MPEG-4 ALS

Ersin Uzun

euzun@ics.uci.edu

***Abstract*: Although modern lossy coding standards such as MP3 or AAC can achieve high compression ratios with transparent subjective quality, they do not preserve every single bit of the original audio data. Lossy coding methods are not suitable for editing or archiving purposes, since multiple coding or post-processing can reveal originally masked distortions. A new emerging standard MPEG-4 ALS introduces an efficient lossless audio coding that enables the compression of digital audio data without any loss in quality and the perfect reconstruction of the original signal. MPEG-4 ALS employs linear prediction over the most recent subset of the audio streams to predict the next stream and encodes the difference between the actual and the predicted values.**

In this paper, we give a detailed understanding of MPEG-4 ALS. We discuss our experimental results that quantify the dependency between block size and residual values in MPEG-4 ALS prediction algorithm to find the optimal block size for better compression. We also experiment on the most popular lossless audio compression products in the market to compare their compression ratios for different audio files.

1 INTRODUCTION

Lossless compression has many applications in recording and distribution of audio. For instance, bit-exact representation is strongly desirable in archival systems, studio operations, for collaborative work or music distribution over broadband links. Even wider use of lossless audio compression in the future should be enabled by continuing growth of capacities of storage devices, transition to high-speed Internet connections, and broadband wireless networks. All these trends and application scenarios makes the lossless audio compression an appealing research and commercial topic.

The MPEG audio subgroup has been working on the standardization of lossless coding techniques for high-definition audio signals since 2002. Initially, MPEG-4 ALS was expected to be finalized by the end of 2004, but even one year after that, it hasn't been standardized yet. However, lossless coding is certainly accepted to become the latest extension of the MPEG-4 audio standard as soon as it is ready.

For experimental purposes, we implement a system that emulates the linear prediction function of the MPEG-4 ALS. We implement the *Levinson-Durbin* [2] algorithm for linear prediction order and coefficients estimation, and try to understand the behavior of the algorithm with different block sizes. In addition, we run the most popular lossless audio compression techniques under their suggested normal use parameters. We experimented on *Apple Lossless* [10], *FLAC* [7], *Monkey's Audio* [8], *La* [11], *Shorten* [6] and *Windows Media Encoder 9 Series* [12] for our comparisons and quantify the performance in terms of achieved compression ratio.

In section 2-4, we present a brief description of MPEG-4 ALS. In section 5, we present our experiments on block size vs. residual variance relation in MPEG-4 ALS prediction accuracy. In section 6, we compare the most popular lossless audio compression products in the market.

2 AUDIO LOSSLESS CODING

Modern audio encoding (e.g. AAC, MP3) applies lossy compression techniques. They achieve high compression ratios, but they do not preserve all the bit information of the original audio data. Lossy coding methods are not suitable for editing or archiving applications, since signal distortions can be revealed through post-processing and multiple coding. Moreover, Lossy compression is not attractive when there is no space constraint for storage or there exists broadband connection for streaming audio. Lossless entropy coding methods such as Lempel-Ziv, Huffman or Arithmetic Coding are not very efficient techniques for audio signals due to the long-time correlations and the high range of values (high entropy).

Forward Adaptive Linear Prediction [3, 13] is one of the methods to beat the restrictions of classic entropy coding methods. In prediction, each sample of the original signal is predicted from previous samples. The difference between original and predicted version is called the residual and if prediction works well, the residuals have a smaller range of values than the original audio signal and can be compressed better. The residual is usually coded using simple entropy coding methods such as Golomb-Rice codes, which are a special case of Huffman codes. Quantization and encoding of LPC coefficients is performed by quantizing the partial correlation (PARCOR) coefficients of the predictor using the arcsine function [1]. How PARCOR coefficients are calculated is explained in Levinson-Durbin algorithm section.

3 MPEG-4 ALS SYSTEM

The MPEG-4 ALS standard not only adopts the concept of forward linear predictive coding but also defines an implementation suitable for multiple computing environments.

