The IJICS (International Journal of Informatics and Computer Science)
Vol 5 No 1, March 2021, Page 29-36
ISSN 2548-8384 (online), ISSN 2548-8449 (print)
Available Online at https://ejurnal.stmik-budidarma.ac.id/index.php/ijics/index
DOI 10.30865/ijics.v5i1.2888

Performance Comparison Analysis of the Elias Delta Code Algorithm with the Even Rodeh Code Algorithm for Compressing Image Files

Elvia Hariska, Ega Yuliani, Surya Darma Nasution*

Prodi Teknik Informatika, Universitas Budi Darma, Medan, Indonesia
E-mail : [1]hariskae@gmail.com, [2]egayuliani7@gmail.com, [3,*]surya.darma.nasution1@gmail.com
Coressponding Author: surya.darma.nasution1@gmail.com
Submitted: **24/02/2021**; Accepted: **09/03/2021**; Published: **29/03/2021**

Abstract−A file or files have a large size, especially an image file. So that it can affect a storage memory and also mess in the speed of data transmission. If an image file has a large capacity it can affect the sending process. So that the sending process has two possibilities, namely the image file sent will require a long process or the image file will experience failure in sending because the storage size exceeds the maximum limit. This problem can be overcome by compressing the data. Data compression can reduce the size of data storage on both internal and external memory, speeding up data transmission due to smaller storage sizes. Data compression has several algorithms that can be used. In this research, the writer will compare the performance of Elias Delta Code algorithm with even Rodeh Code algorithm. Thus obtaining which algorithm is the most efficient used for the image file compression process.

Keyword: Compression; Image Files; Elias Delta Code; Even Rodeh Code; Performance Comparison

1. INTRODUCTION

In today's rapidly developing technology, file compression is very necessary and beneficial. This causes the use of storage media to increase while the storage space is limited so that the media cannot be stored in storage. Files with large storage sizes will take longer to transfer than files with small storage sizes. Therefore compression is needed to minimize file storage size.

Memory or storage capacity is an expensive component to obtain. So it requires a lot of funds if you want to get a memory, while the amount of storage capacity is decreasing every day because it is used every day to store data. In order for storage capacity to be more efficient in use, of course, tools are needed, namely compressing data that aims to reduce the size of files to be stored in storage.

Compression makes problem solving systematic and optimal by using algorithms that are in the internal compression itself, compression has a variety of algorithms so as to get more effective results, such as the Delta Code algorithm, Levenstein algorithm, and others[1].

Until now, there have been many studies that contain file compression. Among them is the research conducted by Ledi Vania Simanjuntak in 2020 regarding "Comparison of the Elias Delta Code Algorithm with Levenstein for Text File Compression" which has the result that the Elias Delta Code algorithm has the highest Space Saving of 6.75%[1]. Agus Adi Pramadi, Surya Darma Nasution, Bister Purba in 2019 regarding "Application of the Even Rodeh Algorithm in Image File Compression Applications" that image file compression can be designed and built using the Microsoft Visual Studio 2008 programming language[2]. Meanwhile, research conducted by Nadia Fariza Rizky, Surya Darma Nasution, Fadlina in 2020 on "Application of the Elias Delta Codes Algorithm in Text File Compression" explains that text files that have a larger size can be compressed into a smaller size[3].

Based on the above discussion, the authors are interested in conducting research to compare the Elias Delta Code Algorithm with the Even Rodeh Code Algorithm for compressing image files so as to know which algorithm is more effective in compressing the image file.

2. RESEARCH METHODOLOGY

2.1 Compression

Compression is the process of converting a set of data into a coded form to save storage space requirements and time for data transmission. Data compression is the process of encoding information using a bit or other information-bearing unit that is lower than the data representation that is not encoded with a particular encoding system[3]. Compression data is an art or sicience that represents information in a more concise form. Data can be in characters in text files, images or sequences of number genereted by other progresses. In the other words, using data compression, a certain file size can be reduced[4].

2.2 Elias Delta Code Algorithm

Elias Delta Code is a compression algorithm created by Peter Elias using the code he previously created, namely Elias Gamma Code, as a building block. In the Gamma code, Elias increases the length of the code in unary (α). In the next code, δ (delta), is added to the code length in binary (β). Thus, the Elias Delta Code, which is also for positive integers, is a little more complex to build[1].

