A Statistical Approach to Adaptive Playout Scheduling in Voice Over Internet Protocol Communication

Priya Chandran[1], Chelpa Lingam[2]
[1]BVIMIT, University of Mumbai, R&D Centre Bharatiar University Coimbatore, India
[2]Department of Computer Engineering, Pillai HOC College of Engineering and Technology, University of Mumbai, India

Article Info

Article history:

Received Nov 30, 2017
Revised Jan 13, 2018
Accepted Sep 5, 2018

Keyword:

Jitter buffer management
Jitter
Playout scheduling
PLC
VoIP

ABSTRACT

Factors like network delay, latency and bandwidth significantly affect the quality of communication using Voice over Internet Protocol. The use of jitter buffer at the receiving end compensates the effect of varying network delay up to some extent. But the extra buffer delay given for each packet plays a major role in playing late packets and thereby improving voice quality. As the buffer delay increases packet loss rate decreases, which in general is a very good sign. However, an increase of buffer delay beyond a certain limit affects the interactive quality of voice communication. In this paper, we propose a statistical framework for adaptive playout scheduling of voice packets based on network statistics, packet loss rate and availability of packets in the buffer. Experimental results show that the proposed model allocates optimal buffer delay with the lowest packet loss rate when compared with other algorithms.

Corresponding Author:

Priya Chandran,
BVIMIT ,University of Mumbai,
R&D Centre Bharatiar University Coimbatore, India.
Email: priyaci2005@gmail.com

1. INTRODUCTION

Voice over Internet Protocol (VoIP) is a technology used for the voice transmission over internet protocol. The wide-spread usage of internet and cheap communication cost made VoIP a very popular technology. In VoIP, voice conversations are digitized and then packetized for transmission. These packets are transmitted every 20 to 40 ms [1] and expect that these packets reach the receiver side in the same interval. If the packets are arrived at the receiver side without any delay variations, they can be played directly. But due to network impairments like delay and latency, the packets are lost or may not arrive in the same order in which the sender sent. This causes the end users to experience a broken, buzzing and delayed speech. Extensive research is going on this area to increase the quality of communication [2-4]. Many researchers proposed receiver side Packet Loss Concealment (PLC) methods [5].

The difference in packets inter-arrival time is known as jitter and to mitigate the effect of jitter, a jitter buffer is used at the receiver side. This jitter buffer holds received packets till its estimated playout time, to accommodate late packets. Thus the playout time of received packets are delayed to reduce packet loss rate. Increasing buffer delay extends playout time of the packets received and hence communication interactivity is reduced. At the same time, this reduces late packet loss rate. But decrease in buffer delay increases late packet loss rate. Many playout scheduling algorithms have been proposed to solve this trade-off. Playout scheduling schemes are classified as static (fixed) and adaptive. Static playout scheduling is the simplest method, where the playout time of all the packets in a session is fixed irrespective of the varying network delay [6]. Since the network conditions are volatile, this method is not very effective in reducing packet loss rate. Keeping a fixed playout scheduling time for all the packets in face of varying network conditions is not a good option.

In adaptive playout scheduling algorithms, the playout time of the packets is adaptively adjusted according to the network delay. i.e., Packets of the same talk spurt experience different playout time and the sequential playout of packets is implemented by PLC mechanisms like extending or compressing the silence periods. PLC approaches like signal reconstruction and timescale modification methods are also widely used for the sequential playout of packets at receiver side. The efficiency of adaptive playout scheduling mechanisms depends on the accurate estimation of the buffer delay needed by the arrived voice packets so that late packets can also be played. In this paper, we propose a framework for adaptive playout scheduling of voice packets by estimating buffer delay of the packets based on the network statistics, packet loss rate and availability of packets in buffer.

Several methods have been developed for estimating the adaptive playout time of packets. Ramjee et al. [7] proposed four algorithms for adaptive playout delay estimation of voice packets. In this method, the playout time of first packet of the talkspurt is estimated using an autoregressive estimate and an offset is added to calculate the playout time of successive packets in the same talkspurt. But the delay adaptation is applied only to the first packet of the talk spurt. Also the buffer size is exponentially increased to store late packets. Pinto et al. [8] proposed an adaptive gapbased algorithm that can be tuned for both end-to-end delay and packet loss to satisfy a user-desired tolerance.

