An Efficient Buffer Delay Correction Algorithm to VoIP

Fabio Sakuray
Computer Science Department
University of Londrina
Londrina, Brazil
Email: sakuray@uel.br

Robinson S. V. Hoto
Mathematics Department
University of Londrina
Londrina, Brazil
Email: hoto@uel.br

Gean D. Breda
and Leonardo S. Mendes
Faculty of Electrical and Computing Engineering
University of Campinas
Campinas, Brazil
Email: gdbreda@gmail.com lmendes61@gmail.com

***Abstract*—Audio applications are widely used on the Internet today. In these applications, packets are considered lost if received after their playout time. Such applications require a playout buffer in the receiver for smoothing network delay variations to enable the reconstruction of a periodic form of the transmitted packets. The objective of buffer delay adjustment algorithms (BDA) is to control the packet loss rate using minimum buffer size to jitter smooth. However, current algorithms fail to obtain a particular packet loss percentage. This paper presents a definition of Optimum Buffer Delay (OBD), used to remove jitter and a technique to correct the buffer delay from any BDA applied between talkspurts, with the purpose of bring the packet loss percentage closer to the value defined by audio applications. This new technique is called Buffer Delay Correction Algorithm (BDCA).**

Keywords–Playout Delay; VoIP; Buffer Delay; MOS.

I. INTRODUCTION

Nowadays, the Internet has been broadly used for voice applications, this can be explained by increase in Voice over Internet Protocol (VoIP) applications efficiency and best network bandwidth to users. Unlike of other applications, VoIP can tolerate some packet loss, but none jitter is allowed [1]. In receiver side of VoIP systems, the audio samples must be played as a continuous stream. This is a challenging process because IP present delay variation (or jitter), this phenomenon results in increase on packet loss rate whenever a packet is received after your playout time [2]. The receiver audio application uses a de-jitter buffer that insert an artificial delay (called Buffer Delay) to reduce this effects, resulting in a controlled packet loss rate that enable a greater communication quality. But long buffer delays can reduce the voice quality in interactive audio applications.

To adapt to network delay variations, the buffer delay needs to be continuously changed in order to reduce packet loss rate. The buffer delay control has been studied in many previous works and several Buffer Delay Algorithms (BDA) have been proposed.

However, these BDAs do not produce the packet loss rate as user requested. This paper presents the formal definition of Optimum Buffer Delay (OBD) to jitter remove, and explain how to use this result in Buffer Delay Correction Algorithm (BDCA), a technique to adjust the buffer delay produced by others BDAs. The BDCA has its focus on shaping the packet loss percentage to follow the one defined by voice service while bringing down one-way delay. This work considers only packet loss caused by jitter.

The remainder of this paper is as follows: Section II presents a review of BDAs, Section III details the features of de-jitter buffer, Section IV presents a definition of optimum buffer delay and the BDC) and Section V demonstrates performance comparisons between BDAs. Concluding remarks and future directions are presented in Section VI.

II. BACKGROUND

Figure 1 shows packets sent between two remote VoIP applications in a regular call, where talkspurts are periods with packets transmission and silence are periods without transmission. In a talkspurt k with n^k packets, a packet i is sent at instant t_i^k, received at instant a_i^k and executed in p_i^k (playout time) [3].

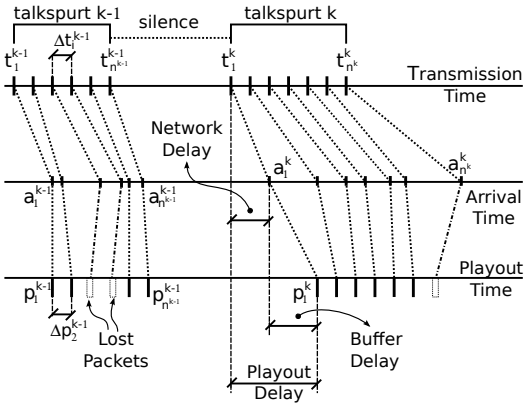


Figure 1. Timings of Packet Audio Transmission.

In the receiver side of VoIP applications, the audio packets must be scheduled to playout with the same temporal spacing used in transmission ($\Delta t_i^k = \Delta p_i^k$). However, jitter makes packets arrive after its playout time and are considered lost because they can not be used when $p_i^k < a_i^k$. To avoid this, most applications use a buffer delay that can be inserted at beginning of each talkspurt (see talkspurt k in Figure 1), which is referred to as "inter-talkspurt" technique, or inserted inside a talkspurt, which is referred to as "intra-talkspurt". This work analyses only algorithms that act in silence periods, since they represent the most of BDA solutions in literature [4].

Lobina in [5], present an important classification of BDAs, as:

1) Packet Loss Intolerant: Algorithms that use high buffer delay values, avoiding packet loss. The sim-

plicity of implementation is the main advantage of these algorithms;
2) Packet Loss Tolerant: audio applications can lose a certain number of packets without affecting audio quality. This class of algorithms adjusts buffer delay to control the packet loss rate;
3) Quality Based: this algorithm class monitors the call quality parameters to adjust the buffer delay.

