Master's Thesis

Title

Adaptive Playout Buffer Algorithm for Enhancing Perceived Quality of Streaming Applications

Supervisor

Professor Masayuki Murata

Author

Kouhei Fujimoto

February 13th, 2002

Department of Informatics and Mathematical Science

Graduate School of Engineering Science

Osaka University

Master's Thesis

Adaptive Playout Buffer Algorithm
for Enhancing Perceived Quality of Streaming Applications

Kouhei Fujimoto

Abstract

An end-to-end packet delay in the Internet is an important performance parameter, because it heavily affects the quality of various applications including real-time and data applications. In particular, it is important to investigate not a whole distribution of the packet transmission delay, but the tail part of the distribution, in order to detect the packet loss. In this thesis, we first analyze the characteristics of the tail part of packet delay distributions by a statistical analytic approach. Our analytic results show that the Pareto distribution is most appropriate in 90–99.9% region of the cumulative distribution of packet transmission delays.

Based on our statistical analysis, we next propose a new playout buffer algorithm, which adjusts the playout delay so that packet loss ratio (due to playout time expiration) can be equal to user's expected value by controlling the trade off between delay and loss ratio from the distribution of packet delays. Numerical examples show that our algorithm adequately control the packet loss ratio according to the user's expected value, and our algorithm can provide stable quality against the varying packet delays of the Internet. Furthermore, we extend the proposed playout buffer algorithm considering user's perceived quality of real-time applications. Perceived quality changes with the combination of delay and packet loss within the network. This extended algorithm finds the best combination. Our simulation and implementation tests show that it can enhance the perceived quality, compared with existing algorithms.

Keywords

Streaming Application, Delay Characteristics, Delay Distribution, Statistical Analysis, Playout Buffer Algorithm, Pareto Distribution, Packet Loss Ratio, Perceived Quality

Contents

List of Figures

List of Tables

1 Introduction

The Internet is now widely deployed and users can easily obtain global accessibility from their home terminals. One of main reasons for the prevalence of the Internet is in its routing mechanisms. Routing of the Internet has two key features; flexibility and scalability. First, the Internet provides the dynamic routing by the exchanging the routing information among routers. Then, if a network link becomes down because of some trouble, an alternative route will be prepared automatically. Second, the packet processing at the routers is simple (e.g., FIFO) to reduce the overhead of packet forwarding at the router.

While it is true that the dynamic routing mechanism is important for IP (Internet Protocol) routing, it is not always helpful for the end users. From the users' point of view, the packet transmission delay is an important performance metric since it directly affects the end-to-end quality. One example is the real-time application using RTP (Real-time Transport Protocol) [1]; a popular protocol for real-time applications in recent years. RTP uses RTCP (Real-time Control Protocol) to control the transmission rate. In RTCP, the sender maintains the transmission delay of packets based on Round Trip Time (RTT) values to control the packet transmission rate. To keep the preferable performance in RTP-based applications, an accurate estimation of the packet transmission delay is essential. However, the dynamic routing of the Internet makes it impossible for the end-users to select the appropriate route for satisfying the users' quality of service (QoS). It is difficult even to predict the transmission delay of the packet, since a simple packet processing at routers provides a variable delay. Furthermore, in real-time applications (e.g., voice communications), it is desirable to separately measure transmission delays of both downstream (sender to receiver) and upstream (receiver to sender) routes because the Internet routes are often asymmetric due to its dynamic routing mechanism [2]. Even if downstream and upstream routes are on the same path, they may have radically different transmission delays due to time-fluctuating queueing delays at the routers. From these reasons, it is necessary to investigate not only the characteristic of RTT but also that of one-way transmission delays in order to develop an accurate delay estimation method.

The studies on the characteristics of the end-to-end packet transmission delay have been made in some literatures [3, 4, 5], but most of those studies have focused on the

entire distributions only. If we want to detect a packet loss, a tail distribution becomes more important. In UDP based real-time applications, a smoothing buffer is typically used at a client host to compensate for variable delays. Received packets are first queued into the smoothing buffer. After several packets are queued, actual decoding is started. Then, the influences of the delay variations within the network can be minimized. (We refer to this delay as the playout delay.) Choosing the playout delay is important because it directly affects the communication quality of the application; if the playout delay is set to be too short, the client application treats packets to be lost even if those packets eventually arrive. On the contrary, the large playout delay may introduce an unacceptable delay that the client users cannot be tolerant. That is, a difficulty exists in determining the playout delay. The packet transmission delay between the server and client may be varied according to the network condition in the Internet, and hence, the adequate playout delay is heavily dependent on variations of packet transmission delays. Therefore, its time-dependent behavior and distribution especially in the tail part are quite important in determining the playout delay.

The issue of playout control has been studied by some previous works [6, 7, 8, 9], and several algorithms controlling the playout buffer (which we will refer to as the playout buffer algorithm (PBA)) have been proposed. Most of those PBAs are however based on a calculation method of the time-out threshold in TCP [10]. For example, Moon et al. [8] trace the packet delays and suggest the playout delay from the distribution of traced delays. However, they only focus on adjustments of the playout delay, and do not consider to control the packet loss ratio. Instead they try to manage the packet loss ratio to be closely zero, which may introduce unacceptable playout delays in spite of the fact that the quality of applications is affected by the playout delay as well as the packet loss ratio. Since recent streaming applications are robust against packet losses with keeping the sufficient reproduction quality, it is possible to shorten the playout delay when we introduce the PBA that allows some marginal packet losses.

Keeping those facts in mind, the first work in this thesis is to analyze the characteristics of the packet transmission delays. We measure the one-way transmission delay as well as the round-trip delay, and determine the suitable distribution function through a statistical analytic approach. We next propose the use of the distribution function to estimate the

playout delay for real-time applications. In an actual situation, some users prefer the real-time reproduction of streaming media even if the packet loss becomes high, and others may want a high quality at the expense of the large playout delay. By taking account of it, we propose a new playout buffer algorithm, which keeps the packet loss ratio according to users' willingness while minimizing the playout delay. The proposed algorithm records the history of one-way delays, and computes the playout delay based on our statistical modeling. Next, we compare the performances of the proposed algorithm and other PBAs by trace-driven simulation to show the effectiveness of our algorithm. The applicability of our proposed PBA is also examined by developing the input plug-in of the streaming application, to validate the computation overhead satisfying the requirements in real-time operations.

The remaining problem is a fact that the packet loss ratio is not a user-friendly metric for the perceived quality in streaming applications. There are many factors affecting the perceived quality in playing audio or speech of the streaming applications. Actually, in addition to the packet loss ratio, other network parameters such as type of codecs, access lines would affect the perceived quality. One important issue is how to map the network metrics into the users' perceived quality of the real-time traffic. Our approach is to utilizing the data set shown in [11], which clarified the relationship between Mean Opinion Score (MOS) of played audio and network parameters (e.g., packet loss, packet transmission delay, transmission rate). Then, we propose another PBA to maximize the MOS index directly for given network parameters.

This thesis is organized as follows. We first show a brief summary of the characteristics of the packet transmission delay and our measurement framework in Section 2. In Section 3, we explain our analytic approach to estimate parameters of distribution functions and select the most appropriate distribution. We next show the result of analysis in Section 4. In Section 5, we propose a new playout buffer algorithm based on the results in Section 4. In Section 6, we evaluate the proposed algorithms through the simulation and implementation. In Section 7, we attend to the users' perceived quality, and examine the relationship between MOS of played audio and network parameters. Then, we modify the proposed algorithm in order to maximize the MOS. Its performance is evaluated through both simulation and implementation experiments. Finally, we summarize our work and

describe our future research topics in Section 8.

2 Measurement Methodologies of Packet Transmission Delay

In this section, we show a brief summary of our measurement methods. We measured two types of the packet transmission delay; the round-trip transmission delay and the one-way transmission delay. We first show the outline of the measurement approach, and next investigate the factors affecting the characteristics of transmission delay.

2.1 Measurement Method of Round Trip Times

There are several tools to measure RTT values. See [12] and references therein. We adopted `pchar` [13] for RTT measurements. `Pchar` (an updated version of `pathchar` [14]) was originally developed to measure the bandwidth of intermediate links between two end hosts. `Pchar` uses the Internet Control Message Protocol (ICMP) Time Exceeded message to measure the RTT. More specifically, `pchar` utilizes the Time To Live (TTL) field in the IP packet. By protocol specification, the router decreases the value of TTL by one before the packet forwarding. If the value of TTL becomes zero, the router sends the ICMP Time Exceeded packet back to the sender. Thus, `pchar` intentionally sets the value of TTL to a smaller value to indicate the number of hops the packet can traverse. After the sender receives the ICMP Time Exceeded packet, the sender can obtain the RTT that is the duration between when the host sends the packet and when it receives the ICMP packet. The advantage of using ICMP messages is that it is not necessary to deploy any other hosts to measure the RTT. In addition, `pchar` provides events of routing changes and the packet loss ratio. Those are the reasons why we use `pchar` for RTT measurements.

2.2 Measurement Method of One-way Delays

To measure the one-way delay, we developed the server-client based tool. By the tool, the sender host records the current time into the packet before sending. When the packet arrives at the receiver host, the delay is calculated using the receiver's clock. In this method, however, time clocks of the sender and receiver should be synchronized in order to measure accurate delays. Unless the two clocks are not synchronized, different two clocks may cause relative offset and skew as illustrated in Figure 1. The relative offset of the two

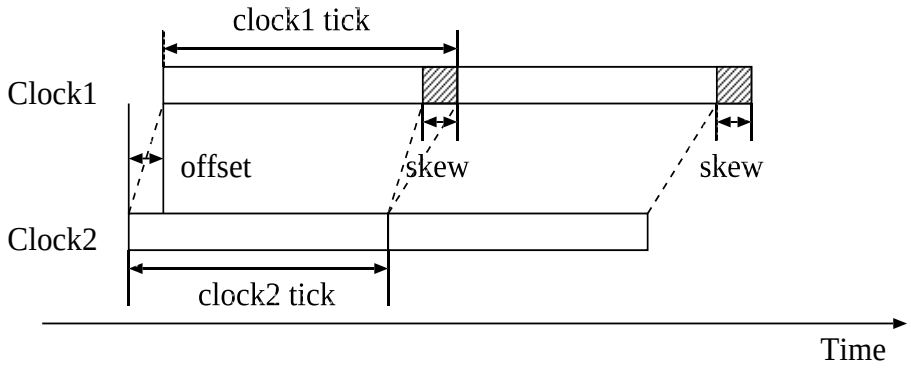


Figure 1: Relative Offset and Skew

clocks is caused when the two clocks have different time. The relative offset introduces a margin of errors in determining the one-way delay. Since, the relative skew is caused by the difference of tick intervals between two clocks, the one-way delay becomes linearly increased or decreased due to it. However, synchronization among distributed hosts in the Internet is difficult. Therefore, in order to avoid the problem in clock synchronization, we first adopt GPS (Global Positioning System) to both end hosts (sender and receiver). Figure 2 shows the measurement infrastructure from GPS. We measure the one-way delays utilizing the clock of GPS on Windows operating system. The GPS provides the time with the accuracy within 100 nsec.

