences satisfy a termination condition. The encoder then proceeds to the next set of speech samples (*i.e.*, the next frame) [33].

We will now describe two modern, state-of-the-art codecs, which we will be using in our experiments, namely *Speex* and *SILK*.

2.2.3 The Speex Codec

The Speex codec is open-source and free from patent royalties. It is designed for packet networks and Voice-over-IP (VoIP) applications, as well as file-based compression. The Speex codec is quite flexible. There are many parameters that can be selected, such as the bit rate and so on. It is also quite robust to packet losses. This property is based on the assumption that in VoIP applications the packets either arrive late or lost, but not corrupted. Below is a list of parameters that can be adjusted during encoding and decoding for the Speex codec [39]:

- **Sampling rate.** The sampling rate is expressed in Hertz (Hz). It indicates the number of samples taken from a signal per second. Speex is mainly designed for three different sampling rates: 8 kHz, 16 kHz, and 32 kHz. These sampling rates are respectively referred to as *narrowband*, *wideband* and *ultra-wideband*.

- **Bit rate.** The bit rate is the speed of the speech signal being encoded. It is measured in *bits per second* (bps). When the speech signal is encoded in narrowband mode, the bit rate can be set from 2.15 kbps to 24.6 kbps; when the speech signal is encoded in wideband mode, the bit rate can be changed in the range from 4 kbps to 44.2 kpbs.

- **Quality.** Speex is a lossy codec. It achieves compression at the expense of the fidelity of the input speech signal. It is possible to control the tradeoff made between quality and the bit rate. In the Speex encoding process, the quality parameter can be changed from 0 to 10.

- **Complexity.** With Speex, it is possible to change the complexity parameter for the encoder. The complexity can be changed from 1 to 10. For normal use, the noise level at complexity 10 is between 1 and 2 dB lower than at complexity 1, but the CPU requirements for complexity 10 is about 5 times higher than for complexity 1. Hence, in practice, the best tradeoff is a setting between 2 and 4.

There exist also other parameters, like discontinuous transmission (DTX), which can be changed when encoding a speech signal. We currently only consider the above mentioned, most commonly used parameters in our system.

2.2.4 The SILK Codec

The SILK codec is preferred for Skype-to-Skype calls. It is a speech codec for real-time, packet-based voice communications. It provides scalability in several dimensions. It supports four different sampling frequencies for encoding the audio input signal. It can adapt to the network characteristics through the control of the bit-rate, the packet rate, the packet loss resilience and the use of DTX. The SILK codec also allows several complexity levels which can be changed to let it take advantage of the available processing power without relying on it. All of these properties can be adjusted while the codec is processing data.

The SILK codec consists of an encoder and an decoder[40]. For the encoder, there exist a number of parameters that can be changed to control the encoding operation.

- **Sampling rate.** SILK can select one of four modes during call setup:
  - Narrowband (NB): 8 kHz sampling rate;
  - Mediumband (MB): 8 or 12 kHz sampling rate;
  - Wideband (WB): 8, 12 or 16 kHz sampling rate; and
  - Super Wideband (SWB): 8, 12, 16 or 24 kHz sampling rate.

  The purpose of the modes is to allow the decoder to utilize the highest sampling rate used by the encoder.

- **Packet rate.** SILK encodes frames of 20 milliseconds each. It can combine 1, 2, 3, 4 or 5 frames in one payload, so each packet corresponds to 20, 40, 60, 80 or 100 milliseconds of audio data. Sending fewer packets per second reduces the bit rate, but it increases the latency and the sensitivity to packet losses since longer packets constitute a bigger fraction of the audio information. In our system we encode one frame into one packet each time.

- **Bit-rate.** The bit-rate can be set to the range from 6 to 40 kbps. A higher bit-rate can improve the audio quality by lowering the amount of quantization noise in the decoded signal. For the narrowband mode, the bit-rate can be changed in the range from 6 kbps to 20 kbps, while for the wideband mode, it can be changed between 8 kbps and 30 kbps.

- **Complexity.** SILK has three complexity levels which can be chosen. A low level can reduce the CPU load by a few times at the cost of increasing the bit-rate by a few percentage points. The three complexity levels are high (2), medium (1) and low (0).

- **DTX.** The DTX function can reduce the bit-rate during silence or background noise. For our tests it is disabled.

On the decoder side, the received packets are split into the number of frames contained in the packet. Each of the frames contains the necessary information to reconstruct the 20 ms frame of the original input signal.

2.2.5 Summary

As described in this section there exist many codecs for audio compression and we only listed some of them briefly. Different codec have different features which make them suitable for different conditions and audio formats. For speech compressors, they can be grouped into three categories as described earlier.

Waveform speech codecs. They produce a good to excellent quality of speech, and the bit rate is between 10 to 64 kbps;

Source codecs. They produce a poor to fair quality of speech, the bit rate can reach to 2 kbps;

Hybrid codecs. They are a combination of the waveform speech codec and the source codec. The speech quality varies from good to fair. The bit rate ranges from 2 to 16 kbps.

For the Speex and the SILK codecs, there are many parameters which can be changed during the encoding process. To be more clear, we summarize the listed audio codecs as Table 2.1.

Parameter	Speex codec	SILK codec
Bit rate (kbps)	1. Narrowband: 2.15 - 24.6 2. Wideband: 4 - 44.2	1. Narrowband: 6 - 20 2. Wideband: 8 - 30
Packet Rate	Frame size is 20 ms long.	Frame size is 20 ms long.
Quality	0 - 10.	No such parameter.
Complexity	It can be changed from 1 to 10. Difference of the noise level between 1 and 10 is only 1 or 2 d-B, while the CPU requirements is 1/5 at complexity 1 compared with complexity 10. In practice, the tradeoff is 3.	0 (low), 1 (medium) and 2 (high). As the difference level for these complexity values are not much, the tradeoff is set to complexity 1 ordinarily.
Sampling Rate	1. Narrowband: 8 kHz 2. Wideband: 16 kHz 3. Ultra-wideband: 32 kHz	1. Narrowband: 8 kHz 2. Mediumband: 8 or 12 kHz 3. WideBand: 8, 12 or 16 kHz 4.Super Wideband: 8, 12, 16 or 24 kHz
Delay	No more than 30 ms.	Around 30 ms for narrowband mode and 34 ms for wideband mode.

Table 2.1: Comparison of the control parameters for the Speex and the SILK codecs.

As Table 2.1 shows, we list the parameters that are used in our system. To be fair in the comparison between the Speex and the SILK codecs, we only change the common parameters for both of them and disable the other parameters. For example, the Speex codec has a *quality* parameter, which can be used to control the tradeoff between the bit rate and the audio quality, while the SILK codec does not have this parameter.

2.3 Error Control Mechanisms

In the following section, we discuss existing techniques for transport protocols that use ARQ (automatic repeat request) and FEC (forward error correction) for providing reliable real-time multicast services.

2.3.1 ARQ-based Error Control Mechanisms

Continuous media (CM) include audio and video data. There are different mechanisms for audio and video error control due to the difference in bit rate requirements.

Dempsey and Liebeherr were the first to investigate the retransmission for CM applications [10, 9] for the case of unicast interactive voice transmissions over local area networks. They proposed the Slack-ARQ approach which performs NAK-initiated retransmissions within a given time duration. Retransmissions are not a feasible option as interactive voice communications often require round-trip delays of less than 200 milliseconds [20]. Recently, a new protocol has been developed for the delivery of non-interactive voice over the Internet to multiple recipients by Xu, Myers and Zhang [43]. Their protocol is named Structure-Oriented Resilient Multicast (STORM). It does local loss recovery to achieve scalability and lower recovery times and its main steps are as follows:

- The receiver detects losses with a method of gap-based loss detection, then uses the NAKs to request the retransmission of the lost packets.
- Each receiver maintains a list of parent nodes from which it chooses one node to send its NAK to. If the node fails to retransmit, the receiver will choose another parent node.
- The NAKs and the transmissions are carried out via unicast to keep the overhead due to loss recovery low.

For video transmissions, as the bit rate requirement is higher than for audio, other mechanisms must be used. An example is the receiver-driven layered multicast (RLM) [23], which uses a hierarchical coding scheme. The signal is encoded in a base layer that provides low quality images and additional complementary layers for improved image quality. Each receiver needs to receive at least the base layer. Different layers of the video are transmitted through different groups. Each client receives the base layer and as many of the additional layers as possible to reconstruct the original video.

Another protocol is named Layered Video Multicast with Retransmissions (LVMR), for non-interactive transmissions of MPEG video to multiple receivers:

- The MPEG stream is separated into three layers: the base layer contains I-frames while the other layers contain B-frames and P-frames, respectively.

- When the loss of a frame is detected, the receiver can send a NAK to ask for the retransmission only if it is still not late to arrive for the play-out of the requested frame. This depends significantly on the network round-trip time.

- With LVMR the loss recovery is local: It knows who the NAKs will be sent to.

- The NAKs and the retransmissions are done via unicast to make sure the overhead is low due to low loss recovery.

