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

Chen-Chi Wu[†], Kuan-Ta Chen[‡], Yu-Chun Chang[†], and Chin-Laung Lei[†]
[†]Department of Electrical Engineering, National Taiwan University
[‡]Institute of Information Science, Academia Sinica
{bipa,congo}@fractal.ee.ntu.edu.tw, ktchen@iis.sinica.edu.tw, lei@cc.ee.ntu.edu.tw

ABSTRACT

VoIP playout buffer dimensioning has been long a challenging optimization problem as the buffer size should maintain a balance between the conversational interactivity and speech quality. One of its challenges comes from the fact that many factors may affect the overall conversational quality, and some of the factors may change over time. Although academic researchers have done numerous efforts in solving this problem, how their research results are applied in practice remains unknown.

In this paper, we investigate whether a gap between VoIP researchers and practitioners, from the perspective of playout buffer dimensioning algorithms, exists. Targeting at three popular VoIP applications, namely, Skype, Google Talk, and MSN Messenger, we design experiments to systematically measure how the applications adjust their playout buffer sizes. By using an objective QoE (Quality of Experience) metric, we show that all of the three applications do not adjust their buffer size very well. In other words, they could provide better QoE to users by simply replacing their buffer dimensioning algorithms. Also, all of them do not adapt the buffer size to the network loss rate, which should also be considered for optimal QoE provisioning.

Categories and Subject Descriptors

H.1.2 [**Models and Principles**]: User/Machine Systems—*Human factors*; K.8.0 [**Personal Computing**]: General—*Games*

Keywords

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

1. INTRODUCTION

VoIP is now becoming an important communication service inside and between enterprises and cooperations, and individuals also start to rely on it in daily communications with their friends and family. The reason is that VoIP now provides low call costs and the voice quality is almost comparable to that of traditional toll telephones. The trend is exemplified by the fact that one of the most widely used VoIP applications, Skype, has 405 million registrars and 15 million online users[1]. Because of the steady growth of VoIP usage, providing reliable service and satisfactory voice quality is now a high-priority task for Internet and VoIP service providers.

There are many factors that can impact the service quality of VoIP, e.g., the speech codec, transport protocol, redundancy/error control, network path selection, and playout buffer dimensioning. While each of the issues deserves a line of research efforts, this work devotes to an empirical investigation of playout buffer dimensioning algorithms employed by popular real-life VoIP applications and its implications.

The basic idea of playout buffering[2] is *to sacrifice the speech conversational interactivity in the exchange for better voice quality.* Normally a voice packet is transmitted from the talker's node to the listener's node every 20 ms or 30 ms in order to maintain continuous and smooth speech conversation. However, as in packet-switched networks the packet queueing delays are variable and hard to predict, some unlucky packets may arrive at the listener's node due to long queueing delays and the speech samples inside the packets will be considered lost. This may result in silent periods, noise, or unclear speeches, depending on the loss concealment algorithm adopted by the voice codec. To reduce the occurrence possibility of this phenomenon, one can employ a playout buffer to temporarily hold a VoIP packet until its scheduled playout time is due. By so doing, packets experiencing 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 regarding the VoIP playout buffer is *how to determine the best buffer size to use.* Generally, a larger buffer size simultaneously leads to a higher sound quality and a lower conversational interactivity. Thus, we can treat buffer size adjustment as an optimization problem where *the optimal buffer size should maintain a balance between the conversational interactivity and speech quality.* The optimal buffer size is jointly affected by several factors, including network delay, delay variability (jitter), re-

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

[2]Since the only buffer we discuss in this paper is the playout buffer, thus we will use "playout buffer" and "buffer" exchangeably in this paper.

dundancy control, error correction, and codec implementations. Moreover, the above factors, especially network delay and network loss, may change over time. Therefore, a good buffer dimensioning algorithm should consider the volatility of network conditions while maintaining the tradeoff between the conversational interactivity and speech quality.