However, in an actual situation, it is impossible that all streaming applications is equipped with GPS. Thus, in order to perform one-way delay measurements without GPS, we remove the effect of relative offset by Paxson's approach [15], and the that of relative skew by Moon's approach [16]. In Paxson's method, one-way delays of upstream and downstream are first measured. Suppose that a packet is sent by host A at time a_s, and transferred to host B at time b_r. Next, a second packet is sent in the reverse direction by host B at time b_s, and transmitted to host A at time a_r. Assuming both paths between host A and B are symmetric, their time-dependent relations are given by

$$b_r = a_s + \Delta T + \Delta C_o, \quad (1)$$

$$a_r = b_s + \Delta T - \Delta C_o, \quad (2)$$

where ΔT is the one-way propagation delay of the path between two hosts, and ΔC_o is

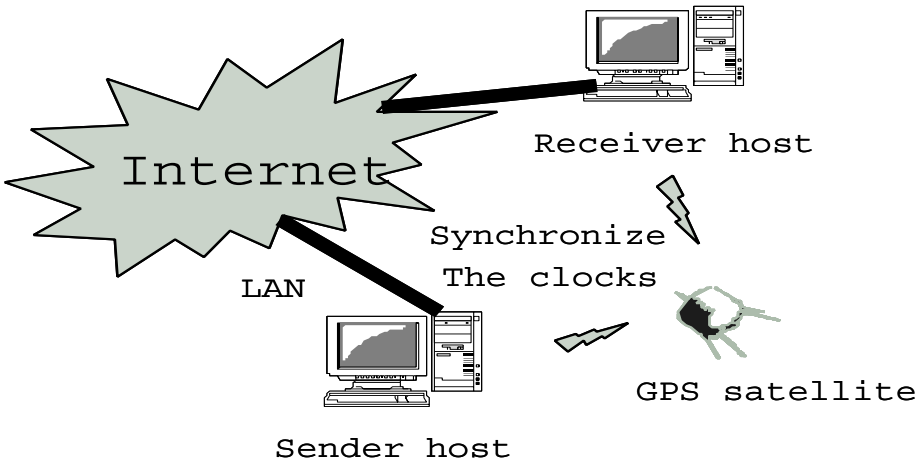


Figure 2: The Measurement Infrastructure with GPS for Measuring One-way Delays

the relative offset between hosts A and B. From Eqs. (1) and (2), the relative offset ΔC_o is calculated by

$$\Delta C_o = \frac{(b_r - a_s) + (a_r - b_s)}{2}. \tag{3}$$

Here, $(b_r - a_s)$ and $(a_r - b_s)$ are measured one-way delays of upstream and downstream path between hosts A and B, respectively. Because ΔC_o in Eqs. (1) and (2) is the one-way propagation delay, we need to obtain the lower bound of the one-way delay including no queueing delay at any intermediate routers. For this purpose, we measure one-way delays several time (10 is enough according to our experience) for each direction, and use the minimum value of one-way delays to calculate ΔC_o in Eq. (3). Then, we obtain the value of relative offset by measuring one-way delays in both of upstream and downstream.

We next summarize the Moon's skew avoidance approach. Figure 3 shows 1,000 samples of one-way delays between the sender and receiver. Measured delays are plotted by cross symbols. In this figure, measured one-way delays are linearly decreased with passing time. The objective of Moon's approach is to decide the slope of line. It first sets the standard point from measured delays. For each samples except the standard point, we get the line between the sample and the standard point, and check whether some samples are below the line or not. If not, i.e., all samples are above or over the line, we select the line as presenting the relative skew. Finally, we obtain the value of the skew from the slope

of the line. In case of Figure 3, the line presenting the relative skew is shown by the solid line, and measured delay after removing the relative skew are plotted by plus symbols.

The Moon's algorithm computes an equation of the line which represent the increase or decrease of delays from the set of measured delays. It fits a line that lies under all the measured one-way delays, but as closely to them as possible, and obtain the value of relative skew from the slope of the line. The result of Moon's algorithm is shown in 3. In this figure, a set of measured one-way delays, a set of delays which are obtained by removing the relative skew from one-way delays according to Moon's approach, and the estimated line are illustrated. This figure show the Moon's algorithm can remove the effect of relative skew on one-way delays adequately.

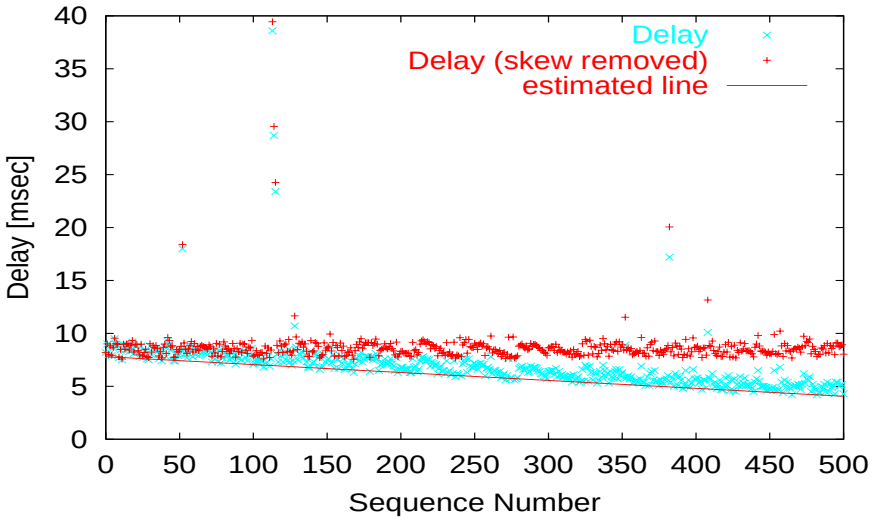


Figure 3: The Result of Skew Removal (One-way Delay, LAN)

2.3 Relative Factors Affecting Delay Characteristics

- **Effects of the "Time of Day":** It is known that the Internet traffic pattern repeats every day [17]. Thus, it is important to investigate the patterns of the suitable distribution function caused by the effects of "time of day".

- **Effects of the Encoder:** There are a lot of audio codecs to reduce the bandwidth usage in real-time applications. The different audio codec may lead the different

Table 1: Payload and Interval of Packet for Each Codec

codec	payload [bytes]	interval [msec]
G.711	160	20
G.728	40	20
G.729	20	20
G.723.1	24	30

delay characteristics because its packet payload sizes and packet sending intervals are determined according to its encoding algorithm.

We will investigate the effects of the following four encoders; G.711, G.728, G.729 and G.723.1 [18, 19, 20, 21]. G.711, generally called Pulse Code Modulation (PCM), is a low complexity codec and widely used in public switched telephone networks (PSTN). It provides toll voice quality at 64 Kbps. G.728 and G.729 are standardized by Telecommunication Standardization Sector of International Telecommunication Union (ITU-T) as low bit rate codecs, which can provide reasonable toll quality at 16 Kbps and 8 Kbps, respectively. G.729 is based on the Conjugate Structure Algebraic Code Excited Linear Prediction (CS-ACELP) method, and used in mobile telephony. G.723.1 is also an ITU-T standardized codec and has the smallest bit rate (6.3 Kbps) in our evaluated codecs. G.723.1 is used in videophone and used as a low bit rate telephone call in Japanese mobile phone. We summarize the size of payload in the packet and the packet transmission interval of each encoder in Table 1.

- **Effect of the Access Line:** Recently, there are various types of access lines for connecting to the Internet. Network characteristics such as the bandwidth are heavily influenced by the type of access lines. Thus, we will investigate the effects of access line on the delay characteristics. In our experiments, we will place the one side of end terminals on our laboratory's private network in Osaka University. We then connect the other side of end terminals by following three types of access lines; Local Area Network (LAN), Asymmetric Digital Subscriber Line (ADSL), and analog dial-up line. Note here that the ADSL and dial-up connections are asymmetric in

terms of the link capacity. The ADSL has 7,864 Kbps for downstream and 768 Kbps for upstream. In the analog modem case, downstream is 56 Kbps and upstream is 33.6 Kbps. Since the capacity of dial-up line is not enough for the G.711 codec, we do not examine the effect of G.711 codec over the dial-up connection.

In the measurements of one-way delay through the asymmetric link, we need to take care of the assumption made in Paxson's measurement method. The relative offset calculation in Paxson's method assumes that the propagation delays of both upstream and downstream are identical. That is, all links between two end hosts have the same bandwidth and the same propagation delay regardless of the direction. If we apply Paxson's method to the asymmetric link such as ADSL, we need to consider the difference of propagation delays between upstream and downstream before calculating the relative offset. Our ADSL had 7,864 Kbps for downstream and 768 Kbps for upstream, thus, the difference of propagation delays is about 1.7 msec. However, such a difference may be negligible, because the percentile of the difference is less than 1% of the value of one-way delay measured through ADSL in our experiments.

- **Effect of the Sampling-scale:** Because the parameters of the distribution function is determined by the measurement results, we need to identify problem how many measurements are required to estimate the delay characteristics accurately. We refer to this number of measurements as "Sampling-scale" in this thesis. In the modeling, if the sampling-scale for parameter estimation of the distribution functions is too small, it may lead to the poor estimation accuracy. Thus, it is essential to investigate the adequate value of sampling-scale for the determination of the suitable distribution function.

We will be back to above-mentioned problems in Section 4.

3 Modeling of Packet Transmission Delay

In this section, we model the distribution of the measured delays, according to the method described in [22] where the authors analyzed characteristics of `telnet` and `ftp` traffic. In modeling, however, we will look at the tail part of delay distribution, because it is important to detect packet loss in streaming applications as described before. Thus, we modify the analysis method for accurate modeling of the tail distribution. In what follows, we summarize our statistical analysis method.

3.1 Candidates of Models for Delay Distribution

We selected four distribution functions as candidates to model the delay distributions. The normal and exponential distributions are given by

$$F(x) = \int_0^x \frac{1}{\sqrt{2\pi}\sigma} exp\Big[\frac{-(y-\zeta)^2}{2\sigma^2}\Big] dy, \tag{4}$$

and

$$F(x) = 1 - \exp(-\frac{x}{\beta}), \quad \beta > 0, \tag{5}$$

respectively. The lognormal distribution is the function, of which variable is the logarithmic variable of the normal distribution, i.e.,

$$F(x) = \int_0^x \frac{1}{\sqrt{2\pi}\sigma y} exp\Big[\frac{-(\log y-\zeta)^2}{2\sigma^2}\Big] dy. \tag{6}$$

The Pareto distribution is widely known to be able to represent a self-similarity [23, 24], which is given by

$$F(x) = 1 - \left(\frac{k}{x}\right)^{\alpha}, \quad x \geq k. \tag{7}$$

3.2 Parameter Estimation

In order to detect the packet loss from the distribution of packet transmission delays, the coincidence at the tail part of distributions is more important, even if the measured data are far from the model distribution function in the other part of the probability distribution. In order to fit the distribution function of the candidates to the delay distribution accurately, we estimate parameters by utilizing only the tail part (e.g., 90–99.9%) of delay distribution. The upper bound is for eliminating the outlier. From our objectives of

applying to the playout control, 90–99.9% is suitable for accurate fitting on the tail part. For parameter estimation of each distribution function, we use the maximum-likelihood-estimator (MLE) method [25]. Parameters of the exponential and normal distributions can be estimated by calculating the mean and variance of measured delays. In the lognormal distribution, two parameters (ζ, σ) are calculated from the following equations;

$$\widehat{\zeta} = \frac{1}{n}\sum_{i=1}^{n} \log x_i, \tag{8}$$

$$\widehat{\sigma}^2 = \frac{1}{n}\sum_{i=1}^{n} (\log x_i - \bar{x})^2, \tag{9}$$

where n is the number of measurements and $\bar{x}$ is the mean of $\log(x_i)$ for all i. Parameters $(\widehat{k}, \widehat{\alpha})$ of the Pareto distribution are obtained from [25];

$$\widehat{k} = \min(x_1, x_2, \cdots, x_n), \tag{10}$$

$$\widehat{\alpha} = n\left[\sum_{i=1}^{n} \log\left(\frac{x_i}{\widehat{k}}\right)\right]^{-1}. \tag{11}$$

3.3 Determination Method of Adequate Distribution

We select the most appropriate probability distribution function by χ^2-test. Noting that a typical application of our analysis is the playout control for streaming applications, the estimation of the value around the target point (e.g., 95%, 99%, 99.9%) on the cumulative distribution of delays should be accurate, since it directly affects the packet loss ratio in streaming applications and the reproduction quality of real-time applications. From this reason, we evaluate the coincidence between the candidate functions and measured delays in 95–99.9% region of the cumulative distribution by the χ^2-test. The lower bound is determined based on the approach in [11].

In the χ^2-test, we investigate the degree of the confidence for fitting by $\widehat{\lambda}^2$, which is calculated as follows. We first separate the range of n measured delays into N subranges. For each subrange i, we obtain the probability p_i that an arbitrary value x belongs to the subrange i and the number of measurements Y_i falling into the subrange i. Then, λ^2 is given by

$$\widehat{\lambda}^2 = \frac{X^2 - K - N + 1}{n - 1}, \tag{12}$$

where

$$X^2 = \sum_{i=1}^{N} \frac{(Y_i - np_i)^2}{np_i}, \tag{13}$$

$$K = \sum_{i=1}^{N} \frac{Y_i - np_i}{np_i}. \tag{14}$$

The distribution having the smallest value of $\widehat{\lambda}^2$ is most appropriate to model the distribution of measured delays. Consequently, we determine the optimum model for the delay distribution.

4 Analytic Results

In this section, we show results of our statistical analysis described in the previous section. Note here that, we define a "target value" as a probability that the packet can be successfully playouted at the client host. We also note that the packet loss within the network is not considered here, and analyze only received packets. We assume that if a user wants to play the streaming audio with 1 % packet loss ratio, the target value should be set to 99 %. We investigate the accuracy of the modeling by using the target value. In experiments, we measured RTTs between our terminal for measurements and several WWW servers in Japan in August 2000. We next prepared servers and clients at Osaka University, Osaka City University, and individual home, and measured the one-way delays between the servers and clients in December 2001.

