An Empirical Evaluation of VoIP Playout Buffer Dimensioning in Skype, Google Talk, and MSN Messenger [*]

Chen-Chi Wu[1], Kuan-Ta Chen[2], Chun-Ying Huang[3], and Chin-Laung Lei[1]

[1]Department of Electrical Engineering, National Taiwan University
[2]Institute of Information Science, Academia Sinica
[3]Department of Computer Science and Engineering, National Taiwan Ocean University

ABSTRACT

VoIP playout buffer dimensioning has long been a challenging optimization problem, as the buffer size must maintain a balance between conversational interactivity and speech quality. The conversational quality may be affected by a number of factors, some of which may change over time. Although a great deal of research effort has been expended in trying to solve the problem, how the research results are applied in practice is unclear.

In this paper, we investigate the playout buffer dimensioning algorithms applied in three popular VoIP applications, namely, Skype, Google Talk, and MSN Messenger. We conduct experiments to assess how the applications adjust their playout buffer sizes. Using an objective QoE (Quality of Experience) metric, we show that Google Talk and MSN Messenger do not adjust their respective buffer sizes appropriately, while Skype does not adjust its buffer at all. In other words, they could provide better QoE to users by improving their buffer dimensioning algorithms. Moreover, none of the applications adapts its buffer size to the network loss rate, which should also be considered to ensure optimal QoE provisioning.

Categories and Subject Descriptors

C.4 [**Performance of Systems**]: Measurement techniques; H.4.3 [**Information Systems Applications**]: Communications Applications—*Computer conferencing, teleconferencing, and videoconferencing*; H.1.2 [**Models and Principles**]: User/Machine Systems—*Human factors*

General Terms

Human Factors, Measurement, Performance

Keywords

E-Model, MOS, PESQ, Quality of Experience, User Satisfaction, VoIP

1. INTRODUCTION

VoIP is becoming an important communication service both within and between enterprises, while individuals are relying on it increasingly for daily communications with family and friends. There are two reasons for this phenomenon: the cost of VoIP calls is low and the voice quality is almost the same as that of traditional toll telephones. The trend is exemplified by the fact that Skype, one of the most widely used VoIP applications, has 405 million registrars and 15 million online users[1]. Because of the steady growth in VoIP usage, providing reliable services with satisfactory voice quality is now a high priority for Internet and VoIP service providers.

A number of factors may affect the service quality of VoIP, e.g., the speech codec, transport protocol, redundancy/error control, network path selection, and playout buffer dimensioning. In this work, we focus on the playout buffer dimensioning algorithms employed by popular VoIP applications.

Basically, *playout buffering*[2] *sacrifices conversational interactivity in exchange for better voice quality.* Normally, a voice packet is transmitted from the speaker's node to the listener's node every 20 ms or 30 ms to maintain continuous and smooth speech conversation. However, in packet-switched networks, packet queuing delays are variable and hard to predict, so some packets may arrive at the listener's node after long delays and the speech samples in the packets will be considered lost. This may result in silent periods, noise, or unclear speech, depending on the loss concealment algorithm adopted by the voice codec. To reduce the frequency of such occurrences, a playout buffer can be employed to hold a VoIP packet until its scheduled playout time. By so doing, packets that experience slightly longer network delays can still be used as long as they arrive at the listener's node ahead of their respective scheduled playout time.

The most challenging issue raised by VoIP playout buffering is *how to determine the most appropriate buffer size for current network conditions.* Generally, a larger buffer size leads to better sound quality, but it reduces conversational

---

[1]http://ebayinkblog.com/wp-content/uploads/2009/01/skype-fast-facts-q4-08.pdf

[2]Since we only playout buffer in this paper, we use "playout buffer" and "buffer" interchangeably hereafter.

interactivity. Thus, we can treat buffer size adjustment as an optimization problem where *the optimal buffer size should maintain a balance between conversational interactivity and speech quality.* The optimal buffer size is affected by several factors, such as network delay, delay variability (jitter), redundancy control, error correction, and codec implementations. The impact of these factors, especially network delay and network loss, may change over time. Therefore, an effective buffer dimensioning algorithm should consider the volatility of network conditions while maintaining the tradeoff between conversational interactivity and speech quality.