Another element of voice call is the phenomenon called spike [6], defined as a sudden and large increase in the end-to-end delay. As result the receiver have an interval without packets followed by a series of packets arriving almost simultaneously. Delay spikes represent a serious problem for VoIP applications, since they lead BDAs to overrated buffer delay values. A BDA must react adequately to the spike by changing your behavior.

Several BDAs has been developed with most of them trying to foresee network delay to set the buffer delay. Now let us consider some examples. The next two algorithms are packet loss intolerant. Ramjee in [7] presents four algorithms to measure the delay variance and estimate the average end-to-end delay, the fourth can detect spike and change the algorithm behavior. Barreto and Arago in [8] present an algorithm based on the standard (Box-Jenkins) linear auto-regressive (AR) model. The playout delay estimated ($\hat{d}^k$) of talkspurts k can be write by:

$$\hat{d}^k = \theta_1^\mu \mu(A^{k-1}) + \theta_1^\sigma \sigma(A^{k-1}) + \theta_2^\mu \mu(A^{k-2}) + ... + \theta_n^\sigma \sigma(A^{k-n}) \quad (1)$$

where A^k is network delay of k-th talkspurt; θ_i^μ and θ_i^σ are weights associated with mean ($\mu(A^k)$) and standard deviation ($\sigma(A^k)$), n is the sliding window size with last talkspurts received.

In a call with M talkspurts, (1) can rewrite by $\mathbf{d} = \mathbf{X}\theta$, where matrix $\mathbf{X} \in \mathbf{R}^{M \times 2n}$ is defined as:

$$\mathbf{X} = \begin{bmatrix} \mu(A^n) & \sigma(A^n) & .. & \mu(A^1) & \sigma(A^1) \\ \mu(A^{n-1}) & \sigma(A^{n-1}) & .. & \mu(A^2) & \sigma(A^2) \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ \mu(A^{M-2}) & \sigma(A^{M-2}) & .. & \mu(A^{M-n-1}) & \sigma(A^{M-n-1}) \\ \mu(A^{M-1}) & \sigma(A^{M-1}) & .. & \mu(A^{M-n}) & \sigma(A^{M-n}) \end{bmatrix}$$

The vectors $\theta \in \mathbb{R}^{2n}$ and $\mathbf{d} \in \mathbb{R}^M$ are: $\theta = [\theta_1^\mu \ \theta_1^\sigma ... \theta_n^\mu \ \theta_n^\sigma]^T$, $\mathbf{d} = [d_{n+1} \ d_{n+2} ... d_{M-1} \ d_M]^T$ with the superscript T denoting matrix transposition.

The estimate of θ is given by $\hat{\theta} = \left[\mathbf{X}^T\mathbf{X}\right]^{-1}\mathbf{X}^T\mathbf{d}$. However, the matrix $\left[\mathbf{X}^T\mathbf{X}\right]$ may be non-invertible, in which case Barreto and Aragao replace it by its regularized version:

$$\hat{\theta} = \left[\mathbf{X}^T\mathbf{X} + \lambda\mathbf{I}\right]^{-1}\mathbf{X}^T\mathbf{d} \quad (2)$$

where $\mathbf{I} \in \mathbb{R}^{2n \times 2n}$ is the identity matrix and $0 < \lambda \ll 1$. The values used by the authors are $\lambda = 0.01$ or $\lambda = 0.001$.

The next three algorithms are packet loss tolerant. Moon *et al.* [3] use the network delay distribution in the last w received packets and a desired packet loss rate. This algorithm can detect spike. Fujimoto *et al.* [9] uses the same idea, but focused on the tail of the network delay probability distribution function. Assuming Pareto distribution for the tail, this approach presents better results when compared with algorithms that use a complete network delay distribution. In [10], Ramos *et al.* present the Move Average Algorithm (MA) to adjusts the playout delay at each new talkspurt given a desired target of average loss percentage (ρ). The authors compute the optimal playout delay (D_k) at the beginning of talkspurt k as:

$$D_k = SORT\left\{Z_i^k\right\} \quad \text{with } i = round(1-\rho)N_k$$

with N^k the number of audio packets received during k-th talkspurt and Z_i^k the variable portion of the end-to-end delay of i-th packet.

The predicted value of D_{k+1}, denoted by $\hat{D}_{k+1}$, is given by

$$\hat{D}_{k+1} = \sum_{l=1}^{M} a_l D_{k-l+1}$$

The coefficients a_l must minimize the mean square error between D_{k+1} and $\hat{D}_{k+1}$. They can by found from solving the equation:

$$\sum_{m=0}^{M} a_{m+1} r_D(m-l) = r_D(l+1) \text{ with } l = 0, 1, ..., M-1.$$

Suppose that it is known the last K values of r_D,

$$r_D(r) \simeq \frac{1}{K-|r|} \sum_{k=1}^{K-|r|} D_k D_{k+|r|}$$

with $r = 0, \pm1, \pm2, ..., \pm(K-1)$. The model's order M is computed as follow: starting with $M = 1$, compute all values of $\hat{D}_k$ and estimate $\mathbb{E}\left[(D_k - \hat{D}_k)^2\right]$, increase M and repeat the process. The model's order is taken equal to the lowest value of M preceding an increase in mean square error.