The method proposed by Liu et al. [9] adaptively adjusts jitter buffer according to an indication of a sequence number of the delaying packet. Approaches based on order statistics based estimation [10], network variations [11-12] and quality-driven playout buffer optimizations [13] are also proposed. B. H. Kim et al. [12] used a static buffer size of 200ms to hold packets. Event counting algorithm proposed in [14] relies on the measure of relative times in handling the object-related events. Normal approximation is used to identify the delay of network in [15]. Modified enhanced normalized least mean squares algorithm (ENLMS) is also used to identify the spike state of the network [16].

Different methods have been proposed by researchers to solve the issues with adaptive playout scheduling in VoIP communication. S.B.Moon et al., [17] computed upper and lower bounds on the minimum average playout delay for a given packet loss by mainitaining delay percentile information. M. K. Ranganathan and L. Kilmartin S [18] proposed a novel fuzzy trend analyzer system to estimate intra-talk spurt playout delay adaptation. Sreenan and Cormac [19] suggested histogram based approaches for synchronizing networked multimedia streams.

2. PROPOSED METHOD

In our proposed method, we perform the following operations to implement adaptive playout scheduling.

a. Computation of network statistics
b. Estimation of playout time based on optimal buffer delay

In the following section, a brief explanation of adaptive playout scheduling is given.

2.1. Adaptive Playout Scheduling

In adaptive playout scheduling algorithms, the playout time of the packets are adaptively adjusted during silence periods and within talkspurts. This is mainly achieved by scaling of voice packets within talkspurts and keeping the synchronous playout of packets. Each packet in the talkspurt experiences different playout time. Buffer delay estimation is a major component in calculating the playout time of packets. The tradeoff between buffer delay and late packet loss rate can be solved using an efficient buffer delay estimation method. We assume that both sender and receiver maintain clock synchronization. As packetization delay and codec delay are codec dependent, they both are not taken into account while computing the playout time of the packets. Playout time of packet i, tp_i is the time packet is played at the receiver and network delay, $dn_{i,}$ is the time gap between sending and receiving time of a packet.

$$tp_i = tr_i + db_i \tag{1}$$

$$dn_i = tr_i - ts_i \tag{2}$$

where, tr_i, db_i and ts_i are receiving time, buffer delay and sending time respectively. Buffering time is estimated for each packet in a talk spurt to calculate the playout time. When the buffering time increases, playout time of the packet also increases. Even though the packet loss rate decreases due to the increase in buffer delay, it reduces the voice quality and listeners experience a broken, buzzing and delayed speech. Total delay experienced by a packet, dt_i , is the difference between the time it is sent from sender and the time it is played on the receiver side. It is the sum of network delay and buffering delay and is given in Equation (3).

$$dt_i = dn_i + db_i \tag{3}$$

The jitter of ith packet, j_i, for a talk spurt is calculated as,

$$j_i = dn_i - dn_{i-1} = (tr_i - ts_i) - (tr_{i-1} - ts_{i-1}) \tag{4}$$

2.2. Computation of Network Statistics and Optimal Buffer Delay

The main idea behind our new approach is to compute the playout time of a packet using buffering delay based on the network statistics of already arrived packets. We use a vector to store the details of already arrived packets. In order to track the latest network conditions, old packet information are removed periodically from vector and updated with latest packet information. It follows a window based approach so that the estimations always adapt to the varying network conditions. In addition to the past network statistics, packets waiting in buffer and late packet loss rate are also considered as factors for buffer delay computation.