A. Structures of the Encoder and Decoder

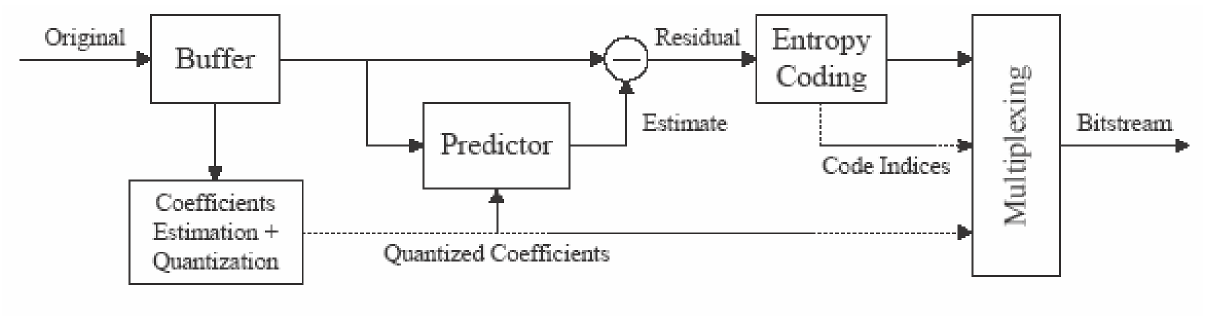


Figure 1 MPEG-4 ALS Encoder

The encoder (Fig. 1) consists of the following blocks:

- *Buffer*: Stores an audio frame, consisting of one or more blocks of PCM samples that represent signals in each audio channel.
- *Coefficients Estimator and Quantizer***:** Estimates and quantizes optimal predictor coefficients for each block.
- *Predictor***:** Calculates the prediction residual from the original signal and the quantized coefficients.
- *Entropy Coding*: Encodes the residual using Golomb-Rice or more efficient entropy codes.
- *Multiplexer***:** Combines the encoded residuals, code indices, predictor coefficients and CRC code in the compressed stream.

For each channel, a prediction residual is calculated using linear prediction with adaptive coefficients. The coefficients are quantized and transmitted as side information. The prediction residual is entrophy coded using Golomb-Rice codes. Finally the multiplexing unit combines all the information to form the compressed bitstream (Fig. 3).

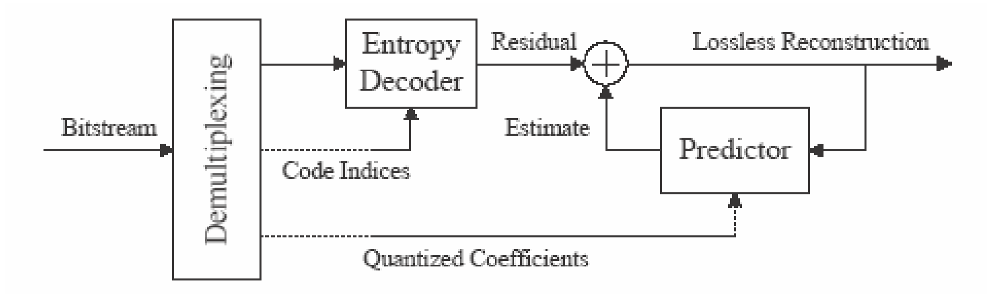


Figure 2 MPEG-4 ALS Decoder

The decoder (Fig. 2) performs exactly the reverse operations on the compressed signal. It first predicts the next signal using the coefficients sent by the encoder and then adds the residual to obtain the original signal.

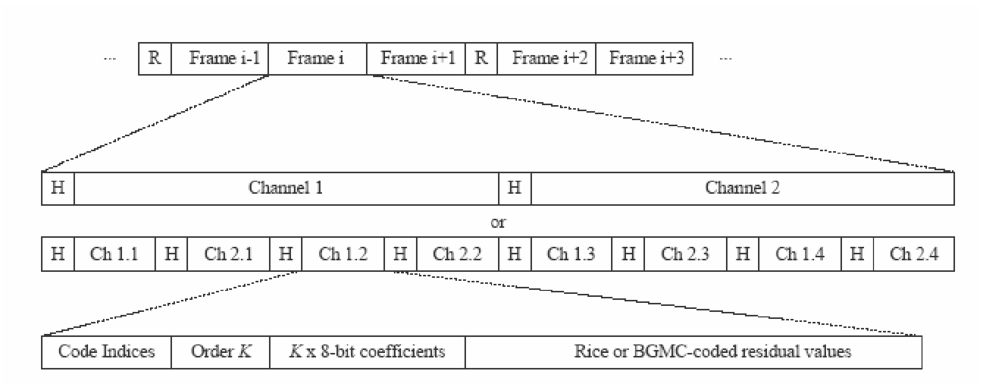


Figure 3 MPEG-4 ALS Bitstream
(R is random access block, H is header)

