International Journal of Research in Engineering, Science and Management
Volume-1, Issue-11, November-2018
www.ijresm.com | ISSN (Online): 2581-5792

Data Compression Using Huffman's Greedy Approach and LZ 77

K. Subha[1], Yash Mohata[2], Jambay Yeshi[3], Ashutosh Digari[4]
[1]*Assistant Professor, Department of Computer Science Engineering, SRM IST, Chennai, India*
[2,3,4]*Student, Department of Computer Science Engineering, SRM IST, Chennai, India*

***Abstract*— Data Compression algorithms have gained immense popularity in recent decades. This sector of Computer Science industry is the new budding field with lots of opportunities. Data is the process of modifying, encoding or converting the bits structure of data compression in such a way that it consumes less space on disk, it enables reducing the storage size of one or more data instances or elements. Data compression is also known as source coding or bit-rate reduction. Data compression has wide implementation in computing services and solutions, specifically Data communications. Data compression works through several compressing techniques and software solutions that utilize data compression algorithms to reduce the data size. The project focuses on enhancing Huffman's Algorithm for greedy approach and combining the LZ77 Algorithm to create a new Algorithm that can perform lossless compression of data. . So, the most important question today is what to do with all this data. Whether to discard it? Certainly not. Whether to keep building new storage units? Well, that can certainly be an option. But what if we can reduce the size of this data? That would be the best alternative to building more powerful storage devices. This is what we call Data Compression. This is how we would be able to solve most of 'ur problems with the management of data that we are facing today. This project deals with the same: 'Data Compression'. Although Data Compression is not something which has been discovered very recently. The concept has been lying about since years. A lot of goals have certainly been achieved, but yet there is a lot to be discovered.**

***Index Terms*— Lempel Ziv 77 Algorithm, Data compression, Huffman's Greedy Approach**

I. Introduction

In signal processing, data compression, source coding, or bit-rate reduction involves encoding information using fewer bits than the original representation. Compression can be either lossy or lossless. Lossless compression reduces bits by identifying and eliminating statistical redundancy. No information is lost in lossless compression. Lossy compression reduces bits by removing unnecessary or less important information. The process of reducing the size of a data file is often referred to as data compression. In the context of data transmission, it is called source coding; encoding done at the source of the data before it is stored or transmitted. Source coding should not be confused with channel coding, for error detection and correction or line coding, the means for mapping data onto a signal.

II. Background

A new prediction algorithm has proposed which predicts whether a file would or would not compress with the LZW Technique. It also predicts the compression Ratio which helps storage systems to decide whether the file should be compressed or not (helping in auto-accommodation of file). This method reduces the compression time by 17.79 %. For energy saving in wireless sensor networks a new lossless compression algorithm has proposed (S-LEC) and compares it with the existing algorithm which are LEC and S-LZW. This method has been applied on wireless data sets like Volcano data and Humidity data. The S-LEC is robust and more efficient than both LEC and S-LZW. In 2008 Aree A. Muhammad and Loay E. George proposed a new scheme for image compression that works on two stages. The first stage uses a lifting scheme wavelet-based transform. For the second stage they developed a modified entropy coding algorithm. Their proposed scheme was tested and showed that the quality maintains the same regardless of the different coding techniques used. They achieved a good compression factor with block sublevel coding algorithm but the computational time was

III. Proposed Method

Huffman's Greedy Approach and LZ77 Algorithms are both efficient algorithms but do not compress data to their full potential. Their compressing power can be increased by combining them together. Huffman's greedy approach is totally based on the frequency of each element in the input text. While LZ77 focuses on reducing redundancy by associating commonly occurring phrases with pointers, thereby making the text itself more compressible to be fed into Huffman's Algorithm. Also, the Huffman's greedy approach is a bottom to top approach. This although an easy to understand and effective approach fails to use the full potential of the tree. And the resultant of this approach is an increased height of the tree. Since the compressibility of the text is inversely proportional to the height of the tree. This is not the best compression which could be deduced from this approach. What is proposed is the creation of a tree in such a manner such that the allocation of memory for the next level does not begin till the nodes of the previous level all have their children nodes.