The rules for encoding a number using the elias delta code are :

a. Write n in binary. The leftmost (most significant) bit will be 1.
b. Count the number of bits, remove the leftmost bit of n and add the calculation in binary to the left of n after the leftmost bit of n is removed.
c. Count the number of bits, remove the leftmost bit of n and add the calculation in binary to the left of n after the leftmost bit of n is removed.
d. Subtract 1 from the calculation in step 2 and add the number of zeros to the code.

When these steps are applied to the 17th integer, the result is: 17 = 10001 (five bits). Delete the leftmost 1 and add 5 = 101 for a result 101 | 0001. The three bits have been added, then add 2 zeros to get the delta code 00 | 101 | 0001[5].
Decode with elias delta code is done in the following steps:
Read the bit from the code until the decode process with gamma code elias is done. This process can be done in the following steps :

a. Count the number of leading zeros from the code then replace the calculation with C.
b. Check the left bit 2C + 1 (C zero, followed by 1, then the rest of the C bit). This is an elias gamma code decode M + 1.
c. Read the next bit M. Call this L.
d. The integer decoded is 2M + L[6].

2.3 Even Rode Code Agorithm

One of the known compression algorithms is the even rodeh code algorithm. This algorithm is a compression algorithm that was disseminated by Shimon Even and Michael Rodeh in 1978. The Even Rode Code algorithm is an algorithm used for data compression whose coding uses characters with several series of bits representing the character based on the frequency of each character[7]. The *even-rodeh code* is almost the same as the omega code, the difference is that the length of the code is sustained until the 3-bit length is reached and becomes the leftmost code group[2]. Here are the steps for compressing files using the even rode code algorithm.

a. Enter the data file
b. Performs a character value reading based on the frequency occurrence
c. Generating event code rode code
d. Calculate bits and sort based on the rodeh code event code
e. Grouping of bit strings
f. The results are compressed.

Below are the stages of building a rode event code with n as the index of the characters as follows :

a. Calculates the length of the bits
b. If the bit length is the value of n, it is converted to binary, by adding a 0 right in front of the binary value so that the bit becomes 3 digits
c. If the bit length is the value of n, it is converted into binary, by adding a 0 just behind the binary value so that the bit becomes 4 digits
d. If the bit length of the value n is converted into binary, by adding 0 to the right of the binary value then the number is added in front of the binary value as many as the number of digits in the binary value[2].

Table 1. Code of Omega Algorithm and Even-Rodeh Codes Algorithm

N	*Even-Rodeh Code*
0	000
1	001
2	010
3	011
4	100 0
7	111 0
8	100 1000 0
15	100 1111 0
16	100 10000 0
32	110 100000 0
100	111 1100100 0
1000	110 1100100 0

Even-Rodeh Code based on character variations:

Table 2. Even-Rodeh code of various characters

N	Number of Even-Rodeh Code bits
1	3

N	Number of Even-Rodeh Code bits
2-3	3
4-7	4
8-15	8
16-31	9
32-63	10
64-127	11
128-255	17
256-512	18

2.4 Image or Image

Image or image is a tool that humans use to convey a message to other humans. Image is a combination of points, lines, and fields and colors to produce an illustration with objects such as physical objects or what is called a human object[8]. Image or image is a very important component to convey messages objectively by visualizing it through the multimedia tools used. Images or images are also very different from other data such as documents or text[2].

3. RESULTS AND DISCUSSION

3.1 Problem Analysis

Problem analysis is a process to identify the causes and consequences of forming a system so that the system can run properly in accordance with the purpose of establishing the system. The problem raised from this research is to get a better compression algorithm between the Elias Delta Code algorithm and the Even-Rodeh Code algorithm which is influenced by the number of hexadecimal values that are in the file to be compressed. In the research, the files to be compressed use a file with the .jpg extension.

Figure 1. Image files to be compressed

3.2 Application of the Elias Delta Code Algorithm

At this stage, compress the file by inserting the image file to be compressed. The next step is to take a sample of the documents to be compressed. The value of 10 hexadecimal can be seen in table 3 below.

Table 3. 10 Hexadecimal Values

Hexa	Freq
FF	2
D8	1
E1	1
27	1
77	1
45	1
78	1
69	1
66	1
Total Hexadecimal	10

Based on the table above, the same hexadecimal values are obtained. Before the compression process, the first step is to read the hexadecimal values then create a hexadecimal value table sorted from the largest frequency value (the same hexadecimal value) to the smallest. The sequence of these values can be seen in table 4 below.

