TEXT FILE COMPRESSION USING HYBRID RUN LENGTH ENCODING (RLE) ALGORITHM WITH EVEN RODEH CODE (ERC) AND VARIABLE LENGTH BINARY ENCODING (VLBE) TO SAVE STORAGE SPACE

S M Hardi*, M Zarlis2, D R P Lubis3, P Sihombing4, M Elveny5

1,2,3,4,5 Faculty of Computer Science and Information Technology, Universitas Sumatera Utara, Medan 20155, Indonesia

*Email: vani.hardi@usu.ac.id

Abstract. The increase of data usage causes problems in data storage, indirectly making the need for data storage also to increase. One alternative solution that can be done is to compress the file so that the file becomes smaller in size so it saves storage space. The algorithm used in this research is the Run Length Encoding algorithm, the Even Rodeh Code algorithm, and the Variable Length Binary Encoding algorithm which are the types of lossless compression. The algorithm will calculate its performance based on Compression Ratio, Ratio of Compression, Redundancy, Compression Time, and Decompression Time. The file that will be used in the data compression process is the file extension *.txt. This study used homogeneous strings (strings that have the same character) and heterogeneous strings (strings that have different characters) in testing the algorithm. In the compression process with a homogeneous string, the combination of the Run Length Encoding algorithm with the Variable Length Binary Encoding algorithm is better than the combination of the Run Length Encoding algorithm and the Even Rodeh Code algorithm with a Compression Ratio of 18.84% and a decompression time of 0.01295 ms. While the compression process on heterogeneous strings from the combination of the Run Length Encoding algorithm with the Even Rodeh Code algorithm is better than the combination of the Run Length Encoding algorithm with Variable Length Binary Encoding algorithms with Compression Ratio of an average of 52.45% and fewer decompression times of 4.93002 ms

1. Introduction

The increase of data usage causes problems in data storage, indirectly making the need for data storage also to increase. The larger the file size, the greater the storage space needed. The file can be text, image, sound, or video. In sending files through the transmission media, the time needed for the process of sending data is getting longer. One alternative solution that can be done is to compress the file so that the file becomes smaller in size so that the storage space becomes bigger and the process of sending files will be faster and save the time. Data compression is a process that can convert an input data stream (original data) into another data stream (compressed data) that has a smaller size [2]. Some examples of algorithms in data compression are Run Length Encoding algorithm, Even-Rodeh Code algorithm, Variable Length Binary Encoding algorithm. The Run Length Encoding (RLE) algorithm is an algorithm that utilizes the same sequence of characters. The Even-Rodeh Code algorithm is a compression algorithm developed by Shimon Even and Michael Rodeh. The Even-Rodeh Code algorithm is almost the same as Elias Omega Code, but the main difference is that it will add 3 bits in length and become the leftmost group of codes [3]. The Even-Rodeh Code Algorithm is a data compression algorithm that encodes each character using several series of bits. Variable Length Binary Encoding (VLBE) algorithm is a binary encoding that can be used for storing or transmitting information. Variable Length Binary Encoding (VLBE) is a code that maps source symbols for a number of bit variables. Variable-length codes can allow sources to be compressed with zero errors (lossless
data compression) and can still be read back by symbols [3]. These algorithms will only be implemented to compress ASCII (*.Txt) text, because the file is one of the standard files commonly used to compare compression algorithms. Based on the background and previous research, the authors are interested in combining the Run Length Encoding algorithm with the Even Rodeh Code algorithm and combining the Run Length Encoding algorithm and the Variable Length Binary Encoding algorithm to find out the best performance of the Run Length Encoding algorithm combination with both algorithms.

2. Method

2.1 Lossless compression
The compression algorithm is said to be lossless when the compressed data can be returned to the original data without reducing the information contained in the data [1]. Lossless compression is also called reversible compression because the original data can be returned perfectly. However, the compression ratio is very low, for example in images such as GIF and PNG. Examples of suitable data are medical images, texts, programs, spreadsheets, and others. Examples of lossless algorithms in data compression are Arithmetic Coding, Huffman Coding, Variable Length Binary Encoding, Even-Rodeh Code, etc.