ARQ mechanisms are not very suitable for live audio and video streaming because they increase the end to end latency as well as they do not scale well to large multicast environments.

2.3.2 FEC-based Error Control Mechanisms

Today, the widespread use of Internet telephony and video-conferencing are limited by the service quality due to losses in congested routers. FEC error control mechanisms have been adopted by many real-time interactive applications with tight delay requirements. There are a number of applications that have been developed with application-specific FEC schemes with good delay properties.

The INRIA freephone [4] encodes audio streams with two different coding standards and by transmitting the encoded samples of the same time interval in subsequent packets, it achieves good delay properties. The data streams of freephone contain a PCM-encoded sample of one time interval in each packet, together with redundant data which is the previous time interval data encoded at a lower bit rate. This FEC scheme adds only a little bandwidth overhead to the PCM audio stream and does not increase the IP packet rate.

An example for a video-specific FEC scheme is the Priority Encoding Transmission (PET) developed at ICSI, Berkeley [2, 1, 38]. This technique allows the user to assign different priorities for each segment of a continuous media stream. PET generates a different amount of redundancy for the segments and disperses user data and these redundant data into subsequent packets according to the assigned priorities. PET can be applied to the transmission of MPEG video streams. For each group of pictures, the I-frames are protected with a higher amount of redundancy than P-frames and B-frames.

The advantage of FEC is that all the loss recovery happens at the receiver side and the sender does not have to known which packets were lost. There is also no time penalty as no retransmissions happen. For this reason, FEC is often preferred for real-time interactive media communications [37, 5, 3]. In contrast, the disadvantage of FEC is that it increases the bandwidth required by the redundant data. Moreover, its ability to recover information is dependent on the characteristics of the underlying network. For example, FEC can not recover from a large burst of packet losses.

In summary, FEC is an attractive alternative to ARQ as it does not increase the latency, but the effectiveness of FEC depends much on the characteristics of the packet loss process in the network. The base layer packet must be received for the FEC mechanism to work. So since both the ARQ and FEC have their own merits and demerits, it is better to make use of their merits together. So maybe we could combine ARQ and FEC together to get a more reliable and effective mechanism for real-time audio and video streaming.

Many proposed FEC mechanisms involve exclusive-OR operations, *i.e.*, the idea is to send every n^{th} packet followed by a redundant packet created by exclusive-ORing the other n packets [36]. This mechanism can recover the loss of one of the other n packets. It increases the number of packets to be sent and the latency. When loss happens, all the n packets have to be received until the lost packet can be constructed. The larger n is, the longer the latency.

2.3.3 Referential Loss Recovery for Application Level Multicast

Streaming media is very sensitive to transmission delays. There is usually a permissible delay time before the media streaming starts. It is called the start-up delay [6]. The referential loss recovery (RLR) [12] method separates the FEC packets from the original media packets and sends them to the receiver side before media streaming starts. The FEC packets are transmitted using TCP in order to make sure that they are received correctly and the usual media packets are still transmitted using RTP and UDP. If media packets are lost, they are recovered by referring to the FEC packets already received at the receiver side. Thus, this method can provide original primary audio without heavily compressed secondary audio even when packet losses occur. It can avoid the increased congestion caused by FEC overhead and make the FEC overhead independent of the streaming session.

With RLR the block delay will be reduced since lost media packets can be recovered using FEC packets already received on the receiver side. In addition, some of the original packets can be generated using the FEC packets received on the receiver side before all the media packets arrive at the client, and this can be used as a solution for delay jitter.

2.3.4 Summary

A lot of research has been performed on error control mechanisms for audio streaming and we only selected a few of them. To be clear, we summarize the above mechanisms into a Table 2.2. For our measurements in this thesis, we will use FEC for SILK to investigate how much the mechanism will affect the speech quality when FEC is enable compared with FEC disabled. For Speex, we will use the PLC (Packet Loss Concealment) mechanism for packet loss control. When packet losses happen, PLC conceals each lost packet with a silent packet. A detailed description will be introduced in the following chapter.

Error Control Mechanism	Features
ARQ-based Error Control Mechanisms	1. Works well for multicast applications as the sender explicitly retransmits the lost packets which have an effect on the next-hop streaming. 2. The disadvantages are that it increases the latency and retransmission may cause more packet loss due to the increasing bandwidth.

FEC-based Error Control Mechanisms	1. One advantage is that it is only visible to the receiver. No latency is caused as no retransmissions happen. 2. The drawback is that it increases the bandwidth for the redundant data. Also, the ability for recovery depends on the loss characteristics of the underlying network. If large bursts of packet losses happen, then FEC will not work.
Referential loss recovery for application level multicast	One advantage is that FEC packets are received ahead of time, so the block delay is reduced as it can recovery directly with the received FEC packets. Another advantage is that it can avoid the increased congestion caused by FEC packets as these packets are already on the receiver side. Moreover with the pre-received FEC packets, we essentially have prediction packets which may be used in future.

Table 2.2: Comparison of different error control mechanisms.

2.4 Speech Quality Evaluation for VoIP Communications

VoIP (Voice-over-IP) refers to one of a family of Internet technologies, communication protocols and transmission technologies for the transmission of human speech over IP networks. Internet telephony is one typical example for the VoIP applications. As IP networks are resource-shared, packet-switched networks, IP-based VoIP applications are cost-effective compared with the traditional resource-dedicated PSTN (Public Switched Telephone Network). It can reduce the communication and infrastructure costs greatly. Moreover, it can provide much more flexible services; for example, it can transmit more than one telephone call over a single broadband connection. On the other hand, current IP networks only provide best-effort services. They lack stringent QoS controls. Communication over the IP networks is less reliable than that of the PSTN. Congestion is inevitable and may result in packet losses, delay and data arrival jitter, which may be directly related to the quality of VoIP applications. In the following sections, we will introduce the speech quality evaluation methods used to measure the quality of VoIP applications.

2.4.1 Subjective Speech Quality Measurements

First, we will introduce subjective speech quality measurements for VoIP applications. Speech quality has subjective characteristics when it is processed by human beings. The level of individual perception of speech quality may be affected by the mood and the interest of the people who evaluate the speech quality. However, to make the evaluation result more objective, the personal point of view should be excluded. Usually, speech quality is measured by MOS (Mean Opinion Score) between 1 ("unacceptable") and 5 ("excellent"). It is an average score from many listeners.

The subjective methods needs great efforts, cost and procedural tools for measuring the subjective ideal of individual preferences for the speech environment, while on the other hand it can provide a much more reliable and actual evaluation for the speech quality. There are many subjective methods for the quality evaluation of speech, such as DCR (Degradation Category Rating), CCR (Comparison Category Rating), and ACR (Absolute Category Rating). Kang *et al.* [19] perform an overall examination of speech quality using ACR which uses the MOS evaluation scale principally.

As these subjective methods are time-consuming and expensive, it is desirable to have methods which can estimate the subjective speech quality from the speech signals, and hence objective speech quality evaluation methods have been developed.

2.4.2 Objective Speech Quality Measurements

Objective measurements give the designer an opportunity to make comparisons based on factors that examine the quality objective and scientific. There are many objective speech quality evaluation methods, such as PESQ, the E-model and so on. Batu and Benjamin [34] evaluated the CVCQ (Conversational Voice Communication Quality) with some objective methods based on packet traces collected on PlanetLab. They employed the PESQ and E-model [30] techniques to evaluate the speech quality for four popular VoIP client systems: Skype (v2.5), Google-Talk (Beta), Windows Live Messenger (v8.0) and Yahoo Messenger with Voice (v8.0). Their experiments suggest that Windows Live messenger is more robust to packet losses by using higher mouth-to-ear delays. Chiang and Xiao used a measurement-based approach to evaluate the performances of Skype and MSN between two hosts [8]. Their experimental results were measured with MOS. They concluded that Skype outperformed MSN from their study. Pietro and Dario [24] have used the PESQ technique to evaluate the objective speech quality in VoIP systems. Leandro and Edjair employed the E-model to evaluate the speech quality [7]. Liu and Wei also employed the E-model for their speech quality evaluation [21]. Xie and Yang [41] studied the VoIP quality of a popular peer-to-peer application, Skype. The quality of a VoIP session in Xie and Yang's paper is measured by the E-model. The quality of VoIP is quantified under the condition that there is no AS constraint and sufficient access capacity exists. They also studied how the AS constraint and access capacity impact the VoIP quality of Skype.

As stated in the previous section, error control mechanisms have been considered for VoIP applications. Huang *et al.* [14] researched Skype's FEC mechanism. They investigated the relationship between redundancy ratio and the network loss rate. They found that when the packet loss rate increased, the redundancy ratio would increase as well. Their experimental results were expressed with MOS. Pentikousis *et al.* [26] conducted a measurement study of Speex and

H.264/AVC video over IEEE 802.16d and IEEE 802.11g. They simultaneously carried emulated H.264/AVC video and Speex VoIP audio on their testbed to get the results for packet losses and one-way delay under both line-of-sight and non-line-of-sight conditions. They emulated multiple Speex VoIP flows with a wideband codec bit rate of 12.8 kbps. The results showed that the testbed's fixed WiMAX uplink was capable of supporting 100 Speex VoIP flows before packet losses happened.