B. Linear Prediction

In general, the current sample of a time discreet signal *x(n)* can be approximately predicted from samples *x(n-k)*. The estimate is given by (1) where K is prediction order and h_k is prediction coefficient.

$$\hat{x}(n) = \sum_{k=1}^{K} h_k \cdot x(n-k), \tag{1}$$

The exact formula used in MPEG-4 ALS [1, 13] to predict y_i from the previous samples y_{i-k} is

$$y_i = \left\lfloor \left(\sum_{k=1}^{M} a_k y_{i-k} + 2^{Q-1} \right) / 2^{Q} \right\rfloor \tag{2}$$

Q is the number of bits for representing the **coefficient *a*;** *M* is the **order** of the predictor (the number of previous samples involved in the prediction); *r* is the **residual**; a_k is the linear prediction coefficients for prediction order *k*. Optimal prediction coefficients are estimated using the **Levinson-Durbin** algorithm which also provides simple means to find the optimal order for prediction.

C. *Coefficient Quantization*

Direct quantization of the predictor coefficients is not very efficient for transmission, since small errors in quantization can produce spectral errors and might yield to instability. Levinson-Durbin Algorithm estimates PARCOR coefficients which are less sensitive for errors and we can obtain the actual predictor coefficients back for the prediction order *i* from the **partial coefficients** $k_1,\ldots, k_i$ in decoding. In order to further reduce the sensitivity to errors, instead of using variable length LAR quantization, the arcsine values of partial coefficients are quantized.

$$arc_i = \arcsin(k_i) \tag{3}$$

At the decoder, an integer-arithmetic function accurately converts the quantized arc_i values to the prediction coefficients α_i.

D. *Residual Entropy Encoding*

The prediction residual is encoding using Golomb-Rice codes [4]. This encoding technique assumes that large integers occur with lower probability. The probability model of such a geometrically distributed source with parameter θ in (0, 1) is

$$P_\theta[i] = (1-\theta)\theta^i \tag{4}$$

Block Length Switching: The encoder processes an input audio signal using fixed-length frames. In the basic mode one frame corresponds to one sample block. The frame length is 2048 bytes for 48 kHz and 4096 bytes for 96 kHz, which corresponds to approximately 43 ms of audio duration. Optionally a frame can be subdivided into 4 sub-blocks for more fine-grained prediction that can better adapt to changes in audio signals. In their last publication [?], MPEG-4 ALS group takes this approach further and the last proposal allows up to 32 hierarchical subdivisions as shown in figure 4.

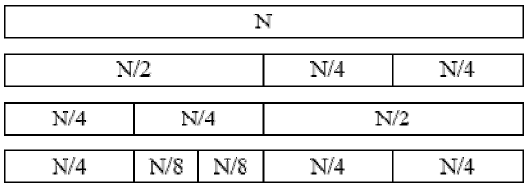


Figure 4 Block length hierarchical subdivisions

Random Access: The standard inserts in the encoded stream random access frames that are not predicted from previous frame. Each random access frame starts with an info field that specifies the distance to the next random frame. Although it is named as random, this approach needs to be started from the first frame and searching in sequential hops between random frames.

Joint Stereo Coding: This function is used to exploit dependencies between the channels. The encoder calculates $d(n) = x_2(n) - x_1(n)$ where x_1 is Left and x_2 is Right channel. The systems then continuously monitor all three of $d(n)$, $x_1(n)$, $x_2(n)$ and encode just two of them that can result in better coding efficiency.