Normal and spike modes of network states for each packet are identified based on the network delay, dn_i, of the packets and the threshold value, th_{spike} [12]. If a packet is received with a delay greater than th_{spike}, current mode is identified as spike. Otherwise, the mode is taken as a normal. The network delay is estimated using equation (5):

$$dn_i = \propto_n * dn_{i-1} + (1 - \propto_n) * dn_i \tag{5}$$

where $\propto_n$ is a weighing factor which depends on delay statistics [7]. Using the above estimated network delay, playout time of packet i is computed as given in equation (6).

$$tp_i = \mathrm{ts_i} + dn_i + \mathrm{db}_i \tag{6}$$

The accurate estimation of buffer delay for playout time calculation helps in allocating late packets in the buffer for playing, and thereby reducing late packet loss rate. For this, we have made the value of db_i proportional to the fluctuating network delay, late loss rate and availability of packets in buffer.
Buffer delay is estimated using equation (7).

$$\mathrm{db}_i = \omega * \mathrm{J_i} * \mathrm{L_{late}} \tag{7}$$

ω is the delay factor which depends on availability of preceding and succeeding packets. The network jitter J of i th packet is estimated as,

$$\mathrm{J_i} = \mathrm{m_i} + \varphi_{\mathrm{i}} * \mathrm{var_i} \tag{8}$$

where φ is the weighting factor of the inter-arrival jitter variance, m and var are the average and variance of inter-arrival jitter, respectively. For every packet that arrive at the receiver, the algorithm checks the current and previous mode, and the value of m and var is calculated differently in each mode using separate equations as given in [12]. This way, quickly varying network conditions can be adapted more accurately for the estimation of playout time. The value of φ is proportional to network characteristics and is calculated as:

$$\varphi_{\mathrm{i}} = \frac{\mathrm{m_{i-1}} - \sigma_{\mathrm{i-1}}}{\mathrm{var_{i-1}}} \tag{9}$$

where σ is the standard deviation. The value of m, var and σ are calculated for all the packets stored in window in order to predict the current network state. The size of window is specified in section 3. In addition to the packet loss rate, our method also considers order of already received packets while predicting buffer delay of ith packet. The calculated jitter can be negative if the packets are received out of order. We use a delay factor ω to adjust buffer delay of expected packet and its value varies based on the above specified scenarios. The value of ω is low, $lower$, $high$ or $higher$ according to the availability of preceding and succeeding packets. For example, if we check the availability of only succeeding and preceding packets received, we assign a value to ω based on the following four different cases.
Both p_{i-1} and p_{i+1} are received, $0 < lower < 1$

p_{i-1} is received and p_{i+1} not received, $0 < low < 1$
p_{i-1} not received, but p_{i+1} is received, $1 < high < 2$
p_{i-1} and, but p_{i+1} not received, $1 < higher < 2$
This way the value assigned to ω is considered as one parameter for controlling the buffer delay of packets. For example, if both the succeeding and preceding packets are already arrived, then a small value is assigned to ω. The proposed algorithm for estimating playout time of packets is given as:

Algorithm: Estimation of Playout time

```
1. Receive packets
2. for (each packet in the talkspurt)
3.         Compute network delay
4.         Estimate the jitter and delay factor
5.         Estimate buffer delay
6.         Calculate playout time
7. end for
```

Packets are discarded based on playout time of late packets and is given as:

store p_k in buffer	If ($tr_k < tp_k$)
discard p_k	Otherwise

3. RESULTS AND ANALYSIS

We have evaluated our proposed algorithm for playout time estimation using an experimental setup [20] as shown in Figure 1. The four modules used for this experimental setup are: Session Initiation Protocol (SIP) application, network traffic emulator, VoIP server and a packet analyzer. We have tested VoIP communication between two endpoints, endpoint1 and endpoint2. A VoIP server installed on a computer is used to test the call flows between two end-points. One endpoint is a mobile device which is registered with the server using a VoIP application. PJSIP application running on a computer is used as another endpoint. Program for Network statistics based playout time estimation is written in C language. Network emulator is used to set different network parameters like jitter, delay, packets reordering and packets loss in RTP packets flow. The two endpoints act as SIP clients and call flows between these two SIP clients were analyzed using packet analyzer.