In the following part, we will specifically introduce the PESQ standard, which will be used to evaluate the speech quality of the streamed audio in our testbed.

The PESQ Standard

There are a lot of technologies which can be used for the measurement for audio quality.

- PSQM (*Perceptual Speech Quality Measure*) is the predecessor technology of PESQ.
- PEAQ (*Perceptual Evaluation of Audio Quality*) is used for various audio measurements, not specifically for speech.
- MOS (*Mean Opinion Score*) is the basic method.
- POLQA (*Perceptual Objective Listening Quality Analysis*) is a very newly developed technology used to measure voice quality for fixed, mobile and IP based networks. POLQA has been selected to form the new ITU-T voice quality testing standard. It is the successor of PESQ and it is expected

to be made available commercially in an aligned release process during September 2010.

From our investigation we found that the PESQ standard is the best choice for our experiments. In the following part, we will introduce the PESQ standard in detail.

PESQ is a set of standards consisting of a testing methodology for the automated assessment of the speech quality experienced by a user of a telephone system. Its standardised form is ITU-T recommendation P.862(02/01). Currently PESQ is widely used as an industry standard for objective voice quality testing. The PESQ model is very suitable for our own testing system. In the audio streaming process, there often occur packet losses and delays due to bad network conditions. Sometimes the delay is quite different among various packets. There is a jitter buffer to reorder the out-of-order packets to improve the audio quality. On the other hand, the jitter re-sizing will lead to a change in the end-to-end audio delay [32]. The PESQ model provides an algorithm for delay assessment, and it enables the reference and degraded signals to be aligned [16]. Hence, the PESQ model is used here for our tests.

The structure of the PESQ model is shown in Figure 2.3. The PESQ score is the combination of the average disturbance value and the average asymmetrical disturbance value. In most cases the output score range will be a MOS-like score between 1 and 4.5.

The level alignment is used to align both the reference signal and the degraded signal to a standard listening level. Then they are fed into an input filter to model

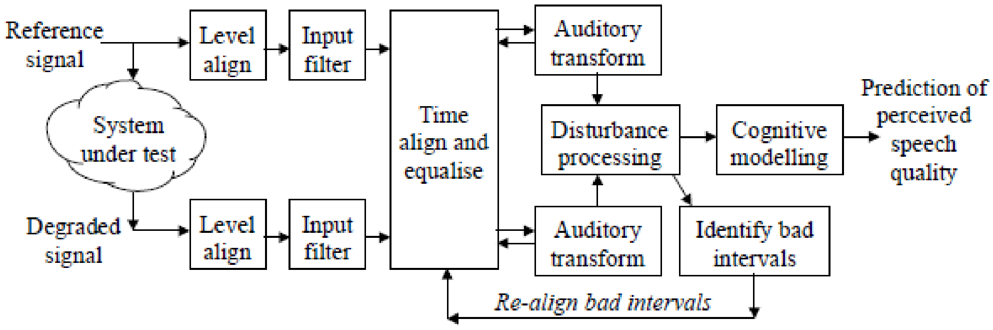


Figure 2.3: Structure of the PESQ model [31].

a standard telephone handset. The signals are aligned in time and processed with an auditory transform. The transmission also involves equalizing for line filtering in the system and for gain variations. Two distortion parameters are then extracted from the disturbance processing. They are aggregated in time and frequency, and then mapped to a prediction of a subjective mean opinion score (MOS).

2.4.3 Summary

As has been described above, both subjective and objective methods have their own advantages and disadvantages. Subjective methods are much more reliable and closer to the actual evaluation, but on the other hand, they are quite time-consuming and expensive to conduct. For the objective methods we only need to feed both the reference signal and degraded signal into the testing model to get the result. This method is much easier, while the result will be an average score and less accurate.

Chapter 3

Experimental Setup and Results

This chapter describes the system that was implemented on PlanetLab to perform the speech quality measurements. With the designed peer-to-peer (P2P) testbed, we simulated real-time speech streaming utilizing both Speex and SILK. To meet the real-time requirements, the latency for the streamed speech transmissions should be as low as possible while at the same time, the speech quality should also be as good as possible. Hence, there exists an essential tradeoff between speech quality and latency. We also conducted LAN experiments on a local machine with the Gilbert Model to simulate a scenario with bursts of packet losses.

3.1 Box Plot Graph Overview

First, we will give an overview of the box plot features that we will be using to present our test result. A box plot is a way of summarizing a set of data values measured on an interval scale. We use the box plot to show the PESQ result for the Speex and the SILK codecs with different configuration parameters. The layout of a basic box plot is shown in Figure 3.1. The median value ($Q2$) is the value of a data point in the test dataset that can separate the dataset into two equal-sized data sub-sets. $Q1$ represents the first quartile or the median of the lower half of the dataset and $Q3$ is the median of the upper half of the dataset.

$$Cutoff1 = Q1 - w \times (Q3 - Q1) \tag{3.1}$$

$$Cutoff2 = Q3 + w \times (Q3 - Q1) \tag{3.2}$$

Min is the actual minimum data point in the dataset which is just above $Cutoff1$ and Max is the actual maximum data point in the dataset which is just below $Cutoff2$. All the data above Max or below Min are considered to be outliers. The default value for w is 1.5.

For our experiments, we obtained a large number of test results. These results represent the speech qualities for the streamed audio files of the peer-to-peer testbed. To show how the results are distributed, we choose the box plot. It can show the value range for the middle 50%, the quality value at points of 25% and 75% of the whole result set as well as outliers.

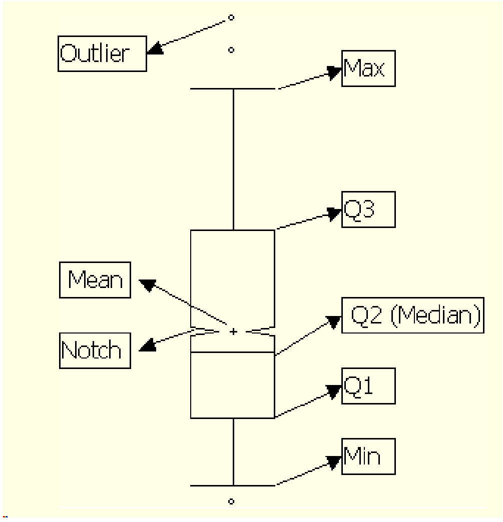


Figure 3.1: Elements and thresholds of a box plot.

3.2 System Design

In this section we describe the structure of our test system, as well as the methods used in our system. The connectivity structure of our test system is a peer-to-peer topology. Each node works both as a server and a client. The two nodes that participate in a transmission act as one server and one client, respectively. Instead of streaming live audio, we simulated the streaming with pre-recorded, off-line audio files between the clients and servers. Next we will describe the audio files that we used for streaming in our testbed.

3.2.1 Test Audio Files

In our experiments, we selected eight 16-bit PCM WAV audio files as input audio sources. Each audio file had a duration of 59 seconds with a bit rate of 128 kbps. They actually included the same content, but with different file names for the convenience of comparing the quality among the streaming nodes under different conditions. We were attempting to reduce the potential factors that may affect the result comparison to as few as possible.

3.2.2 Peer-to-Peer Topology

There are more than 1,000 nodes participating in PlanetLab. For our experiments we selected 325 nodes to create our test system. Among the 325 nodes, we applied a uniform randomized algorithm to send the audio files between each pair of nodes automatically. During each test round each node that works as a server only sends audio files to one client, and at the same time, each node that works as a client can only receive audio files from one node. All the nodes work simultaneously.

3.2.3 The Client/Server Model

There are 325 clients and servers in our system when running one test round. We take one client and one server as an example to present its functions and their working mechanisms.

On the server side, the encoder is using either the Speex or the SILK codec, respectively. It is responsible for reading in the pre-recorded audio files, encoding them into small packets and sending them out to the network using the RTP protocol. On the client side, the first module is a packet collector. It is used to receive the coming packets and put them into a circular buffer. Then the decoder retrieves the packets from the buffer to decode them. At the end of the processing chain, the audio files are re-constructed. After the all these steps, we use the PESQ algorithm to calculate the PESQ value for the received audio files.

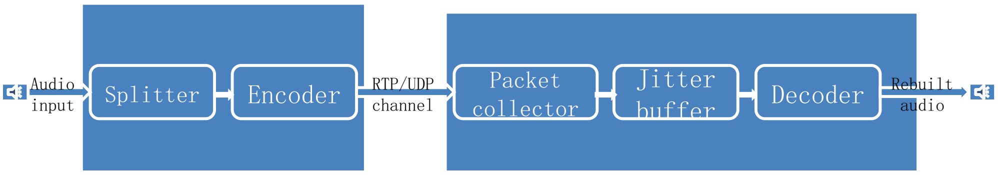


Figure 3.2: The overall structure of the client/server test model.

The original audio files are used as the reference files. The overall structure of our client and server model is shown in Figure 3.2.

