Journal of Techniques, ISSN: 2708-8383, Vol. 5, No. 1, 2023, Pages 52-57
DOI: https://doi.org/10.51173/jt.v5i1.1059

JOURNAL OF TECHNIQUES

Journal homepage*: http://journal.mtu.edu.iq*

RESEARCH ARTICLE - ENGINEERING

Hybrid Lossless Compression Techniques for English Text

Jannat Tariq [1*], Mahmood F. Mosleh [1], Maha Abdulameer [2], Huthaifa A. Obeidat [3], Omar A. Obeidat [4]

[1] Electrical Engineering Technical College, Middle Technical University, Baghdad, Iraq

[2] Middle Technical University, Baghdad, Iraq

[3] Department of Communications and Electronics Engineering, Jerash University, Jerash, Jordan

[4] College of Engineering, Wayne State University, Detroit, Michigan, MI 48202, USA

[*] Corresponding author E-mail: bbc0052@mtu.edu.iq

Article Info.

Article history:

Received
01 November 2022

Accepted
08 January 2023

Publishing
31 March 2023

Abstract

Since the demand for data transfer and storage is always increasing, sending data in its original form will take a long time to send and receive. Compression is an important issue for digital communications systems because it imposes an important rule while reducing complexity and power requirements. The goal of compression is to reduce the file size without compromising the quality of the information, which leads to more capacity saving and reduces the required bandwidth in terms of the communications system. This paper proposes a system that consists of a hybrid of two lossless techniques, including a concatenation of Huffman and LZ4 in order to enhance the traditional techniques. The result of the proposed system demonstrates that the proposed combination techniques reduce the file size significantly, achieving between 73.649 % and 79.708 % in terms of average saving ratio (SR). The above would give us credible, cost-effective, and affordable lossless encoding systems for electronic communication systems.

Keywords: Compression; LCT; CR; SR.

1. Introduction

Handling the increasing volume of data generated by modern day-to-day activities is not an easy task for symmetric communications. According to [1], approximately 2.5 quintillion bytes of data are generated daily. This quantity of data is quite a lot for conventional computing systems to manage. Major corporations are producing an abundance of hardware in an effort to provide a better method for dealing with huge quantities of data; nevertheless, it is nearly impossible to store this data without compression. Data Compression (DC) aims to decrease the file size so that it demands less storage capacity and less transmission bandwidth across data communication channels [2]. By reducing the amount of data to be transmitted over error prone data communication channels, DC reduces the number of errors that occur during data transmission [3]. In wireless communication devices, DC can save considerable amounts of power by compressing data prior to transmission since the power consumed is directly proportional to the amount of data transmitted [4].

Actually, there have been several types of research analyzing compression techniques. Hanumathaiah et al. 2019 [5] proposed a low-cost low-power internet of things system for DC. The work is implemented using a mixture of the Delta technique and Run Length Encoding (RLE). The results indicate a high compression ratio (CR) of 52.67, saving ratio (SR) of 47.33 %, and compression factor of 1.898 for 12bits ADC. Gopinath et al. 2020 [6] addressed the various compression techniques. The results demonstrate that the Lempel-Ziv-Welch (LZW) technique has a better CR than other compression techniques for the same text equaling 4.33 and 76.9% in terms of the SR. However, when calculating execution time, the Delta technique provides better performance than LZW, RLE, and Huffman. Vijayalakshmi et al. 2021 [7] presented a Tamil technique for text compression. Based on the experimental results, they conclude that the proposed technique demonstrates that it outperforms Huffman, LZW, and ZIP by 31.85%, 19.58%, and 3.81%, respectively, in terms of average SR. Sridhar et al. 2021 [8] proposed an idea for the compression of the medical records of patients, which is a large amount of data. The process implementation was done using Python and Hadoop. The experimental results show that the LZW outperformed the Huffman and RLE. Mahammad et al. 2018 [9] describe a parallel paradigm based on open multiprocessing. The results indicated that Arithmetic Coding technology delivers a CR of 46%, which is greater than LZ77's 44% and K-RLE's 37%.

