Volume 3, Issue 3, March 2013 ISSN: 2277 128X
International Journal of Advanced Research in Computer Science and Software Engineering
Research Paper
Available online at: www.ijarcsse.com

A Survey on Different Compression Techniques and Bit Reduction Algorithm for Compression of Text/Lossless Data

Rupinder Singh Brar
Student of masters of technology Computer Science
Department of Computer Science and Engineering
Sri Guru Granth Sahib World University
Fatehgarh Sahib, Punjab, India.

Bikramjeet singh
Student of masters of technology Computer Science
Department of Computer Science and Engineering
Sri Guru Granth Sahib World University
Fatehgarh Sahib, Punjab, India.

Abstract--Compression is useful because it helps us to reduce the resources usage, such as data storage space or transmission capacity. Compression methods have a long list. In this paper , reviews of different basic lossless data and lossy compression techniques are considered. On the basis of these techniques we try to purpose a bit reduction algorithm used for compression of text data which is based on number theory system and file differential technique. It is a simple compression and decompression process which is free from time complexity.

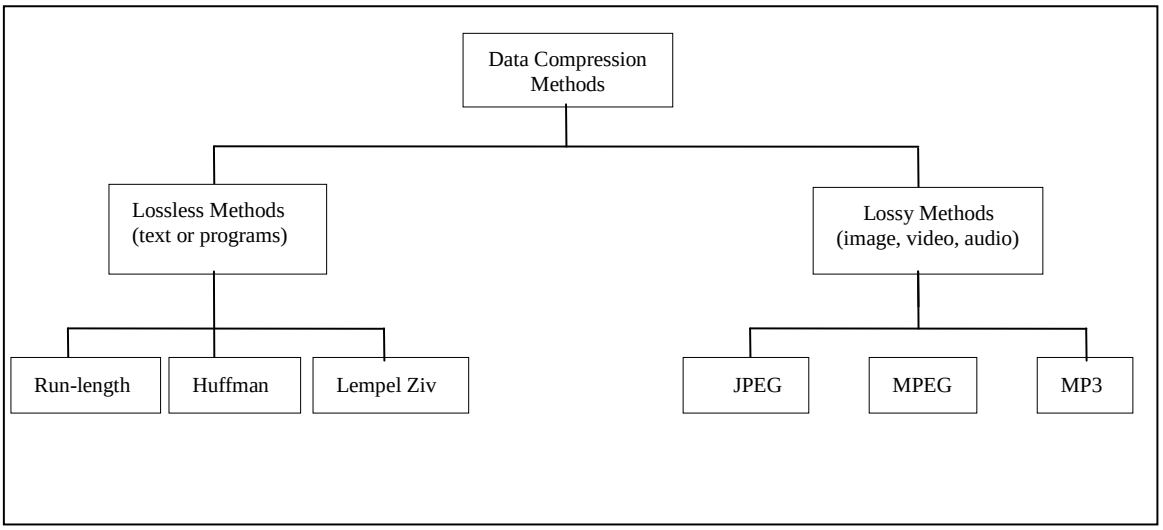
Keywords -- Compression, Number system, lossless compression, lossy compression,Bit Reduction Algo

I. Introduction

COMPRESSION[1]

In computer science and information theory, 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 identifying marginally important information and removing it. The process of reducing the size of a data file is popularly referred to as data compression, although its formal name is source coding (coding done at the source of the data, before it is stored transmitted).

Compression is useful because it helps reduce resources usage, such as data storage space or transmission capacity. Because compressed data must be decompressed to use, this extra processing imposes computational or other costs through decompression, this situation is far from being a free lunch. Data compression is subject to a space-time complexity trade-off. For instance, a compression scheme for video may require expensive hardware for the video to be decompressed fast enough to be viewed as it is being decompressed, and the option to decompress the video in full before watching it may be inconvenient or require additional storage. The design of data compression schemes involve trade-offs among various factors, including the degree of compression, the amount of distortion introduced (*e.g.*, when using lossy data compression), and the computational resources required to compress and uncompress the data. New alternatives to traditional systems, which sample at full resolution then compress, provide efficient resource usage based on principles of compressed sensing. Compressed sensing techniques circumvent the need for data compression by sampling off a cleverly selected basis.



Tree Representation of Compression Methods

A. LOSSLESS[1]

Lossless data compression algorithms usually exploit statistical redundancy to represent data more concisely without losing information. Lossless compression is possible because most real-world data has statistical redundancy. For example, an image may have areas of color that do not change over several pixels; instead of coding "red pixel, red pixel, " the data may be encoded as "279 red pixels". This is a simple example of run-length encoding; there are many schemes to reduce size by eliminating redundancy.