Table 4. Uncompressed Hexadecimal Value

n	Hex	Biner	bit	Frek	Bit x Freq
1	FF	11111111	8	2	16
2	D8	11011000	8	1	8
3	E1	11100001	8	1	8
4	27	00100111	8	1	8
5	77	01110111	8	1	8
6	45	01000101	8	1	8
7	78	01111000	8	1	8
8	69	01101001	8	1	8
9	66	01100110	8	1	8
				Total	80

Counting bits and sorting Elias Delta Code code and obtaining compressed bit file. The compression process can be seen in table 5 below:

Table 5. Elias Delta Code code sequencing

n	Hex	Freq	*Codeword*	Bit	Freq x Bit
1	FF	2	1	1	2
2	D8	1	0100	4	4
3	E1	1	0100	4	4
4	27	1	01100	5	5
5	77	1	01101	5	5
6	45	1	01110	5	5
7	78	1	01111	5	5
8	69	1	00100000	8	8
9	66	1	00100001	8	8
				Total	46

The resulting bit string is "1010001000110001101011100111100100000001000001" with a total of 46 bits.
Padding
46 mod 8 = 6 = N
7 – N + "1"
7 – 6 + "1" = 01
Flagging
9 – N
9 – 6 = 3 = 0011
So that the bit string formed after the addition of padding and flagging is "1010001000110001101011100111100100000001000001010011" so that the total bit length is 52.
Decoding
The last 8 bits = 01010011 => 83
7 + n
7 + 83 = 90
From the compression results with the Elias Delta Codes algorithm above, the compression performance can be calculated, namely :
TB = 46 + 8
= 54 bit
a. *Ratio of Compression (Rc)*
$Rc = \frac{carving\ data\ before\ compresed}{carving\ data\ after\ compresed}$
$Rc = \frac{80\ bit}{54\ bit}$
Rc = 1,481
b. *Compression Ratio (CR)*
$Cr = \frac{carving\ data\ after\ compresed}{carving\ data\ before\ compresed}$ x 100%
$Cr = \frac{54\ bit}{80\ bit}$ x 100%
Cr = 67,5 % bit
c. *Redudancy (Rd)*
$Rd = \frac{carving\ data\ before\ compresed - carving\ data\ after\ compresed}{carving\ data\ before\ compresed}$ x 100%
$Rd = \frac{80\ bit - 54\ bit}{80\ bit}$ x 100%
$Rd = \frac{26\ bit}{80\ bit}$ x 100%

Rd = 32,5 %
Space Saving (Ss)
Ss = (1 – Compression Ratio) x 100%
Ss = (1 – 67,5 %) x 100%
Ss = 66,5 x 100%
Ss = 6,650 %

3.3 Implementation of Even-Rodeh Code Algorithm

Before compressing an image file, an analysis of the image file that will be compressed must be carried out, and in analyzing the image file, a sample image file must be taken. Here are the steps for compressing and decompressing image files using the Even-Rodeh Codes algorithm.

a. Input image file

In compressing image files, you must first include the image files to be compressed.

Figure 2. A compressed image file

b. Reads hexadecimal value and frequency occurrence

The next step is to read the hexadecimal value, in each image file there is a hexadecimal value. In this process, 10 hexadecimal values are generated as sample images to be analyzed. The hexadecimal value can be seen in the image below:

Figure 3. Sample hexadecimal values

The following are the results of image sampling that will be compressed based on the hexadecimal value which will be shown in table 3 below:

Table 6. Value of 10 Hexadecimal samples

FF	D8	FF	E1	27	77	45	78	69	66

Based on the table above, there is a hexadecimal value for the sample image file. For the purposes of manual counting, only take a sample of 10 characters from the hexadecimal value of the sample image file. The hexadecimal values for the sample image files are FF, D8, FF, E1, 27, 77, 45, 78, 69, 66. Next is grouping the hexadecimal values based on frequency values. The sequence of the headesimal value characters can be seen in the following table :

Table 7. Hexadecimal value reading and frequency

Hexadesimal	Frek
FF	2
D8	1
E1	1
27	1
77	1
45	1
78	1

Hexadesimal	Frek
69	1
66	1
Total *Hexadesimal*	10

Based on the table above, the same hexadecimal values are obtained. Before the image file compression process, the first step is to read the hexadecimal value then create a hexadecimal table sorted from the largest frequency to the smallest frequency. The sequence of hexadecimal values can be seen in the table below :