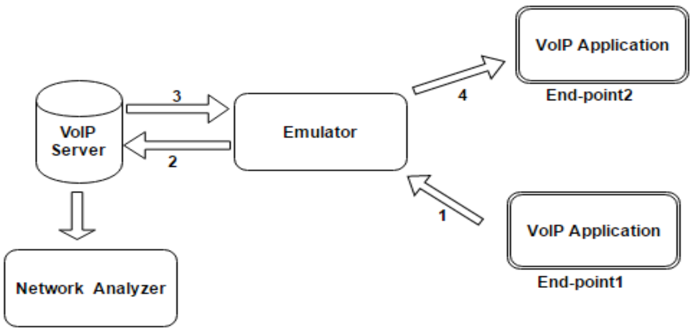


Figure 1. Experimental setup

The Real Time Protocol (RTP) packets sent from endpoint1 is redirected through VoIP server (SIP proxy server) and then sent to endpoint2. Recorded speech sample database, provided by the International Telecommunication Union [21] are taken for our experimental study. The performance of our method is evaluated using the metric, average buffering delay (db_{avg}) and late packet loss rate (L_{late}).

$$db_{avg} = \frac{1}{P}\sum_{i=1}^{n} db_i \tag{17}$$

where P is the set of packets played.

$$L_{late} = \frac{|R-P|}{R} \tag{18}$$

where R the set of packets received.

Also to evaluate communication quality, objective voice quality test is carried out using E-model [22]. The quality of transmission rating in E-Model varies between 0 and 100 and is specified by Transmission Rating factor (R factor). 0 represents bad quality and 100 represents extremely good quality of communication. Then, R factor is mapped to MOS (Mean Opinion Score) scores for finding conversational quality.

We have used network emulator to create traces with varying jitter. One way-delay of the communication is measured for each packet having 160 bytes of payload size and the G.711 coded voice packets are generated at the sender in 30 ms time interval. Performance of our proposed method is compared with two different adaptive playout scheduling schemes. Method 1 uses an autoregressive estimate to predict the network delay and jitter and is given as algorithm 4 in [7], method2 uses self adaptive jitter buffer adjustment mechanism discussed in [9] and method 3 is based on linear prediction with a static buffer size [12]. We denote auto_reg, self_adapt, linear_pred and PM for method1, method2, method3 and proposed method respectively.

Playout delay estimation algorithm is executed for every packet in the talkspurt. Since the packetization interval of audio packets is taken as 30ms [1], the algorithm should be efficient to execute maximum 34 times per second. Also the calculation complexity increases as window size increases while calculating network statistics. At the same time, as the window size decreases, the adaptation is more responsive to the volatile network behavior [7] [14]. So it is important to select the window size which adapts and estimates accurate network characteristics. After experimenting with different window sizes, we kept a window of size 100 packets which resulted in optimal performance. For delay factor calculation, we have checked availability of three succeeding and two preceding packets. Six speech samples were collected for analysis, and Table 1 depicts network trace statistics along with the experimental results. Among the six traces collected, trace1 has small jitters; traces 2, 3 and 4 are having medium jitters, while trace 4 and 6 is having large jitters. Standard Deviation (STD) of network delay of the voice data varies for traces. The maximum jitter experienced in trace1, trace2, trace3 trace4, trace5 and trace6 are 51, 125, 160, 180, 240 and 325 respectively. For the small jitter case (trace1), average buffer delay is nearer to the ideal case (30ms) and as the jitter increases, buffer delay also increases. Performances of six algorithms for six different traces are depicted in Figure 2. For the trace with larger jitter value, 325ms, proposed method gives a high MOS score value compared with other three algorithms.

Table 1. Experimental Results

Trace	STD of network delay (ms)	Maximum Jitter (ms)	Methods used	db_{avg}	L_{late} (%)	MOS
1	9.45	51	auto_reg	42.41	5.46	3.41
			self_adapt	38.53	2.67	3.51
			linear_pred	38.73	2.13	3.78
			PM	35.15	1.32	4.01
2	23.26	125	auto_reg	55.13	7.23	2.91
			self_adapt	49.38	4.25	3.43
			linear_pred	48.21	3.97	3.52
			PM	44.37	2.53	3.72
3	24.11	160	auto_reg	56.28	8.62	2.99
			self_adapt	48.46	4.67	3.76
			linear_pred	49.84	5.14	3.47
			PM	44.72	3.63	3.61
4	25.67	180	auto_reg	56.48	9.01	2.99
			self_adapt	50.76	8.58	3.76
			linear_pred	50.24	8.21	3.47
			PM	44.97	7.03	3.61
5	27.13	240	auto_reg	61.24	9.27	2.42
			self_adapt	55.72	8.24	2.63
			linear_pred	54.57	10.67	2.75
			PM	51.36	7.72	3.02
6	29.81	325	auto_reg	79.63	12.82	2.17
			self_adapt	72.52	10.15	2.64
			linear_pred	72.41	11.21	2.62
			PM	64.56	8.33	2.95

The performance of four adaptive playout scheduling algorithms for all the traces is depicted in Figure 2. In all the cases proposed method results in lowest late loss rate and buffering delay. As the jitter increases, average buffering delay is increased to reduce late packet loss rate.