The next algorithms are quality based. Fujimoto *et al.* [11] shows that jitter, packet loss rate, codec and other parameters can affect call quality. Most solutions only allow packet loss rate setup. The algorithm presented in [11], called E-MOS, utilizes Mean Opinion Score (MOS [1], [12]) classification as input to buffer delay adjust. MOS values are 1 to 5, where 1 is the worst and 5 the best.

Valle *et al.* in [13] present the Dynamic Management of Dejitter Buffer Algorithm (DMDB), that uses MOS rating as input to control the followings algorithms:

1) OpenH323: an open source and packet loss intolerant algorithm, used in CallGen323 application;
2) Window: histogram based algorithm with spike detection, presented in [3];
3) Adaptive: algorithm proposed by [14], which is also reactive and quality based, that tries to maximize the end-user perceiving quality.

III. THEORETICAL ASPECTS OF BUFFER DELAY

In the next Sections, consider n^k the set of packets belonging to k-th talkspurt and p_i^k, a_i^k and t_i^k, respectively, the playout, receiver and transmission time. Using de-jitter buffer (or buffer delay - BD) in receiver side, with dynamic adjustment to each talkspurt, the playout time of i-th packet is:

$$p_i^k = a_1^k + BD^k + (i-1)\Delta t_i^k \quad (3)$$

where $\Delta t_i^k = t_i^k - t_{(i-1)}^k$.

 ISBN: 978-1-61208-411-4

A packet will be lost when it does not meet the jitter restriction [15] [16], i.e., BD is not enough to jitter removal in packet i, then:

$$p_i^k > a_1^k + BD^k + (i-1)\Delta t_i^k \quad (4)$$

Theorem 1 presents a buffer delay value to prevent packet loss by jitter.

Theorem 1: In a talkspurt k, with buffer delay BD^k, no packet is lost by jitter restriction violation if and only if

$$BD^k \quad \geq \max_{i\in\{1,2,...,n^k\}}\{\delta_i^k - (i-1)\Delta t_i^k\}$$

where $\delta_i^k = a_i^k - a_1^k$ for every $i \in \{1,2,...,n^k\}$.

Proof: Since there is no packets loss in talkspurt, this is equivalent to: $p_i^k - a_i^k \geq 0$ for every $i \in \{1,2,...,n^k\} \Leftrightarrow a_i^k \leq p_i^k \Leftrightarrow a_i^k \leq a_1^k + BD^k + (i-1)\Delta t_i^k \Leftrightarrow a_i^k - a_1^k \leq BD^k + (i-1)\Delta t_i^k \Leftrightarrow BD^k \geq (a_i^k - a_1^k) - (i-1)\Delta t_i^k \Leftrightarrow BD^k \geq \max_{i\in\{1,2,...,n^k\}}\{\delta_i^k - (i-1)\Delta t_i^k\}$, for every $i \in \{1,2,...,n^k\}$.

Notice that:

$$BD^k \geq BD_{npl}^k = \max_{i\in\{1,2,...,n^k\}}\{\delta_i^k - (i-1)\Delta t_i^k\}$$

where BD_{npl}^k is the buffer delay which does not present packet loss. ■

Thus, we introduce the notion of limiting due to jitter. In the next definitions consider $N = \{1,2,...,n^k\}$ all packet indexes of the k-th talkspurt.

Definition 1: The BD_c^k is c-th limiting due to jitter, i.e., the value that remove jitter in a set Ω_c of packets of talkspurt k is defined by

$$BD_c^k = \max_{i\in\Omega_c}\{\delta_i^k - (i-1)\Delta t_i^k\},$$

where $\Omega_0 = N$, and $\Omega_c = N - (u_0 \cup u_1 \cup ... \cup u_{c-2} \cup u_{c-1})$ for $c > 0$, and $u_c = \{r_c^1, ..., r_c^{w_c}\}$ are the w_c packets where $p_i^k = a_1^k + BD_c^k + (i-1)\Delta t_i^k$ with $i \in u_c$.

Lemma 1: There is a finite number of jitter limiting values in a talkspurt.

Proof: The first jitter limiting value is: $BD_0^k = \max_{\{i\in\Omega_0=N\}}\{\delta_i^k - (i-1)\Delta t_i^k\}$, used by set of packets $u_0 \subset \Omega_0 = N$. Consider $\Omega_1 = N - u_0 \subseteq \Omega_0$, if $\Omega_1 = \emptyset$, the proof is completed, otherwise it is possible to calculate other jitter limiting value: $BD_1^k = \max_{\{i\in\Omega_1\}}\{\delta_i^k - (i-1)\Delta t_i^k\}$ for which there is a non-empty set $u_1 \subseteq \Omega_1 \subset \Omega_0$ of packets. This reasoning is applied until one is found $\Omega_{m+1} = \emptyset$, then the last jitter limiting value is $BD_m^k = \max_{\{i\in\Omega_m\}}\{\delta_i^k - (i-1)\Delta t_i^k\}$, where $m \leq n$ and, exist $\emptyset \neq u_m \subseteq \Omega_m \subset \Omega_{m-1} \subset ... \subset \Omega_0$ of packets that use that value to remove jitter. Thus, we obtain a finite number of jitter limiting value.