Several VoIP playout buffer dimensioning algorithms have been proposed [9–12]. Most of them adjust the buffer size based on a linear combination of network delay and jitter. The weights assigned to these two factors and the exact adjustment policies may vary according to different optimization goals and design considerations. Although a great deal of research effort has been expended on solving the buffer dimensioning problem, how the research results are applied in practice remains unclear.

In this paper, *we investigate whether a gap exists between the research results on VoIP and how those results are applied in practice from the perspective of playout buffer dimensioning algorithms.* We consider three popular VoIP applications, namely, Skype, Google Talk, and MSN Messenger, and conduct experiments to assess how they adjust their respective playout buffer sizes. We investigate whether they adjust their playout buffers correctly; and, if not, how much their performance differs from the optimum quality. In addition, we present a simple algorithm that computes the optimal buffer size based on objective QoE (Quality of Experience) metrics. Our results indicate that MSN Messenger achieves the best performance in terms of buffer dimensioning due to varying network conditions. Surprisingly, Skype does not adjust its playout buffer size at all.

Our contribution in this work is three-fold:

1. We propose an experiment methodology that can systematically measure the playout buffer size of any VoIP application and investigate the relationship between the measured buffer size and network conditions.

2. By using an objective QoE metric, we show that Google Talk and MSN Messenger do not adjust their buffer sizes appropriately, while Skype does not adjust its buffer at all. All three applications could provide better QoE to users by improving their buffer dimensioning algorithms. Moreover, none of them adapts the buffer size to the network loss rate, but this should also be considered to ensure optimal QoE provisioning.

3. We propose a simple regression-based algorithm that computes the optimal playout buffer size based on the current network conditions. Our approach has three advantages: 1) it is based on an objective user satisfaction measure that considers both conversational interactivity and speech quality; 2) it is simpler than existing approaches, as only a weighted sum operation is needed to compute the optimal buffer size; and 3) it can consider other network factors without making changes to the algorithm.

The remainder of this paper is organized as follows. Section 2 contains a review of related works. We describe the experiment methodology for measuring the playout buffer size of any VoIP application in Section 3, and then analyze the experiment results in Section 4. In Section 5, we detail the proposed approach for predicting the optimal playout buffer size based on current network conditions, and evaluate how appropriately the studied applications adjust their respective buffer sizes. We then summarize our conclusions in Section 6.

2. RELATED WORK

Several VoIP playout buffer dimensioning algorithms have been proposed to improve the audio quality of VoIP communications. In [11], the authors adjust the buffer size based on the EWMA (Exponential Weighted Moving Average) of network delays and their standard deviation (i.e., delay jitter), where the weights of the variables are selected empirically and fixed. Subsequently, [10] extended the above approach by adaptively adjusting the EWMA weight based on the magnitude of the delay jitter. The weight is set higher when the delay jitter is smaller; conversely, it is set lower when the jitter is larger. The reported simulation results show that this approach improves the tradeoff between buffer delay and packet loss significantly. The approaches proposed in [9, 12] further extended [10, 11] by adjusting the buffer size within a speech burst. The objective is to ensure that the playout buffer adapts to varying network conditions more quickly, and thereby improve the conversation quality of VoIP calls.

In recent years, a number of models have been developed to assess the quality of VoIP conversations. For example, PESQ (Perceptual Evaluation of Speech Quality) [8] is widely used to evaluate listening speech quality. PESQ is a signal-based method that compares the original speech signals with the degraded signals, and grades the quality of the latter using a mean opinion score (MOS), which ranges from 1 (Bad) to 5 (Excellent). E-Model [7] is another approach used to evaluate the quality of VoIP conversations. The quality score is the arithmetic sum of the delay impairment factor I_d, the equipment impairment factor I_e, and the factor I_s, which considers the degradation in quality due to speech compression/decompression and quantizing distortions. E-Model outputs a rating factor R (ranging from 0 to 100) that can be converted to an MOS by

$$MOS = \begin{cases} 1 & R < 0 \\ 1 + 0.0035R + R(R - 60)(100 - R) \cdot 7 \cdot 10^{-6} & 0 < R < 100 \\ 4.5 & R > 100. \end{cases} \quad (1)$$

