International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 2 Issue 7, July - 2013

An Algorithm For Lossless Text Data Compression

Rajinder Kaur [1], Er. Monica Goyal [2]

[1] *Student, Department of Computer Science & Engineering,*
Guru Kashi University,
Talwandi Sabo, Punjab, India

[2]*AP, Department of Computer Science & Engineering,*
Guru Kashi University,
Talwandi Sabo, Punjab, India

Abstract
In this paper we reduce the number of bits required to represent a character by using 6-bit binary coding instead of a 8-bit binary coding technique. We use a numbering map for converting the input data for introduce a way to use the binary form in a dynamic way, which has never used for coding data before, and we further use 6-Digits binary representation of alphabet with lowercase and uppercase with some extra symbols that are most commonly used in our text files. We found a new way to decrease the length of the bits string, which is only possible in 6-bit binary representation thus drastically reducing the length of the code. In our technique we also use the Huffman method for increasing our methodology result percentage.

Keywords-Lossless Data, Compression methods, Binary Coding, Decompression.

1. Introduction

Data compression is a method to develop storage capacity by eliminating redundancies that happen in most text files. The compression methods are classified in two ways, lossy and lossless. Lossy compression method reduced file size by eliminating some data that won't be get back by user after decompression, this often used by video and audio files. But Lossless compression method modifies each bit of data inside file for reducing the size without losing any information after

decompression. So this is most useful because if file lost even a single bit after decoding, that mean the file is corrupted. Data compression may be used for network processing in order to save energy because it reduces the amount of data in order to increase the data transmitted bandwidth and decreases transfer time. Most data compression methods consist of one or a combination of different data compression methods. They are using different ideas, which suitable for different types of data, and produce different results, but they are all based on the same base, because they all compress data by removing redundancy from the original data in the input file. The purpose of compression having two components, an encoding algorithm that takes a message and generates a "compressed" output (using lesser bits), and to decoding algorithm that gives the original message or some approximation of it without losing information. These two components are typically attached together since they both have to understand the shared compressed representation.

Huffman algorithm is a technique for generating the minimum redundancy code. It is using three steps for process .The first step is to analysis the file to be compressed and to build the code tree. The second step is to compress the file based on

Huffman codes generated by the code tree in step1. The third step is to decompress the file back to get its original form.

2. Related Work

The data Compression methods have a long list. In this paper we shall consider only the lossless compression methods and not the lossy ones because of related to our work. The work was started with the help of Huffman, under this technique pieces of data (segments or bits) in a file are modified by a code of shorter length. We also read the LZ family compression algorithms which are mostly used in these days. After Huffman for floating values input arithmetic coding is a useful technique that gives good compression ratios. Recently, there has also been a lot of research done for finding the efficient searching algorithms within text files compressed by LZ family. Many searching algorithms use indexing values for better results. But no one of the algorithms provides desirable results. So we work on the text files compression using existing methods for getting good result with the help of our developing method.

A universal method is *optimal* if the compressor produces compression factors that asymptotically meet the entropy of the input data stream for long inputs. *Compression performance*: Several factors are mainly used to express the performance of a compression method. The Compression ratio is one of the main factor to express compression efficiency and is defined as Compression ratio =Size of the output data/size of the input data. A value of 0.7 means that after compression the data occupies 70% of its original size.

3. Existing Method Used By Our Algorithm:-

A basic method for data compression is Huffman coding. It serves for several popular programs used in personal computers. The Huffman algorithm is simple and mainly used for creating a Huffman code tree. The five steps for building this tree is:

1. Start with a list of free nodes each having their frequencies, where each node corresponds to a symbol in the alphabet.
2. Select two free nodes with the minimum frequencies from the list.
3. Create a parent node for these two nodes selected and the frequency is equal to the frequencies of the sum of two child nodes.
4. Remove the two child nodes from the list and add the parent node to the list of free nodes.
5. Repeat the process starting from step-2 until only a single node remains. After building the Huffman tree, the algorithm creates a prefix code for each symbol simply by traversing the binary tree from the root to the node, which corresponds to the symbol. It assigns 0 for a left branch and 1 for a right branch. The Huffman algorithm is called as a *semi adaptive* or *semi-static* because it requires knowledge of frequencies for each node. Along with the compressed output, the Huffman tree with the Huffman codes for nodes or just the frequencies of symbols which are used to create the Huffman tree must be stored. This information is needed during the decompression process.

4. PROPOSED ALGORITHM

To improve the compression ratio for text files, a compression algorithm is designed which is specialized for text application protocols, instead of general purpose compression methods. The basic idea of the specialized compression algorithm is the introduction of a specific encoding scheme which is actually used an improved version of Huffman encoding scheme for all redundant words in a text files.

We have implemented the algorithm on c# platform with vb.net beans framework.