2.2 Lossy Compression
A compression algorithm is said to be lossy when the data experience little or a lot of data loss during compression [1]. Lossy compression is also called irreversible compression because the original data cannot be returned perfectly. Therefore, it is very unlikely that the data compressed with Lossy techniques will be returned like it was before compressed or into the original data. Examples of the data are images, sound, and video. Examples of lossy algorithms in data compression are Wavelet Compression, Fractal Compression, Wyner-Coding (WZC) and etc.

2.3 Run Length Encoding Algorithm
Run Length Encoding Algorithm is an algorithm that performs compression by utilizing characters that repeat in sequence. So that the number of repetitive characters in the data is a determinant of the success of the compression algorithm Run Length Encoding. Each repetitive character sequence needs to add a marker (marker byte) which serves to avoid confusion during decoding.

2.4 Even Rodeh Code Algorithm
The Even-Rodeh Code algorithm is a compression algorithm developed by Shimon Even and Michael Rodeh in 1978 [3]. The Even-Rodeh Code Algorithm is a data compression algorithm that encodes each character using several series of bits.
The steps taken to do compression using the Even Rodeh Code algorithm are as follows:
1. Calculate the length of the bit.
2. If the bit length is \(0 \leq n \leq 3\) then the \(n\) value is changed to binary, add 0 in front of the binary value so that the bit becomes 3 digits.
3. If the bit length is \(4 \leq n \leq 7\) then the \(n\) value is changed to binary, add 0 behind the binary value so that the bit becomes 4 digits.
4. If the bit length is \(n \geq 8\) then the value of \(n\) is changed to binary, add 0 behind the binary value then the number is added to the binary value as many as the number of digits of the binary value.

2.5 Variable Length Binary Encoding Algorithm
Variable Length Binary Encoding (VLBE) Algorithm is a data compression algorithm that encodes each character by sorting characters based on frequency and sorted in a stable manner to form several series of bits.
The steps taken to do compression using the Variable Length Binary Encoding algorithm are:
1. The binary initial value starts from 0 as root for the next binary formation.
2. The previous binary initial value is added with number 1, each number 1 is placed in front of the binary initial value.
3. If binary n is the last, the initial binary value is changed to number 1.

3. Test and Result
The following is an overview of system described through flowchart as shown in Figure 1 below.

![Flowchart System](image)

Figure 1. Flowchart system

the results of application testing performed on a combination of the Run Length Encoding Algorithm with the Even Rodeh Code Algorithm and the Run Length Encoding Algorithm with the C# programming language.

3.1 Compression Processing Time on Homogeneous Strings
The results of the compression process time on homogeneous strings obtained based on the experiment of the 2 character plaintext characters, character A and character B, with the string length of 100000 characters, 200000 characters, 300000 characters, 400000 characters, and 500000 characters as shown in a graph in Figure 2.

![Compression Time Graph](image)

Figure 2. Compression Time Graph on Homogeneous Strings
From the graph in Figure 2, it can be concluded that the Compression Time result from Homogeneous Strings compression with a combination of the Run Length Encoding algorithm and the Variable Length Binary Encoding algorithm is better than the combination of the Run Length Encoding algorithm and the Even Rodeh Code algorithm.

3.2 Compression Processing Time on Heterogeneous Strings

The results of the compression process time on heterogeneous strings obtained based on the experiment of the 10 character plaintext characters, A, B, C, D, E, F, G, H, I, and J with the string length of 100000 characters, 200000 characters, 300000 characters, 400000 characters, and 500000 characters as shown in a graph in Figure 3.

![Figure 3 Compression Time Graph on Heterogeneous Strings]

From the graph in Figure 3, it can be concluded that the Compression Time resulting from the Heterogeneous Strings compression with the combination of the Run Length Encoding algorithm and the Even Rodeh Code algorithm is better than the combination of the Run Length Encoding algorithm with Variable Length Binary Encoding algorithms.