4.1 Effects of "Time of Day"

In this subsection, we investigate the effects of "time of day" on the modeling of delay distribution. First, we show the results of χ^2-test in Table 2. The first and second columns of Table 2 show the type of delays (RTT or one-way delay) and measured time, respectively. Values of $\widehat{\lambda}^2$ for four distributions are shown in columns 3 through 6. The smallest value of $\widehat{\lambda}^2$ among four distributions is shown in bold. As an example, Figure 4(a) compares the distribution of the measured RTTs with four candidates' cumulative distribution functions (CDFs) in busy hours (corresponding to the second row in Table 2). We set the target value to 99%. The distribution labeled by "Sample" is the tail part (90–99.9%) of the cumulative distribution of measured RTTs. The cumulative distribution of RTT values during non-busy hours is shown in Figure 4(b), which corresponds to the eighth row of the Table 2. It also shows the tail part of the measured RTTs' distribution and candidates' CDFs. We can observe from Table 2 that $\widehat{\lambda}^2$ of the Pareto distribution is always smallest in all experiments, i.e., the Pareto distribution is most suitable to estimate the 99% value of CDF in busy hours (e.g., 11 PM) and standard hours (e.g., 2 PM). Here, the reason that the network becomes heavy around 11 PM is NTT (one of largest carriers in Japan) offers the service with unlimited accesses at a fixed charge from 11 PM to 8 AM. Thus, many users connect their terminals to the Internet, and then the traffic increases. The

Table 2: Results on Model Determination (Tail-Part of Delay Distributions, G.728, Dial-up Line) Nor. : Normal, Exp. : Exponential, Lognor. : Lognormal

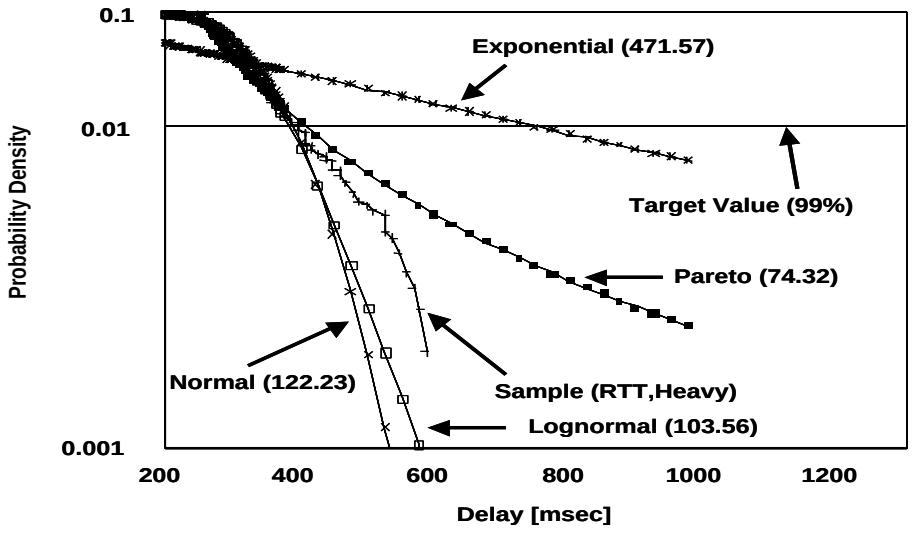
Measurement		Result of χ^2-test			
Delay Type	Hour	Nor.	Exp.	Lognor.	Pareto
RTT	10 PM	332.17	2371.91	266.60	**79.75**
RTT	11 PM	122.22	471.56	103.56	**74.32**
RTT	0 AM	156.09	670.34	128.86	**58.45**
RTT	1 AM	157.21	2189.33	139.47	**49.81**
RTT	2 AM	362.24	1691.48	242.74	**115.28**
RTT	7 AM	292.30	3598.50	240.55	**124.03**
RTT	10 AM	169.64	970.60	360.29	**80.57**
RTT	2 PM	147.02	599.37	250.51	**56.25**
RTT	7 PM	194.33	584.95	257.05	**55.63**
One-way	9 PM	83.82	602.56	71.96	**19.56**
One-way	11 PM	53.86	470.90	49.67	**30.10**
One-way	1 AM	55.06	426.46	49.99	**24.01**
One-way	5 AM	94.45	500.91	85.77	**25.16**
One-way	9 AM	107.76	754.09	98.74	**45.33**
One-way	12 PM	108.66	1218.95	101.09	**30.61**
One-way	3 PM	109.07	336.49	85.41	**21.21**

Table 3: Results on Model Determination (Entire Delay Distributions, G.728, Dial-up Line) Nor. : Normal, Exp. : Exponential, Lognor. : Lognormal

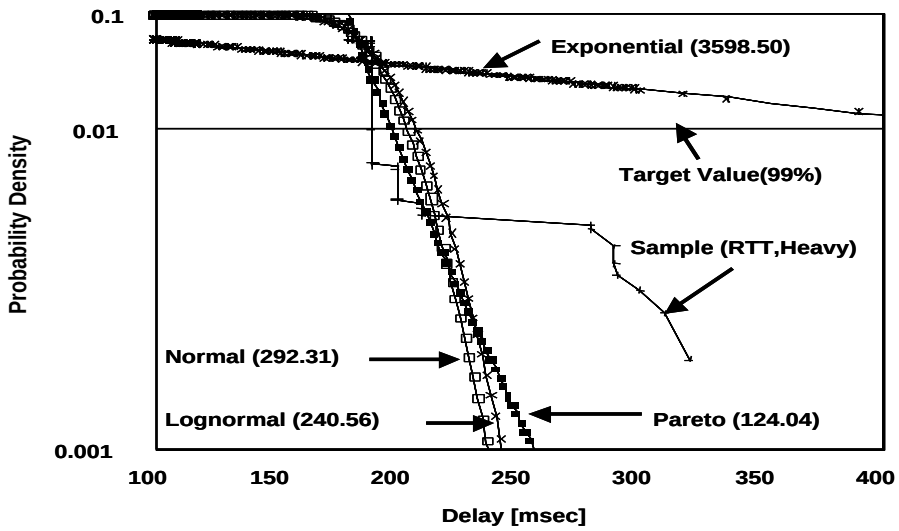
Measurement		Result of χ^2-test			
Delay Type	Hour	Nor.	Exp.	Lognor.	Pareto
RTT	11 PM	173.59	830.91	126.45	**100.22**
RTT	1 AM	164.39	1136.62	**130.49**	130.64
RTT	7 AM	154.59	1780.39	**97.49**	189.54
RTT	10 AM	**21.09**	49.27	32.16	36.873
RTT	2 PM	**22.07**	46.27	26.75	34.51
One-way	2 AM	4.71	25.43	**2.33**	3.51
One-way	4 AM	13.11	84.54	**12.85**	20.75
One-way	10 PM	20.93	268.79	**20.19**	281.81
One-way	5 PM	14.53	149.62	**13.11**	26.17
One-way	8 PM	4.18	5.55	**5.22**	16.96
One-way	11 PM	13.66	33.03	5.31	**3.92**

above observation is applicable to both RTTs and one-way delays.

To illustrate the importance of examining the tail part of the distribution, we next present the case where the χ^2-test is applied to the entire cumulative distribution. Table 3 shows the results. Compared with Table 2, the model determination method picked up different distributions (normal or lognormal distribution), which were not observed when examining only the tail part of distributions. Note that when the network is busy, the Pareto distribution which is heavy-tailed becomes most suitable. It coincides the past researches, which showed that the distribution of packet delays is heavy-tailed as the network becomes congested [22]. In Figure 5, we can observe the significant increase of delays at 11 PM. From 11 PM to around 1 AM, the delay becomes heavy-tailed, because the ratio of long delays is larger then the other period.



(a) Busy Hour (RTT, 11 PM, G.728, Dial-up Line)



(b) Standard Hour (RTT, 2 PM, G.728, Dial-up Line)

Figure 4: Comparisons among Delay Distribution and Candidates' CDFs

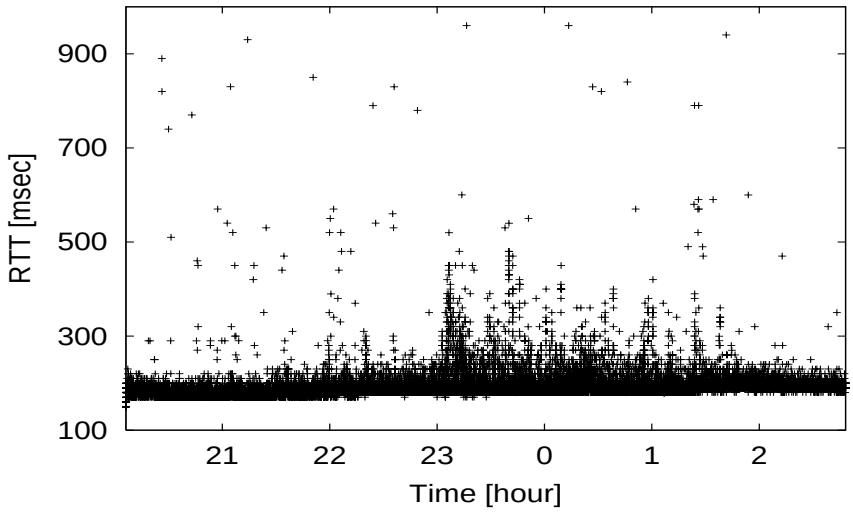
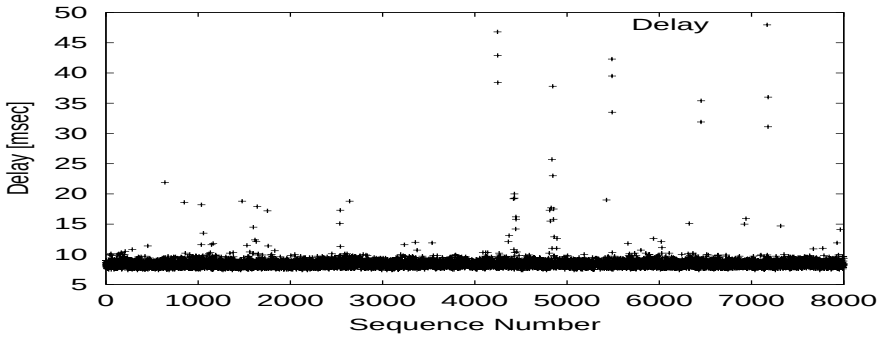


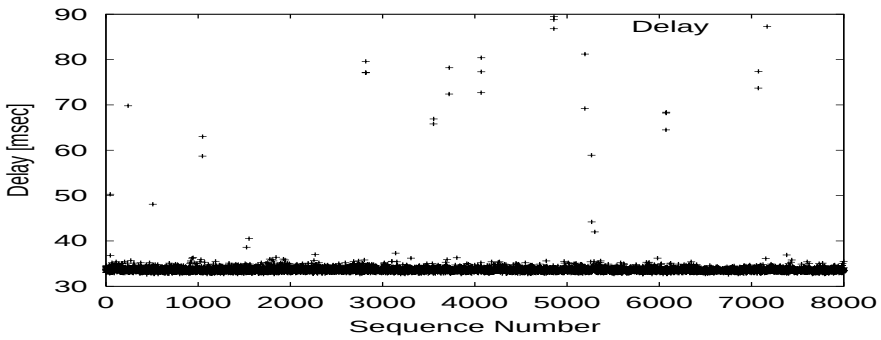
Figure 5: Time Dependent Fluctuations of RTTs

4.2 Effects of Access Line

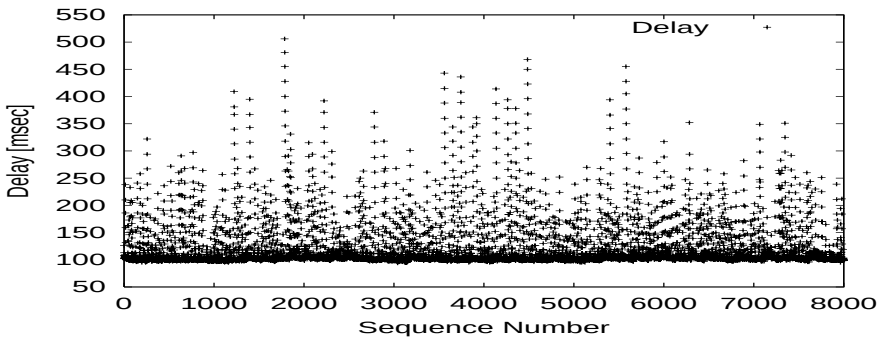
We next show the effects of the access line. First, we compare the results of measuring one-way delays among three types of access lines mentioned in Section 2. Figure 6 shows the time dependent variations of the one-way delays. In all measurements, sending packets are generated by the G.723.1 codec and transmitted at 1 PM. Both of the values of one-way delays via LAN and ADSL are small, and concentrate on the bottom line of the lower bound of measured delays (see Figures 6(a) and 6(b)). On the other hand, the values of one-way delays via the dial-up line are larger than those of LAN or ADSL, and the fluctuation of the delays is always drastic during the measurements. However, by the analysis, we found the Pareto distribution was always most adequate for modeling the one-way delays distribution regardless of the types of access lines. Figure 7 shows the distribution of one-way delays corresponding to Figure 6 and four candidate's CDFs. We also show the results of χ^2-test in Table 4. From Figure 7, we observe that the distributions of the one-way delays for each access line have different curve. However, Table 4 indicates that the adequate model is the Pareto distribution in all cases.



