Proposal of Audio Compression Methods Using Text Coding

Omar Adil Mahdi
Computer Science Department,
College of Education (Ibn - Haitham), University of Baghdad
Ahmed Jasim Mohamed
Computer Science Department,
College of Computer, University of Al-Anbar

ABSTRACT

The proposed system is a new approach to compress audio files, by converting an audio file in to a text file and then compressing the new text file by using the common compression techniques. Two proposed coding methods (4-bit) and (6-bit) are used to convert a digitized Audio file into a text file by using the (4-bit) coding method, it is show that the results are not satisfactory, which make the researcher suggest the other coding method of (6-bit) instead. The proposed system has suggested and carried out the algorithm for audio compression by using CFG after convert the audio file into text file then the production rules have been got and using them for compression instead of the text, which perform a good compression rate 15% and some times it gets to 35% at the expense of time.

1- INTRODUCTION

The basic idea behind any compression-for audio and image- is to find a way to represents data that take up less space [1]. The most important advantage of data compression is reducing the time for data transfer via communication channels [2].

Two families of algorithms exit in compression. when the information can be exactly recovered from the bits, the source coding or compression is called lossless; otherwise, it is called lossy. To achieve higher compression ratios, lossy algorithms remove information from that comes close to the original or that is not perceptible.

Lossless algorithms are typically used for text or executable codes, while lossy are used for images and audio where a little bit of loss in resolutions often undetectable, or at least acceptable. Lossy is used in an abstract sense; it does not mean random loss in samples, but instead means loss of a quantity such as a frequency component, or perhaps loss of noise. [3]

The storage requirements of sound are smaller than those of images or movies, but bigger than those of text, this is why audio compression has become important and has been subject of much research and experimentation throughout 1990s.

The first well known method for effectively coding symbols is now known as Shannon-Fano coding .Claude Shannon at Bell Labs and R.M. Fano at MIT developed this method nearly simultaneously. It depends on simply knowing the probability of each symbol's appearance in a message.

Given the probabilities, a table of codes could be constructed that has several important properties:

- Different codes have different numbers of bits.
- Codes for symbols with low probabilities have more bits, and codes for symbols with high probabilities have fewer bits.
- Through the codes are of different bit lengths, they can be decoded.

Huffman coding shares character tics of Shannon-Fano coding. It creates variable-length codes that are an integral number of bits. Symbols with higher probabilities get shorter codes. Huffman codes have the unique prefix attribute, which means they can be correctly decoded despite being variable length .decoding a stream of Huffman codes is generally done by a binary decoder tree [4].

Several quantities are commonly used to express the performance of compression method. The compression ratio is defined as:

$$\textbf{Compression ratio} = \frac{\text{Size of the output stream}}{\text{Size of the input stream}}$$

The inverse of the compression ratio is called the compression factor

$$\textbf{Compression factor} = \frac{\text{Size of the input stream}}{\text{Size of the output stream}} \qquad [3].$$

2- Conventional Audio Compression Methods

Conventional compression methods, such as RLE, statistical, and dictionary-based, can be used to losslessly compress sound files, but the results depend heavily on the specific sound. Some sounds may compress well under RLE but not under a statistical method. Other sounds may lend themselves to statistical compression but may expand when processed by a dictionary method. Here is how sounds respond to each of the three classes of compression methods. RLE may work well when the sound contains long runs of identical samples.

Statistical methods assign variable-size codes to the samples according to their frequency of occurrence. Dictionary-based methods expect to find the same phrases again in the date. This happens with text, where certain string may repeat often. Lossy sound compression methods discard data to which the human ear is not sensitive. This is similar to lossy image compression, where data to which the human eye is not sensitive is discarded. In both cases we use the fact that the original information (image or sound) is analog and has already lost some quality when digitized. Losing some more data, if done carefully, may not significantly affect the played-back sound, and may therefore be indistinguishable from the original [6].

Because there are so many of data people want to compress, there has been a lot of work over the years on lossless compression techniques. These are general-purpose algorithms that look for patterns in binary data. If they can locate patterns, they can compress the data by replacing blocks of data with codes indicating the pattern. The decompressor (which knows about the same types of patterns) can then undo this process. These algorithms are lossless because the decompressed data will be bit-for-bit identical to the original. This is an important feature for general-purpose algorithms that might be used for any type of data.