However, PESQ and E-Model cannot accurately assess the quality-of-experience (QoE) of VoIP calls. This is because PESQ does not take interactivity into consideration, so the PESQ score can be high even if the end-to-end delay is too long to allow a coherent conversation. On the other hand, E-Model has the following disadvantages: 1) its listening quality assessment is less accurate than that derived by a signal-based algorithm such as PESQ; 2) it does not consider the variability of network delays and loss rates; and 3) it does not take account of the interaction between different factors, e.g., the interplay between network delay and listening quality, or between network delay and loss rates. To address the above problems, Ding et al. [5] proposed a hybrid model that integrates PESQ and E-Model and thereby provides the advantages of both models. We introduce this model in Section 5, and explain how we use it to evaluate the QoE of VoIP communications.
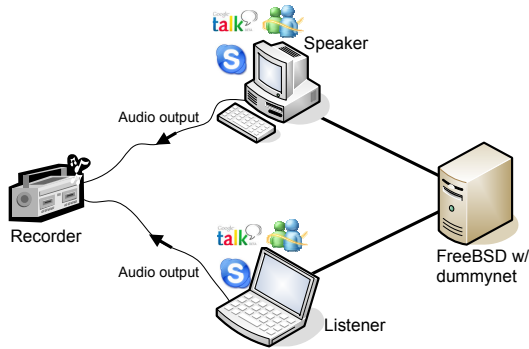
3. EXPERIMENT METHODOLOGY

Figure 1: The experiment setup

In this section, we describe the experiment setup and the procedures for evaluating the playout buffer size of Skype, Google Talk, and MSN Messenger under various network scenarios.

3.1 Experiment Setup

To evaluate the playout buffer sizes of the selected VoIP applications under different network conditions, we set up a FreeBSD 7.0 machine as a router and controlled the pace of traffic flows passing through it by `dummynet`. Two Microsoft Windows XP PCs installed with Skype (version 3.8), Google Talk (version 1.0), and MSN Messenger (version 2009, build 14.0) are connected to each other and to the Internet through the FreeBSD router. In the experiment, we designate one PC as the speaker, and the other as the listener. A speech recording is played continuously on the speaker PC, and via VoIP transmission, the listener PC receives a degraded copy of the content and outputs it as a degraded speech segment. To simulate real-life human conversations, we use speech recordings obtained from the Open Speech Repository [3].

During the experiment, we use a PC with an ESI Maya44 recording card to record the audio output of the speaker (i.e., the original speech segment), and that of the listener (i.e., the degraded speech segment) in a stereo wave file. The recording machine stores the speaker's output in the left channel, and the listener's output in the right channel of the wave file. The setup of the router, call parties, and recorder is illustrated in Fig. 1.

By configuring `dummynet` on the router, we can control the network delay, delay jitter (the standard deviation of the network delay), and the loss rate between the speaker PC and the listener PC. As every packet sent from the speaker must pass through the router to reach the listener, we can examine how the applications' playout buffer sizes change under different network conditions.

In the experiments, we set the network delay and delay jitter between 0 ms and 200 ms with a 25 ms interval, and the packet loss rate between 0% and 10% with a 1% interval. The duration of each VoIP call was 240 seconds, and 10 calls were made under each network setting. To allow sufficient time for the VoIP applications to adapt their playout buffer sizes to the latest network conditions, we started the wave file recording 60 seconds after a call had been established. We assume that the delay jitters follow a Gamma distribution. In addition, as the effect of restricted bandwidth can be simulated by injecting packet loss, the network bandwidth between the two call parties is set to a sufficiently large value, i.e., 1000 Kbps.

3.2 Buffer Size Estimation

To estimate the size of the playout buffers in the compared VoIP applications, we need to determine the end-to-end delay between the time the speaker starts speaking and the time the listener hears the spoken segment. The end-to-end delay can be estimated by computing the delay between the speaker's audio output and the listener's audio output, which are stored in the wave files. Therefore, we calculate the audio delay by searching for the time difference that yields the largest cross-correlation coefficient between the speech recordings output by the two parties [1].

