Analysis for Lossless Data Compression Algorithms for Low Bandwidth Networks

Wogderes Semunigus*, and Balachandra Pattanaik
Department of Electrical and Computer Engineering, Faculty of Engineering and Technology, Bule Hora University, Bule Hora, Ethiopia.
Email: *wogmust@gmail.com

Abstract. Analysis for loss less data compression delivers the relevant data about variations of them as well as to describe the possible causes for each algorithm and best performing data types. It describes the basic lossless techniques of data compression Huffman encodes, Arithmetic Encoding, and Lempel Ziv Encodings then briefly with their effectiveness under varying data types of Latin text, audio and video. These properties give the solution of which lossless compression algorithm more suitable compared to other from the Saving Percentage, compression ratio, time of compression and time of decompression with Low Bandwidth Network. Moreover here Lossless Data Compression Algorithms (LDCA) being implemented and tested Huffman compression, Arithmetic compression, and Lempel Ziv algorithms, the implemented result shows that LZW algorithm saves more size than that of the others two with text file, Huffman compression algorithm saves more file sizes and the time takes to compressed decompress is higher than that of other two for audio file type and finally Huffman performs greater on very huge data compressions that is due to much compressing capability.

Keywords: Saving Percentage, Compression Factor, Compression Ratio, Time of Compression and Lempel Ziv algorithms

1. Introduction
Data compression is the most required concept in the universal issue; source of coding otherwise bit-rate decrease techniques implicates encoding data applying less bits than the most input data. Compressing an existing file to partial of its real size is corresponding to exact replication the capability of the storing data middling [1]. It can then develop reasonable to collection the data at a developed equal, hence quicker, equal of the storing order also decrease the capacity on the input/output channels [2]. This technique can be also dropping. Lossless compression decreases bits through recognizing also excluding arithmetical termination [3-4]. No data is missing in lossless compression. Lossy compression decreases bits through recognising redundant data also confiscating it.

2. Related Works
According to Khalid Sayood paper tries to equate among (RLE also Huffman) processes that are no compression procedures empty texts, consistent with the normal file size. Distribute the assessment [5] among the real file sizes also file size afterward compression exhausting (RLE & HUFFMAN) procedures. The reading shows to the efficiency of the Huffman procedure [6,7]. Good compression whereas while choosing the two techniques, they have big variation gap of compressed data of them text file so that they must include arithmetic encoding techniques to compare effectively. And the better way to have better comparison mode mat lab is advised [8]. Huffman as well as arithmetic coding are equating consistent with their functions. Data compression is a procedure which decreases the data size, eliminating the extreme data. Here et al have not a figure for how their techniques compresses and
decompresses for answering this it is better to use mat lab programming just to have good output of plotting. And finally, they don’t include parameters like compression time and decompression time [9].

j-bit encoding (JBE) operates every bit of information private file to diminish the dimension deprived of trailing some data next deciphering that is categorised to lossless compression [11, 13]. This fundamental procedure is planned to be joining with other data compression procedures to improve the compression ratio. The presentation of this procedure is dignified via equating combination of dissimilar data compression procedures [14]. But this paper includes Run length algorithm that lower the combined compression algorithm which known to be JBE. This investigation efforts to report this request though unmoveing preserving realistic encoding periods [15]. Current subdivision also assemblage examination is applied by code vectoring systems then other improvement examples to recover value also concert inside the next group coding. This investigation saying on regular a 50% reduction in run- time through the encoder with bordering reductions in professed superiority. In this paper considers small size video [12].

3. Measuring Compression Performances Techniques

![System Model Diagram]

Figure 1: System model

Measuring Compression Performances techniques deals lossless data techniques of Huffman, Lempel Ziv Encoding and Arithmetic encoding with parameters of Saving percentage, hardware cost, Compression time and decompression time using excel, for the Saving percentage and Compression ratio, it implemented by using c++ and MATLAB tool [10].

There are numerous standards to assess the function of a compression technique. Though, in anxiety has continually been the space effectiveness as well as time effectiveness. Resulting are several evaluations applied to measure the functions of loss less procedure as shown in figure 1. Saving percentage: it computes the reduction of the sender file.

4. Huffman Algorithms

This type of compression is valuable while annoying to pack additional things on a disk desirable to copy/transmit a file via a network. For example, JPEG, MPEG, or MP3, are a compression arranges, it switches a specific kind of data file. They incline to revenue benefit of recognized structures of such kind of data to compress. Other tools for example, compress, zip, as well as programs similar Zip It can be applied to compress some category of file. Approximately the compression procedures like MPEG,
JPEG are loss—unzipping the compressed outcome doesn't reinvent unspoiled copy of the real. This type of technique compresses through summarizing the data. For video, sound, as well as images, such roughness can be satisfactory since the whole data is sustained also a little wasted pixel. For transcript data, however, a loss procedure typically isn't suitable.

5. Results and Discussion

From the figure 2 above this concludes that compression time for arithmetic compression is very high that of the other two compression algorithms and LZW is good. This is because of increment of file size thus arithmetic compression is based on probability of bit occurrences. Here also based on compression ratio parameter LZW algorithm performs better than the two arithmetic is worst.

![Figure 2: result for 18036byte file size English Text test](image)

It shows the result of ten different tests having varying file sizes for Huffman, arithmetic and Lempel Ziv Welch (LZW) algorithms. The tests are based on a parameter’s compression factor, compression ratio, saving percentage, time of compression as well as time of decompression for all tests.