Many applications– including streaming Internet sound involve transferring sound data and playing it as it is received. Frequently, there are harsh restrictions on how quickly the data can be transferred [1].

3- Proposed System Description

Compression methods are applied on wave file format to produce a compressed file. The proposed schema starts with wave a file .This file is converted into a text file and then coding the new text file using proposed 6-bits characters table then compressed by using the common compression techniques.

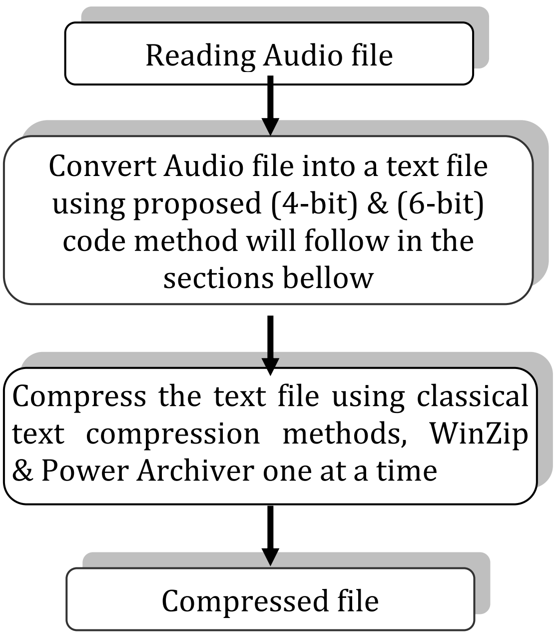

Figure (1)

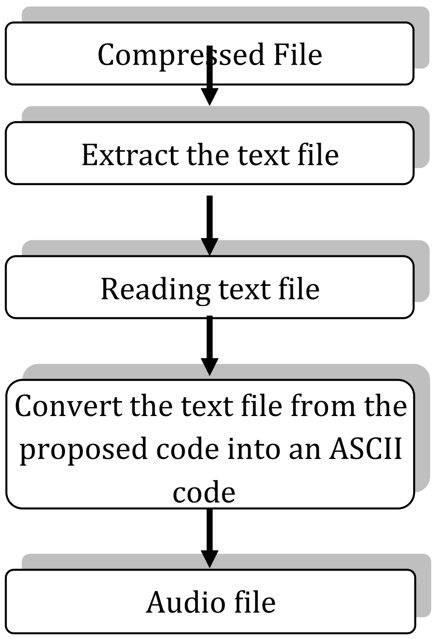

Figure (2)

Figure (1) illustrates the block diagram for the proposed coding compression system.
Figure (2) illustrates the block diagram for the proposed decoding compression system.

It is worthy to mention here that the researcher has used two types of coding which are (4 bits) and (6 bits) coding methods.

4- Proposed 4-bit Code Method Algorithm

In this section the proposed 4- bit code method will be presented as in following algorithm:

Input: Audio file (wave or mp3 format)
Output: Compressed file
Begin
Open Audio file for Input
Open Text file for Output
While not eof (Audio) ***do***
Begin
Read one byte from audio file
Split this byte into two parts
Search in table(1) to find the corresponded character for each part
Save these two characters into the text file

Table (1) Proposal 4 bit Coding Method

Character	Bits	Character	Bits	Character	Bits
A	000000	a	011010	0	110100
B	000001	b	011011	1	110101
C	000010	c	011100	2	110110
D	000011	d	011101	3	110111
E	000100	e	011110	4	111000
F	000101	f	011111	5	111001
G	000110	g	100000	6	111010
H	000111	h	100001	7	111011
I	001000	i	100010	8	111100
J	001001	j	100011	9	111101
K	001010	k	100100	,	111110
L	001011	l	100101	;	111111
M	001100	m	100110		
N	001101	n	100111		
O	001110	o	101000		
P	001111	p	101001		
Q	010000	q	101010		
R	010001	r	101011		
S	010010	s	101100		
T	010011	t	101101		
U	010100	u	101110		
V	010101	v	101111		
W	010110	w	110000		

X	010111	x	110001
Y	011000	y	110010

End
Close Audio file
Close text file
Compress using classical text compression methods
END.