No matter whether the codec is Speex or SILK, the client and server work in the same way.

- On the Server side, the functionality for each part is described below:

  Splitter. The audio files are pre-saved on the machines. We need to split the audio files into small pieces to send. The splitter is used to do this job. It splits audio files into frames where each frame is 20 milliseconds long. The frames are then sent to the encoder for the next step.

  Encoder. The encoder receives the frames from the splitter and encodes them. Subsequently the encoded frames are packed into packets and sent through the network.

- On the client side, the functionality for each part is described below:

  Packet collector. The packet collector is always waiting for the incoming packets from the server. If a new packet arrives, the packet collector will send it to the jitter buffer. Every packet has a unique sequence number. All the received packets will be stored into specific buffer positions according to their sequence numbers.

Jitter buffer. The packets are stored in the jitter buffer before they are decoded. The jitter buffer is used to smooth the packet reception. As the packets are sent using the UDP, some of them may be lost; or due to network latency, the packets will not arrive in sequence. Hence, the jitter buffer is very useful for re-ordering the packets. In addition, with the help of the jitter buffer, the audio quality can be improved to some extent. We can choose the decoding time point. For example, if some packets are arriving out of order, we may choose the decoding time late enough such that the all the out of order packets are decoded correctly. The upper bound of this method is that only after all the packets for an audio file have been received, we begin to send them to the decoder. Considering our real-time requirement, we may choose a point to balance this tradeoff.

Decoder. The decoder is responsible for retrieving the packets from the jitter buffer and decoding them. After this step, the decoded data is stored into a file which represents the reconstructed audio file.

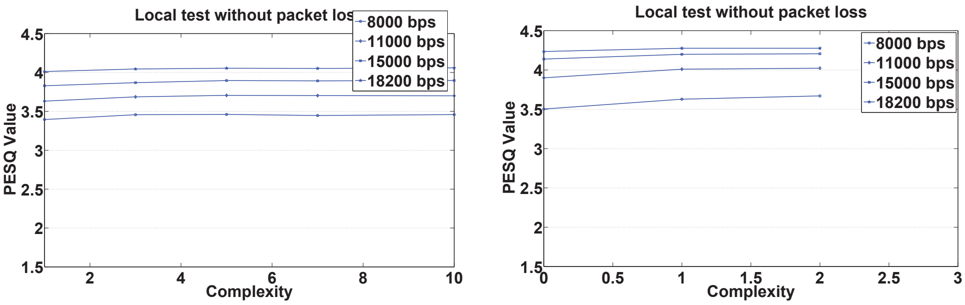
- For the media transmission protocol we use RTP in our system. It is very commonly used in streaming systems. It is often employed together with RTCP. Its lower level protocol used is UDP. UDP provides best-effort, unreliable service.

3.3 Test Results

In this part we will describe the experiments that we performed on PlanetLab as well as on the LAN testbed with both the Speex and the SILK codecs. To achieve repeatable results for our measurements, we used pre-stored audio files as input to simulate live streaming. Our setup allowed us to control the decode time to change the receiver delay of the streaming. Here in our tests we set the latency to be 120 milliseconds. The latency is calculated as follows. Each packet accounts for 20 milliseconds of audio data and we begin to decode when there are at least 6 packets are in the buffer. Different combinations of parameters were tested to explore the design space between the quality and latency requirements. During each tests, we varied exactly one parameter between the Speex and the SILK codecs to observe its influence on the results. Our different tests are introduced one by one in the remaining part.

3.3.1 Tests of Narrowband Mode on PlanetLab

When the test mode is narrowband, the sampling rate of the audio is set to 8 kHz. In narrowband mode, by changing the bit-rate and complexity parameters, we obtained the results as shown in Figures 3.3 up to 3.11 below. For the first set of graphs, we keep the bit rate constant and change the complexity parameter. Here we ran all the combinations of bit rate and complexity on a local machine and network for both the Speex and the SILK codecs to verify the effects of these two parameters.



(a) Speex codec

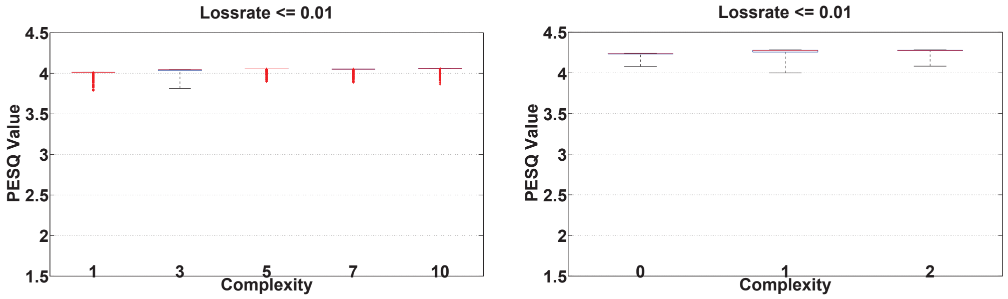
(b) SILK codec

Figure 3.3: PESQ values for all the combinations of bit rate and complexity parameters for the Speex and the SILK codecs.

The Figures 3.3(a) and 3.3(b) show that when the complexity stays the same, with increasing bit rate, the PESQ values will increase accordingly. When the bit rate stays the same, but the complexity increases, the PESQ values will also increase a little bit. From these figures we can observe that when the bit rate is 18,200 kbps, the transmitted audio will reach the highest quality. Hence, we set the bit rate to 18,200 kbps for the PlanetLab tests.

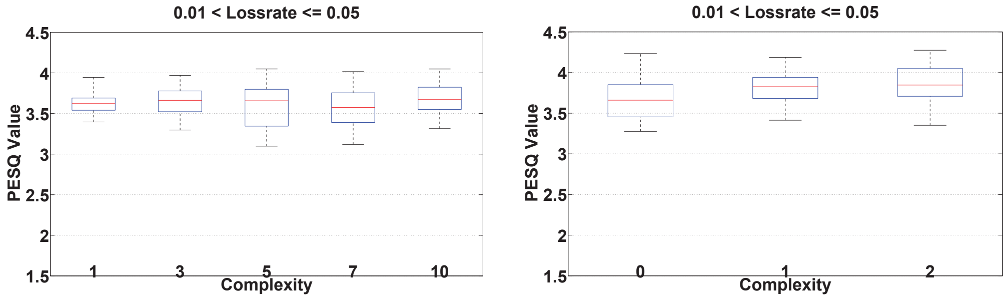
Measurements with Different Complexity

For the Speex codec, the complexity changes from 1 to 10 and the result from PlanetLab is illustrated in Figure 3.4(a). From the figure we can find that the loss rate of the streaming audio has a direct relationship with the PESQ value. When the loss rate is small, it means that there are very few packets lost, hence the PESQ value is high and the quality of the received audio is quite good, and vice versa. On the other hand, when the packet loss rate is high, the PESQ value may also be very high illustrated in the Figures 3.5(a), 3.6(a) and 3.7(a) where the maximum PESQ value is very high. This is quite related with the point in time when the packet loss occurred. If packet losses happened at a time when the speech content is silent, then no matter the loss rate, the PESQ can still be high. If, on the other hand, packet losses happen at a time when the content of the audio is a strong voice signal, then the PESQ will be low. In Figure 3.4(a), there is no box for the box plot, only some lines or red pluses ("+"). This is due to the PESQ values, *i.e.*, when the loss rate is low at 0.01, the audio quality is very good, there are very small packet losses. The PESQ value is high and nearly the same, and the median value, $Q1$ and $Q3$ are nearly the same, so there are no boxes visible in Figure 3.4(a). When the complexity is 10 in Figure 3.4(a), the median values, $Q1$ and $Q3$ are exactly the same, and according to Equations 3.1 and 3.2, all the dataset below $Q1$ and above $Q3$ are outliers, so there are many outliers as defined by the box plot. In fact, they are not outliers, their values are just not within the box boundary. These values are also correct. The whisker set here is 1,000 to make sure that as many values as possible in the data set are treated as normal values. For the SILK codec, the complexity changes from 0 to 2 and the result from PlanetLab is shown in Figure 3.4(b). The description for the SILK codec is very similar to the result for the Speex codec. From these figures about the Speex and the SILK codec when the complexity parameter is changing we can find that when the loss rate is nearly the same, the PESQ values for the SILK codec are higher than those for Speex codec under the condition that both codecs use the same bit rate. The results for the SILK codec are shown in the Figures 3.5(b), 3.6(b) and 3.7(b).



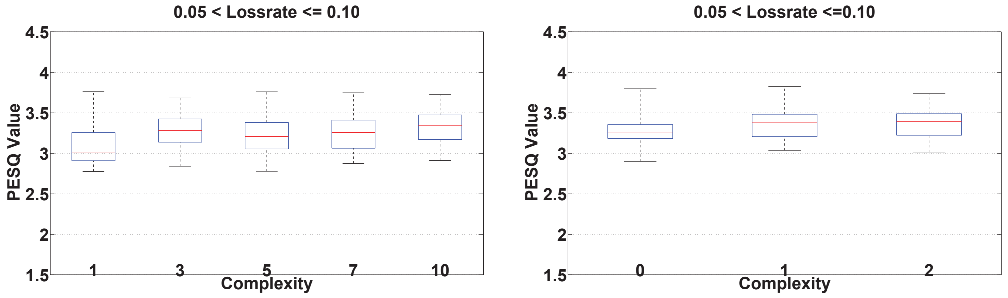
(a) Speex codec (b) SILK codec

Figure 3.4: PESQ values for loss rate ≤ 0.01 when the bit rate is 18,200 kbps for the Speex and the SILK codecs.