In addition to the playout buffer delay, the end-to-end delay of VoIP transmission comprises network delay, coder delay, and packetization delay [4]. Since both call parties are in a LAN, all the network delay components can be controlled. In fact, the propagation delay and transmission delay are very small so they can be neglected. The coder delay and packetization delay are both application-dependent and codec-dependent; thus, we do not have exact information about delays caused by these components. However, a survey of the typical values used by popular codecs [4] shows that the sum of the coder delay and packetization delay is usually around 50 ms. Therefore, we estimate an application's playout buffer size by reducing the measured end-to-end delay by 50 ms and the average network delay induced by `dummynet`. Although the estimate may not be accurate, our objective is to determine how the compared VoIP applications adjust their playout buffer sizes under different network conditions. Thus, the absolute error in estimating the buffer size does not affect the buffer dimensioning behavior we observe or the conclusions we draw.

4. PLAYOUT BUFFER SIZE ADJUSTMENT IN REAL-LIFE APPLICATIONS

In this section, we discuss how Skype, Google Talk, and MSN Messenger adjust their respective VoIP playout buffer sizes under various network conditions.

4.1 Effect of Network Delay and Delay Jitter

The graph in Fig. 2 plots the VoIP playout buffer sizes of Skype, Google Talk, and MSN Messenger when VoIP packets experience different levels of network delay and delay jitter. The vertical bars on the graph represent the 95% confidence band of the average buffer size. Figures 2(a) and (c) show that the curves corresponding to different delays are similar, and the 95% confidence bands merge with each other. This finding indicates that Skype and MSN Messenger do not adjust their playout buffer sizes to compensate for the average network delay. On the other hand, the dissimilarity between the curves in Fig. 2(b) shows that Google Talk considers the average network delay and adapts its playout buffer size accordingly.

We also investigated the impact of delay jitter on the buffer size. As shown in Fig. 2(a), Skype's buffer size remains within the range $(250, 300)$ ms regardless of the magnitude of the delay jitter, which suggests that Skype does not adjust its buffer size in response to network delays. In contrast, both Google Talk and MSN Messenger increase their buffer sizes as the delay jitter increases. This design allows packets that experience longer queuing delays more opportunities to arrive and be used before the scheduled playout
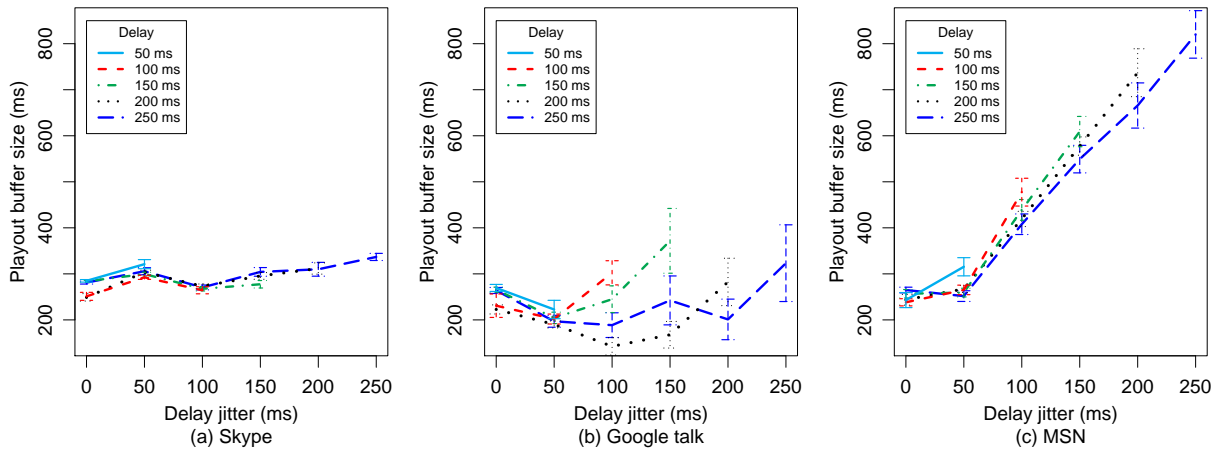
Figure 2: The playout buffer sizes of Skype, Google Talk, and MSN Messenger under different network delays and delay jitters
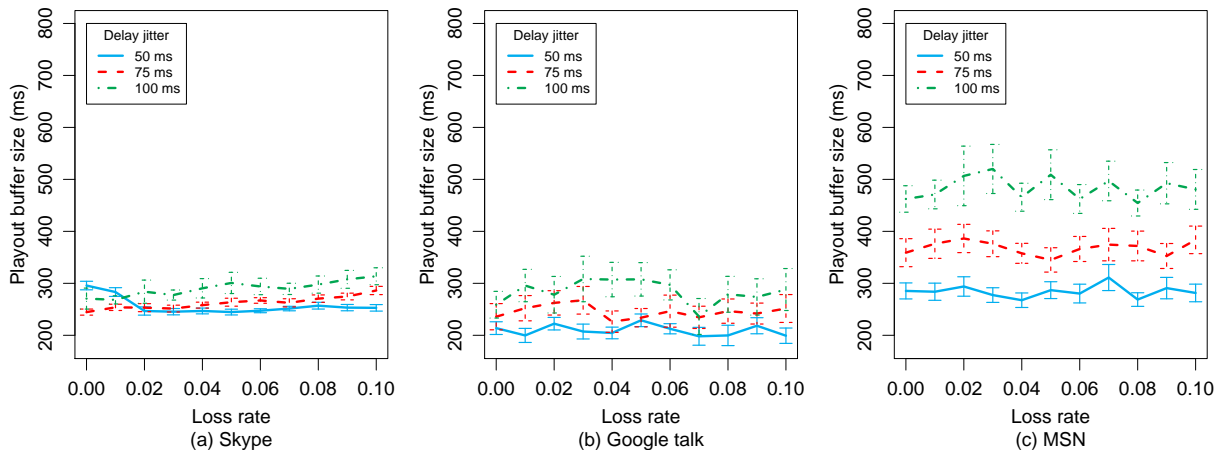