Table (2) Proposal 6 bit Coding Method

Character	Bits
A	0000
B	0001
C	0010
D	0011
E	0100
F	0101
G	0110
H	0111
I	1000
J	1001
K	1010
L	1011
M	1100
N	1101
O	1110
P	1111

5- Proposed 6-bit Code Method Algorithm

In this section the proposed 6-bit code method will be presented as follows:

INPUT: Audio file (wave or mp3 format)
OUTPUT: compress file
BEGIN
OPEN Audio file for input
OPEN text file for output
While not eof (Audio) ***do***
Begin
Read three bytes from Audio file

Split these bytes into four parts 6-bit each
Search in table (3.2) to find the corresponded
character for (decimal value for each part)
save these three characters in the text file

End
CLOSE Audio file
CLOSE text file

Compress the text file using classical text compression methods
END.

6- Experiment And Results

By using the (4 bit) coding method, it is shown that the results are not satisfactory because the duplication of files size. Therefore, our experiments relied on the second method which converts the ASCII (8 bits) to proposal code (6 bits).

The following figures of our experiments show the compression ratio. It shows a graphical representation of the proposed schema compared to the regular compression techniques.

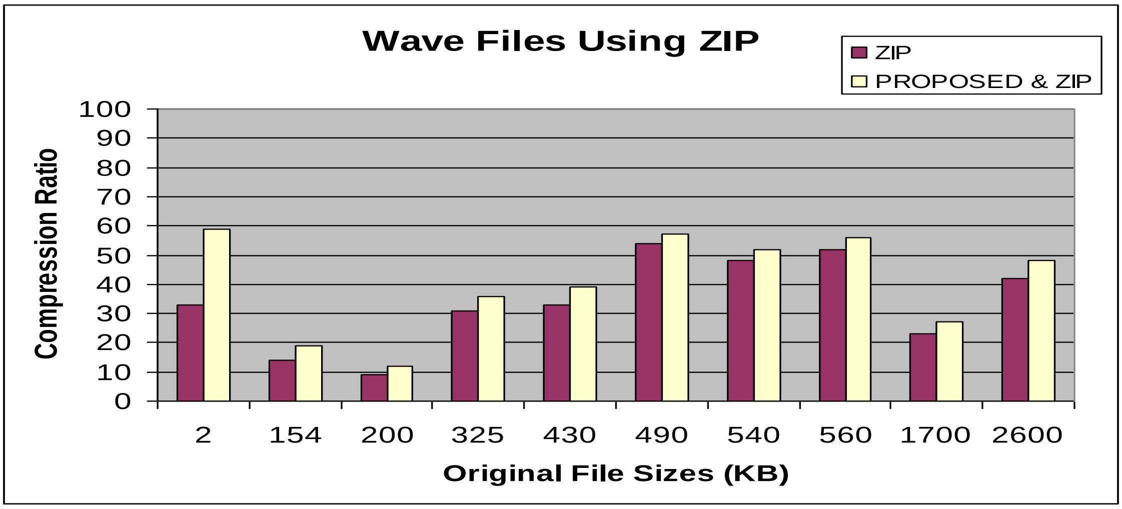

Figure (3) shows the comparison results of WinZip on wave files with & without our proposal coding method.