B. LOSSY[1]

Lossy data compression is contrasted with lossless data compression. In these schemes, some loss of information is acceptable. Depending upon the application, detail can be dropped from the data to save storage space. Generally, lossy data compression schemes are guided by research on how people perceive the data in question. For example, the human eye is more sensitive to subtle variations in luminance than it is to variations in color. JPEG image compression works in part by "rounding off" less-important visual information. There is a corresponding trade-off between information lost and the size reduction. A number of popular compression formats exploit these perceptual differences, including those used in music files, images, and video.

Lossy image compression can be used in digital cameras, to increase storage capacities with minimal degradation of picture quality. Similarly, DVDs use the lossy MPEG-2 Video codec for video compression.

In lossy audio compression, methods of psychoacoustics are used to remove non-audible (or less audible) components of the signal. Compression of human speech is often performed with even more specialized techniques, so that "speech compression" or "voice coding" is sometimes distinguished as a separate discipline from "audio compression".

LOSSLESS COMPRESSION METHODS*[10]*

i. Repetitive Sequence Suppression or Run Length Encoding

The first step in this technique is read file then it scans the file and find the repeating string of characters .when repeating characters found it will store those characters with the help of escape character followed by that character and count the binary number of items it is repeated. This method is useful for image having solid black pixels. This algorithm is also effective for repeating of characters. But it is not effective if data file has less repeating of characters. We can compress the run-length symbols using Huffman coding, arithmetic coding, or dictionary based methods.

ii. Shannon Fano Coding Technique

It is used to encode messages depending upon their probabilities [1]. The method is defined as given below:-

1. For a given list of symbol create a probability table.
2. Sorting the table based on the probability and places the most frequent symbol at the top of a list.
3. The table is divided into equally two halves upper and lower which having a same probability as much as possible.
4. The upper half of the list defined with „0" digit and the lower half with a „1".
5. Repeat the steps 3 and 4 for each of the two halves then further divide the groups and adding bits to the codes and stop the process when each symbol has a corresponding leaf on the tree.

iii. Arithmetic Coding Technique

Arithmetic coding is change the method of replacing each bit with a codeword. So it replaces a string of input data with a single floating point number as a output. The main purpose of this technique is to given an interval to each potential bit data.

1. Arithmetic coding is a more modern coding method that usually than Huffman coding.
2. Huffman coding assigns a codeword to each symbol which has an integral bit length.
3. Arithmetic coding can treat the whole string data as one unit.
4. A message is represented by a half-open interval [a, b) where a and b are real numbers between 0 and 1. Initially, the interval is [0, 1). When the message becomes longer, the length of the interval shorts and the number of bits needed to represent the interval increases.

iv. LZW (Lempel-Ziv Welch) compression method

LZW is the most popular method. This technique has been applied for data compression . The main steps for this technique are given below:-

Firstly it will read the file and given a code to each character. If the same characters are found in a file then it will not assign the new code and then use the existing code from a dictionary. The process is continuous until the characters in a file are null.

Purposed algorithm-BIT REDUCTION ALGORITHM FOR TEXT COMPRESSION

We have implemented the algorithm on java platform with net beans framework.

i. ***Implementation of Bit Reduction Algorithm-***

Data compression is always useful for encoding information using fewer 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 cost too.

This algorithm was originally implemented for use in an SMS application. Using this algorithm, it could send about 256 characters per message (typically 160 characters per message) through the same 7-bit GSM network. The idea is, this program reduces the standard 7-bit encoding to some application 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. Encoding -

Let's assume that we have a string with 8 characters (example: - "abcdefgh"). If we put this on a byte array, we get a byte array with the size of 8. The following diagram shows how these ASCII characters can store in an array.

|97|98|99|100|101|102|103|104|

A single character will need 8 bits if the characters are represented with ASCII. A set of 8 bits can represent 256 different characters. But if we consider the current application, a simple SMS might be included only around 26 different characters. Therefore it is enough to have 5-bit encoding which can give up to 32 different characters to represent. For convert to 5-bit, let's assign new values to the above characters.

| 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 still use 8 bytes for storing the 8 characters. In the next step, we will chop each byte from the position of 3rd bit from the left side and extract the 5 least significant bits. The result will be as follows:

|00001|00010|00011|00100|00101|00110|00111|01000|

We can rearrange these bits in an array of bytes as follows:

|00001000|10000110|01000010|10011000|11101000|

Now we have reduced 8 bytes to 5 bytes. The next section shows how these 5 bytes convert to the 8 bytes and get the original information.