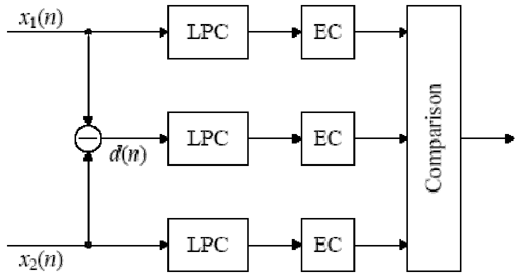


Figure 5 Joint Stereo Coding

4 LINEAR PREDICTIVE CODING

In this section we describe the Linear Predictive Coding analysis/synthesis scheme [2]. According to this method we are able to predict the sample value that succeeds a series of previous M sample values y_{n-1} .. y_{n-M} according to the following linear relation:

$$y_n = \sum_{i=1}^{M} a_i y_{n-i} + G\varepsilon_n \tag{5}$$

M is the order of the predictor and $G\varepsilon_n$ is the gain of the linear filter. We need to pick a such that the mean square error $r = (y_n - y_{n\_original})^2$ is minimized. Equalizing the derivatives of the expected r^2, with respect to the coefficients a to 0, we get M equations.

$$\frac{\partial}{\partial a_j} E\left[\left(y_n - \sum_{i=1}^{M} a_i y_{n-i} + G\varepsilon_n\right)^2\right] = 0 \tag{6}$$

From the partial derivatives above, we get the equations

$$\sum_{i=1}^{M} a_i E\left[y_{n-i} y_{n-j} \right] = E\left[y_n y_{n-j} \right] \qquad (7)$$

If assume that the $\{y_n\}$ sequence is stationary, for R_{yy} as the autocorrelation of the function

$$E[y_{n-i}\, y_{n-j}] = R_{yy}(|i - j|), \qquad (8)$$

Further assuming $\{y_n\}$ is 0 outside the input sample segment of interest for the filter parameters, the autocorrelation function is estimated as,

$$R_{yy}(k) = \sum_{n=n_0+1+k}^{n_0+N} y_n y_{n-k} \qquad (9)$$

where $n_0,$ and N define the segment/block for which we derive filter coefficients.

Defining matrix R as

$$R = \begin{bmatrix} Ryy(0) & Ryy(1) & Ryy(2) & \dots & Ryy(M-1) \\ Ryy(1) & Ryy(0) & Ryy(1) & \dots & Ryy(M-2) \\ Ryy(2) & Ryy(1) & Ryy(0) & \dots & Ryy(M-3) \\ \vdots & \vdots & \vdots & \dots & \vdots \\ Ryy(M-1) & Ryy(M-2) & Ryy(M-3) & \dots & Ryy(0) \end{bmatrix}$$

And A, P as

$$A = \begin{bmatrix} a_1 \\ \vdots \\ a_M \end{bmatrix} \qquad P = \begin{bmatrix} Ryy(1) \\ \vdots \\ Ryy(M) \end{bmatrix}$$

We can write the M equations in matrix form as $RA = P$. Since R is a *Toeplitz* matrix (the diagonal consists of the same element), we can use the efficient **Levinson-Durbin** algorithm to obtain a recursive solution with respect to A.

Levinson-Durbin Algorithm [2], [9]

To compute the coefficients of the M^{th} order filter, we need to compute all the filters of smaller order. Besides the filter coefficient, this algorithm also produces a set of partial correlation coefficients (PARCOR). In the following description α_i^k corresponds to the ith coefficient of the kth order filter