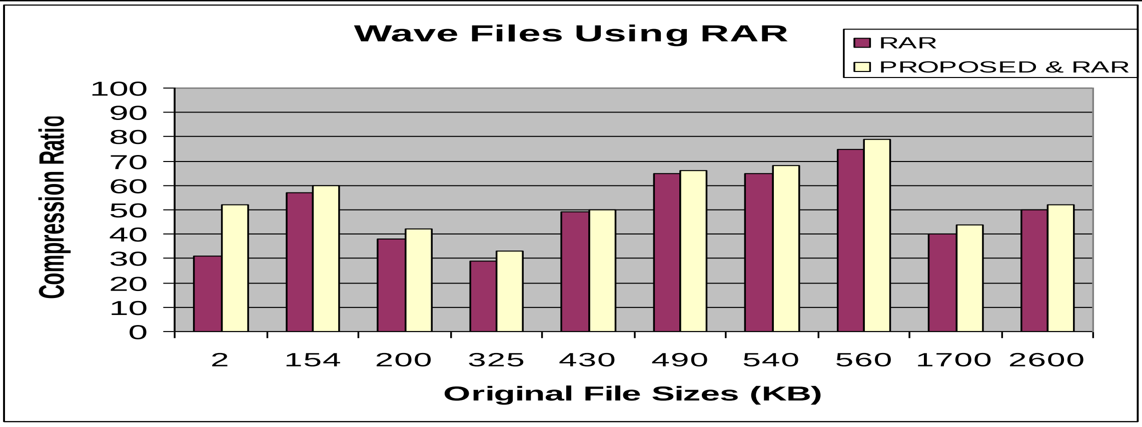

Figure (4) shows the comparison results of Power Archiver .Rar on wave files with & without our proposal coding method.

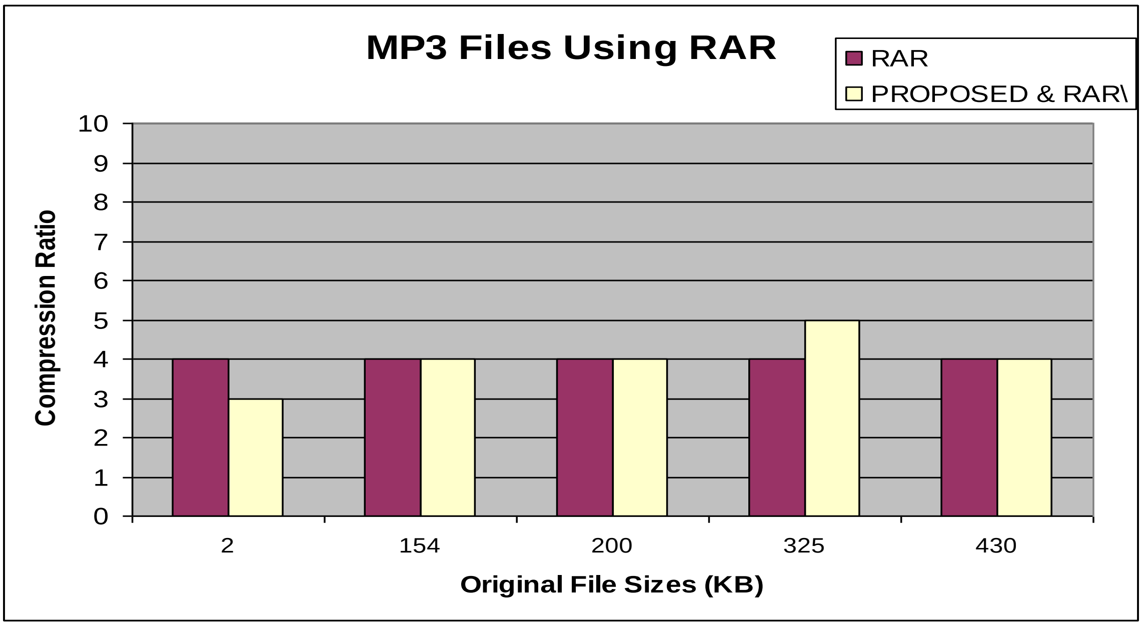

Figure (5) illustrates the comparison results of both the compression rate using Power Archive .Rar on mp3 files and the compression rate using proposal with Power Archiver, Rar on the text file.