Nomenclature			
DC	Data Compression	LZW	Lempel-Ziv-Welch
RLE	Run Length Encoding	PCS	Proposed Combination System
CR	Compression Ratio	LCT	Lossless Compression Techniques
SR	Saving Ratio	LZ4	Lempel-Ziv-4

This paper presents a proposed combination system (PCS) that consists of two traditional serial techniques to increase the storage space and the transfer speed of files by reducing the bit rate. It is worth mentioning the measurement parameters used in this research to evaluate any DC technique, such as CR and SR. In terms of text files, the PCS is applied to English texts with variable text lengths. Finally, compression will be applied to highlight the robustness of the system. The organization of this article is shown as follows: Section 2 presents the classification in DC. Section 3 describes the methodology and PCS. Section 4 shows the results and discussion. In Section 5, the conclusion is provided.

2. Classification in DC

2.1. According to the quality of the data

Techniques can be divided into two categories [10]. The first one is the lossless compression technique (LCT), meaning no data is lost, and the integrity of the data is retained. During compression, the redundancy is removed, and during decompression, it is recreated so that decompression will give the original file. The second type of DC is lossy compression decrease bits by removing unimportant or unnecessary data, as shown in Fig. 1. This strategy is useful in situations where data from certain ranges that the human brain cannot recognize can be deleted, including video, music, and images. In this research, the first type will be implemented in order to regenerate the original data perfectly. The approach taken by this work to address the big data challenge is to reduce the size of data before it is transmitted and employ efficient means of transmitting this data across the network and eventually to its destination.

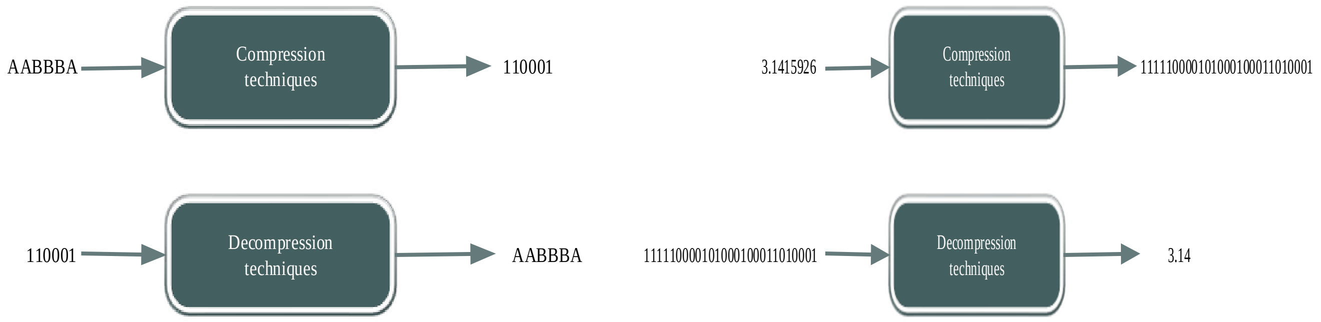


Fig. 1. Lossless and lossy techniques

2.2. According to the coding

In this subsection, the classification of DC models based on famous coding methods such as Shannon-Fano [11], Huffman [12], LZW [13], Lempel-Ziv-4 (LZ4) [14] Arithmetic Coding [15], and RLE [16].

2.3. According to types of data

DC techniques are commonly employed for the compression of data such as text, audio, images, and video.

2.4. According to the application

Numerous techniques can be applied to various application types. For instance, in wireless sensor networks, medical imaging, and other applications.

3. Methodology

Basically, the compression system is made up of two main components: compression and decompression. The compression process receives the input file and encodes it so that it may be transmitted. Upon receiving the transmitted file, the decompression process reconstructs it. The output will be an exact replica of the entry if the process is error-free. Fig. 2 depict the general block diagram of the DC procedure.