There have been numerous proposals regarding VoIP playout buffer dimensioning algorithms [9–11,13]. Most of them adjust the buffer size based on a linear combination of network delay and jitter, where the weights and exact adjustment policies may vary according to different optimization goals and design considerations. While academic researchers have done huge efforts in solving the buffer dimensioning problem, how their research results are applied in practice remains unknown.

In this paper, *we investigate whether a gap between VoIP researchers and practitioners, from the perspective of playout buffer dimensioning algorithms, exists.* Targeting at three popular VoIP applications, namely, Skype, Google Talk, and MSN Messenger, we design experiments to measure how the applications adjust their playout buffer sizes. We analyze if these VoIP applications correctly adjust their playout buffers; and, if not, how much difference is their performance from the best possible quality. Our results indicate that MSN Messenger performs the best in terms of buffer dimensioning due to varying network conditions, while Skype, to our surprise, does not adjust its playout buffer size at all. Finally, we propose a simple algorithm that computes the optimal buffer size based on objective QoE (Quality of Experience) metrics.

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 summarize the relationship between the observed buffer size and network condition.
2. By using an objective QoE metric, we show that all of the studied VoIP applications, Skype, Google Talk, and MSN Messenger, do not adjust their buffer size very well. In other words, they could provide better QoE to users by simply replacing their current buffer dimensioning algorithms. Also, all of them do not adapt the buffer size to the network loss rate, which should also be considered in order for optimal QoE provisioning.
3. We propose a simple regression-based algorithm to compute the optimal playout buffer size based on the current network condition. Our approach has three advantages: 1) it is based on an objective user satisfaction measure which considers both conversational interactivity and speech quality; 2) it is simpler than previous proposals, as only a weighted sum operation is needed to compute the optimal buffer size; and 3) it can easily take more network factors into consideration without 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 well the studied applications adjust their buffer size. Finally, we summarize our conclusions in Section 6.

2. RELATED WORK

There have been several VoIP playout buffer dimensioning algorithms proposed to improve audio quality in VoIP communications. In [11], the authors proposed to adjust the buffer size based on the EWMA (Exponential Weighted Moving Average) of network delays and their standard deviation (i.e., delay jitters), where the weights of variables are fixed and empirically chosen. The work [10] extended [11] by adaptively adjusts the EWMA weights according to the magnitude of delay jitter, where the weight is set higher when the delay jitter is smaller and vice versa. The simulations conducted by the authors show that this approach significantly improves the tradeoff between the buffer delay and packet loss. Later, the works [9,13] further extended [10,11] by adjusting the buffer size within a talk burst. The purpose of such improvements is to make the playout buffer adapt to varying network conditions more quickly, and hopefully achieve a better conversation quality in a VoIP call.

To assess the quality level of a VoIP conversation, a number of models have been developed in recent years. A widely used model for listening speech quality evaluation is PESQ (Perceptual Evaluation of Speech Quality) [8]. PESQ is a signal-based method that compares the original speech signals with the degraded signals, and grade the quality using a mean opinion score (MOS), which ranges from 1 (Bad) to 5 (Excellent). On the other hand, E-Model [7] is used to estimate the VoIP conversational quality, where the quality score is an arithmetic sum of the delay impairment factor I_d, the equipment impairment factor I_e, and the factor I_s which considers the quality degradation due to speech compression/decompression and quantizing distortions. E-Model outputs a rating factor R (ranging from 0 to 100), which can be converted to MOS by

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

We remark that, to accurately assess the quality-of-experience of a VoIP call, neither PESQ nor E-Model is sufficient. The reason is that PESQ does not take interactivity into consideration, so the PESQ score can be high even the end-to-end delays are too long to form a sensible conversation. At the same time, E-Model has the following disadvantages: 1) its listening quality assessment is less accurate than a signal-based algorithm such as PESQ; 2) it does not consider the variability of network delays and loss rate; 3) it does not take account of the interaction between different factors, e.g., the interplay between network delay and listening quality, and that between network delay and loss rate. Due to the above reasons, Ding et al. [5] proposed a combined model that integrates PESQ and E-Model and therefore possesses the advantages of both models. We will introduce this model and use it for our VoIP QoE evaluations in Section 5.