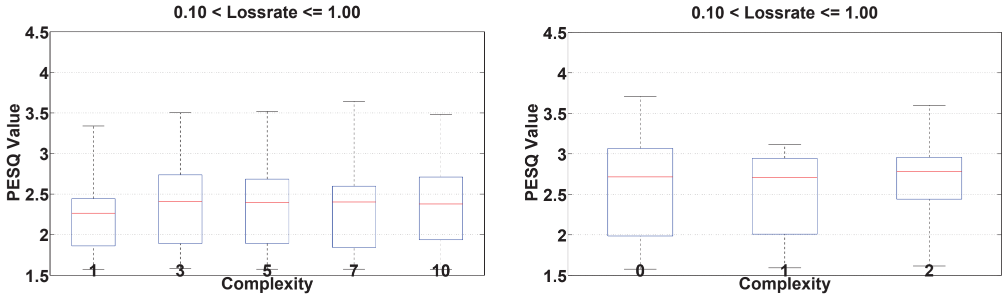
(a) Speex codec (b) SILK codec

Figure 3.5: PESQ values for $0.01 <$ loss rate ≤ 0.05 when the bit rate is 18,200 kbps for the Speex and the SILK codecs.



(a) Speex codec (b) SILK codec

Figure 3.6: PESQ values for $0.05 <$ loss rate ≤ 0.1 when the bit rate is 18,200 kbps for the Speex and the SILK codecs.



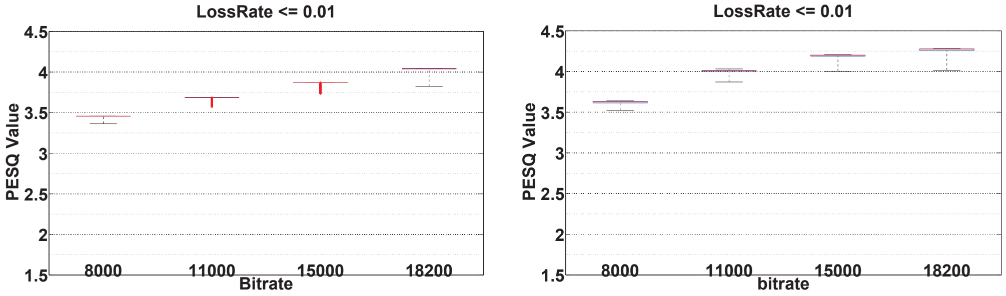
(a) Speex codec (b) SILK codec

Figure 3.7: PESQ values for $0.1 <$ loss rate ≤ 1.0 when the bit rate is 18,200 kbps for the Speex and the SILK codecs.

From the Figures 3.4, 3.5, 3.6 and 3.7, we can find that when the bit rate is constant, the best value for the complexity for the Speex codec is 3 and for the SILK codec it is 1. Overall, the SILK codec performs better than the Speex codec under the condition that both codecs select the same bit rate and keep it constant during this set of experiments.

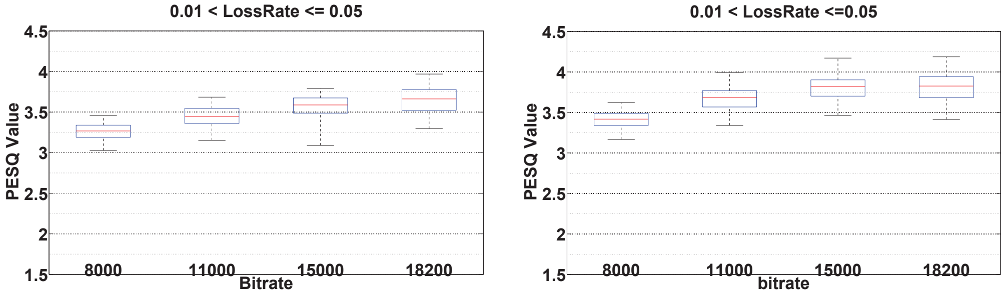
Measurements with Different Bit Rates

For this set of experiments, we will keep the complexity for the Speex codec at 3 and for SILK at 1, then change the bit rate to see the effects. The results for the Speex and for the SILK codecs are shown in Figures 3.8, 3.9, 3.10 and 3.11.



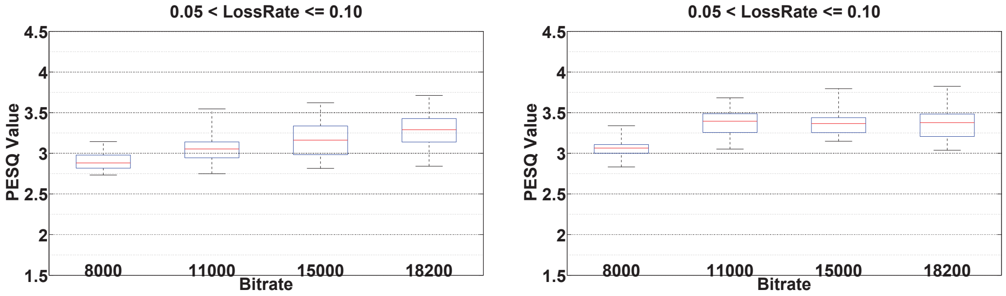
(a) Speex codec with complexity 3 (b) SILK codec with complexity 1

Figure 3.8: PESQ values for loss rate ≤ 0.01 for the Speex and the SILK codecs.



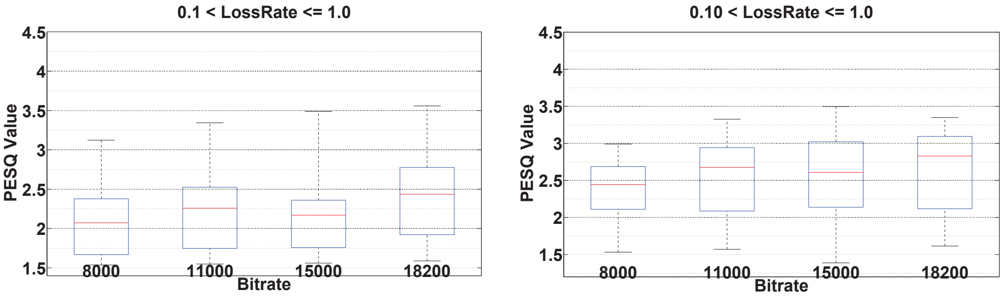
(a) Speex codec with complexity 3 (b) SILK codec with complexity 1

Figure 3.9: PESQ values for $0.01 <$ loss rate ≤ 0.05 for the Speex and the SILK codecs.



(a) Speex codec with complexity 3 (b) SILK codec with complexity 1

Figure 3.10: PESQ values for $0.05 <$ loss rate ≤ 0.1 for the Speex and the SILK codecs.



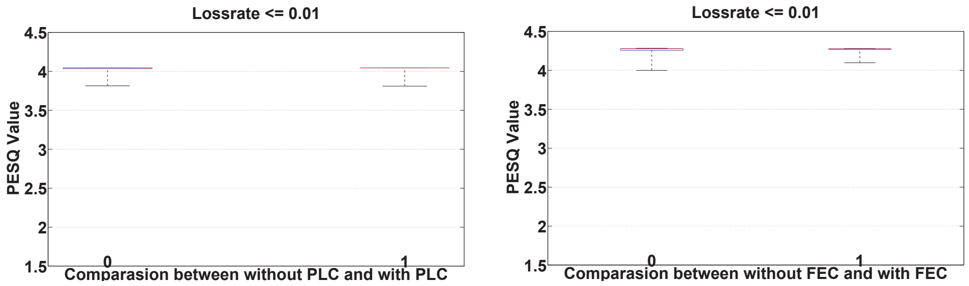
(a) Speex codec with complexity 3 (b) SILK codec with complexity 1

Figure 3.11: PESQ values for 0.1 < loss rate ≤ 1.0 for the Speex and the SILK codecs.

From Figures 3.8, 3.9, 3.10 and 3.11, we can find that when the loss rate is nearly the same for the Speex and the SILK codecs, the PESQ value for the SILK codec is higher than the Speex codec under the condition that the complexity for both codecs does not change during the experiments. For each codec, when the loss rate becomes higher, the PESQ tends to become smaller. This means that the quality of the reconstructed speech is becoming worse. The larger the bit rate, the better the audio quality, and correspondingly the higher the PESQ value. In general, SILK outperforms Speex in this set of experiments.