3.1. System design

In this work, PCS was proposed for use in compression. The system is designed to reduce the data rate, requiring less storage space and transmission bandwidth over a communication channel. Fig. 3 illustrates the procedure of the PCS. The proposed system consists of a hybrid of two LCT to enhance traditional techniques. In this manuscript, we combine Huffman with LZ4. The input of the Huffman part is an English text with different lengths of (111.262, 610.857, 377.11, 53.162, 82.2, 39.612, 71.647, and 49.38) kB. The Huffman technique reduces the original file length based on the repetition of symbols. It utilizes variable-length encoding, and each symbol is assigned variable lengths. The shorter codes will be assigned to symbols that appear more frequently, while symbols that appear less frequently will be assigned longer codes. To eliminate confusion during decoding, this technique follows the prefix-free principle, which ensures that the code allocated to any symbol is not a prefix of another symbol's code [17]. The output of Huffman is fed to the LZ4 part. This work aims to reduce the file's size as much as possible to make it easy to transfer or store on any simple device.

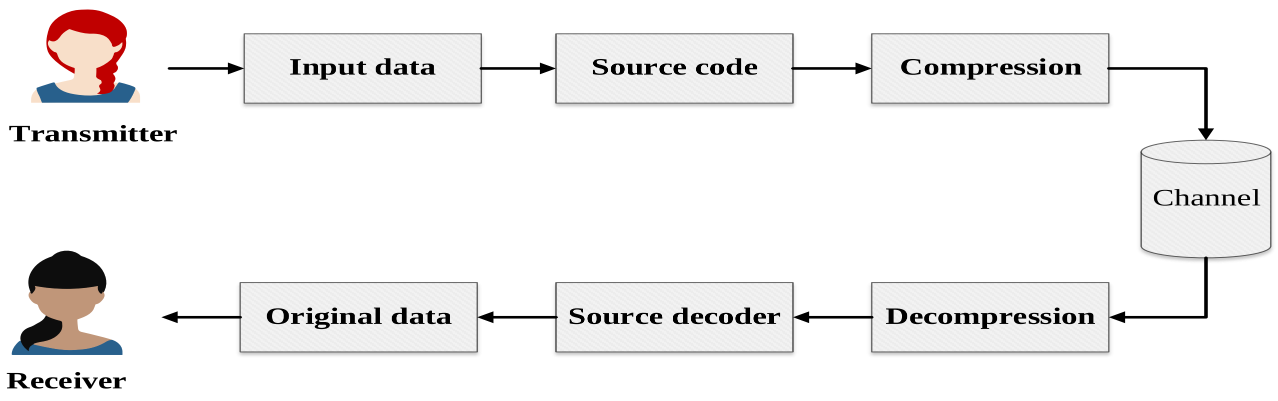


Fig. 2. General block diagram of a DC procedure

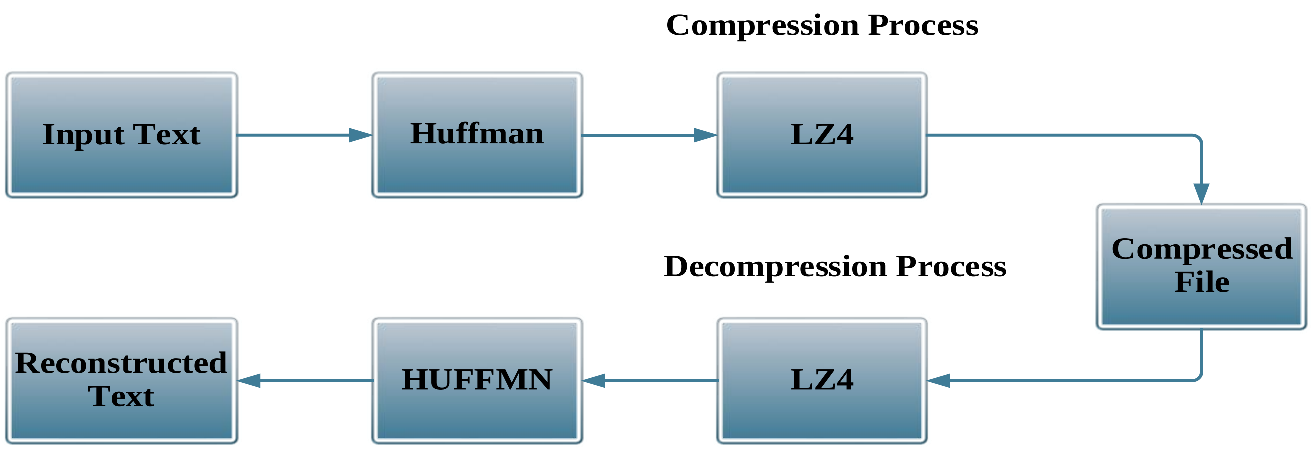


Fig. 3. Procedure of PCS

Python software is used to compress data by entering text. The following is a program display that is used for the compression process. Fig. 4 shows the steps of the program.