(a) LAN

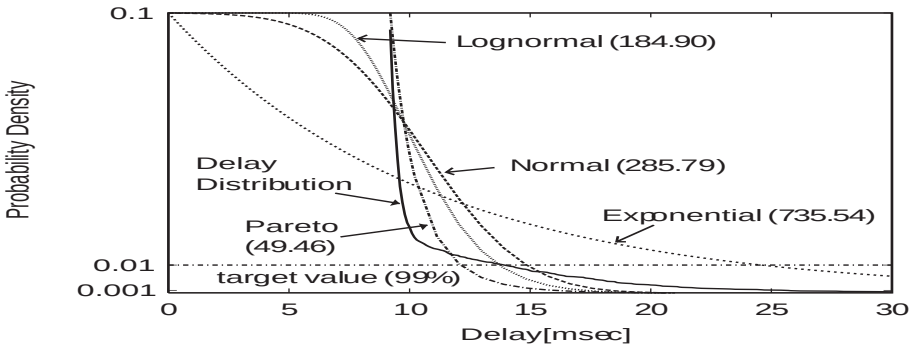


(b) ADSL

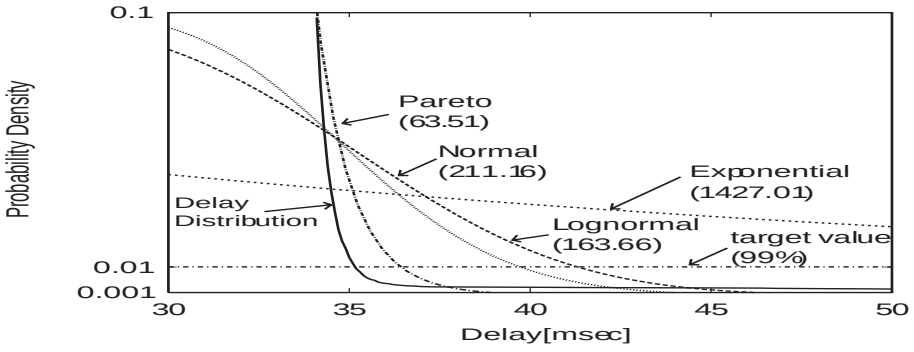


(c) Dial-up

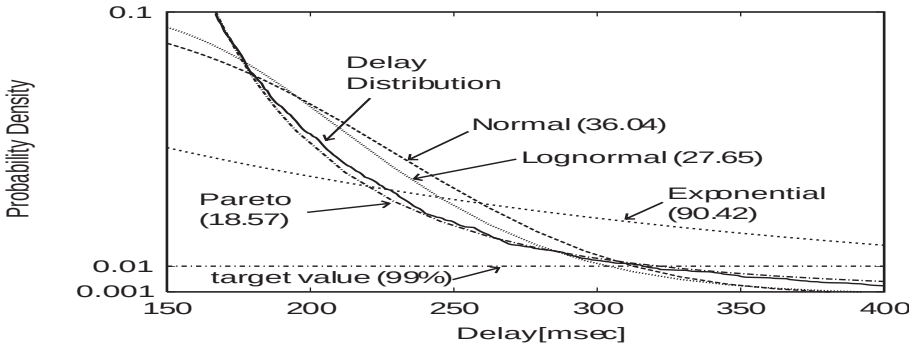
Figure 6: Comparison of the One-way Delay among Different Access Lines



(a) LAN



(b) ADSL



(c) Dial-up

Figure 7: Distribution Comparison (One-way Delay, 3 PM, G.723.1)

Table 4: Results on Model Determination (One-way Delay, G.723.1, 3 PM)

Access Line	Result of χ^2-test			
	Nor.	Exp.	Lognor.	Pareto
LAN	285.79	735.54	184.90	**49.46**
ADSL	211.16	1427.01	163.66	**63.51**
Dial-up	36.04	90.42	27.65	**18.57**

Table 5: Results on Model Determination (One-way Delay, LAN, 3 PM)

Encoder	Result of χ^2-test			
	Nor.	Exp.	Lognor.	Pareto
G.711	364.80	1711.38	303.34	**71.87**
G.728	40.69	167.70	30.00	**15.70**
G.729	74.32	32.04	36.77	**15.05**
G.723.1	285.79	735.54	184.90	**49.46**

4.3 Effects of Audio Encoder

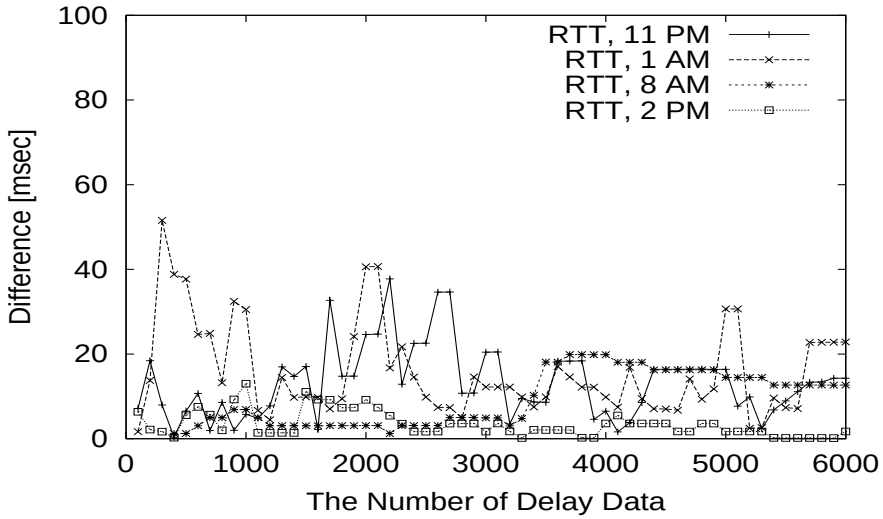
Here, we examine the effects of audio encoders. Table 5 compares the results of χ^2-test among four encoders. From the results in Table 5, we can also confirm that the Pareto distribution is most suitable for the model of the delay distribution as well as "time of day" and the access line.

In summary, we can conclude that the Pareto distribution is always adequate for the model of delay distribution regardless of any factor ("time of day", access line, and audio encoder). It is also worth noting that as we will describe in the next section, we will apply the statistical results presented in this section to on-line estimation of the delay, which is necessary in our adaptive playout buffer algorithm. Thus, we want a light CPU load for the estimation method. If the appropriate model is varied according to the factors cited by use, we need to examine the χ^2-test for each playout controls. However, the computational overhead of χ^2-test is not small, and it is inadequate for real-time applications. Fortunately, however we found that the Pareto distribution is most

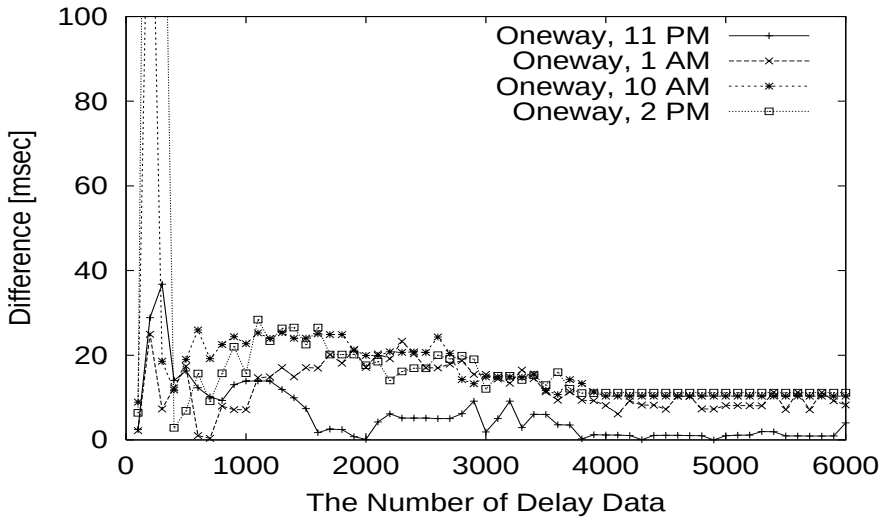
appropriate in any situation. Thus we only have to determine the parameters of the Pareto distribution. This strength is promising to estimate the packet transmission delay with lower CPU calculations.

4.4 Effects of Sampling-scale

We next examine the effects of the sampling-scale. Figures 8(a) and 8(b) show the degree of differences against the sampling-scale when modeling RTT and one-way delays, respectively. We calculate the difference between 99% values of the Pareto distribution and those of the cumulative distribution of measured delays. As shown in the figure, the difference gets remarkable when the sampling-scale is less than 500. On the other hand, we cannot observe critical changes when the sampling-scale is equal to or more than 500. Since our objective is to perform on-line estimation of the delay distribution, it is preferable that the sampling-scale is as small as possible, so that the parameter can be estimated faster with the less sampling-scale. From the results, we can conclude that the required sampling-scale should be equal to or more than 500, in which $500 \times (99.9\% - 90\%) \simeq 50$ delay samples are at least necessary for the accurate parameter estimation of the Pareto distribution. We will evaluate the required sampling-scale through experiments for quick and still accurate parameter estimation in the next section.



(a) RTT



(b) One-way Delay

Figure 8: The Differences between Values of the Pareto Distribution and Measured Data at the Target Value (Target Value=99%, G.728, Modem).

5 Adaptive Playout Buffer Algorithm based on Network Parameters

In Section 4, we investigated the characteristics of the end-to-end packet delays. In this section, we focus on the playout buffer algorithm for streaming applications as an application of our delay analytic results. We propose a new playout buffer algorithm based on the results of our statistical analysis. Our proposed algorithm determines the playout delay so that the packet loss ratio specified by the users is satisfied.

5.1 Existing Playout Buffer Algorithms

For comparison purpose, we also examined four algorithms which have been proposed in [6, 8]. Before introducing our proposed playout buffer algorithm, we describe the brief overviews of each playout buffer algorithm.

Exponential-Average (Exp-Avg): In this algorithm, the playout delay $\widehat{p}_i$ of the ith arrived packet is determined from the approximated values of the mean $\widehat{d}_i$ and variance $\widehat{v}_i$ of one-way delays, which are given by

$$\widehat{p}_i = \widehat{d}_i + 4\widehat{v}_i, \tag{15}$$

$$\widehat{d}_i = \alpha\widehat{d}_{i-1} + (1-\alpha)n_i, \tag{16}$$

$$\widehat{v}_i = \alpha\widehat{v}_{i-1} + (1-\alpha)|\widehat{d}_i - n_i|, \tag{17}$$

where n_i means the one-way delay of ith packet. The value of α is chosen to be 0.998002 according to [6]. Thus, playout time $\widehat{t}_i$ is determined from playout delay $\widehat{p}_i$ and time s_i when the sender host sent packet according to Eq. (18). Here, the playout time means the time when the application client actually starts playing audio data recorded in the packet.

$$\widehat{t}_i = \widehat{p}_i + s_i, \tag{18}$$

Thus, **Exp-Avg** estimates the playout time from means and variances, and does not consider the distribution of the delays.

Fast Exp-Avg (F-Exp-Avg): This algorithm is a modified version of **Exp-Avg**. **F-Exp-Avg** computes the weighted mean of $\widehat{d}_i'$s as

$$\widehat{d}_i' = \begin{cases} \beta \widehat{d}_{i-1}' + (1-\beta) n_i & \text{if } n_i > \widehat{d}_{i-1}', \\ \alpha \widehat{d}_{i-1}' + (1-\alpha) n_i & \text{otherwise,} \end{cases} \tag{19}$$

where α and β are constant values, satisfying $0 < \beta < \alpha < 1$. We set $\alpha = 0.998002$ and $\beta = 0.750000$ following [6].