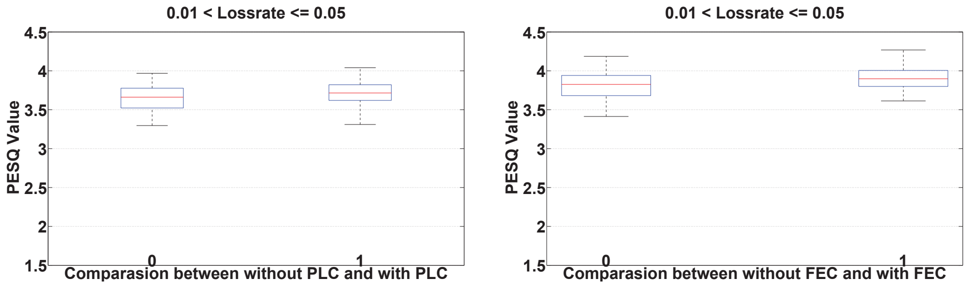
Measurements with FEC/PLC Enabled

To make sure each set of experiments conducted under the condition that only one parameter is different each time, all the above tests are performed without error correction, *i.e.*, FEC for SILK or PLC for Speex. Now we will investigate the effects of FEC for the SILK codec and PLC for the Speex codec. Here we keep the bit rates for both codecs at 18,200 kbps, and complexity for the Speex codec at 3 and for the SILK codec at 1. We obtain the results from PlanetLab as shown in Figures 3.12, 3.13, 3.14 and 3.15.



(a) Speex codec with complexity 3 (b) SILK codec with complexity 1

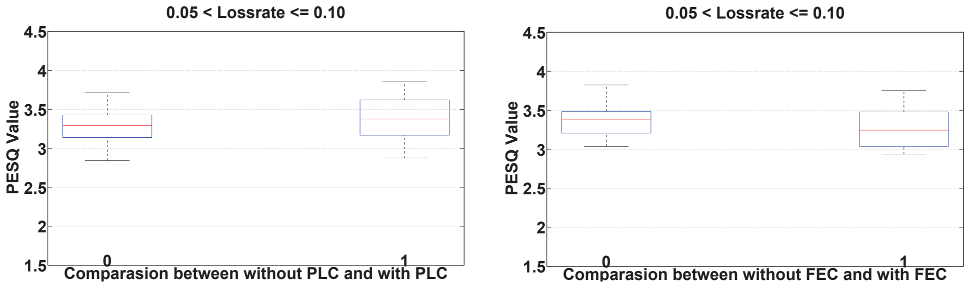
Figure 3.12: PESQ values for loss rate ≤ 0.01 with the bit rate is 18,200 kbps for the Speex and the SILK codecs.



(a) Speex codec with complexity 3 (b) SILK codec with complexity 1

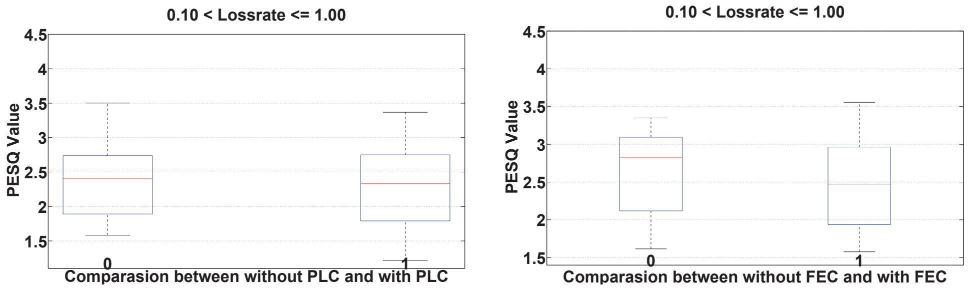
Figure 3.13: PESQ values for $0.01 <$ loss rate ≤ 0.05 with the bit rate is 18,200 kbps for the Speex and the SILK codecs.

From Figures 3.12, 3.13, 3.14 and 3.15 we can find that when PLC for the Speex codec and FEC for the SILK codec are enabled, the overall PESQ value for both codecs increases compared with the previous experiments when PLC for Speex and FEC for SILK were disabled. The boxes of the box plot become narrower and the average value is higher for both codecs, indicating that more values have increased and have more converged. In general, we can find that the SILK codec performs better than the Speex codec.



(a) Speex codec with complexity 3 (b) SILK codec with complexity 1

Figure 3.14: PESQ values for $0.05 <$ loss rate ≤ 0.1 with the bit rate is 18,200 kbps for the Speex and the SILK codecs.

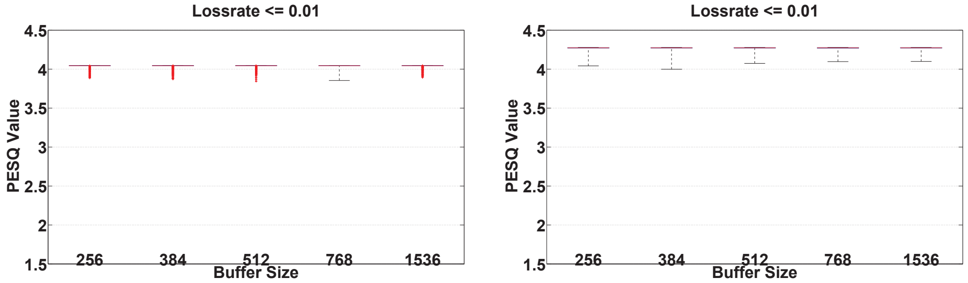


(a) Speex codec with complexity 3 (b) SILK codec with complexity 1

Figure 3.15: PESQ values for $0.1 <$ loss rate ≤ 1.0 with the bit rate is 18,200 kbps for the Speex and the SILK codecs.

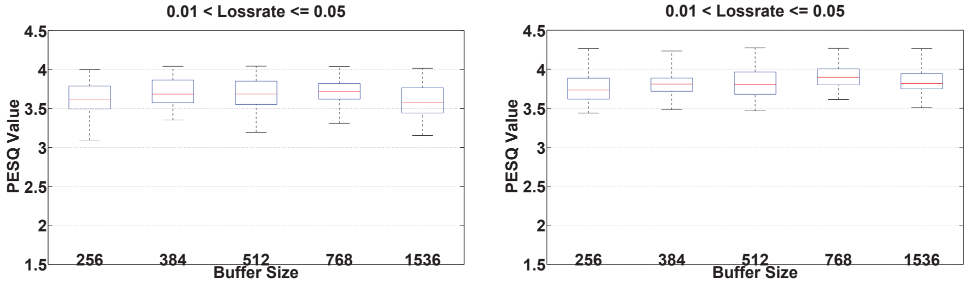
Measurements with Different Buffer Size

The buffer size can be used to control the decoding time. The buffer is used to store the incoming packets. After some time, they will be transmitted to the decoder and it will process these packets. The packets are decoded and stored into audio files. The larger the buffer size, the longer the latency will be. Here, we want to illustrate how the buffer size will affect the audio quality and the PESQ value. The results for Figures 3.16, 3.17, 3.18 and 3.19 were received from PlanetLab. We keep the complexity for the Speex codec at 3 and for the SILK codec at 1. The bit rates for both codecs are set to 18,200 kbps.



(a) Speex codec with complexity 3 (b) SILK codec with complexity 1

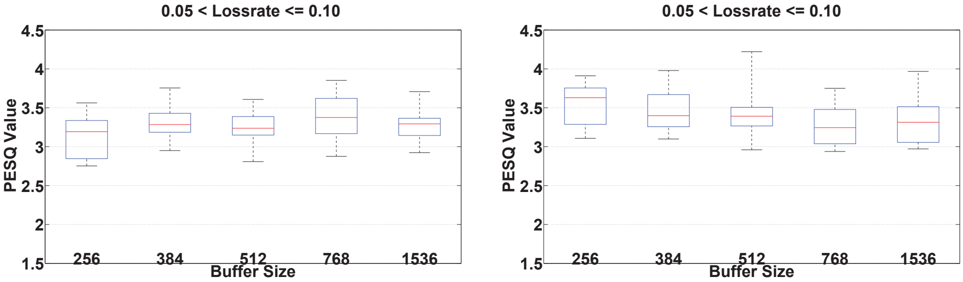
Figure 3.16: PESQ values for loss rate ≤ 0.01 with the bit rate is 18,200 kbps for the Speex and the SILK codecs.



(a) Speex codec with complexity 3 (b) SILK codec with complexity 1

Figure 3.17: PESQ values for 0.01 < loss rate ≤ 0.05 with the bit rate is 18,200 kbps for the Speex and the SILK codecs.

From Figures 3.16, 3.17, 3.18 and 3.19 we can find when the buffer size is increased, the quality of transmitted audio may increase a little bit, but not much. This is due to the decoding process for our system. As long as the number of the packets in the buffer reaches a certain minimum, the decoder will fetch the packets from the buffer and decode them directly one after another. So if the network conditions are good and there is no packet loss – moreover, if the packets are coming in sequence – then the received packets will be decoded in sequence. In this case, the audio quality will be quite good, just as when the buffer size is very large. When the network conditions are not so good, then the larger the buffer size the better the quality will be. As the buffer will be used to re-sequence out-of-order packets and also a longer waiting time means a higher probability to receive incoming packets.