3.3 Decompression Processing Time on Homogeneous Strings

The results of the decompression process in homogeneous strings obtained based on the experiment of the 2 plaintext characters, characters A and character B, with the string length of 100000 characters, 200000 characters, 300000 characters, 400000 characters, and 500000 characters as shown in a graph in Figure 4.

![Figure 4. Decompression Time Graph on Homogeneous Strings]
From the graph in Figure 4, it can be concluded that Decompression Time result from Homogeneous Strings compression with a combination of Run Length Encoding algorithm and Variable Length Binary Encoding algorithm is better than the combination of Run Length Encoding algorithm and Even Rodeh Code algorithm.

3.4 Decompression Processing Time on Heterogeneous Strings

The results of the decompression process on heterogeneous strings obtained based on the experiment of the number of 10 character plaintext characters, A, B, C, D, E, F, G, H, I, and J with the string length of 100000 characters, 200000 characters, 300000 characters, 400000 characters, and 500000 characters as shown in a graph in Figure 5

![Decompression Time Graph on Heterogeneous Strings](image)

From the graph in Figure 4, it can be concluded that the Decompression Time from the compression of the Heterogen String with the combination of the Run Length Encoding algorithm and the Even Rodeh Code algorithm is better than the combination of the Run Length Encoding algorithm with the Variable Length Binary Encoding algorithm

4. Conclusion

1. Algorithm Run Length Encoding is effective when combined with other algorithms on homogeneous strings. In the process of compression and decompression, the combination of the Run Length Encoding algorithm with the Even Rodeh Code algorithm and the combination of the Run Length Encoding algorithm with the Variable Length Binary Encoding algorithm are influenced by the number of variations in character.

2. The test results of compressing *.txt text with homogeneous strings (strings that have the same character) based on the variable Compression Ratio (Cr), Ratio of Compression (Rc), Redundancy (Rd) and compression time indicate that the combination algorithm Run Length Encoding with the Variable Length Binary Encoding algorithm has a Compression Ratio of 18.84% better than the combination of the Run Length Encoding algorithm with the Even Rodeh Code algorithm with Compression Ratio an average of 37.57%

3. The test results of compressing *.txt text with heterogeneous strings (strings that have different characters) based on the variable Compression Ratio (Cr), Ratio of Compression (Rc), Redundancy (Rd) and compression time indicate that the combination of Run Length Encoding algorithm and algorithm The Rodeh Code event has a Compression Ratio of 52.45% better than the combination of the Run Length Encoding algorithm with a Variable Length Binary Encoding algorithm with a Compression Ratio of 53.74% on average.
4. The test results of decompressing *.txt text files with homogeneous strings (strings that have the same character) indicate that the combination of the Run Length Encoding algorithm with Variable Length Binary Encoding algorithms requires an average time of 0.01295 ms less than the combination of the Run Length Encoding algorithm with Even the Rodeh Code algorithm in returning the compressed text file to the original text file, with an average time of 0.01793 ms.

5. The test results of decompressing *.txt text with heterogeneous strings (strings that have different characters) show that the combination of the Run Length Encoding algorithm with the Even Rodeh Code algorithm requires an average time of 4.93002 ms less than the combination of the Run Length Encoding algorithm with a Variable Length Binary Encoding algorithm in returning the compressed text files to the original text file, with an average time of 5.17042 ms.

References

[1] Gozali, F. & Mervyn. 2004. Analisis Perbandingan Kompresi Data dengan Teknik Arithmetic Coding dan Run Length Encoding. JETri 4(1): 37-52

[2] Salomon, David. 2007. Data Compression The Complete Reference. Third Edition. New York: Springer-Verlag

[3] Salomon, D. & Motta, G. 2010. Handbook of Data Compression. Fifth Edition. Springer: London.