■

Lemma 2: The jitter limiting values are presented in the format $BD_j^k < BD_{j-1}^k$ to $j = 1, ..., m$.

Proof: Considering $BD_j^k < BD_{j-1}^k$, then:

$$BD_j^k = \max_{\{i\in\Omega_j\}}\left\{\delta_i^k - (i-1)\Delta t_i^k\right\}$$

$$BD_{j-1}^k = \max_{\{i\in\Omega_{j-1}\}}\left\{\delta_i^k - (i-1)\Delta t_i^k\right\}$$

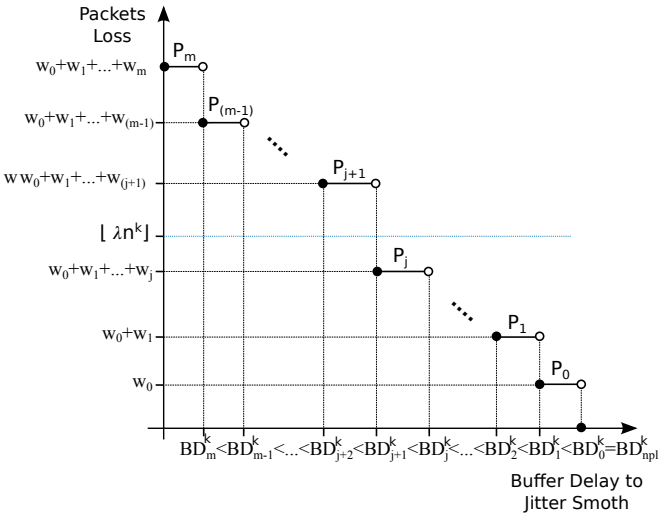


Figure 2. Steps due to jitter in a talkspurt.

where $\Omega_j = N - (u_0 \cup ... \cup u_{j-2} \cup u_{j-1})$ and $\Omega_{j-1} = N - (u_0 \subset \cup ... \cup u_{j-2})$, so $\Omega_j \subset \Omega_{j-1}$, then $BD_j^k < BD_{j-1}^k$. If $BD_j^k = BD_{j-1}^k$ then $u_j \cap < u_{j-1} \neq \emptyset$ ■

In a talkspurt, we have the following ordering $BD_m^k < BD_{m-1}^k < ... < BD_0^k$.

Definition 2: Intervals of type $P_m = \left[0, BD_m^k\right), P_{m-1} = \left[BD_m^k, BD_{m-1}^k\right), ..., P_0 = \left[BD_1^k, BD_0^k\right)$ will be referenced as steps due to jitter.

Definition 3: At each step, due to jitter we have associated a number named degree, given by:

$$degree(P_j) = \sum_{c=0}^{j} w_c$$

where $BD_{m+1}^k = 0$, and $j = 0, ..., m$.

Each degree is unique by definition. Besides $w_c \geq 1$ for each c, from what we can conclude that:

$$0 < degree(P_0) < ... < degree(P_m)$$

The lemma 3 allows monitoring the packet loss behaviour with each BD^k value used to jitter remove.

Lemma 3: Using BD^k in a talkspurt, then the number of lost packets is equal to the degree to which the step belongs.

Proof: In a talkspurt, we have the degrees P_j, with $j = 0, ..., m$, due to lemma 2 we have that $0 = BD_{m+1}^k < BD_m^k < ... < BD_{j+1}^k \leq BD^k < BD_j^k < ... < BD_1^k < BD_0^k$. Then, $\max_{\{i\in\Omega_{j+1}\}}\left\{\delta_i^k - (i-1)\Delta t_i^k\right\} = BD_{j+1}^k \leq BD^k < BD_j^k < ... < BD_0^k$ and:

$$\begin{aligned} BD_j^k &= \delta_r^k - (r-1)\Delta t_r^k,\ r \in u_j \\ BD_{j-1}^k &= \delta_r^k - (r-1)\Delta t_r^k,\ r \in u_{j-1} \\ &\vdots \\ BD_0^k &= \delta_r^k - (r-1)\Delta t_r^k,\ r \in u_0 \end{aligned}$$