3. EXPERIMENT METHODOLOGY

In this section, we describe the experiment setup and procedures for measuring the applications' playout buffer sizes in various network scenarios.

3.1 Experiment Setup

To measure the selected VoIP applications' playout buffer sizes under different network conditions, we setup a FreeBSD 7.0 machine as a router and control the pace of traffic flows

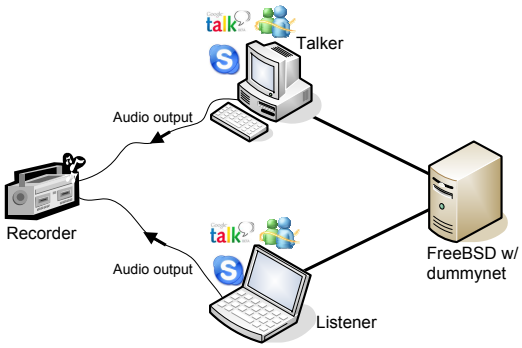

Figure 1: The experiment setup

passing through by `dummynet` [12]. 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. We designate one of the PCs as the talker, and another one as the listener in the experiment. A speech recording will be continuously played on the talker, so that the listener will receive a degraded copy of the speech recording via VoIP transmission. To simulate real-life human conversation, the speech recordings we used were downloaded from the Open Speech Repository [3].

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

By configuring `dummynet` on the router, we control the network delay, delay jitter (the standard deviation of network delays), and loss rate between the talker and listener PC. As every packet sent from the talker must pass through the router before reaching the listener, we can therefore inspect how the applications' playout buffer sizes change due to the various network conditions we conduct.

In the experiments, we separately set the network delay and delay jitter between 0 ms and 200 ms with a 25 ms spacing, and set the packet loss rate between 0% and 10% with an 1% spacing. We set the duration of each VoIP call 240 seconds, and make 10 VoIP calls with each network setting. To allow sufficient time for the VoIP applications to adapt their playout buffer size to the latest network conditions, the wave file recording was started after a call is established for 60 seconds. Without loss of generality, we assume that the delay jitters follow a Gamma distribution. Also, as the effect of restricted bandwidth can be emulated by injecting packet loss, the network bandwidth between the two call parties is set to a sufficiently large value, 1000 Kbps.

3.2 Buffer Size Estimation

To estimate the size of the playout buffers inside the proprietary VoIP applications we studied, first we need to know the end-to-end delay between the time the talker speaks and the time the listener hears. The end-to-end delay can be estimated by computing the audio delay between the talker's audio output and the listener's audio output, which are recorded in the wave files. Therefore, we calculate the audio delay by searching for the time difference which yields the largest cross-correlation coefficient between the speech recordings output from two parties [1].

However, the end-to-end delays of VoIP transmission comprise not only the playout buffer delay, but also network delay, coder delay, and packetization delay [4]. Since both call parties are inside a LAN, all the network delay components are under control, where the propagation delay and transmission delay are so small so that they can be neglected. Other sources of end-to-end delay, i.e., the coder delay, and packetization delay, are application- and codec-dependent, thus we do have the exact information about these two delay components. Have referenced to the typical values used by popular codecs [4], we find that the sum of the coder delay and packetization delay is typically around 50 ms. Therefore, we estimate the applications' playout buffer size by subtracting the measured end-to-end delay by 50 ms plus the average network delay induced by `dummynet`. We acknowledge that the estimate may not be 100%-accurate. However, as our focus is on how the VoIP applications adjust their playout buffer sizes due to different network conditions, the absolute error in estimating the buffer size will not affect the buffer dimensioning behavior we observed nor the conclusions we made.