1. Open PCS software.
2. Select the file to be compressed.
3. Click the "RUN" button.
4. Wait for the result to appear in the command window.

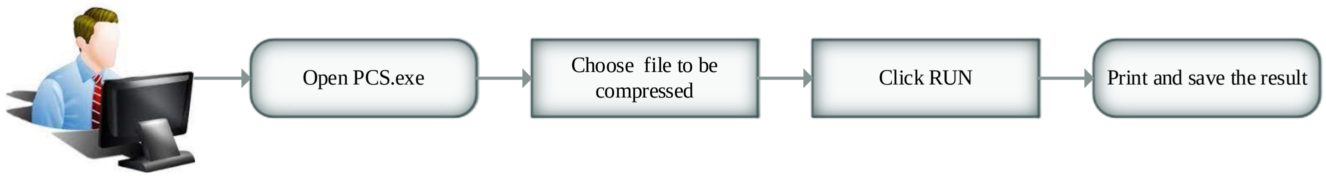


Fig. 4. Flow chart of compression program

3.2. Performance evaluation

In this section, certain experimental results are displayed and explained. To demonstrate the effectiveness of our PCS, we will use various criteria to evaluate the performance of the technique. The following is a brief explanation of CR and SR.

3.2.1. Compression Ratio (CR)

When transmitting files over a network, bandwidth is typically limited. Therefore, it is vital to decrease the bit rate as much as feasible. CR provides an indication of the capacity of the compression technique to compress a variety of network-shared file types [18]. The definition of CR is as follows (1):

$$CR=\frac{\text{original file size}}{\text{compressed file size}} \tag{1}$$

3.2.2. Saving Ratio (SR)

The basic objective of any DC technique is to minimize storage space. The ability of the compression technique to reduce the bit rate which can be measured by calculating its compressing efficiency in terms of SR [19]. According to the literatures, any compression technique that provides a higher SR should be robust and effective for saving memory space and lowering transmission costs [20]. This subsection evaluates and illustrates the compression efficiency of the PCS, which was determined using (2) [21]:

$$\text{SR}=\frac{\text{original file size-compressed file size}}{\text{original file size}}\times 100\% \qquad (2)$$

4. Results and Discussion

4.1. Our Experiments

All experiments were performed using Python version 3.10.0 on a personal computer with specifications including an Intel Core i5 2.5 GHz CPU, 16 GB of RAM, and a 64-bit Windows 10 operating system. Our proposed method is applicable to both large and small files. We have analyzed eight considerable text files from [22]. Table 1 illustrates the different lengths in files of the proposed work.

The previous datasets were compressed in order to evaluate the PCS depending on the CR and SR. Before calculating CR and SR, it is necessary to calculate the compressed size of each file in the datasets. Table 2 and Fig. 5 show the results of the dataset compression. The CR has been calculated using (1) and the SR of the PCS is calculated using (2) for each file, and the results are presented in Table 2.

Table 1. The compression file, CR, and SR of the PCS

File name	Input data (kB)	PCS (kB)	CR	SR
bib	111.262	27.44575	4.0539	75.332 %
book2	610.857	155.2375	3.935	74.587 %
news	377.11	99.37225	3.7949	73.649 %
paper1	53.162	11.864625	4.4807	77.682 %
paper2	82.2	16.679625	4.9282	79.708 %
progc	39.612	8.508125	4.6558	78.521 %
progl	71.647	14.54975	4.9243	79.692 %
progp	49.38	10.735	4.5999	78.26 %

In all cases, the proposed system performed better. More specifically, PCS has reduced the size of the small file to 8.5 kB while giving 155.2 kB for the input large file.