A. Algorithm

- Start
- Input Data
- Application Of Lp77
- Finding The Frequency Of Elements
- Sorting In Descending Order
- Starting Allocation Of Tree
- Do While (Node(Level-1) ==True)
- If (Node(Level-1) Has All Children)
- Continue Allocation To The Next Level Else
- Allocate Children Nodes To The Unallocated Node(Level-1)

IV. Implementation

To understand the probability-based approach let us take an easy example below is a simple table describing the decoding of a simple textual data consisting of five characters: A, B, C, D, E. We have already given a table describing the probabilities of finding those characters and we are going to use these values throughout our calculations. The entities mentioned in the table are as follows:

- *Character*: Represents the character under consideration
- *Probability*: Represents the probability of finding the character
- *Representation*: The representation of the character according to the common representation or with the help of the binary tree
- *Cost/Bit*: Effective Cost/Bit
- *Total Cost*: Total Cost

The formula for calculating other quantities are as follows:
Total Cost = Probability * (Cost/Bit)
Final Cost= ∑ Total Cost

A. Normal Approach to Compression

In this approach we use the normal method of representation of characters during algorithm. To represent characters in bits, the no. of bits depends upon the no. of characters. Suppose we want to transmit two characters a and b: then we need one single bit: 0 and 1. But as we have a third character to transmit c: we would need two bits at a time to transmit each of a, b and c: 00 01 10. So, since we have only 5 characters here we can easily represent them with the help of permutation of 3 bits.

TABLE I
NORMAL APPROACH

Character	Probability	Representation	Cost/Bit	Total cost
A	0.04	000	3	0.12
B	0.12	001	3	0.36
C	0.45	010	3	1.35
D	0.23	011	3	0.69
E	0.16	100	3	0.48
Total Cost				3.0

The total cost by this method comes out to be 3.0

B. Huffman's Greedy Approach

For this approach we arrange the characters in a binary tree in the decreasing order of their probability, from top to down. The right branch sums up for a 1-bit representation and the left one for a 0-bit representation. Thus if we tabulate our observation from the above approach we get the following:

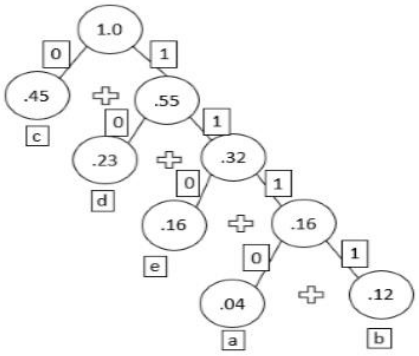

Fig. 1. Huffman's Greedy Approach

TABLE II
HUFFS MAN'S GREEDY APPROACH

Character	Probability	Representation	Cost/Bit	Total cost
A	0.04	1110	4	0.16
B	0.12	1111	4	0.48
C	0.45	0	1	0.45
D	0.23	10	2	0.46
E	0.16	110	3	0.48
Total Cost				2.03

By using the Huffman's greedy approach, we get a Total Effective final cost of 2.03.
The Huffman's greedy approach is a bottom to top approach. This although an easy to understand and effective approach fails to use the full potential of the tree. And the resultant of this approach is an increased height of the tree. Since the compressibility of the text is inversely proportional to the height of the tree. This is not the best compression which could be deduced from this approach.

C. Proposed Approach

For this approach we arrange the characters in a binary tree in the decreasing order of their probability, from top to down. The right branch sums up for a 1-bit representation and the left one for a 0-bit representation. But we would not be moving on to the next level of the tree till the entire level is filled. Since the compressibility of the text is inversely proportional to the height of the tree. Huffman's is not the best compression which

could be deduced from this approach. What is proposed is the creation of a tree in such a manner such that the allocation of memory for the next level does not begin till the nodes of the previous level all have their children nodes.

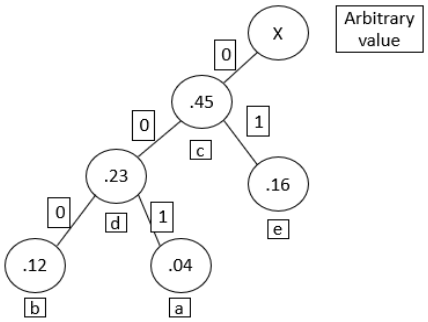

Fig. 2. Proposed approach

TABLE III
PROPOSED APPROACH

CHARACTER	PROBABILITY	REPRESENTATION	COST/BIT	TOTAL COST
A	0.04	001	3	0.12
B	0.12	000	3	0.36
C	0.45	0	1	0.45
D	0.23	00	2	0.46
E	0.16	01	2	0.32
TOTAL COST				1.71