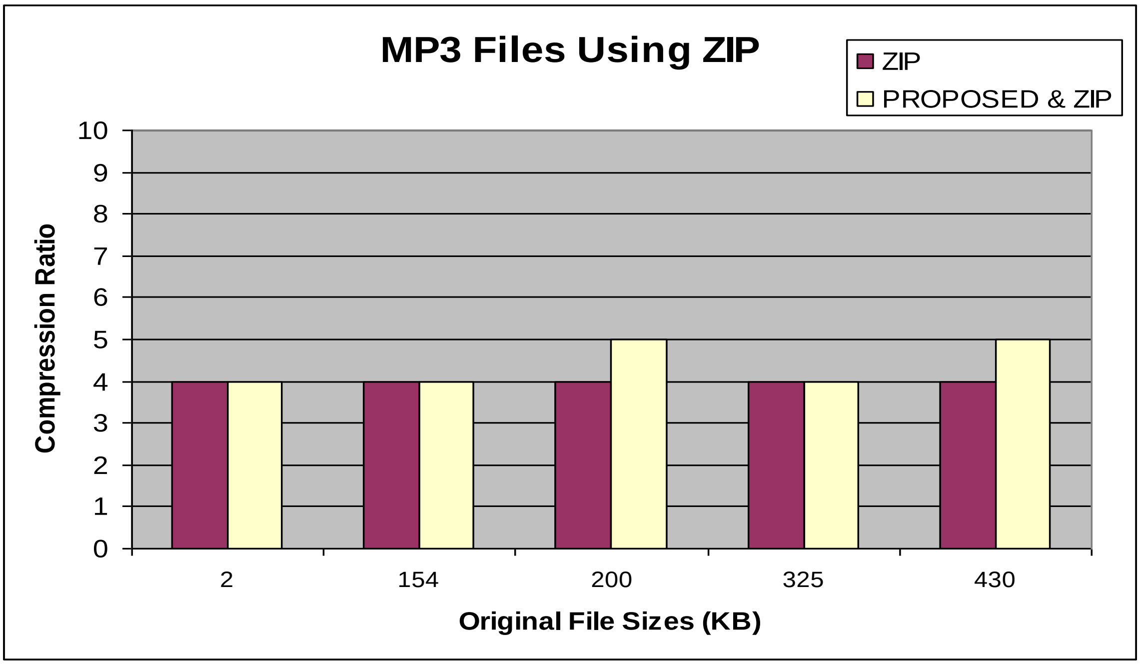

Figure (6) illustrates the comparison results of both the compression rate Using Win Zip on mp3 files and the compression rate using proposal with Win Zip on the text file.

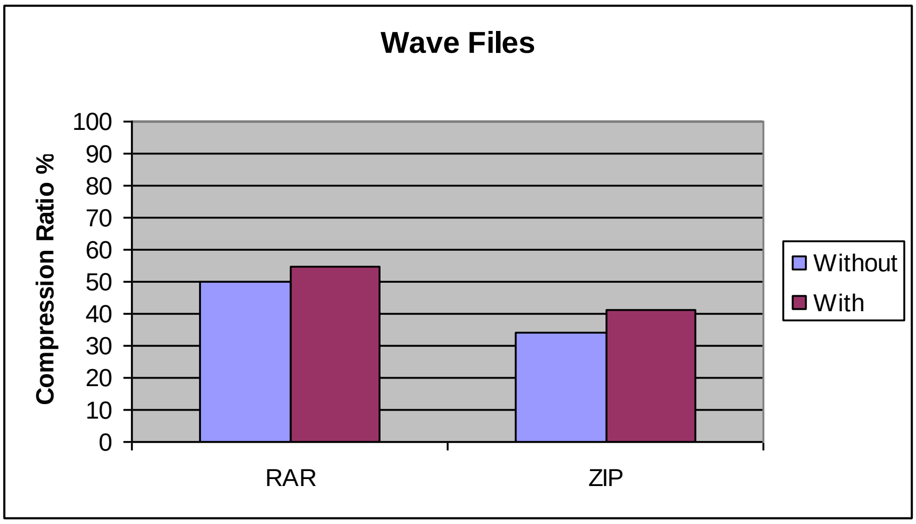

Figure (7) illustrates the average of compression ratio on WAVE files with WinZip (.zip) & Power Archiver (.RAR) before and after using the proposed method.

