



International Journal of Emerging Technologies in Computational and Applied Sciences (IJETCAS)

(Open Access, Double Blind Peer-reviewed, Refereed and Indexed Journal)

www.iasir.net

File Compression

Pulkit Mittal, Kenil Patel, Abhishek Mitra, Divyanshu Jalthar
School of Computer Science and Engineering,
VIT University, Vellore, Tamil Nadu, INDIA.

Abstract: File compression is of utmost importance in today's world where data is stored digitally forming groups of large Information systems. These data take up large amount of storage space. So there is an increasing need to compress files. It can save millions of terabytes of information and data, reducing the need for storage space and fast network transmission. In this paper we will not only study the existing algorithms for compression but also modify them and extend them to other file formats.

Keywords: Compression, Decompression, Encryption, Decryption, Compression Ratio

I. Introduction

The software proposed in this paper is an extremely useful utility that uses sophisticated algorithm to compress and decompress a file. Different algorithms and techniques can be applied to compress not only text but also image, video and audio files. Multiple file compression algorithms and techniques are available but the best algorithm is the one which not only takes the least time to compress but also provides the best compression ratio among other algorithms available. Thus, keeping these factors in mind, the best algorithm needs to be selected for a given scenario. Compressing a file using file compression algorithms creates another encoded version of the file (compressed version). This compressed file can be sent over a network, say Internet and at the receiving end a decoder algorithm decodes the entire file and converts it into a user-readable format. But, sending a compressed version of the file over the Internet is not safe and so this software also has an encryption module in it. If the user is to send some highly private information over a network in compressed format, then the user can select the encryption feature after compressing the file, and then provide an encryption key to the compressed file before sending the file over the network. The receiver should have the same software installed in his/her machine in order to first provide the decryption key, which is nothing but the same encryption key provided by the sender and then if the decryption key is authentic i.e., if the decryption key is same as the encryption key provided by the sender, then the software uses decompression algorithms to decompress the file in human-readable format with minimal loss of information. Thus, the file can only be opened by those, with whom the sender has shared the encryption key. This feature maintains privacy of information while transferring files over the Internet.

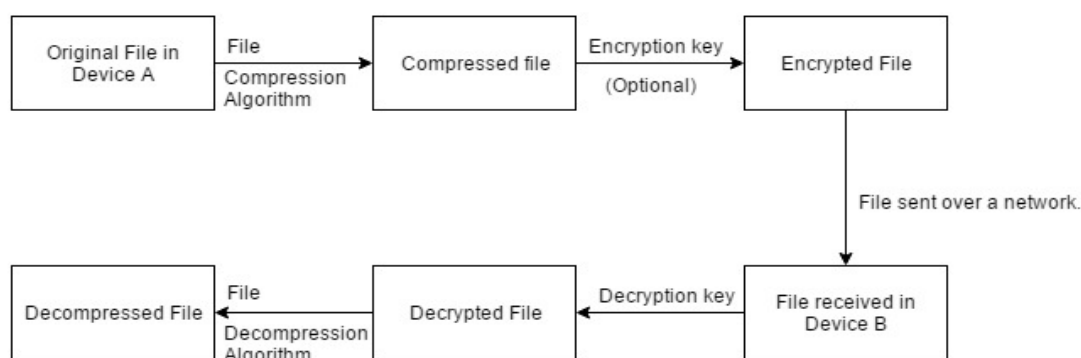


Fig. 1- Flow of modules

II. Related Works

In the research paper [1], a combination of Dynamic bit reduction and Huffman coding, also referred to as hybrid compression has been proposed. Use of hybrid technique ensures that the algorithm consumes less time and provides appreciable compression ratio. The research paper [2] elucidates on two major compression techniques: Huffman and arithmetic coding. Both the techniques are compared based on their performances. [3] is a review paper describing various compressing and decompressing techniques for textual data. [4] has proposed an

approach for lossless image compression in spatial domain using a technique called image folding. Also this technique has been compared with other standard algorithms in terms of performances.