Figure 3: The playout buffer sizes of Skype, Google Talk, and MSN Messenger under different network loss rates

time. Specifically, MSN Messenger adjusts its buffer size linearly to adapt to increasing delay jitter, while Google Talk only increases its buffer size by a small amount, even when the amount of delay jitter is large. The diverse behavior of the two applications may result in different overall quality levels. We discuss this aspect further in Section 5.

4.2 Effect of Network Loss

Figure. 3 shows the applications' playout buffer sizes under different network loss rates, where the delay jitter is set to 50, 75, and 100 ms, with an average delay of 100 ms. Our objective is to determine whether Skype, Google Talk, and MSN Messenger take account of the network loss rate in their buffer dimensioning algorithms. The curves in Figs. 3(a)-(c) do not exhibit any significant adjustments to compensate for changes in the network loss rate, which suggests that none of the applications considers this factor in their buffer dimensioning algorithms.

In summary, our experiment results show that Skype maintains the same playout buffer size regardless of the network delay, delay jitter, and loss rate, while MSN Messenger's buffer size grows linearly as the delay jitter increases. In contrast, Google Talk adjusts its buffer size gradually based on the average network delay and delay jitter. None of the compared applications considers the packet loss rate in their

buffer dimensioning algorithms. Even so, we do not know which application's policy is the most effective, or how user satisfaction can be achieved through the applications' respective design choices. To address these issues, we propose a VoIP QoE measurement model to evaluate the effectiveness of the applications' buffer dimensioning algorithms in the following section.

5. PLAYOUT BUFFER OPTIMIZATION BASED ON USER SATISFACTION

In this section, we propose a methodology that derives the optimal VoIP playout buffer size based on an objective user satisfaction measure. We then compare the optimal buffer size with the buffer sizes measured for Skype, Google Talk, and MSN Messenger in Section 4 to determine whether the applications adjust their buffer sizes appropriately. Finally, we develop a regression-based algorithm that computes the optimal playout buffer size given the network configuration.