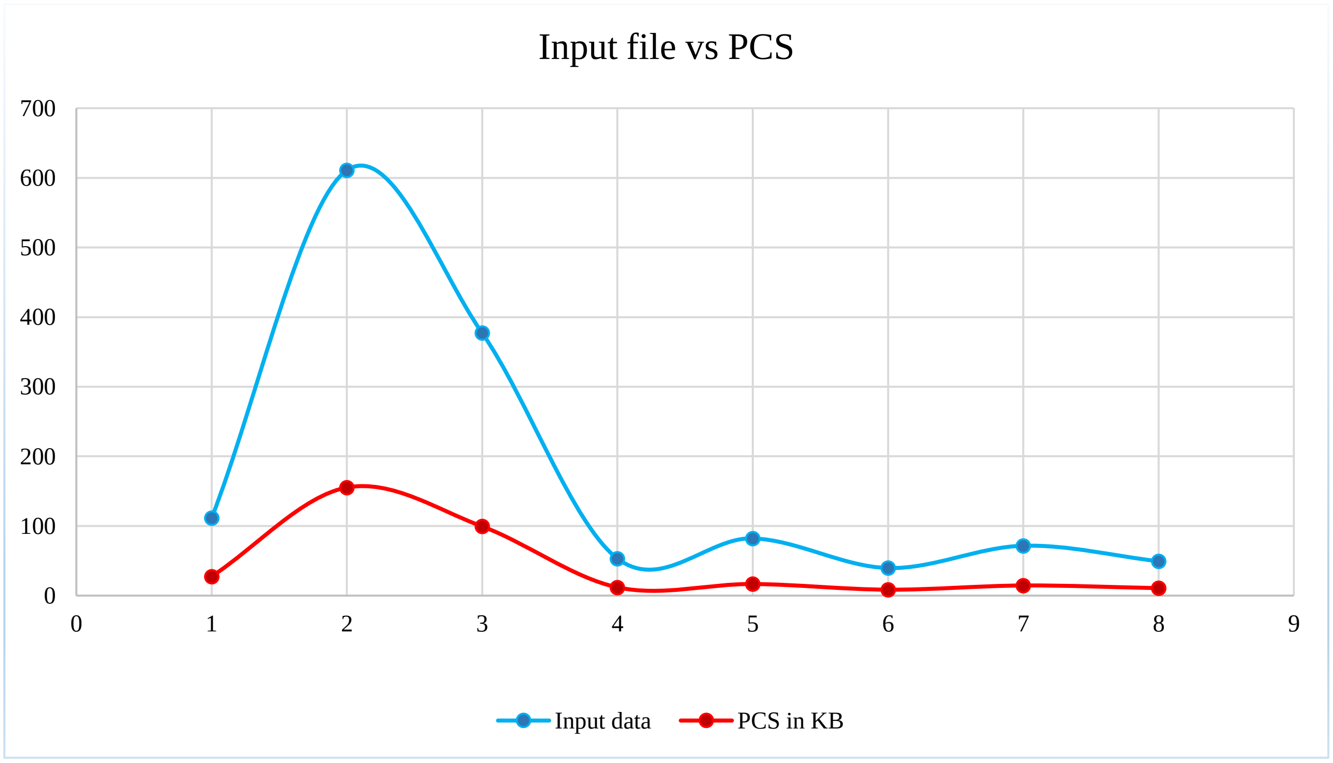
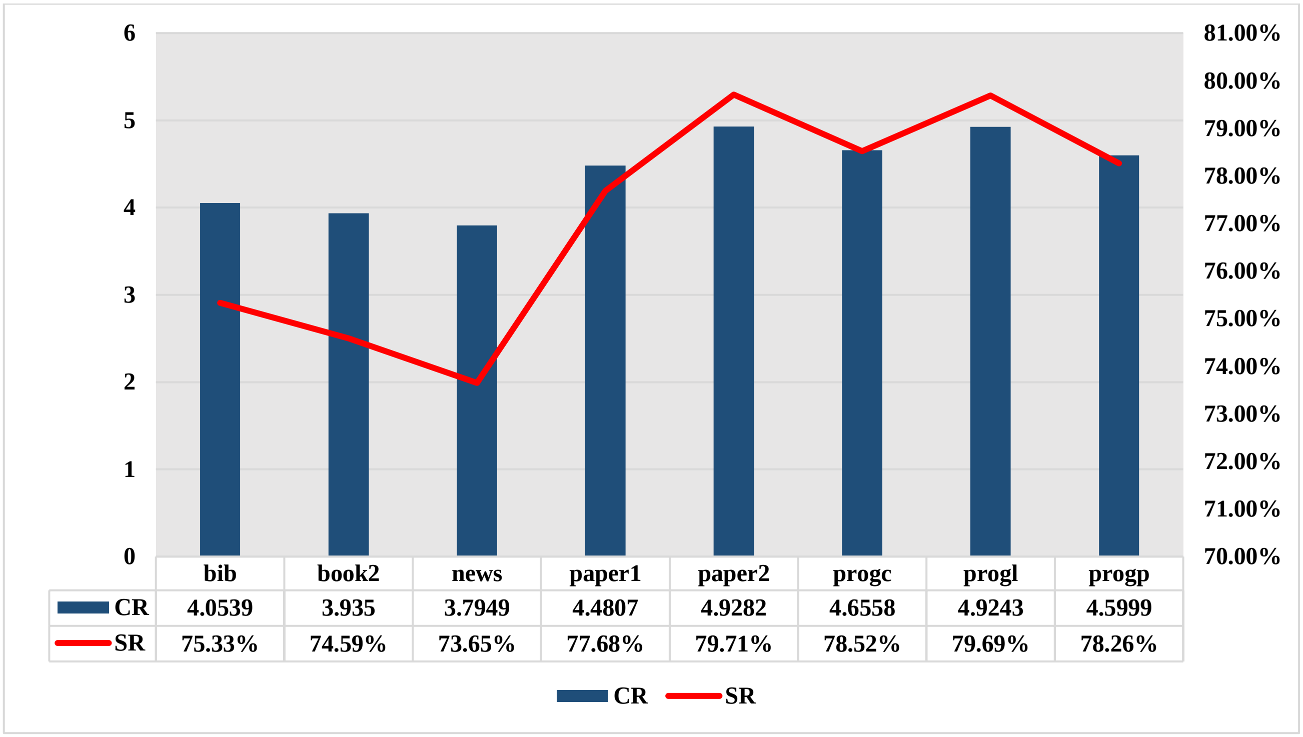