Huffman's Greedy Approach and LP77 Algorithms are both efficient algorithms but do not compress data to their full potential. Their compressing power can be increased by combining them together. Huffman's greedy approach is totally based on the frequency of each element in the input text. While LP77 focuses on reducing redundancy by associating commonly occurring phrases with pointers, thereby making the text itself more compressible to be fed into Huffman's Algorithm. Also, the Huffman's greedy approach is a bottom to top approach. This although an easy to understand and effective approach fails to use the full potential of the tree. And the resultant of this approach is an increased height of the tree. Since the compressibility of the text is inversely proportional to the height of the tree. This is not the best compression which could be deduced from this approach. What is proposed is the creation of a tree in such a manner such that the allocation of memory for the next level does not begin till the nodes of the previous level all have their children nodes.

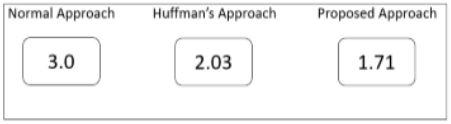

Fig. 3. Comparison

V. CONCLUSION

In this paper, to achieve better compression rates, we combined the Huffman's Greedy Approach and Lempel-Ziv 77 Algorithms. The Huffman's Greedy Algorithm has a Probability based approach while LP77 focuses on reducing the redundancy of the file. Therefore, we intend to first preprocess the data into a more compressible form and the compressing it by feeding it into our Algorithm. We have proposed the creation of a tree in such a manner such that the allocation of memory for the next level does not begin till the nodes of the previous level all have their children nodes.

VI. FUTURE WORK

In this paper there's lack of sample space to compare our proposed model with the existing models. Hence, we will be focusing on coming up with larger and more varied sample spaces to get an accurate comparison and analyze the results. Currently we are facing a lack of appropriate hardware to actually test the true potential of our proposed algorithm and also compare it with the test results of the existing approaches. So, testing our proposed algorithm on appropriate hardware and analyzing the results of the same is something we'd like to accomplish in the near future.

REFERENCES

[1] Komal Sharma, Kunal Gupta. "Lossless Data Compression Techniques and Their Performance." Department of Computer Science, Amity University, Noida, Uttar Pradesh, ICCCA 2017.
[2] Chang-Wen Chen, Yi-Cheng Kong and Kuen-Jong Lee. "Test Compression with Single-Input Data Spreader and Multiple Test Sessions." Department of Electrical Engineering, National Cheng Kong University, IEEE 26th Asian Test Symposium, 2017.
[3] Amit Jain, Ravindra Patel. "An Efficient Compression Algorithm (ECA) for Text Data." International Conference on Signal Processing Systems, 2017.
[4] Dapeng Dong and John Herbert. "Compressed Domain-Specific Data Processing and Analysis." Department of Computer Science, University College Cork, Ireland. IEEE International Conference on Big Data 2017
[5] Devi Dath and Vinitha Panicker J. "Enhancing Adoptive Huffman Coding Through Word by Word Compression for Textual Data." International Conference on Communication and Signal Processing, April 6-8, 2017, India.
[6] S Jancy, Dr. C Jayakumar. "Various Lossless Compression Techniques Surveyed." Professor, Department of Computer Science, Sathyabama University, Chennai, India. ICONSTEM 2017.
[7] Adam Gleave, Christian Steinruecken. "Making Compression Algorithms for Unicode Text." University of Cambridge. Data Compression Conference 2017.
[8] Koichi Marumo, Shinichi Yamagiwa. "Time Sharing Multithreading on Stream-based Lossless Data Compression." Department of Computer Science/Faculty of Engineering, Information and System. 5th International Symposium on Computing and Networking 2017.
[9] Ahmed S. Farhan, Fouad H. Awad-MIEEE, Khitam Abdulbasit, Mohammed Adeeb. "Proposed Two Shift-Coding Based compression Techniques." College of Business Informatics, University of Information Technology & Communication, Iraq/Computer Science Department, College of Computer-University of Anbar,Iraq. 2017 ICCCIT, Slemani, Iraq.
[10] M M Kodabagi, M.V Jerabandi, Nagaraj Gadagin. "Multilevel Security and Compression of Text Data using Bit Stuffing and Huffman Coding." Dept. of Computer Science and Engineering KLE Institute of Technology, Hubli, India. 2015 ICATCCT.