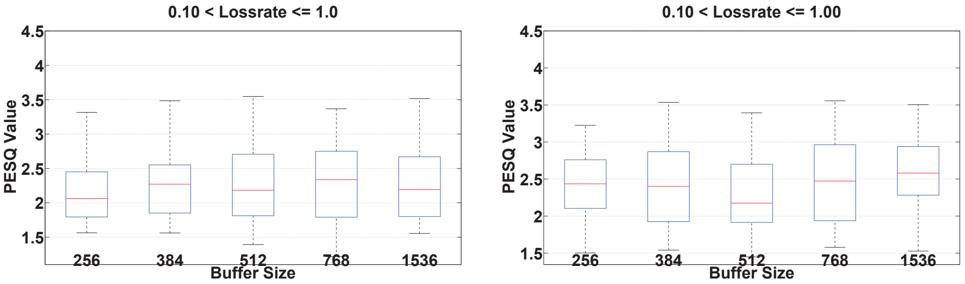


(a) Speex codec with complexity 3 (b) SILK codec with complexity 1

Figure 3.18: PESQ values for $0.05 <$ loss rate ≤ 0.1 with the bit rate is 18,200 kbps for the Speex and the SILK codecs.

3.3.2 Tests on Wideband Mode on PlanetLab

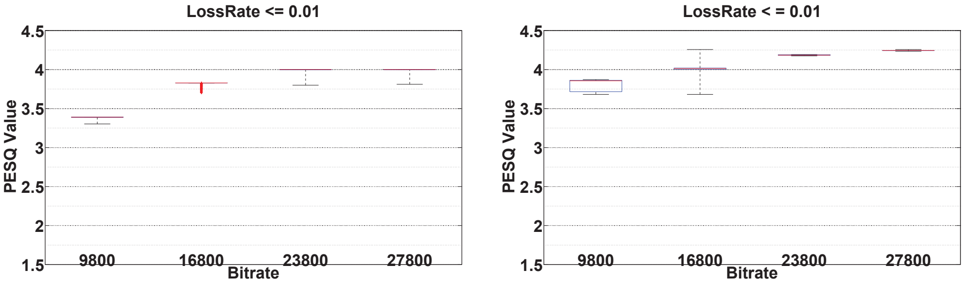
So far we have explored the quality of speech streamed on PlanetLab for both Speex and SILK in narrowband mode. In this section, we investigate the



(a) Speex codec with complexity 3 (b) SILK codec with complexity 1

Figure 3.19: PESQ values for 0.1 < loss rate ≤ 1.0 with the bit rate is 18,200 kbps for the Speex and the SILK codecs.

wideband mode. While in wideband mode, the sampling rate of the audio must be set at 16 kHz. We will keep the complexity for the Speex codec at 3 and for the SILK codec at 1. By changing the bit rate, we are interested to see which codec performs better. We observe the results as shown in Figures 3.20(a) and 3.20(b) for the Speex and the SILK codecs, accordingly.



(a) Speex codec with complexity 3 (b) SILK codec with complexity 1

Figure 3.20: PESQ value without packet loss for the Speex and the SILK codecs in wide band mode.

From Figures 3.20(a) and 3.20(b), we can find that when bit rate is 16,800 kbps, the PESQ value range for SILK is larger than that for Speex, but the middle 50% of the results for SILK are larger than those for Speex. This may be due to some errors. These values can be treated as outliers. In general, we can see that SILK performs better than Speex. From this set of results we can conclude from all the experiments introduced in the previous section that, in general, SILK outperforms Speex. However, the difference is not that large.

3.3.3 Tests on Narrowband Mode on LAN Testbed

All the previous experiments are conducted on PlanetLab where we have no control over how packet losses happen. We can only collect the result data according to the momentarily occurring loss rates. Here, in this section, we will use a packet loss model, the Gilbert Model [11], to simulate the packet loss in a LAN testbed. With this model, we can control the loss rate.

The Gilbert Model is a simple model to induce burst packet losses. There are two states for the model as shown in Figure 3.21. G represents the good state which means that expected packets reach the destination, and B represents the bad state which means that the expected packets are lost. The probability for the two states should satisfy $e_G < e_B$. For our LAN testbed, we set $e_G = 0$, which means the good state is error free and we will only consider packet loss scenarios. In this model, p and r are the probabilities to change from the good state to the bad state and vice versa.

The average packet loss rate is calculated as denoted in Equation 3.3 and the burst packet loss rate is modeled by choosing appropriate values for p and r.

$$P = \frac{p}{p+r} \tag{3.3}$$

For our LAN experiments and for the burst loss case, we choose several sets

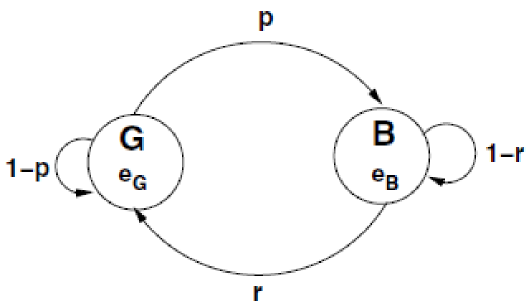


Figure 3.21: The two-state Gilbert Model [13].

of p and r to control the loss rate. This set of experiments were carried out under the condition that the complexity for Speex is set at 3 and for SILK set at 1, while the bit rate stays at 18,200 bps. Our test results are shown in Table 3.1. From this table, we can find that when using the Gilbert Model to simulate the burst length of packet losses, the quality of the transmitted speech becomes quite poor. The PESQ value is low. Under similar conditions, we can see that SILK outperforms Speex. When we compare this set of data with the previous results we obtained, we can find that the PESQ values for the burst length of packet losses in the LAN environment is in the same range as the results we received from PlanetLab.

Table 3.1: PESQ values for burst length of packet loss with the Gilbert Model on LAN.

Loss rate	Speex codec	SILK codec
1%	3.6164	3.8784
2%	3.3971	3.6758
5%	3.0796	3.3297
10%	2.6734	2.9092

3.3.4 Summary

From the results we measured we found that the SILK codec performs better than Speex codec in general. For each codec, when the audio quality is better, it will have a higher PESQ value and the PESQ value can to some extend measure the audio quality for our streaming system. We also obtained a set of PESQ values which are calculated on a local machine where the local machine works both as a client and a server under zero loss conditions as illustrated in Tables 3.2, 3.3 and 3.4. Overall, no matter whether we simulated packet losses on the LAN or not, the test results were in the same range as the results we received from the real network, the PlanetLab.

Table 3.2: PESQ values when the complexity stays the same in narrow band mode for LAN.

Bit rate	Speex codec	SILK codec
8,000	3.4568	3.6364
11,000	3.6864	4.0116
15,000	3.8687	4.1983
18,200	4.0447	4.2779

Table 3.3: PESQ values when the bit rate stays the same in narrow band mode for LAN.

Complexity	Speex codec	SILK codec
0		4.2551
1	4.1075	4.2862
2		4.2855
3	4.1267	
5	4.1496	
7	4.1418	
10	4.1468	

Table 3.4: PESQ values when the complexity stays the same in wide band mode for LAN.

Bit rate	Speex codec	SILK codec
9,800	3.3896	3.7997
16,800	3.8284	4.0186
23,800	4.0003	4.1823
27,800	4.0003	4.2439

Chapter 4

Conclusions and Future Work

4.1 Conclusions

Over the last decade streaming media has become a topic of much interest for researchers. The study of quality measurements of streaming media also has attracted interests. For this thesis we developed a small peer-to-peer testing system designed for PlanetLab to stream audio files with the aim of evaluating the speech quality of two widely used VoIP codecs, Speex and SILK.

We first introduced our motivation and objectives as well as the contributions of the thesis. Then, we performed a thorough literature survey on audio codecs and measurements studies of VoIP applications, including subjective and objective measurements. In Chapter 3, we explained in detail our testing system as well as all our experiments that we executed on PlanetLab and a LAN testbed. We selected different combinations of parameters for each set of experiments, both for Speex and SILK, under the condition that only one parameter was changed at a time. We studied how the complexity, the bit rate, the buffer size and the methods of PLC for Speex or FEC for SILK affected the quality of the streamed audio. All the test results were evaluated based on the PESQ metric and graphed using box plots. From these results, we were able to conclude that SILK outperforms Speex in general by a measurable, though not very large, margin.

4.2 Future Work

Though we have successfully fulfilled our initial objectives, we found some limitations in our testing system. Currently, our system uses point-to-point connections, *i.e.*, at a client it can only receive audio from one server, and conversely, it can also only send to one client from a server. In order to make the testing system easier to use and test multi-point conferencing scenarios, it may be useful to enhance the software to be able to send audio to more clients simultaneously as a server and receive audio from several servers at a client at the same time.

Bibliography

[1] A. Albanese, J. Blomer, J. Edmonds, M. Luby, and M. Sudan. Priority Encoding Transmission. *Information Theory, IEEE Transactions on*, 42(6):1737–1744, 1996.