4. PLAYOUT BUFFER SIZE ADJUSTMENT IN REAL-LIFE APPLICATIONS

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

4.1 Effect of Network Delay and Delay Jitter

As shown in Fig. 2, we plot the VoIP playout buffer sizes in Skype, Google Talk, and MSN Messenger when their VoIP packets experience different levels of network delays and delay jitters. The vertical bars on the graph represent the 95% confidence band of the average buffer size. We can see that, in Fig. 2(a) and (c), the curves corresponding to different delays are similar, and the 95% confidence bands collide with each other. This phenomenon indicates that Skype and MSN Messenger do not adjust their playout buffer sizes due to the average network delay. On the other hand, the dissimilarity between the curves in Fig. 2(b) evidences that the average network delay is included in the consideration how Google Talk adapts its playout buffer size.

We then investigate the impact of delay jitter on the buffer size adjustment. As shown in Fig. 2(a), Skype's buffer size remains within the range $(250, 300)$ ms regardless of the magnitude of delay jitter. This observation suggests that Skype does not adjust its buffer size due to network delays. On the contrary, both Google Talk and MSN Messenger increase their buffer sizes as the delay jitter increases. This design allows packets that experience longer queueing delays more chances to be arrive before the scheduled playout time and be used. Specifically, we observe that MSN Messenger adjusts its buffer size linearly due to increasing delay jitter, while Google Talk merely increases its buffer size by a small amount even when the delay jitter is large. The different behavior of the two applications may likely lead to different overall quality levels, which we will investigate in Section 5.

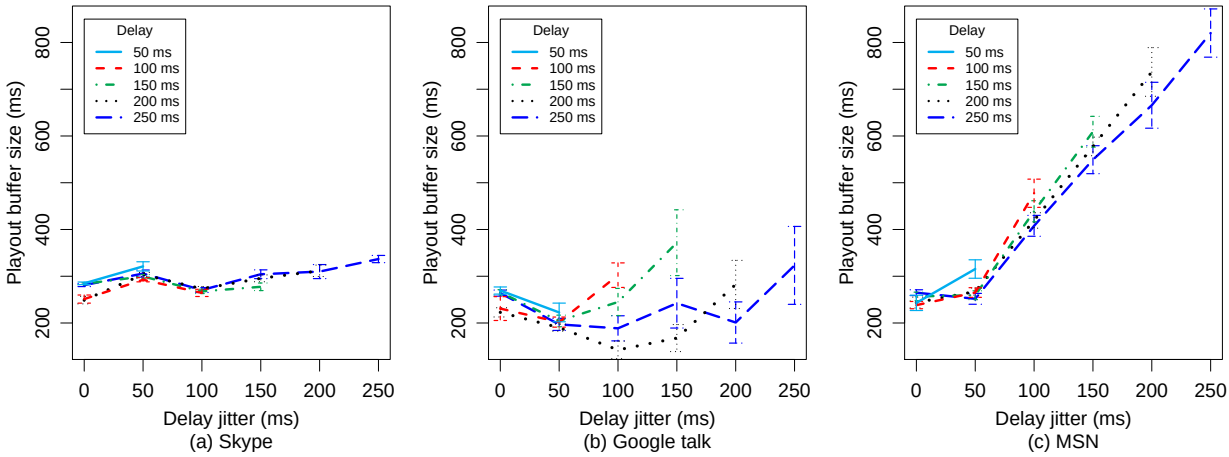

Figure 2: The playout buffer sizes of Skype, Google Talk, and MSN Messenger under different network delays and delay jitters