Bit Reduction Algorithm-

Data compression is always useful for encoding information using lesser number of bits than the original representation it would use. There are many applications where the size of information would be critical. In data communication, the size of data can affect the storage cost. This algorithm was originally implemented for use in

a text file like message communication application. The idea in is this program reduces the standard 8-bit encoding to some application using specific 5-bit encoding system and then pack into a byte array. This method will reduce the size of a string considerably when the string is lengthy and the compression ratio is not affected by the content of the string.The Algorithm

1. Compression: -

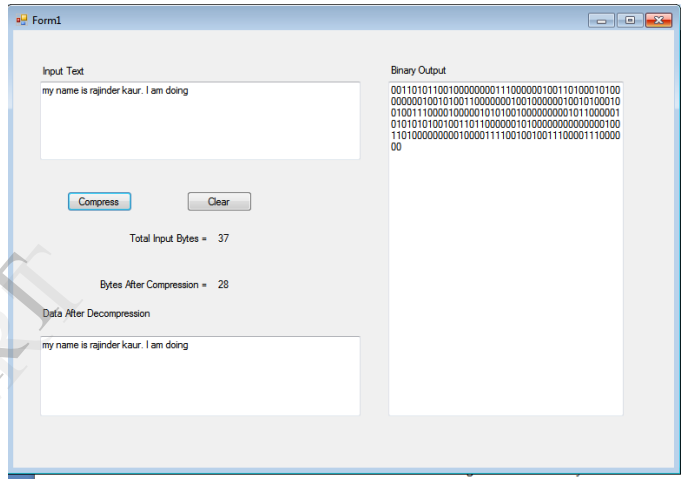
Let's assume that we have a input string with 8 characters. If we put this on a byte array, we get a byte array with the size of 8. A single character will need 8 bits if the characters are represented with ASCII values. A set of 8 bits can represent 2^8 different characters. But if we consider the application, a simple text data might be included only around 26 different characters. Therefore it is need to have 5-bit encoding which can give up to 2^5 different characters to represent. For converting into the new 5-bit encoding, we assign new values to the alphabet characters like | a = 1 | b=2 | c=3 | d=4 | e=5 | f=6 | g=7 | h=8 |. If we look more closely at the new byte array, it will look like the following (the values of characters are in binary representation). 00000001|00000010|00000011|00000100|00000101|00000110|00000111|00001000| But we use 8 bytes for storing the 8 characters. In the next step, we will cut three bits from the position of 3rd bit from the left side and extract the 5 least significant bits. The result will be shows as follows:
|00001|00010|00011|00100|00101|00110|00111|01000|.Now we have reduced 8 bytes to 5 bytes. The next step shows how these 5 bytes convert to the 8 bytes and we get the original information.

2. Decompression: -

When an array of bytes is shown, each character should be represented in the binary form. Then all the 1's and 0's should be arranged as their index values and all data split to the sets of five bits. After splitting data, it will be as follows: Code |00001 000|10 00011 0|0100 0010|1 00110 00|111 01000| then these sets converted to decimal values represent the characters that we have compressed. Code |00001 = 1(a) 000|10 = 2 (b) 00011 = 3(c) 0|0100 = 4 (d) 0010|1 = 5 (e) 00110 = 6 (f) 00|111 = 7 (g) 01000| = 8 (h). Then the information will be shown in original form as "abcdefgh".

5. Result:-

Snapshot



Lenth of Text	No. of Experiments	Out Put	Bytes Saved
100	5	75	25
149	15	112	98
427	33	321	106
572	19	429	143
700	4	525	175

6. Conclusion

In our paper a new text data compression algorithm is produced. The most important feature of our algorithm is its simplicity. An entirely different technique is developed to decrease the size of text files. The technique of 'saving space' have shown in this algorithm. Since every character is taken care of, so the output codes do not depend upon the

redundancy, like the traditional compression algorithms. After the code formulation, ASCII code modifies the binary numbers, which finally reduce the file size. A lot of research and findings led to the conclusion that there are no such algorithms in data compression that emphasis on different compression based on number theory and bit reduction.

7. References:-

1.S. S. Nayak, S.P Sahoo, R. Matta, and S. k. Pattanayak,"A Modified Approach to Lossless data compression method" International Journal of Research in Engineering, IT and Social Sciences,November 2012

2. N. PM, Dr. R.M.Chezian,"A Survey on lossless dictionary based data compression algorithms" (Februray 2013)

3. P.Yellamma Dr.N. Challa," Performance Analysis Of Different Data Compression Techniques On Text File", International Journal of Engineering Research & Technology (IJERT) Vol. 1 Issue 8, October – 2012

4. J.Abel, W.Teahan,"Universal Text Preprocessing For Data Compression"(IEEE)

5. S Sankar ,Dr. S Nagarajan," A Comparative Study: Data Compression on TANGLISH Natural Language Text", *International Journal of Computer Applications (0975 – 8887) Volume 38– No.3, January 2012*

6. Md. R. Hasan, " Data Compression using Huffman based LZW Encoding Technique ", International Journal of Scientific & Engineering Research, Volume 2, Issue 11, November-2011

7. M. Gupta B. Kumar," Web Page Compression using Huffman Coding Technique", *International Conference on Recent Advances and Future Trends in Information Technology (iRAFIT2012*).