5.1 QoE Measurement

As mentioned in Section 2, the widely used E-Model cannot accurately estimate user satisfaction with VoIP conversations. Therefore, we employ the QoE measurement model proposed by Ding et al. [5] to quantify the overall QoE pro-
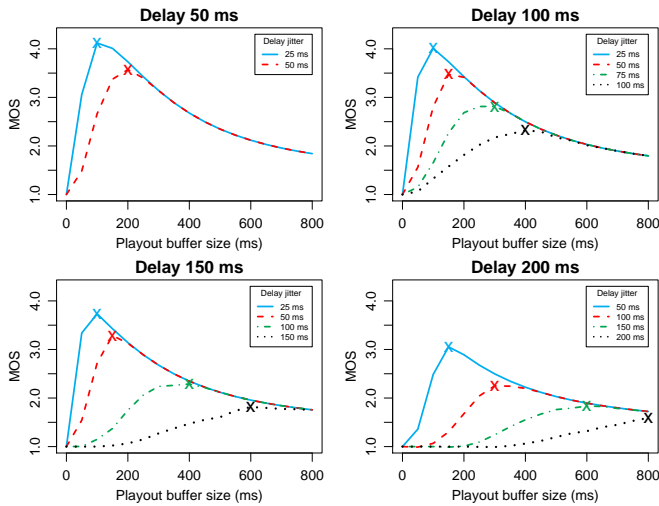
Figure 4: The simulation result of inferring the optimal buffer sizes for different network delays and delay jitters

Figure 5: The simulation result of inferring the optimal buffer sizes for different network loss rate

vided by a VoIP application. The major advantage of this model is that it combines the accurate listening quality assessment of PESQ and the interactivity assessment of E-model. Given an original audio clip and its degraded version, we compute the MOS score by the following procedures:

1. Apply PESQ to the original and degraded audio clips, and convert the resulting MOS score to an R score using the formula in ITU-T G.107, Appendix I [7].

2. Compute the delay impairment I_d in the E-Model based on the network delay. Other parameters of the E-Model remain unchanged.

3. Subtract the R score derived in step 1 from I_d obtained in step 2, and convert the resulting R score into a MOS score by Equation 1.

5.2 Optimal Buffer Size Derivation

The optimal playout buffer size yields the highest user satisfaction in a VoIP call. To derive the optimal buffer size under a given network condition, we designed a simulator that evaluates the quality of a VoIP conversation given the network configuration and the playout buffer size. Based on the VoIP QoE measurement model, we define the optimal playout buffer size as the size that yields the highest MOS score. The steps for determining the optimal buffer size are as follows:

1. Encode an audio clip into a sequence of VoIP frames by using the encoder library in the Intel Integrated Performance Primitives Library [2].

2. Simulate network packet loss with the Gilbert model. A packet will be dropped if the model is in the "Error" state; otherwise, it will be retained.

3. Introduce network delay to each packet via a Gamma distribution. If a packet's delay is longer than the current playout buffer size, it will be dropped; otherwise, it will be retained.

4. Decode the resulting stream of frames into a degraded audio clip.

5. Apply the VoIP QoE measurement model with the required inputs, i.e., the network delay together with the
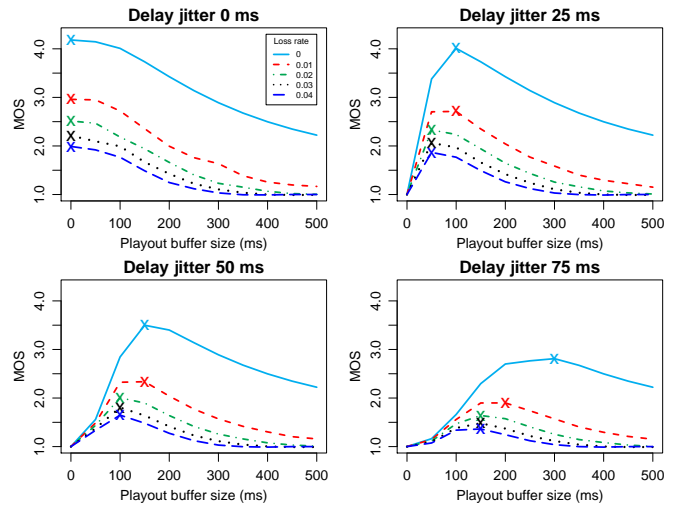
original and degraded audio clips, to derive the MOS score.