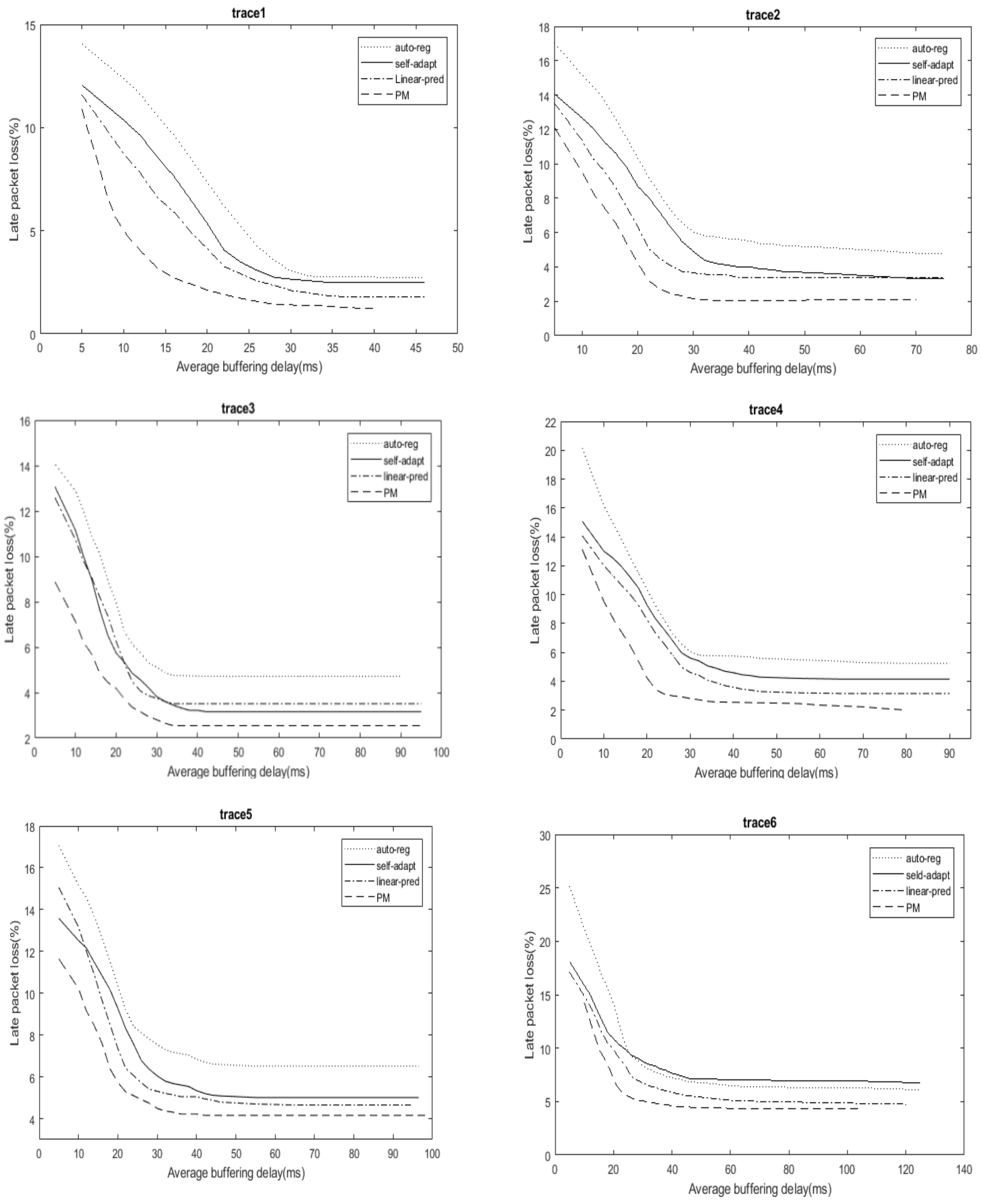
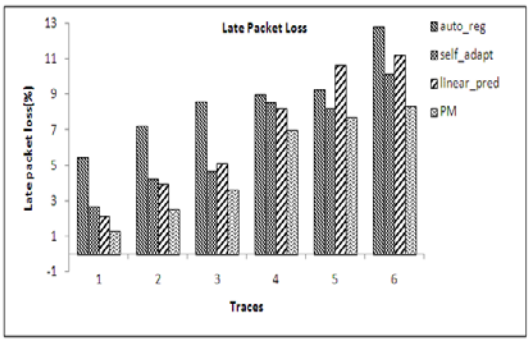
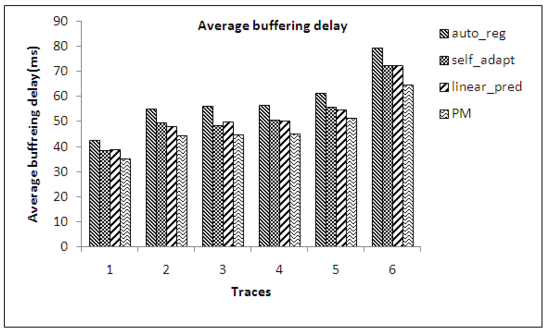