Spike Detection (SPD): This algorithm focuses on *spike* which represents a sudden and large increase in delays over a sequence number of packets. Examples of spikes are shown at 4,650, 4,800, and 5,000 in Figure 10(a). **SPD** usually obtains the playout delay from Eq. (16), which is same as **Exp-Avg**. During spike, however, **SPD** uses the following equation;

$$\widehat{d}_i = \widehat{d}_{i-1} + n_i - n_{i-1}, \tag{20}$$

to catch up the sudden increase of delays. In **SPD**, we use $\alpha = 0.875$ following [6].

Window: This algorithm proposed in [8] intends to detect a spike as **SPD**. During the spike, the first packet in the spike is used as a playout delay. After spike, the playout delay is chosen by finding the delay corresponding to the qth quantile of the distribution of the last N packets received by the receiver. In our evaluation, a value of 0.99 is used for q, and 10,000 for N, which are used in [8].

5.2 Proposed Algorithm

To provide a high-quality communication in streaming applications, it is desirable that the packet loss ratio and playout delay are kept small. However, there is a critical trade-off between packet loss ratio and the length of the playout delay. Existing algorithms aim at minimizing the packet loss ratio with a minimum playout delay. On the other hand, the main goal of our algorithm is to control the packet loss ratio while keeping the playout delay as small as possible. We utilize the results obtained through the statistical analysis presented in the previous section to determine the optimum playout delay. The key idea is that we determine the playout delay from the Pareto distribution so that the packet loss ratio becomes equal to the ratio preferred by the user.

We show the design of our proposed playout buffer algorithm more specifically. Our playout buffer algorithm records the history of one-way delays of packets. On each packet arrival, parameters (k, α) of the Pareto cumulative distribution function $F(x)$ are updated to estimate the playout delay p_i from the equation $F(p_i) = X$, where X is a target value. The target value is equal to the reproduction ratio of packets preferred by the user. From the Pareto CDF in Eq. (7), our proposed algorithm determines the playout delay by

$$\widehat{p}_i = \frac{k}{\sqrt[\alpha]{1 - \frac{X}{100}}}. \tag{21}$$

We consider 95, 99, and 99.9% as the target value X through our numerical results. For example, if we choose $X = 95\%$, our algorithm tries to minimize the playout delay while keeping the packet loss to be 5%. Note that of course, if the packet loss within the network exceeds 5%, our method has no means to keep the packet loss to be 5%. In what follows, we will assume that the packet loss ratio in the network does not exceed the target value. We refer to our proposed playout buffer algorithm as the loss control playout buffer algorithm (**Loss-Control**).

6 Performance Evaluation of Playout Buffer Algorithm

In this section, we evaluate the performances of playout buffer algorithms by the trace-driven simulation, and we investigate an effectiveness of the proposed algorithm.

6.1 Simulation Method

First, we prepared a set of one-way delays of packets for our trace-driven simulation. We measured the one-way delays with various combinations of "time of day", encoders, and access lines. Second, in our simulation, the recorded one-way delays are traced one by one, and the playout delay p_i of the ith packet is estimated according to each algorithm for all measured delays. Then, we check whether the delay of the next packet is smaller than the estimated playout delay or not. If the delay is larger than the estimated playout delay, the packet is treated to be lost. After tracing all measured delays, average playout delay and packet loss ratio are computed as the output.

6.2 Parameter Setting

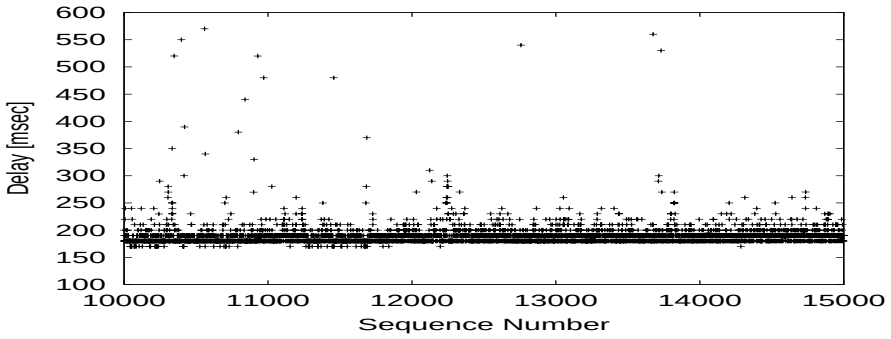
In **Loss-Control**, the number of delays for the parameter estimation is a dominant factor affecting the accuracy of loss control. Generally, the accuracy of parameter estimation can be improved by increasing the number of delay samples. However, if the number of delays for parameter estimation becomes larger, the effect of one delay value on the parameter estimation becomes less respectively. It means that the variation of the delay have little impact on parameter estimation, and the estimated Pareto CDF does not reflect the changes of the recent delays. However, the network condition sometimes changes dynamically, typically found in the *spike*. Thus, it is likely that large number of the delays is not suitable for parameter estimation in **Loss-Control**. From these reasons, it is necessary to investigate how many number of delays is adequate for **Loss-Control**. Note that the minimum number of delays should be 500 to estimate accurate parameters of the Pareto distribution as confirmed in the Subsection 4.4.

We now demonstrate the influence of the number of delays for parameter estimation on **Loss-Control**. The results of experiments are shown in Figure 9. The traced one-way delays are plotted in Figure 9(a). In our experiment, we change the number of delays

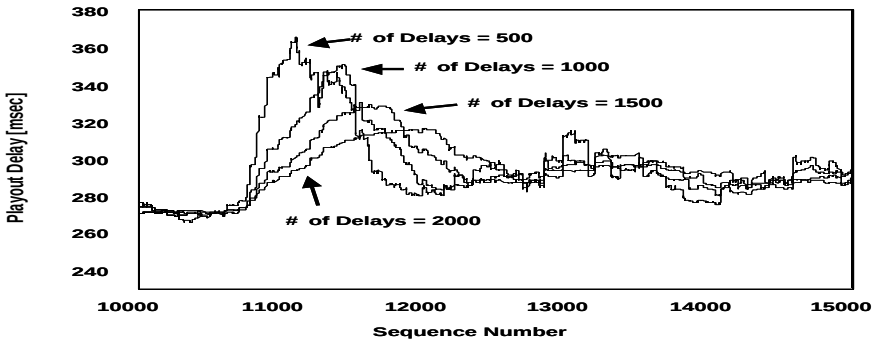
to be 500, 1,000, 1,500 or 2,000. Figure 9(b) shows the four variations of playout delays evaluated by **Loss-Control**, dependent the samples of delays for parameter estimation. From Figure 9(b), it is clear that the smaller the number of delays is, the more quickly the playout delay changes. It is clearly shown during period from sequence number 11,000 to 12,000. This result indicates the smaller number of delays is better for the playout delay evaluated by **Loss-Control** in order to follow the changes of delays. Next, in Figure 9(c), the packet loss ratios dependent on the number of delays are plotted. In this figure, we plot four cases in the busiest (11 PM) and non-busy (2 PM, 3 AM, 9 AM) hours, and we set the target packet loss ratio to be 1%. In Figure 9(c), we cannot observe significant improvements at any case even if the number of delays becomes larger. The less number of delays in parameter estimation also has an advantage of the less computing, which would be desirable for the real-time applications. Based on above results, the number of delays is set to be 500 in **Loss-Control** in the below.

6.3 Evaluation Results

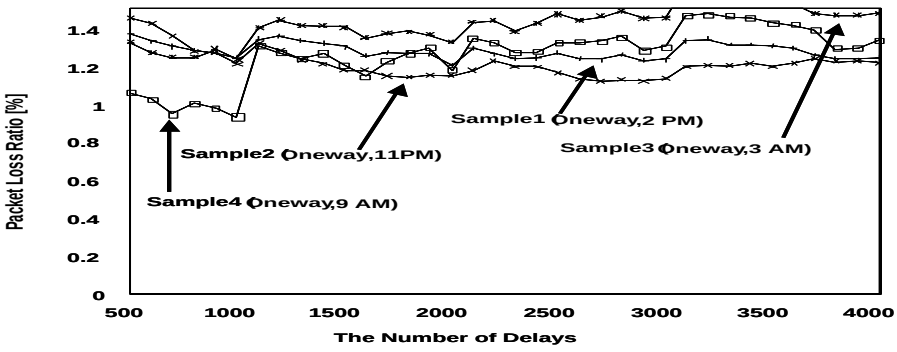
Table 6 compares packet loss ratios (PLRs) and mean values of the playout delays evaluated by simulation in three cases. The first case is "dynamic", in which the values of one-way delays often change, and many spikes are observed in this case. The packets were sent by the G.723.1 encoder at 2 PM and delivered to the receiver via the dial-up line. The second is "moderate" case, in which there are a small number of spikes. We used the G.711 encoder and ADSL and the delays were measured at 1 PM. The last case is "quiet", where no dynamic change of delays were observed. These delays were sent by the G.723.1 encoder at 2 PM and delivered to the receiver through LAN. In **Loss-Control**, we used 95, 99, and 99.9% as the target values. From this table, we can observe a clear trade-off between PLRs and the playout delays. Note again that the purpose of our proposed **Loss-Control** is that the ratio of the playouted packet can be kept close to the target value requested by users. The results in Table 6 show that PLRs of **Loss-Control** are almost close to the intended packet loss ratio $(1 - X)$, although there are a few differences between actual PLRs and target ratios. On the other hand, the objective of **Exp-Avg**, **F-Exp-Avg**, or **SPD** is to minimize the playout delay while keeping the PLR to be 0%. In the "quiet" case, these algorithms can accomplish their purposes. However, in the



(a) Variations of One-way Delays



(b) Variations of Playout Delays



(c) Variations of Packet Loss Ratio

Figure 9: Effects of the Number of Delays for Parameter Estimation in Loss-Control

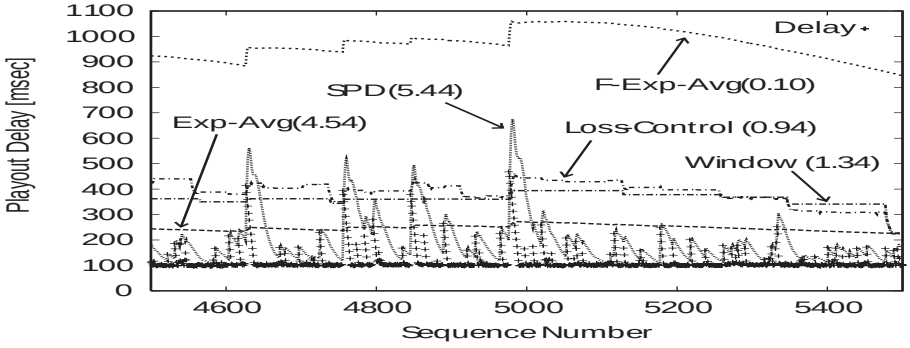
Table 6: Comparison of PLR and Mean Playout Delay

Case	Algorithm	Target	PLR [%]	Mean of d_i [msec]
"dynamic" G.723.1 Dial-up Line 2 PM	Loss-Control	95%	5.7	227.92
		99%	0.94	387.12
		99.9%	0.12	770.44
	Exp-Avg	-	4.54	247.91
	F-Exp-Avg	-	0.10	970.34
	SPD	-	5.44	198.74
	Window	99%	1.34	362.57
"moderate" G.711 ADSL 1 PM	Loss-Control	95%	6.02	40.61
		99%	1.77	58.45
		99.9%	0.60	375.28
	Exp-Avg	-	4.93	39.79
	F-Exp-Avg	-	0.04	102.26
	SPD	-	3.08	39.74
	Window	99%	2.33	48.60
"quiet" G.723.1 LAN 2 PM	Loss-Control	95%	3.94	9.40
		99%	0.72	9.87
		99.9%	0.22	10.53
	Exp-Avg	-	0.18	10.49
	F-Exp-Avg	-	0.01	29.53
	SPD	-	0.77	10.19
	Window	99%	1.05	9.76

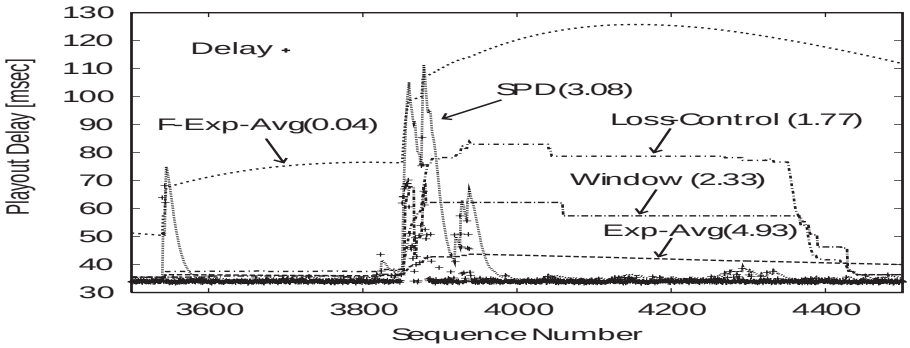
"dynamic" case, the PLRs of **Exp-Avg** and **SPD** cannot achieve the objective PLR. On the contrary, although the PLR of **F-Exp-Avg** is close to 0%, the playout delay is too large, not suitable for "real-time" application. Of course, the PLRs of **Exp-Avg**, **F-Exp-Avg**, and **SPD** might be controlled by changing the multiplier of $\widehat{v}_i$, which is currently set to be 4 (see Eq. (16)). However, the fundamental problem is that there is no way to map the multiplier $\widehat{v}_i$ to the value of PLR in those algorithms. **Window** keeps the playout delays as small as possible, and the PLRs are close to 1%. However, in the "dynamic" and "moderate" cases, **window** cannot accurately control the playout delay.