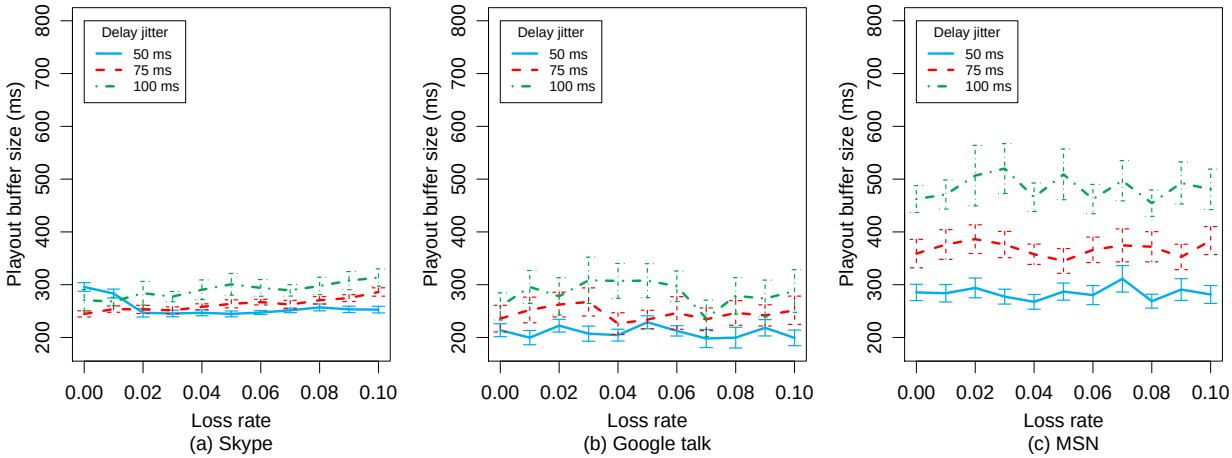

Figure 3: The playout buffer sizes of Skype, Google Talk, and MSN Messenger under different network loss rates

4.2 Effect of Network Loss Rate

Fig. 3 shows the applications' playout buffer sizes with 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 investigate whether Skype, Google Talk, and MSN Messenger take account of the network loss rate in their buffer dimensioning algorithms. Obviously, the curves in Fig 3(a)-(c) do not exhibit any significant change tread due to the change in the loss rate. The phenomenon suggests that all of the three popular VoIP applications do not consider the network loss rate in their buffer dimensioning algorithms.

In summary, our experiment results reveal that Skype keeps the same playout buffer size regardless of network delays, delay jitters, and loss rates, while MSN Messenger's buffer size basically grows linearly as the delay jitter increases. As to Google Talk, it adjusts its buffer size gently based on the average network delay and delay jitter. For all the three applications, the packet loss rate is not taken into consideration in their buffer dimensioning algorithms. Even so, we do not know which application's policy is the best, and how is the user satisfaction achieved due to their design choices. Thus, we will introduce an VoIP QoE measurement model and employ the model to evaluate the goodness of these real-life applications' buffer dimensioning algorithms in the following section.

5. PLAYOUT BUFFER OPTIMIZATION BASED ON USER SATISFACTION

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

5.1 QoE Measurement Model

As mentioned in Section 2, E-Model, though commonly used, has accuracy problems in estimating the user satisfaction in VoIP conversations. Therefore, here we employ a QoE measurement model, which was proposed by Ding