[5] addresses the problem of lossless compression of text files with an added security. [6] Proposes a network conscious text compression system for transmitting huge volumes of data over a network. This method is also integrated with different text transfer protocols. It applies LIPT encoding along with BZIP2 and GZIP for compression. In order to speed up pattern matching by text compression [7] proposes Byte pair encoding (BPE). It's decompression is faster than most other algorithms and requires small work space. It highlights the advantage and practicality of BPE to find patterns without decompressing the files.

[8] gives a new model to reduce the space taken by files and time taken to answer a query by compressing not only the text but also index of the text retrieval system. [9] uses a dictionary based model for compression. Instead of the encoder working online, here it uses the full message to infer a complete dictionary in advance, and include an explicit representation of the dictionary in compressed file. [10] uses StarNT, a dictionary-based fast lossless text transform algorithm. It achieves a superior compression ratio than almost all the other recent efforts based on BWT and PPM. [11] It uses Quine-McCluskey algorithm which is capable of minimizing logical relationships for any number of inputs. It considers every 16-bit block from the input file, as one chunk called a vector, and finds the minimized Boolean function for it which helps to remove redundant data in order to reduce the size of a data file.

[12] This paper discusses some new interpretations and uses of the Burrows–Wheeler transform. It introduces some lossless compression. PPM (Prediction by Partial Matching) is best method for lossless or text compression. In this paper they used modified Burrows–Wheeler transform (Combined PPM with Burrows–Wheeler compression). [13] A set of selected algorithms are examined and implemented to evaluate the performance in compressing text data. It discusses run length encoding, Huffman encoding, arithmetic encoding, the Lempel Ziv Welch Algorithm.

[14] This paper presents a study of transform methods used in loss-less text compression in order to pre-process the text by exploiting the inner redundancy of the source. The transform methods are Burrows-Wheeler Transform (BWT, also known as Block Sorting), Star Transform and Length-Index Preserving Transform (LIPT). BWT converts the original blocks of data into a format that is extremely well suited for compression.

[15] J-Bit Encoding was proposed by AgusDwiSuarjaya which is a lossless compression technique. The algorithm controls bits of information to reduce the size and optimize the contribution for other algorithm by dividing the input data into two information where the primary information will contain non-zero byte and the second information will contain bit esteem clarifying position of nonzero and zero bytes. This algorithm has better compression ratio for 24-bit images because the image has more complex data than 8 bit since it stores more color. 5 types of files with 50 different sizes for each type were used to test the algorithm which showed that the algorithm gives better compression ratio when inserted between move to front transform (MTF) and arithmetic coding (ARI).

[16] In this paper the author has provided a method which is capable of compressing images without degrading its quality which is achieved through minimizing the number of bits required to represent each pixel. In this paper lossless image compression with four modular components: pixel sequence, pre-diction, error modeling, and coding is discussed. The method which is proposed in this paper is a lossless image compression scheme which is applied to all types of image based on Lempel-Ziv-Welch algorithm which reduces the repeated value in image and Bose, Chaudhuri and Hocquenghem codes that detect or correct the errors. After testing the proposed algorithm, the result concludes that it improves the compression of images as compared to LZW algorithms, RLE and Huffman because its average compression ratio is 1.636383 which is quite better than standard lossless image compression.

[17] In this paper, a novel concept of image folding involving lossless image compression in spatial domain or continuous tone images. Corrugation/ Data folding exploits pixel redundancy in 2D images by subtracting even pixels from odd pixels and store the difference data in a buffer. This algorithm which is talked about is appropriate for those pictures whose pixel values shift gradually in the 2D-space, i.e., the image doesn't have an excessive number of high frequency contents—like edges, corners and so forth in it. After certain levels of data folding compression efficiency gets saturated.