In our simulations, we use G.711, the most widely used codec in digital speech applications. We also employ the set of speech recordings [3] that we used to estimate the buffer sizes in the compared VoIP applications.

Simulations were conducted to observe the impact of different playout buffer sizes on the MOS scores with different network delays and delay jitters, as shown in Fig. 4. The MOS score varies significantly as the buffer size increases from 0 ms to 800 ms, which demonstrates the importance of using an effective buffer dimensioning algorithm to improve VoIP conversation quality. We define the buffer size with the highest MOS score as the optimal buffer size and annotate it with a check mark on the graphs. For example, the optimal buffer size is 100 ms when the delay and delay jitter are 50 ms and 25 ms respectively. The figure shows that as the delay jitter increases, a larger buffer size is normally required to provide the best QoE to users. However, an unreasonably large buffer size may degrade the overall quality because a long buffering delay will affect the conversational interactivity.

We also performed simulations to determine the optimal VoIP buffer sizes with different delay jitters and packet loss rates. All the simulations were run with the network delay set to 100 ms. From the results plotted in Fig. 5, we observe that when the delay jitter is small, network loss rates do not affect the optimal buffer size significantly. However, when the delay jitter is large, a higher packet loss rate may lead to a shorter optimal buffer size. We believe the reason is that increasing the buffer size does not allow more packets to arrive and be used when the network loss rate is high. Instead, by reducing the buffer size in order to increase the conversation interactivity, the overall QoE can be enhanced. However, this behavior can be changed if a redundancy control algorithm is introduced [6], so that a VoIP frame can be transmitted several times to cope with high packet loss rates. We will investigate the impact of redundancy control on the optimal playout buffer in a future work.

5.3 Evaluation of Buffer Dimensioning Algorithms in Real-Life Applications
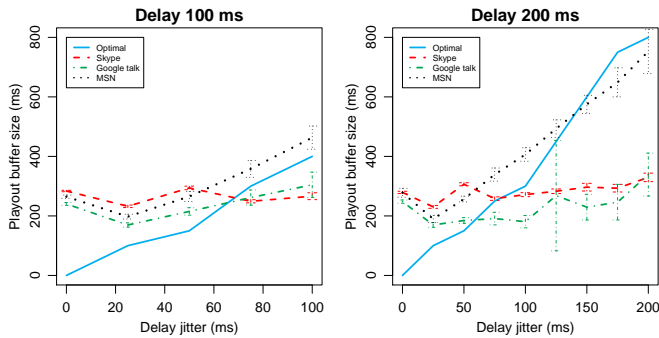
Figure 6: Comparison of the optimal buffer size we derived and the estimate buffer sizes of Skype, Google Talk, and MSN Messenger

Table 1: Coefficients of the Model

Variable	Coef	Std. Err.	t	$Pr > |t|$
$(constant)$	157	20	7.54	$< 2 \times 10^{-9}$
$delay$	-1.05	0.21	-4.78	$< 2 \times 10^{-5}$
$delay \cdot jitter$	0.02	< 0.01	17.25	$< 2 \times 10^{-16}$
$delay \cdot jitter \cdot plr$	-0.57	0.04	-11.65	$< 5 \times 10^{-15}$

Having derived the optimal VoIP playout buffer sizes and determined how Skype, Google Talk, and MSN Messenger adjust their buffer sizes, we can evaluate whether the applications' buffer dimensioning algorithms are optimal. Since the applications do not consider the network loss rate, we only compare their respective buffer sizes with the optimal buffer size under different network delay and delay jitter conditions. As shown in Fig. 6, MSN Messenger's buffer dimensioning algorithm is relatively better than those of Skype and Google Talk, which are too conservative to adjust their playout buffer sizes. We believe that all three applications could provide users with better QoE by improving their buffer dimensioning algorithms.