1. Set $E_0 = R_{yy}(0),\ i = 0,\ \alpha_0 = 1$
2. Increment i by one.
3. Calculate

$$k_i = -\left(\sum_{j=0}^{i-1} a_j^{i-1} Ryy(i-j)\right) / E_{i-1}$$

4. Set $a_i^i = k_i$
5. Calculate for $j = 1, 2, \ldots, i-1$.

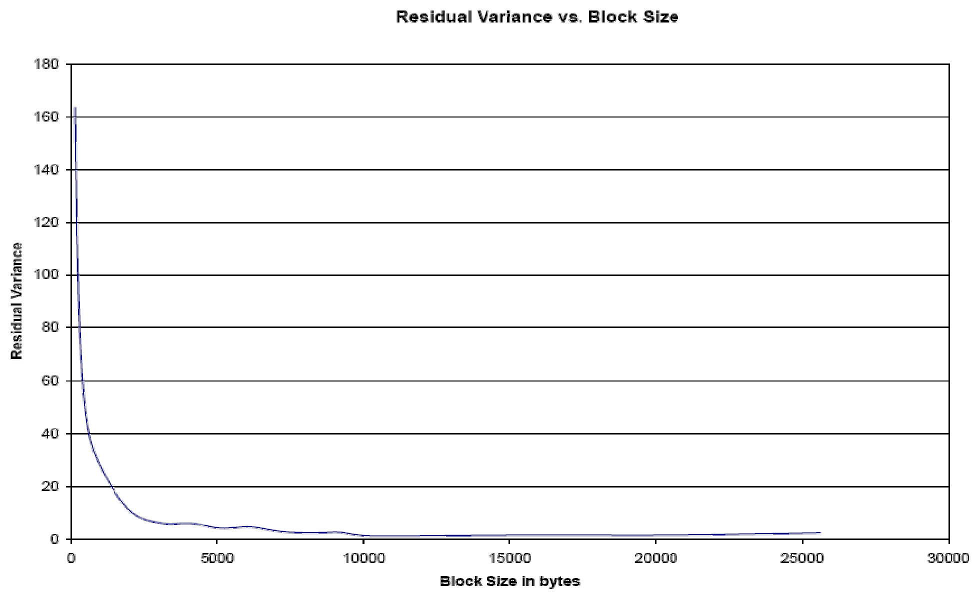
$$a_j^i = a_j^{i-1} + k_i a_{i-j}^{i-1}$$

6. Calculate $E_i = (1 - k_i^2)\, E_{i-1}$
7. If $i < M$, go to step 2.

For each order i we estimate the variance of the residual that results from the use of the determined coefficients for the i^{th} order, and therefore we can estimate the bit rate for the residual. The variance is estimated over the whole block except the first i samples in it. Considering the bit rate for transmitting coefficients, we are able to determine the optimal order as well. This is the order for which the residual variance no longer decreases.

5 Experimentations on linear prediction

We use Vivaldi Winter Concerto No. 4 classical song in the experiments. We generate an 8-bit audio PCM input stream from the part of a song (file size app. 50KB) and experimented on that chunk. We analyze the variance of residual values that result from applying linear prediction over the varying block sizes. Our experiments aim to quantify the dependency between block size and residual values to determine the optimal block size that can yield low entropy residuals. The graph on Fig. 6 shows this dependency. The x-axis represents the block size N and the y-axis represents the variance of the residual over the whole file. The residual variance provides us with an estimate of the required bit-rate for transmitting the residuals. Prediction is performed using Equation (1).



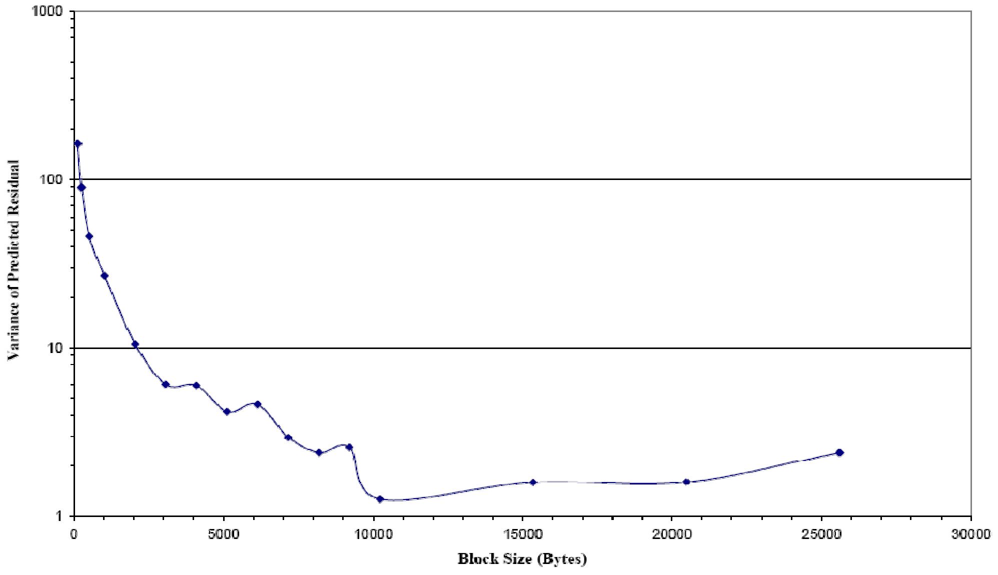


Figure 6 Residual Variance vs. Block Size Relation. First is normal and second is logarithmically scaled on residual variance.