Fig. 5. The difference between original file size and compressed file

It is clear from Table 1 & Fig. 6 that the proposed techniques provide better compression, which indicates an average CR of 4.42. Based on the results, it is clear that the PCS provides better results, which indicates that it achieves between 73.649 % and 79.708 % in terms of average SR.



	bib	book2	news	paper1	paper2	progc	progl	progp
CR	4.0539	3.935	3.7949	4.4807	4.9282	4.6558	4.9243	4.5999
SR	75.33%	74.59%	73.65%	77.68%	79.71%	78.52%	79.69%	78.26%

Fig. 6. Illustrates the SR and CR

5. Conclusion

Modern daily activities produce massive amounts of data, which creates a significant challenge for file storage and transmission. Compression is a useful technology that increases the capacity of memory and reduces the data rate to reduce the bandwidth of data transfer. This work proposes a system to improve text compression in the English language. The PCS consists of two serial compression techniques: Huffman and LZ4. The input data is represented by the datasets in eight different sizes. The system is modelled using the Python package. The result demonstrates that the proposed combination techniques reduce the file size significantly, achieving an average SR of between 73.649 % and 79.708 %. In the future, the above technique can be used in practise with an FPGA to verify the above results.

Acknowledgement

We would like to thank the electrical engineering technical college dean for their support and facilities that enabled us to apply our experiment. In addition, thanks to the department of computer technical engineering for their help and encouragement to finish our research.