Assuming that $BD^k < (a_r - a_1) - (r-1)\Delta t_r^k$ for all $r \in u_0 \cup ... \cup u_j$, thus we have, $a_1 + BD^k + (r-1)\Delta t_r^k < a_r$, for $r \in u_0 \cup ... \cup u_j$, i.e., the jitter restriction is broken for all $r \in u_0 \cup ... \cup u_j$, then packets $r_j, ..., r_0$ are lost. On the

other hand, with $BD^k \geq (a_r - a_1) - (r-1)\Delta t_r^k$, for all $r \in \Omega_{j+1} \supset \Omega_{j+2} \supset ... \supset \Omega_m \supset \Omega_{m+1}$.Then $p_r \geq a_r$ for all $r \in \Omega_j$, and $r \in \Omega_{j+2}$, so on for all $r \in \Omega_m$. With $u_{j+1} \subseteq \Omega_{j+1}, ..., u_m \subseteq \Omega_m$, the packets $r_{j+1}, ..., r_m$ not be lost, and $\{u_0, ..., u_m\}$ a subset of *N*, the total number of packets is $w_0 + ... + w_j = \sum_{c=0}^{j} w_c = degree(P_j)$. ■

IV. Buffer Delay Correction Algorithm

Prior to present the BDCA, an important definition is presented that relates buffer delay and target packet loss. This value is named OBD. In the previous Section, we can see that there is a certain limit to Buffer Delay (BD^k), and above this level there is no packet loss. On the other hand, the good quality of voice communication admits a certain limit of packet loss. Therefore, let us suppose a $\lambda \in (0, 1)$ of packets loss in a talkspurt, i.e., at most $\lfloor \lambda n^k \rfloor$ packets can be lost (see Figure 2) where $\lfloor x \rfloor$ is the floor function (greater integer smaller than or equal to x). In this case we are interested in solving (5) bellow.

$$\min \left\{ f(BD^k) = BD^k \mid \Psi(BD^k) \leq \lfloor \lambda n^k \rfloor, BD^k \in [0, +\infty) \right\} \quad (5)$$

That is the optimum buffer delay (OBD_λ^k), which represents a minimum delay value inserted in a talkspurt k, with target loss factor λ.

Theorem 2: In a talkspurt that use BD^k, n' packets will be lost, if and only if, BD^k belongs to degree of with step n'.

Proof: When $n' = 0$, i.e., no packet is lost, the theorem 1 assure this proof. If $n' > 0$, consider $W = \{w_0, w_0 + w_1, ..., w_0 + ... + w_m\}$ a set of all packets lost by jitter, if $n' \in W$ with $n' = w_0 + w_1 + ... + w_j$ for any j, then $BD^k \in P_j$. If $BD^k \in P_h$ for $0 \leq h < j$, less than n' packets would be lost, on other hand, if $j \leq h < m$, more than n' would be lost. With $degree(P_j) = w_0 + ... + w_j$, the BD^k belongs to a degree, with step n'. ■

Looking for theorem 2 and $BD^k \subset \{P_m, ..., P_0\}$ with $P_i \in [0, +\infty)$ we can write (5) as follow:

$$\min \left\{ \min\{ f(BD^k) = BD^k | \Psi(BD^k) \leq \lfloor \lambda n^k \rfloor, BD^k \in I \} \right\} \quad (6)$$

where $I \in \{P_m, ..., P_0\}$ and $\min\{f(BD^k) = BD^k \mid \Psi(BD^k) \leq \lfloor \lambda n^k \rfloor, BD^k \in I\}$ can be solved by Weierstrass Theorem, because in this case, I is compact and f is continuous.

The BDCA is a method to adjust the value presented by one BDA, i.e., approaching BD_{BDA}^k to OBD_λ^k, with packet loss rate in λ. To apply BDCA over talkspurt k, the Adjust Factor (AF) is computed as:

$$AF(k) = \frac{1}{Z} * \sum_{i=(k-1-Z)}^{i=(k-1)} \frac{OBD_\lambda^i}{BD_{BDA}^i} \quad (7)$$

The window size (Z) has the last 40 received talkspurts to reduce computational costs, values greater than 40 do not change significantly the results. To find OBD_λ^i, the following elements are needed:

- Packets transmitted until talkspurt $(i-1)$;

$$N_{i-2} = \sum_{j=1}^{i-2} n^j \quad (8)$$

- Number of packets lost from talkspurts 1 to $(i-1)$;
- Target Packet loss rate (λ).

The OBD_λ^i should be used in i-th talkspurt to bring the packet loss closer to λ. The BD_{BDA}^i is the value computed by selected BDA. Equation (9) shows Buffer Delay adjusted:

$$BD_{BDCA}^k = BD_{BDA}^k * AF(k) \quad (9)$$

Then, the adjusted playout time ($\hat{p}_i^k$) is defined by:

$$\hat{p}_i^k = a_1^k + BD_{BDCA}^k + (i-1)(\Delta t_i) \quad (10)$$

Consider the talkspurt $(k-1)$ received, the BDCA to playout time adjust of talkspurt k is showed in Figure 3.

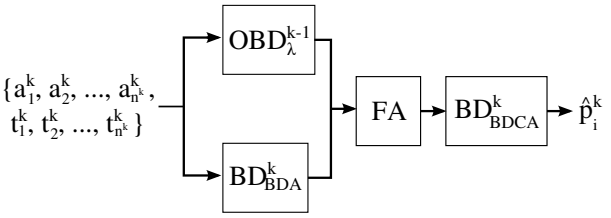


Figure 3. BDCA.