[18] The principle target of this paper is to compress images by reducing the number of bits per pixel required to present it and to diminish the transmission time for transmission of images and after that recreating back by decoding the Huffman codes. Two fundamental components of compression are: Redundancies reduction which aims at removing duplication from the signal source Irrelevancy reduction which omits parts of the signal that will not be noticed by the signal receiver. The experiment presented a new compression and decompression technique based on Huffman coding and decoding for scan testing to reduce test data volume, test application time.

III. Methodology

On successful installation of the software, the user is provided with a Home screen of the Data Handling Solutions Software as shown in Fig-2. The user can now login and continue or else continue as a guest and perform the compression or decompression actions as per the need. Upon selecting the compression option, the user is redirected to the compression page as shown in Fig-3. Here, the user first needs to browse the file which needs to be compressed and upload it. The user can even select the file format, i.e .rar or .zip or .tar or .huf, in which the compressed file should be converted into. Apart from this, the user can select the encrypt file option. With the help of this option, the user can protect the file from unwanted viewers thereby protecting the privacy of the file. If the user selects the encrypt file option, then the user has to provide with a encryption key, and this key needs to be shared with every user who is to view the file. Upon successful compression of the file, the Data Handling Software generates a summary of the compression done, showing details such as the filename, input file size, output file size, compression ratio of the file and also the data loss as percentage as shown in Fig-4. After the user has compressed the desired file, the user can either transfer the file over a network or by any other communication media to the target users. In order for these receivers to view the information in the file, they need to first either login or continue as a guest in the Data Handling Software. Then the user needs to decompress the file by first selecting the decompression option and then browsing and uploading the desired compressed file which needs to be decompressed as shown in Fig-5. If the compressed file was previously encrypted by a key, then the user needs to enter the private key as shared to him/her by the sender in order to view the file. If the user enters the correct private key for the file, then the Data Handling Software will successfully decompress the file and allow the user to view it.



Fig-2. Home Page of the Software



Fig-3. Compression Module with built-in encryption feature

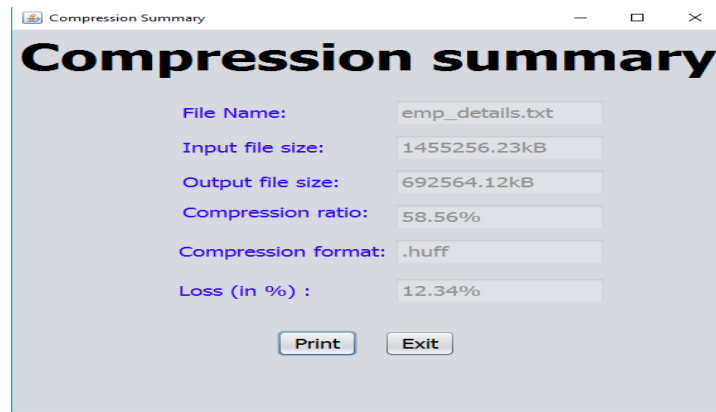


Fig-4. Compression Summary



Fig-5. Decompression module with private key checking

Algorithm

The main function that builds Huffman tree

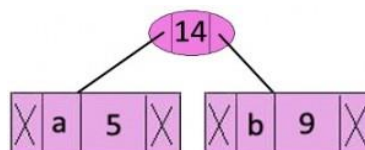
- Step 1: Create a min heap of capacity equal to size. Initially, there are nodes equal to size.
Iterate while size of heap doesn't become 1
- Step 2: Extract the two minimum freq items from min heap.
- Step 3: Create a new internal node with frequency equal to the sum of the two nodes frequencies.
Make the two extracted node as left and right children of this new node. Add this node to the min heap.
- Step 4: The remaining node is the root node and the tree is complete.
Assign 1 to right edge and recur.
If this is a leaf node, then it contains one of the input characters, print the character and its code from arr[]

IV. Numerical Illustration:

Character	Frequency
a	5
b	9
c	12
d	13
e	16
f	45

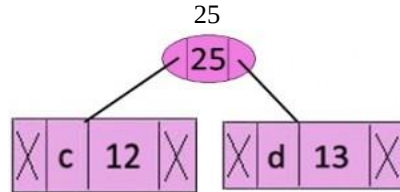
Step 1. Build a min heap that contains 6 nodes where each node represents root of a tree with single node.