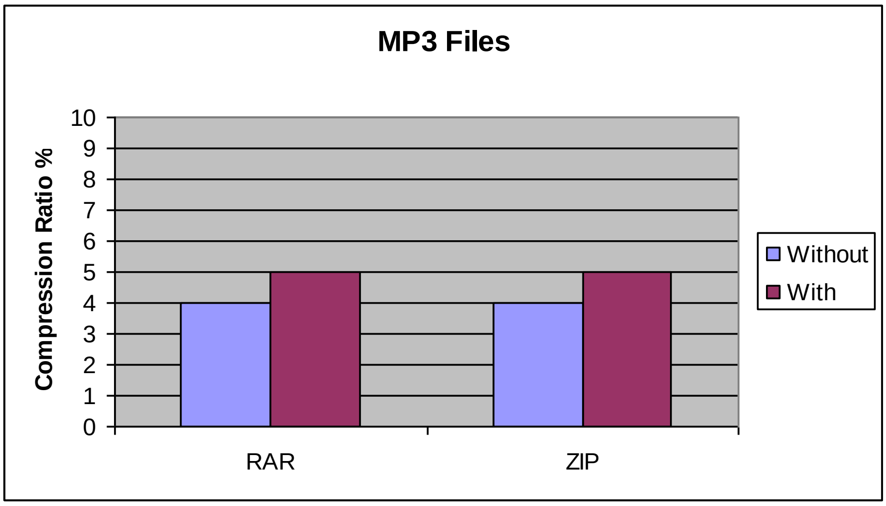

Figure (8) illustrates the average of compression ratio on MP3 files with WinZip (.zip) & Power Archiver (RAR) before and after using the proposed method.

7- Conclusion

In general lossy compression methods are used on sound files (speech & audio) in order to obtain good compression ratio, it is obvious that the amount of information (audio) loss will have a minor effect on the outcome. In our proposed method we realized that there is some loss of information since the file we are using is a digital file obtained from an analog file. So our proposed method should not lose any information (audio) the following conclusion are drawn:

1. The proposed coding methods (4-bit) and (6-bit) to convert a digitized audio file into a text file and according to our experiments the (6-bit) coding method was better than (4-bit) coding method.
2. When Power Archiver is applied to a WAVE files with our proposed method good compression ratio 55% are obtained, while the average compression ratio of a direct compression of the wave files is 50%.
3. When Win Zip is applied to a WAVE files with our proposed method good results 42%, are obtained while the average compression ratio of the WAVE files 34% without the proposed method.
4. When Win Zip is applied on MP3 files with our proposed method we also get good results 5%, while the maximum compression ratio of the direct compression of the MP3 files is 4%.
5. The results show that the Power Archive get better results than Win Zip. At the same time both of them give better results than the direct compression of the audio files (WAVE & MP3). Generally, WAVE files compression gives better results than MP3 files.

REFERENCES

[1] Tim Kientzle “A Programmer’s Guide to Sound” An Imprint of Addison Wesley Longman Inc., 1998.
[2] J. A. Seidler, “Information System and Data Compression”, University of Salzburg, Boston / Dordreocht / London 1997.
[3] D.L. Donoho, M.Vetterli, R.A. DeVore, and I. Daubechies, “Data Compression and Harmonic Analysis”gd,, July 1998.
[4] Nelson, Mark, and Jean-Loup Gailly “The Data Compression Book” 2nd Edition, New York, M&T Books, 1996.
[5] Storer, James A., (ed.), " Data Compression" Conference, Los Alamitos, CA, IEEE Computer Society Press.1997.
[6] David Salomon, “Data Compression the Complete Reference” 2nd Edition, Spring Verlag, New York, 2000.

المستخلص

يعد النظام المقترح اسلوب جديد لضغط الملفات الصوتيه (audio files) وذلك عن طريق تحويل ملفات الصوت الى ملفات نصيه (text file) ومن ثم ضغط الملف الناتج الجديد بواسطة تقنيات الضغط الشائعه. للنظام المقترح اسلوبين (4-bit) و (6-bit) لتحويل ملف الصوت الى ملف نصي. وبأستخدام أسلوب (4-bit) كانت النتائج غير مرضيه مما حدا بالباحث استخدام اسلوب جديد هو (6-bit). تم اقتراح وتنفيذ خوارزمية ضغط ملفات الصوت بأستخدام CFG وذلك بعد تحويله الى ملف نصي ثم استخراج production rules والاستعانه بها بدلا من text مما حقق نسبة ضغط جيده 15% وفي بعض الحلات حقق نسبة ضغط 35% ولكن على حساب الوقت.