Figure 10 compares the variations of playout delays for five algorithms corresponding to Table 6. The target value X of **Loss-Control** is 99%, and the PLR of each algorithm is also shown by the number in parentheses. From these figures, we can find that **F-Exp-Avg** has a tendency to overestimate the playout delays, which is larger than twice of playout delays of **Loss-Control** in any case. Especially, in Figure 10(a), it is undesirable that the playout delay increases up to 800 msec. On the other hand, **Exp-Avg** and **SPD** always compute the small playout delay, which leads to many packet losses. Additionally, Figure 10(a) shows that **Exp-Avg** cannot follow the variation of packet delays adaptively. Therefore, **Exp-Avg** is not suitable in the "dynamic" case. Figure 10(a) also shows that **SPD** can follow the drastic changes of delays because of the spike detection. As shown in Figures 10(b), however, **SPD** is too sensitive to the spike, then it sometimes misleads to unnecessary increase of playout delay. The variations of playout delays of **Window** and **Loss-Control** are similar with each other, because both utilize the distribution of one-way delays. They can detect the spike, and change their playout delays for keeping a stable loss ratio. Moreover, we can see that they are keeping the playout delay small.

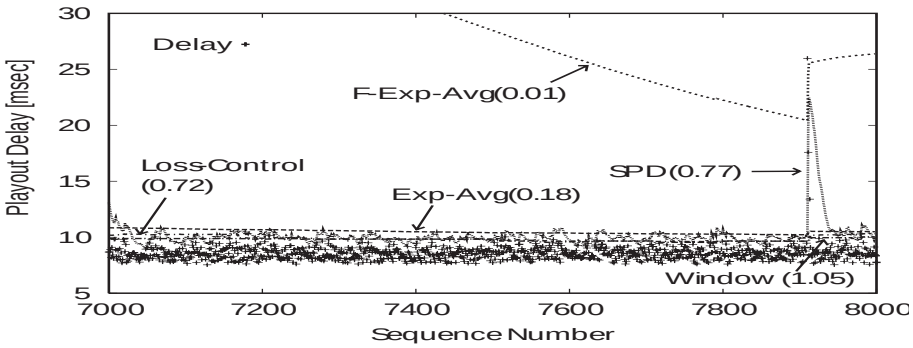
In summary, our simulation results show that **Exp-Avg**, **F-Exp-Avg**, and **SPD** cannot keep the small PLR by the heavy fluctuation of packet transmission delays. The PLRs of **Window** are different from 1% in some cases. Thus, it is likely that **Window** does not control the playout delay with the accuracy as same as **Loss-Control**. On the other hand, we can conclude that our algorithm is superior to others in that **Loss-Control** provides a stable and desired PLR specified by the user regardless of variations of one-way delays.



(a) "dynamic"



(b) "moderate"



(c) "quiet"

Figure 10: Playout Delay Variations

Figure 11: Operation Window of Client[†]

6.4 Performance Evaluation through Implementation Experiments

We developed a streaming client in which our PBA is implemented, and verified the applicability of our algorithm by running the application. More specifically, we implemented our PBA as an input plug-in of `Winamp` [26], which is one of major frontends in real-time applications today.

We placed the streaming server at Osaka University, which sends audio packets generated by the G.711 or G.728 encoder (the size of the packet and transmission interval are shown in Table 1). Packets are transmitted via the Internet and transferred to our developed client. In the client, the smoothing buffer is adjusted based on the playout delay calculated by our PBA (**Loss-Control**). Arrived packets are stored into the buffer and then the client starts playing after the playout interval. The client application can also control its target value through a GUI Interface. Figure 11 shows the operation window of our client.

Our implementation experiments include (1) to check whether the packet loss ratio

[†]©Nullsoft Inc. 2002

Table 7: Playout Delays and PLRs of Loss-Control through Implementation Tests (G.728, Dial-up Line, 4 PM)

Situation	Target Value [%]	PLR [%]	Average Playout Delay [msec]
G.728	95%	6.53	134.16
Dial-up Line	99%	0.82	264.57
4 PM	99.9%	0.12	774.17
G.711	95%	5.02	9.71
LAN	99%	1.61	11.01
4 PM	99.9%	0.03	25.52

measured by the application satisfies the target value, (2) to verify whether the computational overhead of calculating the playout delays is enough small to operate our PBA in *real-time*, and (3) to investigate whether the application can actually control the perceived quality according to end users specifications.

Table 7 shows results of our implementation tests. We can observe similar results to those shown in Table 6, and our streaming client can control its packet loss ratio by setting the target value. We also measured the computation time for calculation of playout delays in our client program. The platform was Microsoft Windows 98 operating system on Intel Pentium III 750 MHz CPU. In this case, the computation overhead was about 0.02 msec for each packet arrival; 0.1% of packet transmission interval in G.711, which is sufficiently small overhead. Note that we also confirmed that the audio playing is not interrupted by any other factors except packet losses.

7 Extension of Playout Buffer Algorithm to Maximize the Perceived Quality

In Sections 5 and 6, we have proposed a new playout buffer algorithm (**Loss-Control**), and demonstrated that **Loss-Control** can control the packet loss ratio as users like. As described in Section 1, however, there is a high possibility that the delay and other network parameters (type of codecs, access lines) in addition to the PLR would affect the perceived quality. The end users can choose the preferable playout quality, but there are still many configurable parameters left to the end users. It is hence necessary to introduce a simpler index for the end users, which directly indicates the perceived quality in multimedia communications. Today, many metrics to express the playout quality are proposed and evaluated. Those are categorized into two major types; subjective and objective. Objective indexes are based on the bit-level difference between the original source and the playouted data. Typical examples include Signal-to-Noise Ratio (SNR) and Segmental Signal-to-Noise Ratio (SSNR) [27]. Objective approaches are universal in the sense that indices are simply obtained by numerical calculation and independent of human preferences. However, there is still a critical problem in these metrics. The perceived quality is varied according to an individual difference and/or type of applications, even if objective metrics of playouted data are the same. Namely, the end users are still forced to control the perceived quality through trial and error. On the other hand, the subjective metrics that we adopt in this thesis are more user-friendly because it is based on scores made by users according to listening and/or watching the played media. Mean Opinion Score (MOS), Diagnostic Acceptability Measure (DAM), Perceptual Speech Quality Measure (PSQM), Measuring Normalizing Blocks (MNB), and Perceptual Analysis Measurement System (PAMS) are categorized in the latter class [28, 29].

Our objective in this section is to maximize the subjective index of the perceived quality for given network parameters, which are automatically measured. We first model relations between the MOS and network parameters shown in [11] into mathematical formulas. After modeling, we obtain the MOS-relative form, which gives the appropriate packet loss ratio and playout delay according to the MOS value. We then modify our **Loss-Control** PBA to apply above MOS-relative function. Numerical comparisons are

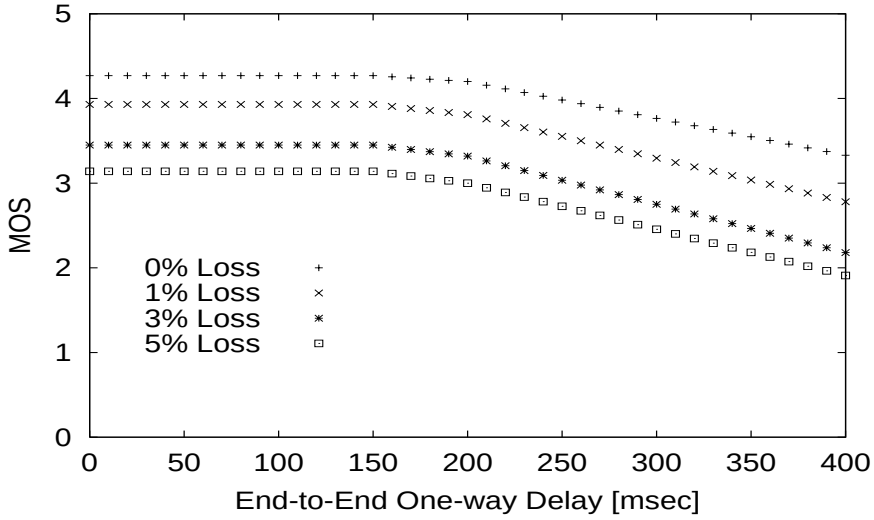


Figure 12: Effects of PLR and Delay (Encoder: G.711)

shown in the last part of this section.

7.1 Effects of Packet Loss Ratio and Delay on MOS

To clear the relation between the MOS value and network parameters, we pick out the data from [11], which show the effects of network parameters on MOS. We show one referred data in Figure 12. Each plot shows the relation between MOS and end-to-end one-way delay in given loss ratio. From the model of one-way delay distribution described in Section 4, we can get the feasible combinations of playout delay and the packet loss ratio to maximize the MOS index. We describe our modeling method in the next subsection.

7.2 Modeling Methods of MOS Functions

The first step of our modeling is to formulate relations among MOS, packet loss ratio, and playout delay approximately. That is, we plot MOS curves shown in Figure 12 with mathematical notations given by our modeling method. Of course, our formulas depends on the relation shown in Figure 12. However, our modeling approach is also applicable to another result of relation.

From Figure 12, we can have the following assumptions.

- Four curves shown in Figure 12 are in parallel. It means that the packet loss ratio and one-way delay, and hence the playout delay, affect MOS independently. From this assumption, we can consider the effect of packet loss ratio and one-way delay on modeling MOS separately.

- The degree of degradation in MOS values is proportional to the packet loss ratio, and it does not depend on the playout delay.

Under above-mentioned assumptions, we can obtain the MOS function $M(p, d)$ for given packet loss ratio p and the playout delay d from $M(p)$ and $M(d)$ separately. We now determine the MOS function as follows.

We first model the MOS function $M(d)$ for given playout delay d with a three-dimensional polynomial approximation, where the packet loss ratio p is assumed to be zero (shown in the cross point in Figure 12). Parameters of the polynomial are obtained by curve fitting. We then obtain the MOS function $M(d)$ as

$$M(d) \approx 4.10 + 2.64 \times 10^{-3} d - 1.86 \times 10^{-5} d^2 + 1.22 \times 10^{-8} d^3. \tag{22}$$

We then get the MOS curve by sliding inversely in the horizontal direction. By applying the second assumption described above, the degree of degradation of MOS values is proportional to the packet loss ratio. We calculate the parameter of the function by the least linear square method. The MOS function $M(p)$ for given packet loss ratio p is hence expressed as

$$M(p) \approx 4.10 - 0.195p, \tag{23}$$

where the playout delay is set to $d = 0$.

Because we assume that the packet loss ratio and the playout delay affect the MOS value independently, we can obtain $M(p, d)$ for given p and d by combining Eqs. (22) and (23), i.e.,

$$\begin{aligned} M(p, d) &= 4.10 - 0.195p + 2.64 \times 10^{-3} d \\ &\quad -1.86 \times 10^{-5} d^2 + 1.22 \times 10^{-8} d^3. \end{aligned} \tag{24}$$

In Figure 13 we add the solid curves by Eq. (24) to Figure 12, and we can observe our approximate modeling provides an agreement with the original ones.