The OBD algorithm presents linear time and can run in parallel with BDCA, this makes BDCA defined by BDA's computational complexity.

V. Implementation and Results

In this Section, a performance analysis of the proposed algorithm is presented. The BDAs used for comparison are:

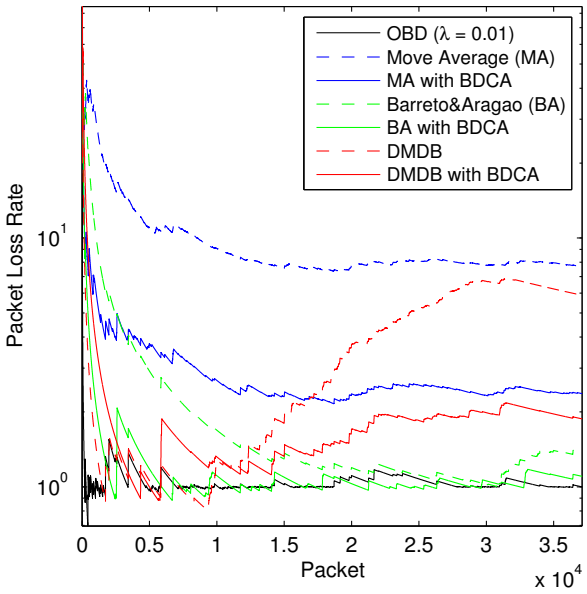
- Move Average Algorithm (MA) [10] a loss-tolerant technique;
- Algorithm from Barreto and Aragao (BA) presented in [8], classified as loss-intolerant technique;
- Dynamic Management of Dejitter Buffer (DMDB) presented in [13], considered a quality based technique.

For the tests, we consider the traces described in [3]. The traces contain the sender and receiver timestamps of transmitted packets. One 160 bytes audio packet is generated approximately at every 20 ms when there is speech activity [17]. The number of concurrent applications, network protocols or other elements of network environment may change the packet delay, but do not affect the BDCA. This enable the use of traces in simulation tests. A description of the traces is depicted in Table I.

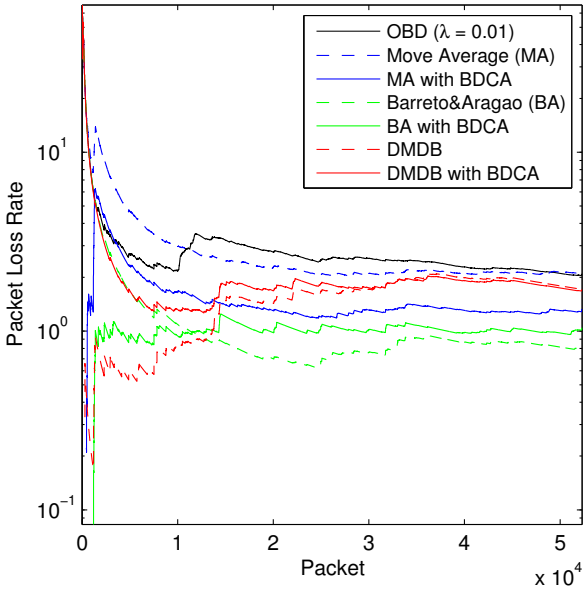
TABLE I. Traces Description.

trace	*Talkspurts*	*Packets*	*Length (s)*
A	536	37104	165.696
B	540	52296	174.604

To assess the performance of BDCA, we focus in packet loss rate, buffer delay average and quality of call (MOS). Considering N packets in a session, M the number of talkspurts, n^k the number of packets in talkspurt k, r_i^k the success indicator with values $r_i^k = 0$ when the packet is lost ($p_i^k < a_i^k$) or $r_i^k = 1$ when packet is available in receiver on playout time



(a) Trace A



(b) Trace B

Figure 4. Traces A and B with $\lambda = 0.01$

$(p_i^k \geq a_i^k)$. The total number of packets played out in an audio session is given by