As can be seen in Fig. 6, variance decreases with increasing block size. For a small block size of 128-bytes the variance is extremely high indicating that small block strategies are not effective. For large block sizes (above 3KB) the variance drops below 6. In general, the larger is N, the more accurate is the coefficient estimation. However, the prediction is not efficient above the block sizes of 15K. This is because the estimation cannot capture short-lived changes in the audio anymore. Therefore we recommend block sizes between 10K and 15K for classical kind of well patterned audio input. We note here that larger block size implies increased computation time and larger frame size which requires larger buffer requirements at the receiver. Moreover, the optimal block size should change depending on the audio input. The more transient changes the input has, the smaller should be the block sizes for better compression ratios.

6 Comparison among popular lossless audio compression schemes

Initially, we aimed to experiment on MPEG-4 ALS to find the optimum parameters and fine-tune the whole algorithm for better performance. After doing so, we were aiming to compare it with the other popular techniques to evaluate its performance in real life. However, the MPEG group didn't allow us to use their implementation either in binary or source code format. Since it was not feasible to implement it all from scratch, especially in its current not standardized form, we gave up with our initial plans and didn't involve MPEG-4 ALS to our comparisons. However, in [1] the authors compare the performance of their MPEG-4 ALS implementation to *Shorten* [6]**,** *FLAC* [7]**,** *Monkey's Audio* [8]**.** According to their results; *Shorten* generally exhibits smaller compression/decompression times by 50-70%, but substantially increased compression ratio by 10-20%. *FLAC* shows slightly higher ratio, approximately 30% higher encoding time, and slightly lower decoding times. *Monkey's Audio* has almost equal compression ratios, lower compression time, but 3 times more decompression time. Since we involved these 3 programs into our comparison as well, reader can tentatively place MPEG-ALS into the graph represented in Fig. 7 to compare its performance to the other state of the other schemes.

In our comparison, we used 9 different audio files having sizes between 52 and 840 megabyte. All these audio files are created from different music CDs from different artist and genres. First all audio samples are ripped into wav files and then compressed using six popular lossless audio compression programs under their suggested normal use parameters. The graph (Fig. 7) shows the achieved compressed file sizes for six different programs.

According to our comparisons the La gives the best compression ratio almost on all inputs of different size and kinds. On the other hand Monkey's audio consistently gets the second place and shorten has the worst compression rate among all.