5.4 Optimal Buffer Size Modeling

Though our methodology can determine the optimal VoIP playout buffer size via simulations, the procedure is time consuming; therefore, it cannot be used in real time. For this reason, we propose a regression-based algorithm to determine the optimal buffer size given a network scenario. Using a polynomial regression approach, we can develop a model based on the simulation results. The model can derive the optimal playout buffer size by computing

$$(constant) + coef_{delay} \cdot delay +$$
$$coef_{delay \cdot jitter} \cdot delay \cdot jitter +$$
$$coef_{delay \cdot jitter \cdot plr} \cdot delay \cdot jitter \cdot plr,$$

where $delay$ denotes the average network delay, $jitter$ denotes the standard deviation of network delays, and plr denotes the packet loss rate. The coefficients for G.711 are listed in Table 1. The R^2 value of the regression model is 0.885, which indicates that the model can predict the optimal buffer size with a high degree of accuracy. One advantage of our model is that we can easily include more network factors and extend the approach to other audio codecs. Our methodology is effective and efficient because it computes the optimal buffer size with a very low computational overhead, and it takes account of users' perceptions of conversa-

tion interactivity and speech quality.

6. CONCLUSION AND FUTURE WORK

In this paper, we investigate the playout buffer dimensioning algorithms applied in popular VoIP applications. We propose an experiment methodology to determine whether Skype, Google Talk, and MSN Messenger adjust their playout buffer sizes appropriately under different network conditions. Our results indicate that MSN Messenger yields the best performances in terms of buffer dimensioning to suit varying network conditions. Surprisingly, Skype does not adjust its playout buffer size at all. Finally, we propose a simple algorithm that computes the optimal buffer size based on an objective QoE metric that considers both conversational interactivity and speech quality.

In our future work, we will pursue the following research avenues. 1) We will consider more factors in order to understand the buffer dimensioning behavior of real-life VoIP applications, e.g., the frame size, redundancy control algorithm, and speech codec used. 2) As the proposed buffer dimensioning algorithm is based on objective QoE metrics, we expect that it should be able to achieve near-optimal VoIP quality under different network scenarios. Thus, we will conduct real-life network experiments to assess the algorithm's performance.

7. REFERENCES

[1] Audio Signal Delay Project. http://www.cs.columbia.edu/irt/software/adelay/report.html.
[2] Intel Integrated Performance Primitives (Intel IPP). http://www.intel.com/support/performancetools/libraries/ipp/.
[3] Open Speech Repository. http://www.voiptroubleshooter.com/open_speech/.
[4] Cisco. Understanding delay in packet voice networks. http://www.cisco.com/en/US/tech/tk652/tk698/technologies_white_paper09186a00800a8993.shtml.
[5] L. Ding and R. A. Goubran. Assessment of effects of packet loss on speech quality in VoIP. In *Proceedings of the 2nd IEEE International Workshop on Haptic, Audio and Visual Environments and Their Applications*, pages 49–54, 2003.
[6] T.-Y. Huang, K.-T. Chen, and P. Huang. Tuning the redundancy control algorithm of Skype for user satisfaction. In *Proceedings of IEEE INFOCOM 2009*, 2009.
[7] ITU-T Recommendation G.107. The E-model, a computational model for use in transmission planning, Mar. 2005.
[8] ITU-T Recommendation P.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, Feb. 2001.
[9] Y. J. Liang, N. Farber, and B. Girod. Adaptive playout scheduling and loss concealment for voice communication over ip networks. *IEEE Transactions on Multimedia*, 5:532–543, 2003.
[10] M. Narbutt and L. Murphy. VoIP playout buffer adjustment using adaptive estimation of network delays. In *Proceedings of 18th International Teletraffic Congress (ITC-18)*, pages 1171–1180, 2003.
[11] R. Ramjee, J. Kurose, D. Towsley, and H. Schulzrinne. Adaptive playout mechanisms for packetized audio applications in wide-area networks. In *Proceedings of the IEEE INFOCOM 1994*, pages 680–688, 1994.
[12] C. J. Sreenan, J.-C. Chen, P. Agrawal, and B. Narendran. Delay reduction techniques for playout buffering. *IEEE Transactions on Multimedia*, 2:88–100, 2000.