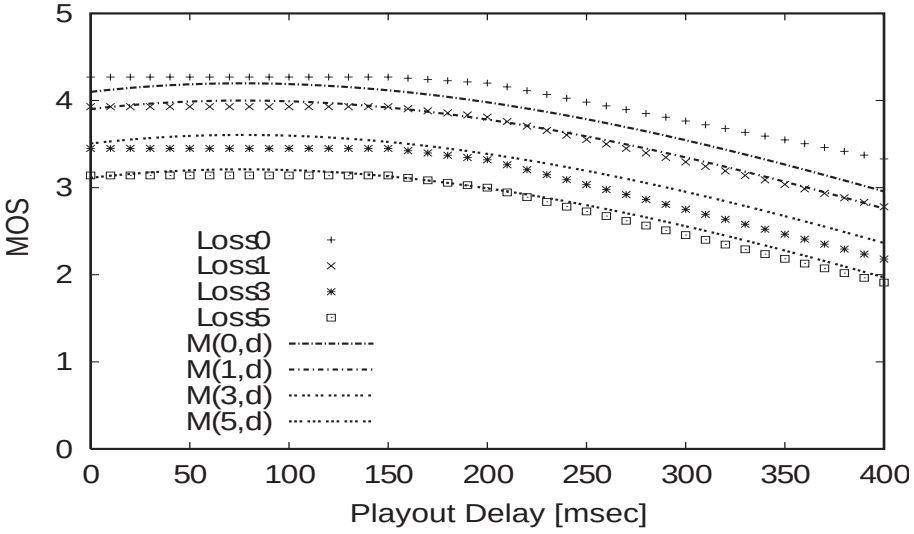


Figure 13: Results of Modeling MOS values and Network Parameters (Encoder: G.711)

In the real network, however, there is a correlation between the packet loss ratio p and the playout delay d. In streaming applications, the packet loss ratio p is a summation of (1) packet loss ratio caused by packet drops within the network (referred to as p_n), and (2) ratio of late arriving packets exceeding playout threshold (p_d). That is,

$$p = p_n + p_d. \tag{25}$$

From Eq. (7) in Section 3, we have a relation between p_d and d as follows;

$$p_d = 100\left(\frac{k}{d}\right)^{\alpha}. \tag{26}$$

By applying Eq. (26), Eq. (24) can be rewritten as

$$\begin{aligned} M(p_n, d) &= 4.10 - 0.195\left(p_n + 100\left(\frac{k}{d}\right)^{\alpha}\right) + 2.64 \times 10^{-3} d \\ &\quad -1.86 \times 10^{-5} d^2 + 1.22 \times 10^{-8} d^3. \end{aligned} \tag{27}$$

As shown in Eq. (27), two parameters d and p_n affect the MOS value. For the streaming application, however, only the playout delay d is controllable. We therefore consider Eq. (27) as a function of d, denoted by $Q(d)$, i.e.,

$$\begin{aligned} Q(d) &= 4.10 - 0.195 p_n - 19.5\left(\frac{k}{d}\right)^{\alpha} + 2.64 \times 10^{-3} d \\ &\quad -1.86 \times 10^{-5} d^2 + 1.22 \times 10^{-8} d^3. \end{aligned} \tag{28}$$

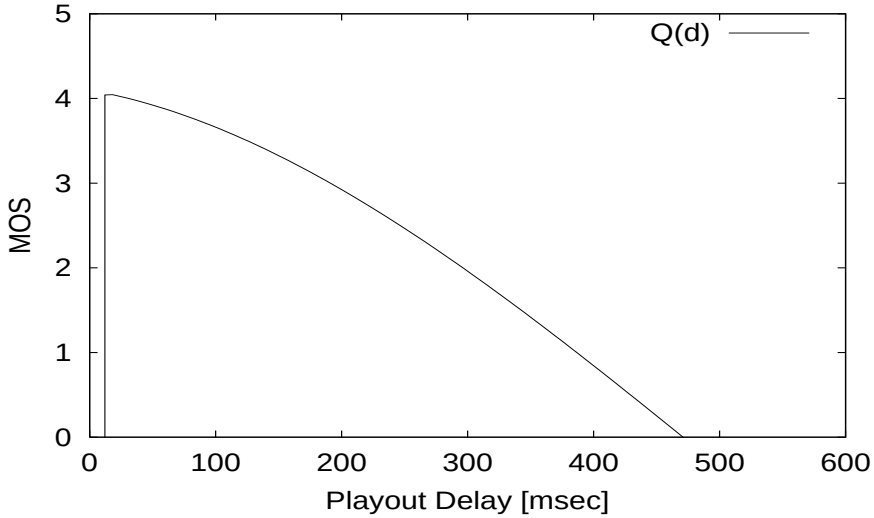


Figure 14: MOS Function $Q(d)$ (Encoder: G.711)

We now examine $Q(d)$. If $d = 0$, all packets are treated as packet loss, and no packet is played. Thus, we set $Q(0) = 0$. As the playout delay is increased, $Q(d)$ takes a smaller value. However, when the playout delay is too large, $Q(d)$ is again degraded due to the large delay for playing. Therefore, there exists an optimum point of d that provides the maximum value of $Q(d)$. Figure 14 shows the example of variation in $Q(d)$ dependent on the playout delay d, where α and k in Eq. (28) are set to 9.10 and 15.53 from measured data, respectively. Calculating the optimal d is realized by the *false position method* [30] utilizing the differential equation of $Q(d)$. Because $Q(d)$ is the convex function, we can determine the optimal d from the x-intercept of the differential equation of $Q(d)$ by the false position method.

7.3 Modified Playout Buffer Algorithm for Enhancing MOS Index

We modify our **Loss-Control** PBA presented in Section 5 to realize the MOS-based control. In our **Loss-Control** algorithm, the playout delay was determined from the target packet loss ratio. On the other hand, our new algorithm is to control the playout delay to maximize the MOS value $Q(d)$. More specifically, our new PBA consists of the following steps;

(1) Measure the transmission delays of arrived packets

(2) Calculate the parameter of Pareto distribution (α, k) by the MLE method (see Subsection 3.2)

(3) Assign the value of (α, k) into the MOS function $Q(d)$

(4) Obtain the optimal value of d, maximizing $Q(d)$, by utilizeing the *false position method* applied to the differential equation of $Q(d)$

(5) Set the playout delay to d

(6) Return to Step.1

We refer to this new playout buffer algorithm as enhanced MOS-based playout buffer algorithm (**E-MOS**).

7.4 Performance Evaluation

Now, we evaluate the performance of **E-MOS** by simulations. Table 8 compares packet loss ratios (PLRs), mean values of the playout delays, and MOS evaluated by simulation in the following three cases; The first case is "dynamic", in which the values of one-way delays often change, and many spikes are observed. The packets were sent by the G.723.1 encoder at 2 PM and delivered to the receiver via the dial-up line. The second is a "moderate" case, in which there are a several number of spikes. We used the G.711 encoder on ADSL and the delays were measured at 1 PM. The last case is "quiet", where no dynamic change of delays were observed. These delays were sent by the G.723.1 encoder at 2 PM and delivered to the receiver through LAN. In **Loss-Control**, we used 95, 99, and 99.9% as the target values. The MOS values shown in the last column of Table 8 are evaluated by Eq. (24) from the PLR and playout delay. The maximum value of MOS among all PBAs is shown in bold.

Results in Table 8 indicate that **E-MOS** can provide the highest perceived quality for users in any network conditions. Looking at the playout delays and PLRs of **E-MOS**, we can find that **E-MOS** has a tendency to minimize the PLR when the one-way delays are small ("moderate" and "quite" cases in Table 8). From Figure 12, we can observe

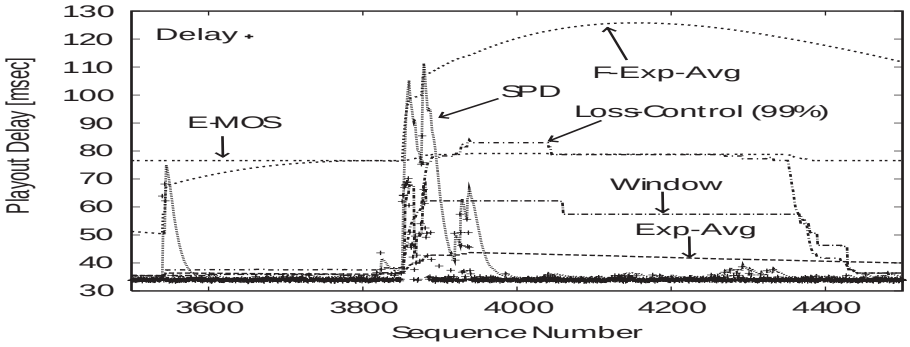
Table 8: Comparison of PLR and Mean Playout Delay and MOS

Case	Algorithm	Target	PLR [%]	Mean of d_i [msec]	MOS
"dynamic"	Loss-Control	95%	5.7	227.92	2.22
		99%	0.94	387.12	2.41
		99.9%	0.12	770.44	0.59
	E-MOS	-	2.95	294.75	**2.49**
	Exp-Avg	-	4.54	247.91	2.38
	F-Exp-Avg	-	0.1	970.34	0.10
	SPD	-	5.44	198.74	2.33
	Window	99%	1.34	362.57	2.47
"moderate"	Loss-Control	95%	6.02	40.61	2.99
		99%	1.77	58.45	3.83
		99.9%	0.60	375.28	3.61
	E-MOS	-	0.10	77.71	**4.17**
	Exp-Avg	-	4.93	39.79	3.21
	F-Exp-Avg	-	0.04	102.26	4.13
	SPD	-	3.08	39.74	3.57
	Window	99%	2.33	48.60	3.72
"quiet"	Loss-Control	95%	3.94	9.40	2.94
		99%	0.72	9.87	3.60
		99.9%	0.22	10.53	3.70
	E-MOS	-	0.00	51.92	**3.77**
	Exp-Avg	-	0.18	10.49	3.71
	F-Exp-Avg	-	0.01	29.53	3.76
	SPD	-	0.77	10.19	3.59
	Window	99%	1.05	9.76	3.53

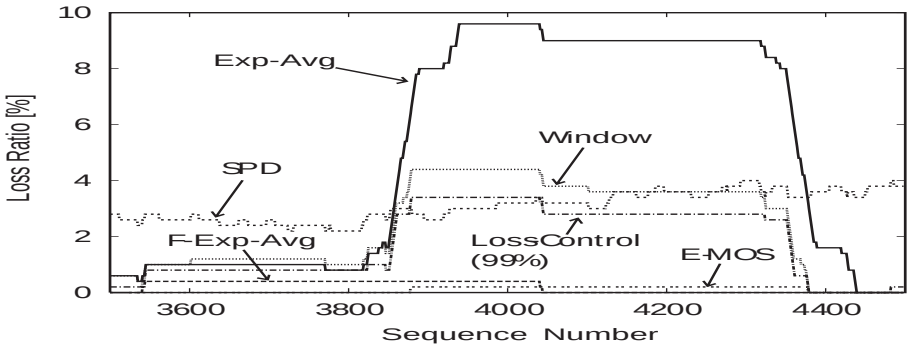
that the effect of introducing the playout delay is quite limited when the playout delay is small (less than 200 msec). In this region, it is effective to prevent the packet loss by lenghtening the playout delay. However, as the one-way delay becomes large, **E-MOS** tries to intentionally bear the increasing PLR for reducing the playout delay. It is a good slution for improving the users' perceived quality. Other PBAs have their own ground. For example, **Window** gives a good result in the "dynamic" case, but it is worse than **F-Exp-Avg** in other cases. However, **F-Exp-Avg** provides quite poor performance in the "dynamic" condition. **Exp-Avg** and **Loss-Control (99.9%)** give a passable performance in all cases. However, these methods cannot attain the improvement of the perceived quality as **E-MOS**. Furthermore, the **Loss-Control** method has a disadvantage that it tries to shorten the playout delay and forces to abandon a packets even if the playout delay is enough short (less than 200 msec). Thus, **Loss-Control** is not suitable in low packet transmission delay environments.

Figure 15 shows the time-dependent behavior of the playout delay, PLR, and MOS for PBAs. Here, the target value of **Loss-Control** is set to 99%. The playout delays of **E-MOS** are larger than the other algorithms except **F-Exp-Avg**, where the one-way delays are small. From this figure, we find that **E-MOS** indends to minimize the PLR when the one-way delay is less than 200 msec. On the other hand, in the condition that the one-way delay is over 200 msec, **E-MOS** tries to increase the PLR for reducing the playout delay in order to enhance the MOS. That is, **E-MOS** can archieve a good balance of the playout delay and PLR based on Eq. (28).