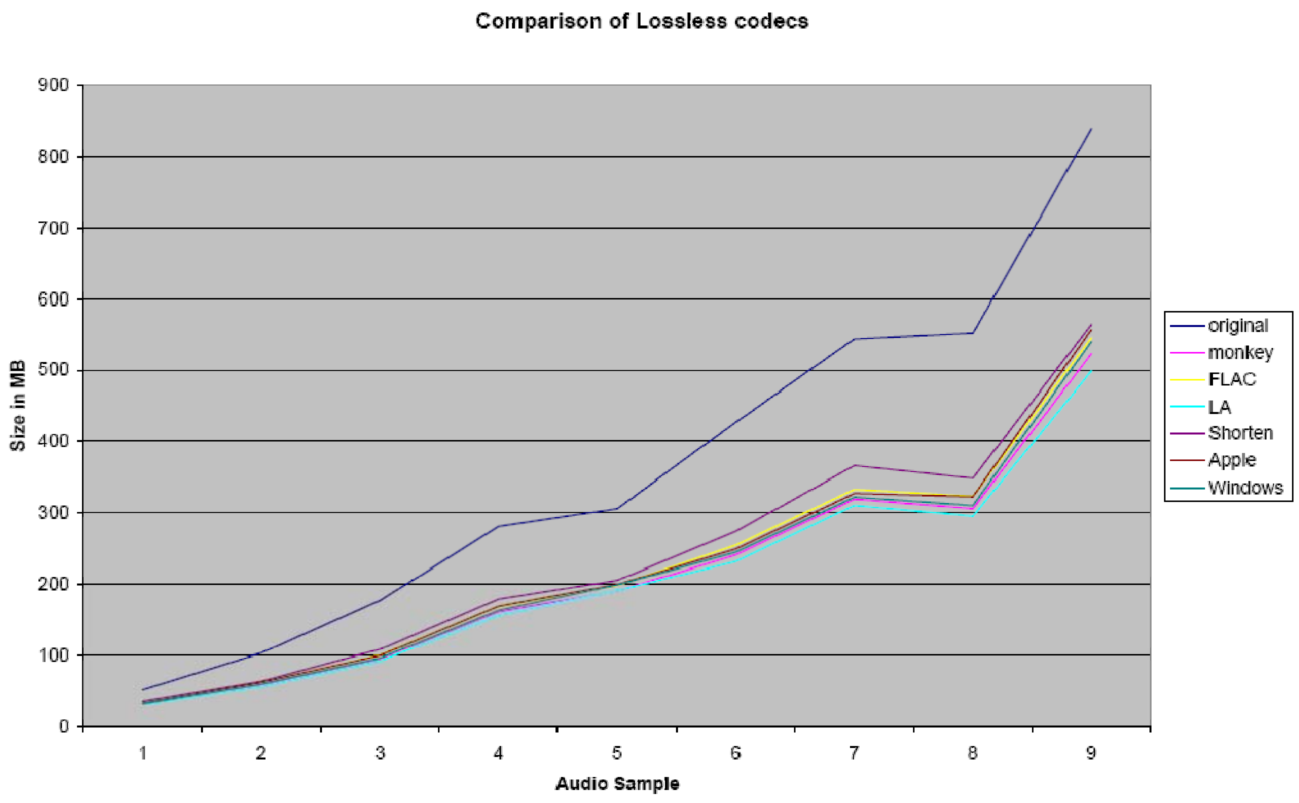


Figure 7 Comparison of compressed data file sizes among popular lossless audio compression programs

The average compression rates achieved by the programs over nine input samples are

Monkey's Audio	FLAC	La	Shorten	Apple Lossless	Windows Media 9
0.582618	0.610584	0.565402	0.648867	0.610018	0.593746

Figure 8 Avarage Compression Ratio of the Programs

Considering the results published in [1], it would not be wrong to put MPEG-4 ALS between La and Windows Media Codec 9 series in the comparison graph (fig. 7) and expect an average compression ratio between 0.56 and 0.59 from it.

References

[1] Liebchen, T. Reznik, Y.A., "MPEG-4 ALS: an emerging standard for lossless audio coding", in Data Compression Conference, 2004. Proceedings. DCC 2004 439-448.

[2] Khalid Sayood, "Introduction to Data Compression", Morgan Kaufman Publishers, Second Edition 2003, 502-509

[3] B. S. Atal and S. L. Hanauer, "Speech Analysis and Synthesis by Linear Prediction of the Speech Wave" in J. Acoust. Soc. Am. 50 (2) (1971) 637-655.

[4] S. W Golomb, "Run length Encodings", IEEE Trans. Inf. Theory, 12 (7) (1966) 399-401.

[5] Liebchen, T. Moriya, T. Harada, N. Kamamoto, Y. Reznik, Y., "The MPEG-4 Audio Lossless Coding (ALS) Standard - Technology and Applications", in 119th AES Convention, New York, 2005.

[6] T. Robinson, "SHORTEN, Simple Lossless and Near-lossless Waveform Compression", Tech. Rep. CUED/F-INFEG/TR.156, Campridge University, UK, 1994..

[7] "Monkey's Audio Compression Program", www.monkeysaudio.com

[8] "FLAC open-source audio compression program", http://flac.sourceforge.net

[9] "Lecture notes", http://www.cs.tut.fi/sgn/arg/8003051/LP_03_en.pdf

[10] "Apple Lossless Audio Codec", http://www.apple.com/itunes

[11] "La Lossless Audio", http://www.lossless-audio.com

[12] "Windows Media Encoder 9 Seies", http://www.microsoft.com/windows/windowsmedia/9series/encoder/default.aspx

[13] Liebchen, T. Reznik, Y. , "Improved Forward-Adaptive Prediction for MPEG-4 Audio Lossless Coding", in 118th AES Convention, Barcelona, 2005

*The emulated reference format is IEEE transactions and journals.