Figure 2. Performance of four adaptive playout scheduling algorithms for six different traces

The comparison of late packet loss and average buffering delay estimation of algorithms for traces is depicted in Figure 3(a) and Figure 3(b) respectively. In all the cases proposed method results in lowest late loss, buffer loss and buffering delay.





(a) (b)

Figure 3. (a) Late packet loss rate (b) Average buffering delay of six traces

Figure 4 shows MOS score of all the traces with small, medium and large jitter cases. MOS score obtained from the experiments indicates that for all the traces speech quality is significantly improved by proposed algorithm. Experimental results show that proposed method estimates playout time of packets to match the jitter variations while keeping buffering delay minimum for maintaining interactive voice quality.

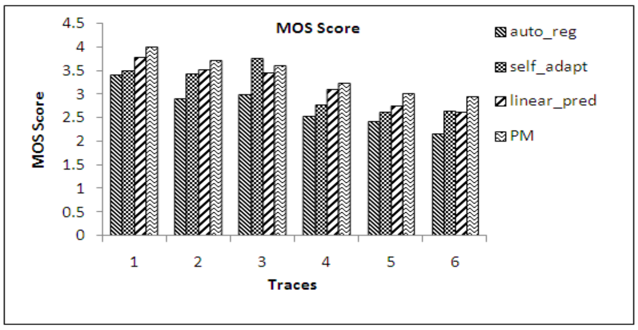


Figure 4. MOS score of different algorithms

4. CONCLUSION

Widespread use of internet replaces traditional telecommunication system by the VoIP services. Rendering high quality service to the users is a crucial challenge faced. Network delay is one major impairment factor which affects the quality of communication. Network delay causes the late arrival or loss of packets and there by exacerbating the voice quality. In this paper, we propose a model based on network statistics, packet loss rate and packet availability in the buffer to estimate playout time of each packet. We have compared our algorithm with existing three adaptive playout scheduling algorithms and the results indicate that our method solves the trade-off between buffer delay and packet loss in a better way than the existing adaptive playout scheduling methods with minimum packet loss rate and buffering delay. Also the communication quality is evaluated using an objective voice quality test, E-model and our algorithm achieves a higher MOS score, for all the traces, than the other three algorithms.

REFERENCES

[1] ITU-T Recommendation G.114. *One way transmission time*. 2000.

[2] Kolhar, Manjur, Mosleh M. Abualhaj, and Faiza Rizwan. *QoS Design Consideration for Enterprise and Provider's Network at Ingress and Egress Router for VoIP protocols. International Journal of Electrical and Computer Engineering (IJECE)*. 2016: 6.1: 235.