Step 2 Extract two minimum frequency nodes from min heap. Add a new internal node with frequency $5 + 9 = 14$.



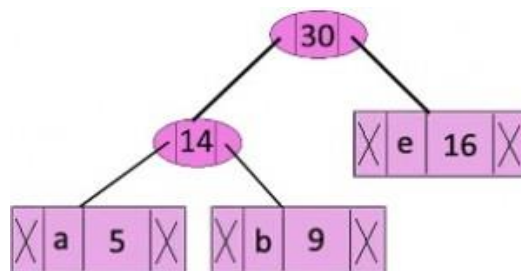
Character	Frequency
c	12
d	13
Internal Node	14
e	16
f	45

Step 3: Extract two minimum frequency nodes from heap. Add a new internal node with frequency $12 + 13 =$



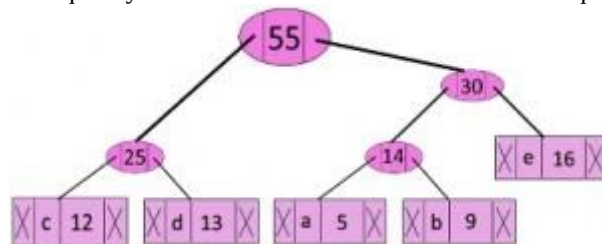
Character	Frequency
Internal Node	14
e	16
Internal Node	25
f	45

Step 4: Extract two minimum frequency nodes. Add a new internal node with frequency $14 + 16 = 30$



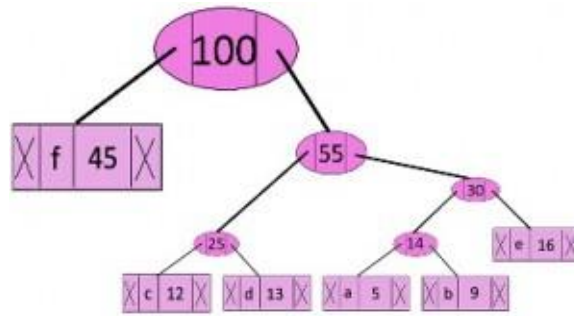
Character	Frequency
Internal Node	25
Internal Node	30
f	45

Step 5: Extract two minimum frequency nodes. Add a new internal node with frequency $25 + 30 = 55$



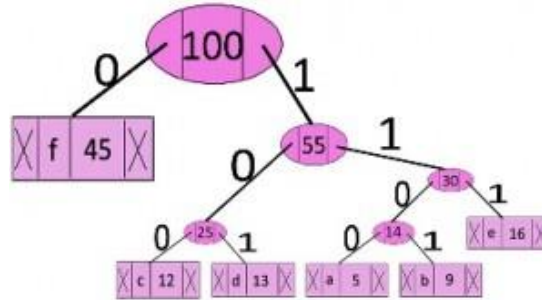
character	Frequency
f	45
Internal Node	55

Step 6: Extract two minimum frequency nodes. Add a new internal node with frequency $45 + 55 = 100$



Since the heap contains only one node, the algorithm stops here.