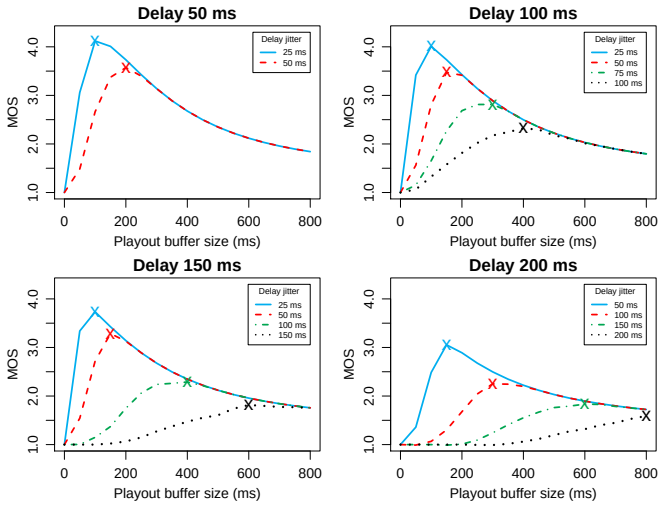

Figure 4: The simulation result of inferring the optimal buffer sizes for different network delays and delay jitters

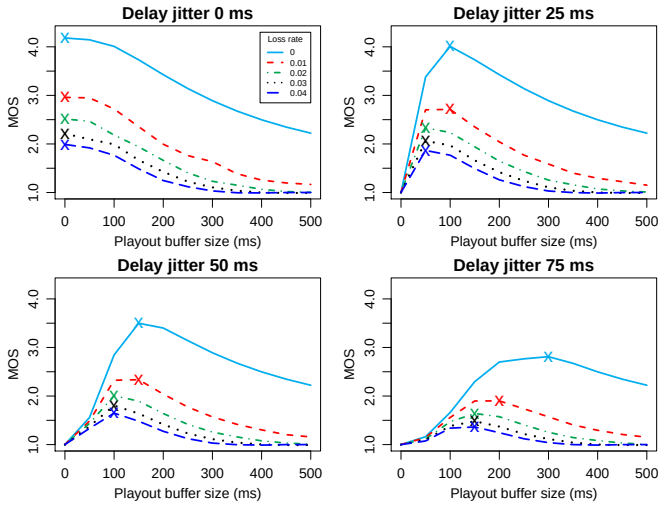

Figure 5: The simulation result of inferring the optimal buffer sizes for different network loss rate

et al. [5], to quantify the overall QoE provided by a VoIP application. The best feature of this model is that it takes advantage of 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 result MOS score to a R score using the formula in ITU-T G.107 Appendix I [7].
2. Compute the delay impairment I_d in E-Model based on the network delay. Other parameters of the E-Model remain unchanged.
3. By subtracting the R score obtained in step 1 with I_d obtained in step 2 we obtain the result R score, and convert this score to MOS score using Equation 1.

5.2 Optimal Buffer Size Derivation

We define that the optimal playout buffer size denotes the buffer size which leads to the highest user satisfaction in a VoIP call. To derive the optimal buffer size under a given network condition, we design a simulator which can evaluate the quality of VoIP conversation given a network configuration and a playout buffer size. Based on the VoIP QoE measurement model, we derive the optimal playout buffer size as the buffer size that yields the highest MOS score. Our steps for obtain the optimal buffer size are as follows:

1. Encode an audio clip into a sequence of VoIP frames by using the encoder library in Intel Integrated Performance Primitives Library [2].
2. Simulate network packet loss with the Gilbert model, where a packet is dropped if the model is in the "Error" state and retain it otherwise.
3. Introduce network delay to each packet, where the delays are generated using a Gamma distribution. Following that, if a packet's delay is longer than the current playout buffer size, it will be dropped; otherwise, it is retained.
4. Use the corresponding speech decoder to 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, original and degraded audio clip, to derive a MOS score.

In our simulations, we use G.711, the most common codec used in digital speech applications. Also, the same set of speech recordings [3] is employed as we used in estimating the buffer sizes in real-life VoIP applications.

We conduct simulations to observe the impact of playout buffer sizes on the MOS scores with different delays and delay jitters, as shown in Fig. 4. We can see that the MOS score varies significantly as the buffer size increases from 0 ms to 800 ms, which indicates that the importance of a good buffer dimensioning algorithm to the 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 figures show that as the delay jitter increases, a larger buffer size is normally required to provide the best QoE to users. At the same time, a unreasonably large buffer size may degrade the overall quality because the long buffering delay will also lower the conversational interactivity.