References

[1] J. Howarth, “Top 2022 big data statistics,” Available online: https://explodingtopics.com/blog/big-data-stats .
[2] I.M. Pu, “Fundamental Data Compression; Butterworth-Heinemann,” Oxford, UK, 2005.
[3] D. Salomon, G. Motta, “Handbook of Data Compression,” London, New York, Springer, 2010.
[4] S. Porwal, Y. Chaudhary, J. Joshi, and M. Jain, “Data compression methodologies for lossless data and comparison between algorithms,” Int. J. Eng. Sci. Innov. Technol. (IJESIT) 2013, 2, 142–147.
[5] A. Hanumanthaiah, A. Gopinath, C. Arun, B. Hariharan and R. Murugan, “Comparison of Lossless Data Compression Techniques in Low-Cost Low-Power (LCLP) IoT Systems,” 2019 9th International Symposium on Embedded Computing and System Design (ISED), 2019, pp. 1-5, doi: 10.1109/ISED48680.2019.9096229.
[6] A. Gopinath and M. Ravisankar, “Comparison of Lossless Data Compression Techniques,” 2020 International Conference on Inventive Computation Technologies (ICICT), 2020, pp. 628-633, doi: 10.1109/ICICT48043.2020.9112516
[7] B. Vijayalakshmi and N. Sasirekha. "Comparative Analysis of Lossless Text Compression Methods with Novel Tamil Compression Technique." vol 9 (2021): 38-44.
[8] A. P. Sridhar and P. V. Lakshmi, “An Efficient Lossless Medical Data Compression using LZW compression for Optimal Cloud Data Storage,” vol. 25, no. 6, pp. 17144-17160, 2021.
[9] F. S. Mahammad and V. M. Viswanatham, “Performance analysis of data compression algorithms for heterogeneous architecture through parallel approach, ” 2018.
[10] Z. N. Li, M. S. Drew, and J. Liu, “Fundamentals of multimedia,” Springer, 2004.
[11] R. M. Fano, “The Transmission of Information,” Massachusetts Institute of Technology, Research Laboratory of Electronics, 1949.

[12] D. A. Huffman, “A Method for the Construction of Minimum-Redundancy Codes,” in Proceedings of the IRE, vol. 40, no. 9, pp. 1098-1101, Sept. 1952, doi: 10.1109/JRPROC.1952.273898.
[13] J. Ziv and A. Lempel, “A Universal Algorithm for Sequential Data Compression,” IEEE Trans. Inf. Theory, vol. 23, no. 3, pp. 337–343, 1977, doi: 10.1109/TIT.1977.1055714.
[14] W. Liu, F. Mei, C. Wang, M. O’Neill and E. E. Swartzlander, “Data Compression Device Based on Modified LZ4 Algorithm,” in IEEE Transactions on Consumer Electronics, vol. 64, no. 1, pp. 110-117, Feb. 2018, doi: 10.1109/TCE.2018.2810480.
[15] I. H. Witten, R. M. Neal, and J. G. Cleary, “Arithmetic coding for data compression,” Commun. ACM, vol. 30, no. 6, pp. 520–540, 1987, doi: 10.1145/214762.214771.
[16] A. H. Robinson and C. Cherry, “Results of a prototype television bandwidth compression scheme,” in Proceedings of the IEEE, vol. 55, no. 3, pp. 356-364, March 1967, doi: 10.1109/PROC.1967.5493.
[17] M. Nelson and J.-L. Gailly, “The Data Compression Book Chapter 1 Introduction to Data Compression,” 2007.
[18] W. Zhan and A. El-Maleh, “A new scheme of test data compression based on equal-run-length coding (ERLC),” vol. 45, no. 1, pp. 91-98, 2012.
[19] S. T. Klein and D. Shapira, “Practical fixed length Lempel–Ziv coding,” vol. 163, pp. 326-333, 2014.
[20] S. Roman, “Introduction to coding and information theory,” vol. 34, no. 09. 1997.
[21] T. C. Bell, J. G. Cleary, and I. H. Witten, “the Canterbury Corpus,” Available online: https://corpus.canterbury.ac.nz/.