[2] A. Albanese and M. Luby. PET-Priority Encoding Transmission. In *Proceedings of the 2nd International Workshop on Architecture and Protocols for High Performance Networks: High-Speed Networking for Multimedia Applications*, page 265. Kluwer, BV, 1995.

[3] J. Bolot, S. Fosse-Parisis, and D. Towsley. Adaptive FEC-based Error Control for Internet Telephony. In *INFOCOM'99. Eighteenth Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings. IEEE*, volume 3, pages 1453–1460. IEEE, 2002.

[4] J. Bolot and A. Vega-Garcia. The Case for FEC-based Error Control for Packet Audio in the Internet. *ACM Multimedia Systems*, 1997.

[5] J. Bolot and A. Vega-Garcia. Control Mechanisms for Packet Audio in the Internet. In *INFOCOM'96. Fifteenth Annual Joint Conference of the IEEE*

Computer Societies. Networking the Next Generation. Proceedings IEEE, volume 1, pages 232–239. IEEE, 2002.

[6] G. Carle and E. Biersack. Survey of Error Recovery Techniques for IP-based Audio-visual Multicast Applications. *IEEE Network*, 11(6):24–36, 1997.

[7] L. Carvalho, E. Mota, R. Aguiar, A. Lima, and J. de Souza. An E-Model Implementation for Speech Quality Evaluation in VoIP Systems. In *Computers and Communications, 2005. ISCC 2005. Proceedings. 10th IEEE Symposium on*, pages 933–938. IEEE, 2005.

[8] W. Chiang, W. Xiao, and C. Chou. A Performance Study of VoIP Applications: MSN vs. Skype. *Proc. of MULTICOMM*, 2006.

[9] B. Dempsey, J. Liebeherr, and A. Weaver. On Retransmission-based Error Control for Continuous Media Traffic in Packet-Switching Networks. *Computer Networks and ISDN Systems*, 28(5):719–736, 1996.

[10] B. Dempsey, M. Lucas, and A. Weaver. An Empirical Study of Packet Voice Distribution over a Campus-Wide Network. In *Proceedings of the IEEE 19 th Conference on Local Computer Networks*. Citeseer, 1994.

[11] E. Gilbert et al. Capacity of a Burst-noise Channel. *Bell Syst. Tech. J*, 39(9):1253–1265, 1960.

[12] M. Hayasake, M. Gamage, and T. Miki. Referential Loss Recovery for Streaming Audio using Application Level Multicast. In *Communications, 2005 Asia-Pacific Conference on*, pages 264–268. IEEE, 2005.

[13] O. Hohlfeld, R. Geib, and G. Haßlinger. Packet Loss in Real-time Services: Markovian Models Generating QoE Impairments. In *Quality of Service, 2008. IWQoS 2008. 16th International Workshop on*, pages 239–248. IEEE, 2008.

[14] T. Huang, P. Huang, K. Chen, and P. Wang. Could Skype be more Satisfying? A QoE-centric Study of the FEC Mechanism in an Internet-scale VoIP System. *Network, IEEE*, 24(2):42–48, 2010.

[15] C. Huitema. RFC3605: Real Time Control Protocol (RTCP) Attribute in Session Description Protocol (SDP). *RFC Editor United States*, 2003.

[16] R. ITU-T. 862-Perceptual Evaluation of Speech Quality (PESQ): An Objective Method for End-to-End Speech Quality Assessment of Narrow-band Telephone Networks and Speech Codecs. *International Telecommunication Union-Telecommunication Standardisation Sector (ITU-T)*, 2001.

[17] N. Jayant et al. *Signal Compression: Coding of Speech, Audio, Text, Image and Video*. World Scientific, 1997.

[18] K. Jonas, P. Kanzow, and M. Kretschmer. Audio Streaming on the Internet. Experiences with Real-time Streaming of Audio Streams. In *Industrial Electronics, 1997. ISIE'97., Proceedings of the IEEE International Symposium on*, volume 1. IEEE, 2002.

[19] J. Kang, I. Na, Y. Choi, and J. Kim. A Study of Subjective Speech Quality Measurement over VoIP Network. In *Info-tech and Info-net, 2001. Pro-*

ceedings. ICII 2001-Beijing. 2001 International Conferences on, volume 5, pages 311–316. IEEE, 2002.

[20] E. Klemmer. Subjective Evaluation of Transmission Delay in Telephone Conversations. *Bell System Technical Journal*, 46(6):1141–1147, 1967.

[21] J. Liu and G. Wei. Model the Packet-loss Dependent Effective Equipment Impairment Factor in Speech Quality Estimation in VoIP and its Realization. In *Advanced Computer Control (ICACC), 2010 2nd International Conference on*, volume 4, pages 359–362. IEEE, 2010.

[22] S. McCanne and V. Jacobson. VIC: A Flexible Framework for Packet Video. In *Proceedings of the third ACM international conference on Multimedia*, pages 511–522. ACM, 1995.

[23] S. McCanne, V. Jacobson, and M. Vetterli. Receiver-driven Layered Multicast. In *Conference proceedings on Applications, technologies, architectures, and protocols for computer communications*, pages 117–130. ACM, 1996.

[24] P. Paglierani and D. Petri. Uncertainty Evaluation of Objective Speech Quality Measurement in VoIP Systems. *Instrumentation and Measurement, IEEE Transactions on*, 58(1):46–51, 2008.

[25] P. Parnes. RTP Extension for Scalable Reliable Multicast. *Work in Progress*, 1996.

[26] K. Pentikousis, J. Pinola, E. Piri, and F. Fitzek. A Measurement Study of Speex VoIP and H.264/AVC Video over IEEE 802.16d and IEEE 802.11g. *Computers and Communications*, pages 19–24, 2008.

[27] J. Postel et al. Internet Protocol, 1981.

[28] J. Postel and U. Protocol. RFC 768. *User Datagram Protocol*, 1980.

[29] L. Rabiner and R. Schafer. *Digital Processing of Speech Signals*, volume 100. Prentice-hall Englewood Cliffs, NJ, 1978.

[30] I. Rec. G. 107-The E Model, a Computational Model for use in Transmission Planning. *International Telecommunication Union*, 2003.

[31] A. Rix, J. Beerends, M. Hollier, and A. Hekstra. Perceptual Evaluation of Speech Quality (PESQ)-A New Method for Speech Quality Assessment of Telephone Networks and Codecs. In *IEEE International Conference on Acoustics Speech and Signal Processing*, volume 2. Citeseer, 2001.

[32] A. Rix and M. Hollier. The Perceptual Analysis Measurement System for Robust End-to-End Speech Quality Assessment. In *IEEE International Conference on Acoustics Speech and Signal Processing*, volume 3. IEEE; 1999, 2000.

[33] D. Salomon. *Data Compression: The Complete Reference.* Springer-Verlag New York Inc, 2007.

[34] B. Sat and B. Wah. Evaluation of Conversational Voice Communication Quality of the Skype, Google-Talk, Windows Live, and Yahoo Messenger

Voip Systems. In *Multimedia Signal Processing, 2007. MMSP 2007. IEEE 9th Workshop on*, pages 135–138. IEEE, 2007.

[35] H. Schulzrinne, S. Casner, R. Frederick, and V. Jacobson. RTP: A Transport Protocol for Real-time Applications, 1996.

[36] N. Shacham and P. McKenney. Packet Recovery in High-speed Networks using Coding and Buffer Management. In *Proc. IEEE Infocom*, volume 1, pages 124–131, 1990.

[37] N. Shacham and P. McKenney. Packet Recovery in High-speed Networks using Coding and Buffer Management. In *INFOCOM'90. Ninth Annual Joint Conference of the IEEE Computer and Communication Societies.'The Multiple Facets of Integration'. Proceedings., IEEE*, pages 124–131. IEEE, 2002.

[38] R. Storn. Modeling and Optimization of PET-Redundancy Assignment for MPEG Sequences. *International Computer Science Institute-Publications-TR*, 1995.

[39] J. Valin. The Speex Codec Manual Version 1.2 Beta 3. *Xiph. org Foundation*, 2007.

[40] K. Vos, S. Jensen, and K. Soerensen. Silk Speech Codec. Technical report, IETF Internet-Draft draft-vossilk-00. txt, 2009.

[41] H. Xie and Y. Yang. A Measurement-based Study of the Skype Peer-to-Peer VoIP Performance. In *Proc of IPTPS*. Citeseer, 2008.

[42] A. Xu, W. Woszczyk, Z. Settel, B. Pennycook, R. Rowe, P. Galanter, J. Bary, G. Martin, J. Corey, and J. Cooperstock. Real-time Streaming of Multichannel Audio Data over Internet. *Preprints-Audio Engineering Society*, 2000.

[43] X. Xu, A. Myers, H. Zhang, and R. Yavatkar. Resilient Multicast Support for Continuous-Media Applications. In *Proceedingsof the 7th International Workshop on Network and Operating Systems Support for Digital Audio and Video (NOSSDAV97), Washington University in St. Louis, Missouri.* Citeseer, 1997.