We also perform simulations to infer the optimal VoIP buffer sizes with different delay jitters and packet loss rates, and plot the results in Fig. 5. All the simulations were run with the network delay set to 100 ms. 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, while network loss already degrade the speech quality, increasing the buffer size does not help much; instead, by decreasing the buffer size to increase the conversation interactivity, the overall QoE can only be increased. However, this behavior can be changed if any redundancy control algorithm is introduced [6], where a VoIP frame may be transmitted several times to cope with high packet loss rates. We consider investigating the impact of redundancy control on the optimal playout buffer size as part of our future study.

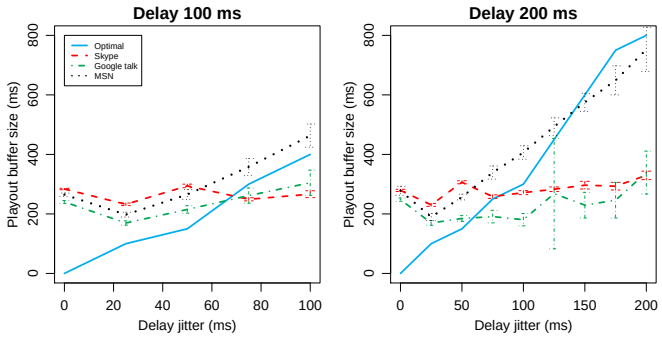

Figure 6: Comparison of the optimal buffer size we derived and the estimate buffer sizes of Skype, Google Talk, and MSN Messenger

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	$< 5 \times 10^{-15}$

5.3 Evaluation of Buffer Dimensioning Algorithms in Real-Life Applications

Now that we have derived the optimal VoIP playout buffer sizes and known how Skype, Google Talk, and MSN Messenger adjust their buffer sizes, we can evaluate whether the buffer dimensioning algorithms of these applications are optimal. Since we observe that the studied applications do not take account of network loss rates, we only compare the buffer size of studied applications with optimal buffer size with different delays and delay jitters. As shown in Fig. 6, we find that MSN Messenger implements the relatively better buffer dimensioning algorithm, while the other two applications are way too conservative to adjust their playout buffer sizes. We suggest that these applications could provide users with better quality of experience by improving their buffer dimensioning algorithms.

5.4 Optimal Buffer Size Modeling

Though our methodology is able to derive the optimal VoIP playout buffer size via simulations, the procedure is time consuming and therefore not possible to 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 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 very well. One of the advantage of our model is that we can easily include more network factors and extend this approach to other audio codecs. Overall, our methodology is advantageous in that it not only computes the optimal buffer size with a very low computation overhead, but also takes account of the user-perceived interactivity and speech quality.

6. CONCLUSION AND FUTURE WORK

In this paper, we aim to investigate whether a gap between the VoIP researchers and practitioners, from the perspective of playout buffer dimensioning algorithms, exists. By using the experiment methodology we proposed, we analyze if Skype, Google Talk, and MSN Messenger correctly adjust their playout buffers. Our results indicate that MSN Messenger performs the best in terms of buffer dimensioning due to varying network conditions, while Skype, to our surprise, 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 which considers both the conversational interactivity and speech quality.

In the future, we plan to pursue the following two directions: 1) We will consider more factors in understanding the buffer dimensioning behavior of real-life VoIP applications, such as frame size, redundancy control algorithm, and speech codec used; 2) As our algorithm is based on objective QoE metrics, we expect that it should achieve a near-optimal VoIP quality given a specific network scenario. Thus, we will conduct real-life network experiments to verify how our proposed buffer dimensioning algorithm works.

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.

[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.

[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] L. Rizzo. Dummynet: A simple approach to the evaluation of network protocols. *ACM SIGCOMM Computer Communication Review*, 27(1):31–41, 1997.

[13] 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.