$$\Upsilon = \sum_{k=1}^{M} \sum_{i=1}^{n^k} r_i^k \tag{11}$$

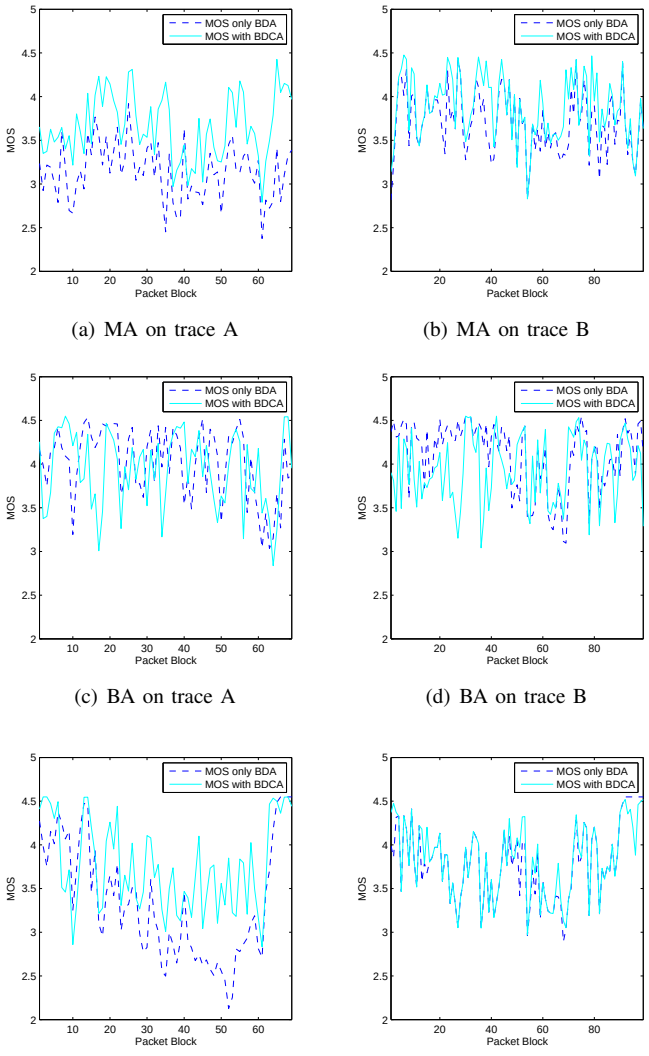
We consider in this work the average buffer delay to remove jitter (BD_{av}), shown in (12).

$$BD_{av} = \frac{1}{\Upsilon} \sum_{k=1}^{M} \sum_{i=1}^{n^k} r_i^k \left(p_i^k - a_i^k\right) \tag{12}$$

The percentage of packets not used in audio application on the receiver side (ω) is obtained by the (13).

$$\omega = \frac{N - \Upsilon}{N} * 100 \tag{13}$$

In graphs of Figure 4 we use the terms "With BDCA" to represent the original BDA running with BDCA. The target percentage of packets loss is 1%. These graphs are showing the evolution of packet loss rate in a voice call. For interactive audio, packet loss rate is considered adequate up to 1% of call [18], [19]. The Figure 5 are showing MOS values computed using PESQ algorithm over selected BDAs.



(a) MA on trace A (b) MA on trace B

(c) BA on trace A (d) BA on trace B

(e) DMDB on trace A (f) DMDB on trace B

Figure 5. Evolution of MOS.

TABLE II. PACKET LOSS TARGET (λ) IN 1%.

trace	BDA	Only BDA			BDA with BDCA			OBD		
		ω	BD_{av}	MOS	ω	BD_{av}	MOS	ω	BD_{av}	MOS
A	MA	7.70	116.55	3.13	2.37	180.52	3.65	0.99	121.74	4.01
	BA	1.39	216.59	4.01	1.10	401.93	3.93	0.99	121.74	4.01
	DMDB	5.88	191.97	3.34	1.87	290.62	3.75	0.99	121.74	4.01
B	MA	2.13	42.45	3.69	1.31	61.62	3.87	2.08	30.78	3.74
	BA	0.83	65.94	4.10	1.03	68.99	3.93	2.08	30.78	3.74
	DMDB	1.72	64.22	3.78	1.70	79.81	3.82	2.08	30.78	3.74

 ISBN: 978-1-61208-411-4

The Table II shows the results of packet loss target percentage (1%), the columns ω and BD_{av} are expressed in percentage of transmitted packets and milliseconds, respectively. The MOS column present an average value computed using PESQ algorithm [20], on blocks of 3000 packets, with shift of 500 packets to next window. The tests were made in Matlab [21].

VI. Conclusion

In this paper, we presented the Buffer Delay Correction Algorithm (BDCA) to reduce the difference between packet loss rate of any BDA and the Optimum Buffer Delay (OBD). We have compared the BDA with and without BDCA using 1% of packet loss rate.

Figures 4(a) and 4(b) show that packet loss percentage (ω) with BDCA are closer to values from OBD than running only BDA. But any greater buffer delay is able to produce a reduced packet loss rate. The BDCA uses only the necessary buffer delay to regulate the packet loss rate to closer to target value. This can be viewed in Table II and graphics of Figure 5, call quality is best or equal the results "without BDCA" (or only BDA) in most parts of calls.

We are currently expanding the definitions of Buffer Delay Adjustment to reach packet loss caused by latency, i.e., including the sum of packet discarded with playout time greater than the maximum threshold (L). To reach this new restriction, we are working in a new formulation of Adjust Factor.

References

[1] W. C. Hardy, VoIP Service Quality: Measuring and Evaluating Packet-Switched Voice). McGraw-Hill, 2003.