Table 8. Uncompressed hexadecimal

N	*Hexadesimal*	Biner	Bit	Frek	Bit x Frek
0	FF	11111111	8	2	16
1	D8	11011000	8	1	8
2	E1	11100001	8	1	8
3	27	00100111	8	1	8
4	77	01110111	8	1	8
5	45	01000101	8	1	8
6	78	01111000	8	1	8
7	69	01101001	8	1	8
8	66	01100110	8	1	8
				Total	80

Based on the table above, one hexadecimal value is worth eight bits of a binary number. So that 10 samples of hexadecimal values have a binary value of 80 bits.

c. Even-Rodeh Codes coders

The compression of this image file uses the Even-Rodeh Codes algorithm. This compression is done by multiplying the hexadecimal frequency value by the bit frequency of the Even-Rodeh Codes. The rules for forming the Even-Rodeh Codes code can be seen in the analysis and discussion in the previous chapter. The values of the Even-Rodeh Codes are as follows :

Table 9. Even-Rodeh Codes

N	*Even-Rodeh Codes*
0	000
1	001
2	110
3	011
4	100 0
7	111 0
8	100 1000 0
15	100 1111 0
16	100 10000 0
32	110 100000 0
100	111 1100100 0
1000	110 1100100 0

d. Counting image bits and even-rodeh code sequencing and obtaining compressed file bits.

This compression process is to compress the value of the hexadecimal sample with the Even-Rodeh Codes code value. The compression process can be seen in the table below:

Table 10. Compression Process with Even-Rodeh Codes code values

N	Hexadesimal	Frek	Codeword	Bit	Frek x Bit
0	FF	2	000	3	6
1	D8	1	001	3	3
2	E1	1	010	3	3
3	27	1	011	3	3
4	77	1	1000	4	4
5	45	1	101 0	4	4
6	78	1	110 0	4	4
7	69	1	111 0	4	4
8	66	1	100 1000 0	8	8
				Total Bit	39

Based on the table above, a new compressed bit value can be formed from the hexadecimal value arrangement of the sample before compression, namely FF, D8, FF, E1, 27, 77, 45, 78, 69, 66.

e. Gets the bit string and added padding and flagging .

This process is to convert a hexadecimal value into a binary value bit string as shown below. "000001010011100010101100111010010000". The padding and flagging bits are added first before the compression results are written into the file. The addition of padding and flagging bits is done by referring to the remaining number of bits divided by 8. The number of compression results is 39 bits, because 39 is not divisible by 8, it can be obtained padding in the form of bits, namely "" so that it is divisible by 8.

The resulting bit string is "000001010011100010101100111010010000" with a total of 39 bits.

Padding
39 mod 8 = 7 = N
7 – N + "1"
7 – 7 + "1" = 1

Flagging
9 – N
9 – 7 = 2 = 0010

So that the bit string formed after the addition of padding and flagging is "00000101001110001010110011101001000010010" so that the total bit length is 41.

Decoding
The last 8 bits = 00010010 => 18
7 + n
7 + 18 = 25

From the compression results with the Even-Rodeh Codes algorithm above, the compression performance can be calculated, namely :

TB = 39 + 8
= 47 bit

a. *Ratio of Compression (Rc)*

$Rc = \frac{carving\ data\ before\ compresed}{carving\ data\ after\ compresed}$

$Rc = \frac{80\ bit}{47\ bit}$

Rc = 1,702

b. *Compression Ratio (CR)*

$Cr = \frac{carving\ data\ after\ compresed}{carving\ data\ before\ compresed}$ x 100%

$Cr = \frac{47\ bit}{80\ bit}$ x 100%

Cr = 58,75 bit

c. *Redudancy (Rd)*

$Rd = \frac{carving\ data\ before\ compresed\ - carving\ data\ after\ compresed}{carving\ data\ before\ compresed}$ x 100%

$Rd = \frac{80\ bit - 47\ bit}{80\ bit}$ x 100%

$Rd = \frac{33\ bit}{80\ bit}$ x 100%

Rd = 41,25 %

Space Saving (Ss)

Ss = (1 – *Compression Ratio*) x 100%
Ss = (1 – 58,75) x 100%
Ss = 57,75 x 100%
Ss = 5,775 %

3.4 Implementation of the Exponential Method

In calculating and comparing the compression process of the two algorithms:

a To determine an alternative to analyze the speed comparison between the Elias Delta Code algorithm and the Even Rodeh Code algorithm in compression, it is necessary to determine which algorithm will be used as the compression algorithm.