[3] Audah, L., Sun, Z. Cruickshank, H. *QoS based Admission Control using Multipath Scheduler for IP over Satellite Networks. International Journal of Electrical and Computer Engineering (IJECE)*. 2017; 7.6: 2958-2969.

[4] Wheeb, Ali Hussein. *Performance Evaluation of UDP, DCCP, SCTP and TFRC for Different Traffic Flow in Wired Networks. International Journal of Electrical and Computer Engineering (IJECE)*. 2017; 7.6: 3552-3557.

[5] Maheswari, K., and Punithavalli, M., *Receiver based packet loss replacement technique for high quality VoIP streams,* In Nature & Biologically Inspired Computing, NaBIC 2009. World Congress on, IEEE, pp. 1669-1672.
[6] Alvarez-Cuevas, Felipe, et al. *Voice synchronization in packet switching networks. IEEE Network*. 1993; 7.5: 20-25.
[7] R. Ramjee, J. Kurose, D. Towsley and H. Schulzrinne.*Adaptive playout mechanisms for packetized audio applications in wide-area networks*. Proc. IEEE INFOCOM, Toronto, Canada. 1994.
[8] Pinto, Jesus, and Kenneth J. Christensen. *An algorithm for playout of packet voice based on adaptive adjustment of talkspurt silence periods. Local Computer Networks*, LCN'99. Conference on. IEEE. 1999.
[9] E Liu, G Shen, S Jin, L Gui. *Self-adaptive jitter buffer adjustment method for packet-switched network*. U.S.10 July 2012; Patent No. 8, 218, 579.
[10] Y.J.Liang, N. Farber, and B. Girod. *Adaptive playout scheduling and loss concealment for voice communication over IP networks. IEEE Transactions on Multimedia*. 2001; Vol. 5(4): pp. 532-543.
[11] L Repele, R Muradore, D Quaglia. *Improving performance of networked control systems by using adaptive buffering. IEEE Transactions on Industrial Electronics*. 2014; 61.9: 4847-4856.
[12] Byeong Hoon Kim, Hyoung-Gook Kim, Jichai Jeong and Jin Young Kim. *VoIP receiver-based adaptive playout scheduling and packet loss concealment technique. IEEE Transactions on consumer Electronics*. 2013; 59.1: 250-258.
[13] Sun, Lingfen, and Emmanuel C. Ifeachor. *Voice quality prediction models and their application in VoIP networks. IEEE transactions on multimedia*. 2006; 8.4: 809-820.
[14] Y Xie, C Liu, MJ Lee, TN Saadawi. *Adaptive multimedia synchronization in a teleconference system. Multimedia Systems*. 1999; 7.4: 326-337.
[15] Gibbon, John F., and Thomas D. C. Little. *The use of network delay estimation for multimedia data retrieval. IEEE Journal on Selected Areas in Communications*. 1996; 14.7: 1376-1387.
[16] Shallwani, Aziz, and Peter Kabal. *An adaptive playout algorithm with delay spike detection for real-time VoIP. Electrical and Computer Engineering*. IEEE CCECE 2003. Canadian Conference on. 2003; Vol. 2.
[17] Moon Sue B., Jim Kurose, and Don Towsley. *Packet audio playout delay adjustment: performance bounds and algorithms. Multimedia systems*. 1998; 6(1):Pp. 17-28.
[18] M.K.Ranganathan and L.Kilmartin. *Neural Fuzzy Computation Techniques for Playout Delay Adaptation in VoIP Networks. IEEE Transactions on Neural Networks*. 2005; 16(5).
[19] CJ Sreenan, JC Chen, P Agrawal. *Delay reduction techniques for playout buffering. IEEE Trans. Multimedia*. 2000; vol. 2: pp. 88–100.
[20] Chandran P, Lingam C. *Performance evaluation of voice transmission in Wi-Fi networks using R-factor*. In 2015 International Conference on Information Processing (ICIP), IEEE. 2015; pp. 481-484.
[21] ITU-T, Coded-Speech Database, Supplement 23 to ITU-T P-Series Recommendations, International Telecommunication Union, 1998.
[22] ITU-T Rec G.107. *The E-model: a computational model for use in transmission planning*. 2014.