[2] Z. Qiao, R. K. Venkatasubramanian, L. Sun, and E. C. Ifeachor, "A new buffer algorithm for speech quality improvement in voip systems," Wirel. Pers. Commun., vol. 45, no. 2, apr 2008, pp. 189–207. [Online]. Available: http://dx.doi.org/10.1007/s11277-007-9408-7

[3] S. B. Moon, J. Kurose, and D.Towsley, "Packet audio playout adjustment: performance bounds and algorithms," ACM/Springer Multimedia Systems, vol. 6, no. 1, january 1998, pp. 17–28. [Online]. Available: http://dx.doi.org/10.1007/s005300050073

[4] Y. Zhang, D. Fay, L. Kilmartin, and A. W. Moore, "A garch-based adaptive playout delay algorithm for voip," Computer Networks: The International Journal of Computer and Telecommunications Networking, vol. 54, no. 17, dec 2010, pp. 3108–3122. [Online]. Available: http://dx.doi.org/10.1016/j.comnet.2010.06.006

[5] L. Atzori and M. L. Lobina, "Playout buffering in ip telephony: a survey discussing problems and approaches," Communications Surveys Tutorials, IEEE, vol. 8, no. 3, 2006, pp. 36–46. [Online]. Available: http://dx.doi.org/10.1109/COMST.2006.253269

[6] C. Perkins, RTP: Audio and Video For The Internet. Addison Wesley, 2012.

[7] R. Ramjee, J. Kurose, D. Towsley, and H. Shulzrinne, "Adaptive Playout Mechanisms for Packetized Audio Applications in Wide-Area Networks," in Proceedings of IEEE Infocom, vol. 2, Montreal, Canada, 1994, pp. 680–688. [Online]. Available: http://citeseer.nj.nec.com/ramjee94adaptive.html

[8] J. B. A. Jr. and G. A. Barreto, "Novel approaches for online playout delay prediction in voip applications using time series models," Computers and Electrical Engineering, vol. 36, no. 3, 2010, pp. 536 – 544. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S004579060900113X

[9] K. Fujimoto, S. Ata, and M. Murata, "Statistical analysis of packet delays in the internet and its application to playout control for streaming applications," IEICE Transactions on Communications, vol. E84-B, no. 6, june 2001, pp. 1504–1512. [Online]. Available: http://www-ana.ist.osaka-u.ac.jp/achievements/web2001/papers/k-fujimo01ieice-ModelingPlayout.pdf

[10] V. M. R. Ramos, C. Barakat, and E. Altman, "A moving average predictor for playout control in voip," in Proceedings of the 11th international conference on Quality of service, ser. IWQoS'03, vol. 2707. Berlin Heidelberg: Springer-Verlag, June 2003, pp. 155–173. [Online]. Available: http://dl.acm.org/citation.cfm?id=1784037.1784049

[11] K. Fujimoto, S. Ata, and M. Murata, "Adaptive playout buffer algorithm for enhancing perceived quality of streaming applications," Telecommunication Systems, vol. 25, no. 3-4, april 2004, pp. 259–271.

[12] ITU-T P.800, Recommendation ITU-T P.800 - Methods for subjective determination of transmission quality, Telecommunication Standardization Sector of International Telecommunication Union (ITU), august 1996. [Online]. Available: http://www.itu.int/rec/T-REC-P.800

[13] R. F. Valle, L. S. G. de Carvalho, R. B. Aguiar, E. S. Mota, and D. Freitas, "Dynamical management of dejitter buffers based on speech quality," in Proceedings of the The IEEE symposium on Computers and Communications, ser. ISCC '10. Washington, DC, USA: IEEE Computer Society, 2010, pp. 56–61. [Online]. Available: http://dx.doi.org/10.1109/ISCC.2010.5546799

[14] L. Sun and E. C. Ifeachor, "Prediction of perceived conversational speech quality and effects of playout buffer algorithms," in Communications, 2003. ICC '03. IEEE International Conference on, vol. 1, 2003, pp. 1–6.

[15] F. Sakuray, R. S. V. Hoto, and L. S. Mendes, "Analysis and estimation of playout delay in voip communications," International Journal of Computer Science and Network Security, vol. 8, no. 3, March 2008, pp. 98–105. [Online]. Available: http://paper.ijcsns.org/07_book/200803/20080315.pdf

[16] D. Florencio and L.-W. He, "Enhanced adaptive playout scheduling and loss concealment techniques for voice over ip networks," in Circuits and Systems (ISCAS), 2011 IEEE International Symposium on, 2011, pp. 129–132. [Online]. Available: http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=5937518

[17] H. Schulzrinne, "Voice communication across the internet: a network voice terminal," Tech. Rep., july 1992. [Online]. Available: http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.120.1343

[18] S. Nagireddi, VoIP Voice and Fax Signal Processing, 1st ed. Wiley Publishing, 2008.

[19] TIA/EIA 116A, Telecommunications-IP Telephony Equipment - Voice Quality Recommendation for IP Telephony, Telecommunication Industry Association, 2006.

[20] ITU-T P.862, Recommendation ITU-T P.862 - Perceptual evaluation of speech quality (PESQ): An objective method for end-to-end speechquality assessment of narrow-band telephone networks and speech codecs, Telecommunication Standardization Sector of International Telecommunication Union (ITU), february 2001. [Online]. Available: http://www.itu.int/rec/T-REC-P.862

[21] MATLAB, MATLAB and Statistics Toolbox Release 2012b. Natick, Massachusetts: The MathWorks Inc., 2012.

 ISBN: 978-1-61208-411-4