b Determine the criteria for comparing the two algorithms. Furthermore, determining the criteria in analyzing the process and how it works. The criteria can be seen in the following table :

Table 11. Determination of Criteria

Criteria	Information
Ratio Of Compression (RC)	The comparison value between the data bit size before being compressed with the data bit size that has been

Criteria	Information
	compressed.
Compression Ratio (CR)	Percentage comparison between uncompressed data and uncompressed data.
Redundancy (RD)	The results of the assessment of the ratio value with the results of the compression ratio value.
Space Saving (SS)	The difference between uncompressed data and the size of the compressed data.

c. Assignment of each criterion that has been established. This value is taken based on the analysis of the Elias Delta Code algorithm and the *Even Rodeh Code algorithm*.
d. Determine the results or priority decisions based on the value of each alternative. The results of the priority decisions can be seen in the table below:

Table 12. Priority Decisions

Alternative	RC	CR	SS	Rank
Elias Delta Algorithm	1,481	67,5%	6,650%	1
Even Rodeh Algorithm	1,702	58,75%	5,775%	2

The table above explains that the alternative that has the highest total value (%) will be ranked first, this is because the bigger (SS) the compression is getting better and faster. Based on this analysis, the Elias Delta Code algorithm is the best algorithm in performing the compression process.

4. CONCLUSION

Based on the research that has been done on file compression using a comparison technique that was built, it can be concluded from the trials that have been done using the Elias Delta Code algorithm and the Even Rodeh algorithm, the program can compress files that have a .jpg extension. A file comprsession program that uses the hxd application to get hexadecimal values is quite helpful in the compression process. The compression ratio value is affected by the contents of the compressed file. The more repetitions of characters in a compressed file, the higher the compression ratio. From the explanation above, it can be concluded that the Elias Delta Code algorithm is the most efficient in compressing image files.

REFERENCES

[1] L. V. Simanjuntak, "Perbandingan Algoritma Elias Delta Code dengan Levenstein Untuk Kompresi File Teks," *J. Comput. Syst. Informatics*, vol. 1, no. 3, pp. 184–190, 2020.
[2] A. A. Pramadi, S. D. Nasution, B. Purba, A. Event, and R. Code, "PENERAPAN ALGORITMA EVEN-RODEH PADA APLIKASI KOMPRESI FILE," *KOMIK (Konferensi Nas. Teknol. Inf. dan Komputer)*, vol. 3, pp. 73–84, 2019, doi: 10.30865/komik.v3i1.1570.
[3] N. F. Rizky and S. D. Nasution, "Penerapan Algoritma Elias Delta Codes Dalam Kompresi File Teks," *Build. Informatics, Technol. Sci.*, vol. 2, no. 2, pp. 109–114, 2020.
[4] G. A. NABILA, "ANALISIS PERBANDINGAN ALGORITMA BOLDI VIGNA ζ1 CODE DAN ALGORITMA EVEN-RODEH CODE PADA KOMPRESI FILE TEKS," UNIVERSITAS SUMATERA UTARA MEDAN, 2019.
[5] L. Marlina, A. Putera, U. Siahaan, H. Kurniawan, and I. S. Sulistianingsih, "Data Compression Using Elias Delta Code," *Int. J. Recent Trends Eng. Res.*, pp. 210–217, 2017, doi: 10.23883/IJRTER.2017.3406.TEGS6.
[6] F. Jaya and I. Harefa, "Penerapan Algoritma RC6 dan Algoritma Elias Delta Code Pada Aplikasi Pengamanan dan Kompresi Short Message Service (SMS) Berbasis Android," *J. Inf. Sist. Res.*, vol. 1, no. 3, pp. 155–161, 2020.
[7] M. H. D. A. L. I. Subada, "ANALISIS PERBANDINGAN ALGORITMA EVEN-RODEH CODE DAN ALGORITMA FIBONACCI CODE UNTUK KOMPRESI FILE TEKS," UNIVERSITAS SUMATERA UTARA, 2018.
[8] M. Simangunsong, "Perbandingan Algoritma Elias Delta Code Dan Unary Coding Dalam Kompresi Citra Forensik," *RESOLUSI Rekayasa Tek. Inform. dan Inf. ISSN*, vol. 1, no. 1, pp. 18–26, 2020.