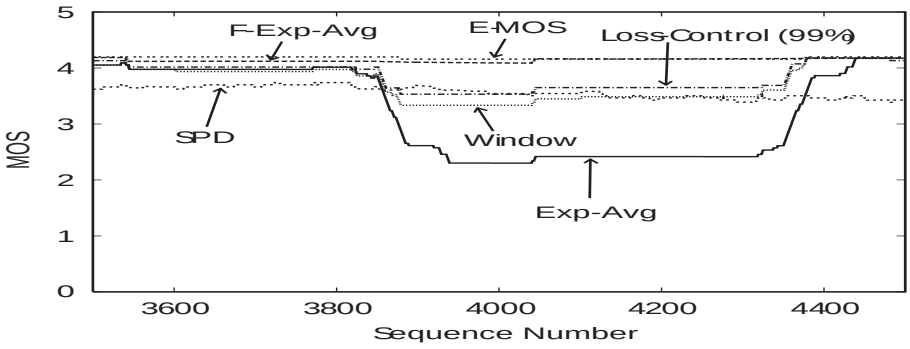
Because **E-MOS** has the largest computational burden among PBAs, it is important to estimate its computational overhead. We implemented **E-MOS** as an input plug-in of `Winamp`, and verified its effectiveness. The experimental setting was same as that of **Loss-Control** (see Subsection 6.4). We show our experimental environment in Figure 16. From the results of implementation tests, we find that **E-MOS** runs in "real-time" as the other algorithms. The computational overhead was about 0.02 msec for each packet arrival; 0.1% of packet transmission interval in G.711, which is sufficiently a small overhead. We also confirmed that the audio playing is not interrupted by any other factors except packet losses.



(a) Comparison of Playout Delay



(b) Comparison of PLR



(c) Comparison of MOS

Figure 15: Performance Evaluation of Each PBA ("moderate" case)

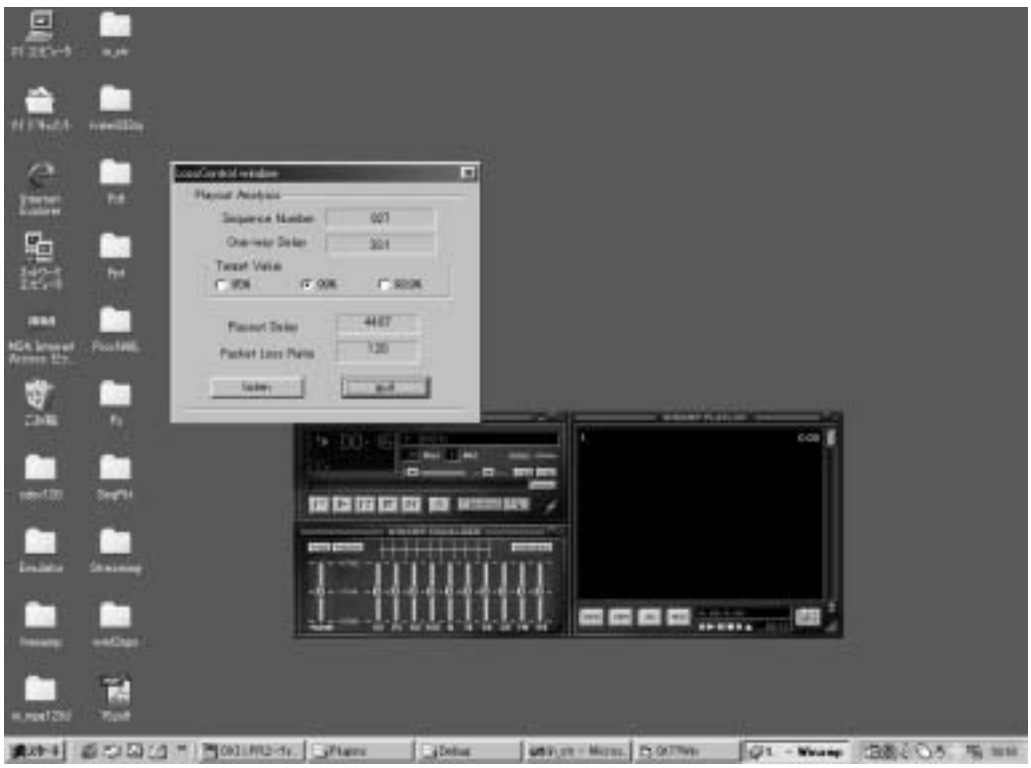


Figure 16: The Snapshot of Implementation Experiments[†]

[†]©Nullsoft Inc. 2002

8 Concluding Remarks

In this thesis, we have measured packet transmission delays and analyzed their characteristics by taking into account the network parameters. We have next described a method for modeling the tail distribution of the delays. From the results of statistical analysis, we have found that the Pareto distribution is most appropriate as the model of the one-way delay distribution as well as RTT distributions in any network conditions.

Then, we have proposed a new playout buffer algorithm based on our statistical analysis. Numerical examples have shown that our proposed method can control the playout buffer while satisfying the target packet loss probability, which is typically specified by the user's preference. Furthermore, we have implemented the proposed algorithm as an input plug-in of a generally used application. Implementation experiments have shown the effectiveness of our proposed algorithm.

Moreover, we have modified our proposed algorithm to be able to directly specify the perceived quality of streaming applications, which is represented by MOS. Simulation and implementation experiments have shown that the modified algorithm performs the highest quality among all of PBAs that we have tested.

For future research topics, it is necessary to improve the accuracy of our model for representing the delay distributions. To achieve it, it might be useful to test another heavy-tailed probability functions as the model of delay distributions. Moreover, though no serious problem occurs at the client of **E-MOS**, less CPU load would be more comfortable for users. The more effective calculation method for **E-MOS** is necessary.

Acknowledgements

I would like to express my sincere and deep gratitude to my supervisor, Professor Masayuki Murata of Osaka University, who has guided me through this work. His continuous advice and encouragements are acknowledged and greatly appreciated. Thanks for leading me into the area of computer networks including the subjects in this thesis.

Special thanks go to Research Associate Shingo Ata of Osaka City University, for his valuable comments and suggestions. Without his support, this thesis would not have been possible.

I would like to express my gratitude to Professor Hideo Miyahara, Assistant Professor Naoki Wakamiya, Research Associate Hiroyuki Ohsaki, Research Associate Go Hasegawa, and Research Associate Shin'ichi Arakawa of Osaka University, for their comments and suggestions on the thesis proposal.

Finally, I thank Mr. Kazumine Matoba and Mr. Ryo Kawabe of my research group, many friends, and colleagues in the Department of Informatics and Mathematical Science of Osaka University for their generous help, enlightening and valuable suggestions.

References

[1] H. Schulzrinne, S. Casner, R. Frederick, and V. Jacobson, "RTP: A transport protocol for real-time applications," *RFC 1889*, January 1996.

[2] S. Savage, A. Collins, and E. Hoffman, "The end-to-end effects of Internet path selection," in *Proceedings of ACM SIGCOMM '99*, pp. 289–299, September 1999.

[3] D. Cohen, "Issues in transnet packetized voice communications," in *Proceedings of Fifth Data Communications Symposium*, pp. 6–10–6–13, September 1977.

[4] J.-C. Bolot, "Characterizing end–to–end packet delay and loss in the Internet," in *Proceedings of ACM SIGCOMM '97*, pp. 289–298, September 1997.

[5] V. Paxson, "End–to–end Internet packet dynamics," in *Proceedings of ACM SIGCOMM '97*, pp. 139–152, September 1997.

[6] R. Ramjee, J. Kurose, D. Towsley, and H. Schulzrinne, "Adaptive playout mechanisms for packetized audio applications in wide–area networks," in *Proceedings of IEEE INFOCOM '94*, pp. 680–688, April 1994.

[7] B. J. Dempsey and Y. Zhang, "Destination buffering for low-bandwidth audio transmissions using redundancy-based error control,," in *Proceedings of LCN, 21st Annual Conference on Local Computer Networks*, pp. 345–354, October 1996.

[8] S. B. Moon, J. Kurose, and D. Towsley, "Packet audio playout delay adjustment: performance bounds and algorithms," *ACM/Springer Multimedia Systems*, vol. 5, pp. 17–28, January 1998.

[9] S. Mohamed, F. Cervantes-Pérez, and H. Afifi, "Integrating networks measurements and speech quality subjective scores for control purpose," in *Proceedings of IEEE INFOCOM 2001*, April 2001.

[10] J. Postel, "Transmission control protocol specification," *RFC 793*, September 1981.

[11] C. Savolaine, "QoS/VoIP overview," in *IEEE Communications Quality & Reliability (CQR 2001) International Workshop*, April 2001.

[12] "Caida measurement tool taxonomy." available at `http://www.caida.org/tools/taxonomy/index.xml`.

[13] B. A. Mah, "pchar: A tool for measuring Internet path characteristics." available at `http://www.ca.sandia.gov/~bmah/Software/pchar/`.

[14] A. B. Downey, "Using pathchar to estimate Internet link characteristics," in *Proceedings of ACM SIGCOMM '99*, pp. 241–250, August 1999.

[15] V. Paxson, "On calibrating measurements of packet transit times," in *Proceedings of ACM SIGMETRICS '98*, pp. 11–21, June 1998.

[16] S. B. Moon, P. Skelly, and D. Towsley, "Estimation and removal of clock skew from network delay measurement," in *Proceedings of IEEE INFOCOM '99*, pp. 227–234, March 1999.

[17] K. Thompson, G. J. Miller, and R. Wilder, "Wide–area Internet traffic patterns and characteristics," *IEEE Network*, pp. 10–23, November 1997.

[18] International Telecommunication Union, "Pulse code modulation (PCM) of voice frequencies," Recommendation G.711, Telecommunication Standardization Sector of ITU, Geneva, Switzerland, November 1988.

[19] International Telecommunication Union, "Coding of speech at 16 kbit/s using low-delay code excited prediction," Recommendation G.728, Telecommunication Standardization Sector of ITU, Geneva, Switzerland, September 1992.

[20] International Telecommunication Union, "Coding of speech at 8 kbit/s using conjugate-structure algebraic-code-excited linear-prediction," Recommendation G.729, Telecommunication Standardization Sector of ITU, Geneva, Switzerland, March 1996.

[21] International Telecommunication Union, "Dual rate speech coder for multimedia communications transmitting at 5.3 and 6.3 kbit/s," Recommendation G.723.1, Telecommunication Standardization Sector of ITU, Geneva, Switzerland, March 1996.

[22] V. Paxson and S. Floyd, "Wide-area traffic: The failure of Poisson modeling," in *Proceedings of ACM SIGCOMM '94*, pp. 257–268, August 1994.

[23] W. E. Leland, M. S. Taqqu, W. Willinger, and D. Wilson, "On the self-similar nature of Ethernet traffic," in *Proceedings of ACM SIGCOMM '93*, vol. 2, pp. 183–193, February 1993.

[24] M. E. Crovella and A. Bestavros, "Self-similarity in World Wide Web; traffic evidence and possible causes," in *Proceedings of ACM SIGMETRICS '96*, pp. 160–169, May 1996.

[25] V. Brazauskas and R. Serfling, "Robust and efficient estimation of the tail index of a one-parameter pareto distribution," *North American Actuarial Journal available at* `http://www.utdallas.edu/~serfling/`, April 2000.

[26] NULLSOFT, "WINAMP.COM | now featuring self-transforming mechanical elves." available at `http://www.winamp.com`.

[27] M. Unser, B.L.Trus, J.Frank, and A.C.Steven, "The spectral signal-to-noise ratio: computational efficiency and statistical precision," *Ultramicroscopy*, vol. 30, pp. 429–434, July 1989.

[28] International Telecommunication Union, "Objective quality measurement of telephone-band (300–3400Hz) speech codecs," Recommendation P.861, Telecommunication Standardization Sector of ITU, Geneva, Switzerland, February 1998.

[29] S. Voran, "Objective estimation of perceptual speech quality, part I: development of the measuring normalizing block technique," *IEEE Transactions on Speech and Audio Processing*, vol. 7, pp. 371–382, July 1999.

[30] W. H. Press, B. P. Flannery, S. A. Teukolsky, and W. T. Vetterling, *Numerical Recipes in C; The Art of Scientific Computing*, ch. 9.2, pp. 263–266. Cambridge University Press, 1988.