Character Frequency
Internal Node 100



Character	Code-word
f	0
c	100
d	101
a	1100
b	1101
e	111

V. Results

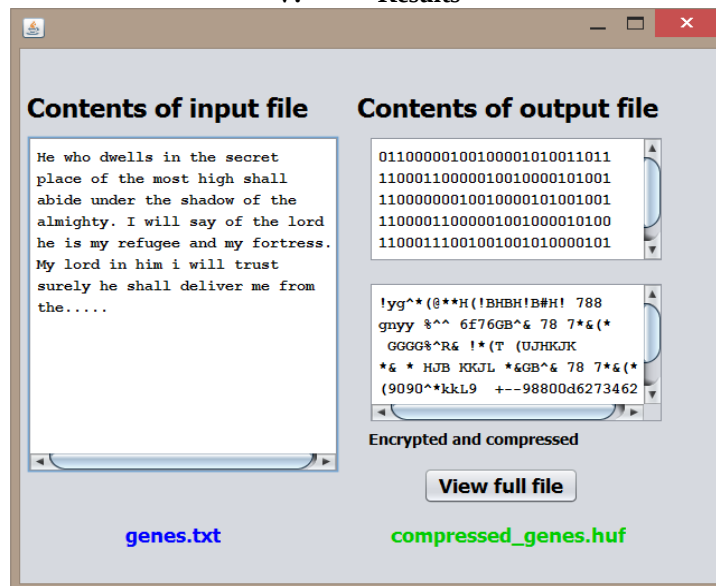


Fig. 6. Output Interface

The above figure shows how the output file will be generated. The file is successfully compressed and encrypted for further use.

VI. Conclusion

Thus, we have proposed and implemented software which facilitates the user to compress and decompress files. It even enables the user to encrypt the files that they are compressing. Thus, the user can encrypt the files and send it to other peers who can view the files after providing the correct key and thereby decompressing it.

VII. Future Scope

The proposed system can further be upgraded by using more complex and optimized compression and decompression schemes. Algorithms should be selected in such a way that they provide as much lossless compression as possible. Advanced encryption and decryption methods can also be incorporated into the software so as to protect the privacy of the user's documents from outside unwanted viewers and moderators.

References

- [1] Research Paper on Text Data Compression Algorithm using Hybrid Approach, JCSMC, Vol. 3, Issue. 12, December 2014
- [2] Data Compression Methodologies for Lossless Data and Comparison between Algorithms International Journal of Engineering Science and Innovative Technology (IJESIT) Volume 2, Issue 2, March 2013
- [3] A Review of Various Data Compression Techniques to form a New Technique for Text Data Compression
- [4] Lossless Image Compression based on Data Folding CRE ST, KPIT Cummins Info systems Ltd. Pune, India
- [5] An Intelligent text data encryption and compression for high speed and secure data transmission over internet
- [6] Network conscious text compression system (NCTCSys), N. Motgi
- [7] Speeding Up Pattern Matching by Text Compression, Yusuke Shibata, Takuya Kida, Shuichi Fukamachi, Masayuki Takeda, Ayumi Shinohara, Takeshi Shinohara, Setsuo Arikawa
- [8] Compression: A Key for Next Generation Text Retrieval Systems, Nivio Ziviani, Edleno Silva de Moura
- [9] Off-Line Dictionary-Based Compression, N. JESPER LARSSON AND ALISTAIR MOFFAT
- [10] Dictionary-based fast transform for text compression, Weifeng Sun
- [11] Compression of small text files, Jan Platoš, Václav Snášel, Eyas El-Qawasmeh
- [12] Word-based text compression using the Burrows–Wheeler transform, Peter Fenwick
- [13] Fast and efficient log file compression, Przemysław Skibiński and Jakub Swacha
- [14] Transform Methods Used in Lossless Compression of Text Files, Radu RADESCU
- [15] International Journal of Advanced Computer Science and Applications, Vol. 3, No.8, 2012
- [16] Lossless Image Compression Technique Using Combination Methods, June 22nd, 2012
- [17] Lossless Image Compression based on Data Folding, Suresh Yerva, Smita Nair, Krishnan Kutty, 04 August 2011
- [18] A new lossless method of image compression and decompression using Huffman coding techniques, Jagadish H. Pujar, Lohit M. Kadlaskar