![Figure 3: result for saving percentage parameter test](image)

The figure 3 shows that the percentage of size saved from the original file size using the three lossless algorithms. From this can conclude that LZW algorithm saves more size than that of the others. Thus for smaller file sizes and bigger file sizes an LZW algorithm is better.
Table 1: Result of all six input images compression factor

| No_ | Image name | size  | Huffman coding | Arithmetic coding | LZW coding |
|-----|------------|-------|----------------|-------------------|------------|
|     |            |       |                |                   | dictionary size=4096 |
| 1   | Baby       | 32x32 | 1.4006         | 1.2132            | 1.0343     |
| 2   | Bird       | 48x48 | 1.6250         | 1.6837            | 1.5329     |
| 3   | Aircraft   | 64x64 | 1.2846         | 1.0093            | 1.8576     |
| 4   | Veggie     | 128x128 | 1.0164 | 1.0212 | 2.1874 |
| 5   | Pet        | 160x160 | 1.0855 | 1.1379 | 3.4409 |
| 6   | Night      | 240x240 | 1.3454 | 1.1169 | 1.8954 |

Based on table 1 a parameter compression ratio LZW compression algorithm is averagely better than that of Huffman and arithmetic compression algorithms. LZW algorithm performs better result on image sizes of 64x64, 128x128, 160x160 and 240x240. Thus compression ratio for a single frame of a video is about 0.133, compression factor is 7.5 and saving percentage is about 86% of original data size which is grouped as a very good compression time.

![Comparison of Huffman, LZW and arithmetic video](image)

**Figure 4:** Comparisons of Huffman LZW and arithmetic video frame

6. Arithmetic Video Compression Result
Here figure 4 based on the parameters used for comparison, arithmetic compression has better saving percentage, compression factor and compression ratio and to put a point compression based on arithmetic calculation is performs better. And in other point of view since Huffman compression scheme does on removing redundant bit thus if the frame is composing don single colour it performs better than that of arithmetic compression.

7. Conclusion
Many images, text and video lossless compression algorithms are proposed and implemented. Here in this algorithm explained in detail, implemented, and tested using different input types like text, image, audio and video file types with different file sizes. Thus, arithmetic data compression technique is performed better of text file type, and LZW encoding performs good on highly redundant bit sequence, Huffman performs greater on ridiculously huge data compressions that is due to much compressing capability. LZW algorithm saves more sizes than that of the other two with text file, Huffman compression algorithm saves more file sizes, and the time takes to compress and decompress is higher than that of
other two for audio file type. This paper concludes that all data compression algorithms have their own advantage or disadvantage, but they will perform better as respective task. Fora large size data their execution time is highly increasing. Thus, only focus on their compression ratios, saving percentage, compression time, compression factor, and decompression time.

References
1. Perkins, C., Belding-Royer, E., & Das, S. (2003). Ad hoc On-Demand Distance Vector (AODV) Routing: RFC Editor.
2. Morales-Sandoval, M. (2004). Hardware architecture for elliptic curve cryptography and lossless data compression. Computer Science Department National Institute for Astrophysics, Optics and Electronics Tonantzintla. Puebla México.
3. Kumar, MJ& Baskaran, R 2015, ‘Performance Analysis and Comparison of Congestion Control in Wired and Wireless Environment’, International Journal of Control Theory and Appliciations, vol. 8, no.5, pp. 1743-1757.
4. Nancy, V. (2018). A Security for MANET Interruption Recognition & Preclusion approaches for Network Layer Attacks. International Journal of Applied Engineering Research, 13(12), 10702-10706.
5. Cheng, Y., Huang, C., & Shi, W. (2007, September). Trusted dynamic source routing protocol. In 2007 International Conference on Wireless Communications, Networking and Mobile Computing (pp. 1632-1636). IEEE.
6. Sinha, S. R., Khatri, P., & Gupta, S. (2016, April). Improvisation of qos using compression on scalable multimedia manet. In 2016 International Conference on Computing, Communication and Automation (ICCCA) (pp. 645-650). IEEE.
7. Kumaki, T., Kuroda, Y., Koide, T., Mattausch, H. J., Noda, H., Dosaka, K., ... & Saito, K. (2005, May). CAM-based VLSI architecture for Huffman coding with real-time optimization of the code word table [image coding example]. In 2005 IEEE International Symposium on Circuits and Systems (pp. 5202-5205). IEEE.
8. Kumar, MJ& Baskaran, R (2017), Analyzing a Personalized Network System through NetFlow’, Institute of Integrative Omics and Applied Biotechnology (IIOAB), 7(7), pp. 1-7.
9. Huffman, K. (1991). Profile: David A. Huffman. Scientific American, 54-58.
10. Gawthrop, P. J., & Wang, L. (2005). Data compression for estimation of the physical parameters of stable and unstable linear systems. Automatica, 41(8), 1313-1321.
11. Suarjaya, I. M. A. D. (2012). A new algorithm for data compression optimization. arXiv preprint arXiv:1209.1045.
12. Jan, Z., Azeem, N., & Zahoor, F. (2015, November). Digital watermarking using combination of Ranklets and wavelets. In2015 First International Conference on Anti-Cybercrime (ICACC)(pp. 1-5). IEEE.
13. Kwong, S., & Ho, Y. F. (2001). A statistical Lempel-Ziv compression algorithm for personal digital assistant (PDA). IEEE Transactions on Consumer Electronics, 47(1), 154-162.
14. Mehta, R. (2020). Optimal Huffman Coding Performance of Ad-Hoc Networks Based on Cross-Layer Design. Journal of Information Science & Engineering, 36(6).
15. Niu, D., Zhang, Y., Zhao, Y., & Yang, M. (2009, December). Research on Routing Protocols in Ad Hoc Networks. In 2009 International Conference on Wireless Networks and Information Systems (pp. 27-30). IEEE.