2. Decoding -

When an array of bytes is given, each byte should be represented in to binary. Then all the 1's and 0's should be arranged as their index and then can be split to the sets of five bits. After splitting, it will be as follows:

Collapse | Copy Code

|00001 000|10 00011 0|0100 0010|1 00110 00|111 01000|

These sets can be converted to decimals and these values represent the characters that we have encoded.

Collapse | Copy 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 can be decoded as "abcdefgh".

ii. Bit Reduction Algorithm in steps-

1. Pick up a plain text or file which contain the text.

2. Apply compression algorithm-BIT REDUCTION ALGO

3. Bit Reduction algorithm-Which is more efficient and less complex than Huffman coding algorithm and also from that algorithm which is defined in base paper.
4. Steps to be followed in Bit reduction algorithm-

- Pick up characters from the text file which is to be encoded and obtain their ASCII code correspond to them.
- Obtain the binary code from these ASCII key codes corresponding to each character.
- Then put these binary no. into array of byte(8bit array).
- Chop up extra bits from binary no like extra 3 bits from the front.
- Then rearrange these into array of byte and maintain the array.
- Final text will be encoded and as well as compression will be achieved.
- Now decompression will be achieved in reverse order at the client-side.

II. Conclusion

In this paper a new data compression algorithm is introduced. The unique feature of this algorithm is its simplicity. An entirely different technique is employed to reduce the size of text files. The technique of 'saving bits' is employed in this algorithm. Since every character is taken care of, so the output codes do not depend upon the repetition, like most of the other compression algorithms. Different combination of characters can be represented by fewer numbers of bits. After the code formation, ASCII replaces the binary numbers, which finally reduces the file size. The compression algorithm takes O(n) time, where n is the total number of characters in the file. Since the differential breaking follows Divide and Conquer policy, it takes O(nlogn) time. So, the total computation time required for this algorithm is proportional to O(nlogn). Quite a lot of research and findings led to the conclusion that there are no such algorithms in data compression that lay emphasis on differential compression based on number theory and bit reduction. Future work can be done on coding of special character which are not specified on key-board so that better results are revised.

References

[1] S.Gavaskar, Dr.E.Ramaraj, R.Surendiran, "A Compressed Anti IP Spoofing Mechanism Using Cryptography ," *IJCSNS International Journal of Computer Science and Network Security,* VOL.12 No.11, November 2012 .

[2] I Made Agus,Dwi Suarjaya ," A New Algorithm for Data Compression Optimization, "*International Journal of Advanced Computer Science and Applications (IJACSA)*, Vol. 3, No.8, 2012.

[3] Mark Daniel Ward ," Exploring Data Compression via Binary Trees1," *International Journal of Advanced Computer Science and Applications (IJACSA)*, Vol. 3, No.8, 2012.

[4] Manjeet Gupta Brijesh Kumar ," Web Page Compression using Huffman Coding Technique," *International Conference on Recent Advances and Future Trends in Information Technology (iRAFIT2012) Proceedings published in International Journal of Computer Applications® (IJCA),*2012.

[5] Zhenhai Duan, Xin Yuan, Jaideep Chandrashekar," Controlling IP Spoofing Through Inter-Domain Packet Filters," *IEEE TRANSACTIONS ON DEPENDABLE AND SECURE COMPUTING,*VOL. 5, NO. 1, JANUARY-MARCH 2008.

[6] Cedric Westphal," Improvements on IP Header Compression," *Global Telecommunications Conference, (GLOBECOM '03)*, January 2003

[7] Anat Bremler- Barr, Hanoch- Levy," Spoofing Prevention Method," *PODC'04*, July 25–28, 2004, St. Johns, Newfoundland, Canada.

[8] L. Gao and J. Rexford," Stable internet routing without global coordination," *IEEE/ACM Transactions on Networking*, vol. 9, no. 6, Dec.2001.

[9] K. Park and H. Lee ," On the effectiveness of route-based packet filtering for distributed DoS attack prevention in power-law internets*," In Proc. ACM SIGCOMM*, San Diego, CA, Aug. 2001.

[10] Rajinder Kaur ,Mrs. Monica Goyal ,"A Survey on the different text data compression techniques ," *International Journal of Advanced Research in Computer Engineering & Technology (IJARCET)*, Volume 2, Issue 2, February 2013 .

[11] An introduction to IP header compression, URL: WWW.EFFNET.COM.

[12] Cisco Content Services Switch SSL Configuration Guide-*CSS Compression Overview*.

