Web Page Compression using Huffman Coding Technique

Manjeet Gupta
Assistant professor,Department of CSE
JMIT ,Radaur

Brijesh Kumar
associate professor, Department of IT
Lingyaya's university

ABSTRACT
Compression helps in reducing the redundancy in the data representation so as to reduce the storage requirement of it. Compression is an important technique used to improve web retrieval latency Compression is an important technique used to improve web retrieval latency. Plethora of algorithms is available for compressing the data. Some of the algorithms helps in achieving lossless compression and some are good at lossy compression. The objective of this study is to analyze the amount of compression that can be achieved by the use of existing Huffman Coding on web pages and to suggest further improvement that can be done to compress web pages so as to achieve better compression ratio and compression efficiency. The proposed technique may works even better with large files. How d o u know . This paper also outlines the comparison of various compression methods using different parameters.

General Terms
Huffman codes, Compression algorithm.

Keywords
Compression , Compression ratio, Redundancy codes, lossless.

1. INTRODUCTION
The lavishing amount of web pages on internet has increased the demand for bandwidth requirement. Especially with the growing technology where lots and lots of web pages containing every field of knowledge, commerce, and communication have become available. Over the last decade there has been an unprecedented explosion in the amount of digital data transmitted via the Internet in the form of text, images, video, sound, computer programs, etc. If this trend expected to continue, then it will be necessary to develop a compression algorithm that can most effectively use available network bandwidth by compressing the data at maximum level. Along with this it is mandatory to consider the security aspects of the compressed data transmitting over Internet, as most of the text data transmitted over the Internet is very much vulnerable to an attack. Compression can be done on Image files and data files. Image compression is the application of Data compression on digital images [6]. Compression is the process of reducing the amount of data needed for storage or transmission of a given piece of information (text, graphics, video, sound, etc.), typically by use of encoding techniques [4]. Data compression stands for compressing data or files containing data so that they can be stored in much less memory space that they had been stored in their original form. It lets you store more stuff in the same space, and it helps to transfer that stuff in less time, or with less bandwidth [3]. The task of compression consists of two components, an encoding algorithm that takes a message and generates a "compressed" representation (hopefully with fewer bits), and a decoding algorithm that reconstructs the original message or some approximation of it from the compressed representation. These two components are typically intricately tied together since they both have to understand the shared compressed representation. There are two types of compression algorithms – lossy and lossless [7]. Lossless algorithms, which can reconstruct the original message exactly from the compressed message, and lossy algorithms, which can only reconstruct an approximation of the original message. Lossless algorithms are typically used for text, and lossy for images and sound where a little bit of loss in resolution is often undetectable, or at least acceptable.

Huffman algorithm [2] is a method for construction of minimum redundancy codes .It is a three step process .The first step is to analyze the file to be compressed and to built the code tree. The second step is to compress the file based on Huffman codes generated by the analyzation step. The third step is to decompress the file back in to its original form[1]. Huffman developed a coding procedure for statistically independent source in order to minimize the average code length. A technique is introduced which compresses a web page at the server –side and decompresses again on the client side so that the same page can be transferred to client- side in lesser time[8]. The technique has been tested using a java based program. The program firstly asks for the web page to be compressed, analyze it, compresses the contents using Huffman Encoding Scheme and stores the compressed file in an output html file. The file also contains the decoding information in it. When the browser request this page from web server, the compressed page is transferred in lesser time to the client side, the page is loaded in to the browser, but before actually displaying the page, a small script is executed, to decode the page. The decoded page is actually shown to the client. Many web page compression tools are available in the market. Most of them include two processes for compressing a web page - removing extra blank spaces and removing comments.
To compress a web page using Huffman
* build a table of bit encoding, table giving a sequence of bits that's used to encode characters or may build the table from a Huffman coding tree.

* Read the web page to be compressed and process one character at a time. To process each character find the bit

sequence that encode s the characters using the table built in previous step and write this bit sequence to the compressed file.

* The page is then loaded on the web server so that clients can fetch them.

When the compressed page is fetched by the user through a web browser, the compressed web page is transferred to client side in lesser time using lesser bandwidth because of lesser size. As the browser starts rendering the web page, a small java script is executed automatically, which decodes the web page contents by replacing the encoded characters with actual words. Because the major chunk of the time is consumed in data transmission, rather than decoding the web page, the page is displayed in lesser time. Compression has been claimed to be an attractive solution to save energy consumption in high-end servers and data centers. A comprehensive evaluation of energy consumption for various file compression techniques implemented in software has been already presented. Various compression tools available on Linux are applied to a variety of data files, and tried on server class and workstation class systems. Their energy and performance results are compared against raw reads and writes. The results revealed that software based data compression cannot be considered as a universal solution to reduce energy consumption. Various factors like the type of the data file, the compression tool being used, the read-to-write ratio of the workload, and the hardware configuration of the system impact the efficacy of this technique. In some cases, however, it is found that compression save substantial energy and improve performance.

2. RELATED WORK

Several transformations have been applied to pre-process the text to make it more compressible by existing algorithm [9]. The transformed text can be compressed better when most of the available compression algorithms are applied. The three transformations called Star Encoding where a word is replaced by characters '*' and at most two characters of the original word, Fixed Context Length Preserving Transformation (LPT) where strings of * characters are replaced by a fixed sequence of characters in alphabetic order sharing common suffixes depending on the lengths of the strings (viz. 'stuvw' or 'qrstuvw' etc), Fixed Context Reverse LPT (RLPT) which is same as LPT with the sequence of characters reversed and shortened-context LPT (SCLPT) where only the first character of LPT is kept, which uniquely identifies the sequence. All of these transforms improve the compression performance and uniformly beat almost all of the best of the available compression algorithms. It is estimated that in the year 2004 the National Service Provider backbone has estimated traffic about 30000Gbps and that the growth continues to be 100% every year[10]. The text data competes for 45% of the total internet traffic. A number of algorithms for lossless compression, out of which Huffman , BWT and PPM out perform the classical algorithms like arithmetic and LZ families of Gzip[6]. Whereas for image compression various algorithms are implemented like RLE(Run length Encoding), JPEG 2000, Wavelet Transform, SPIHT(Set Partition in Hierarchical Trees) it has been observed that can achieve higher compression ratio for MRI, Ultrasound ,CT scan and iris images by SPIHT method[11]. Furthermore we also observe that for MRI image wavelet compression method has higher compression ratio and has good PSNR value for iris image than JPEG method Compression ratio is almost same of iris and MRI image[3]. For CT scan image JPEG compression method outperforms the PSNR and degree of compression than wavelet compression method. SPIHT is the most efficient method in respect of compression ratio and PSNR vale. With the increase in the degree of compression SPIHT keeps the image quality [2]. In case of Huffman compression the adaptive Huffman coding technique outperforms non adaptive Huffman [2] coding. In an experiment both were tested against the same text and image files. The text file was a DBase I11 file, copied to an ASCII file with blank separators, containing hospital biomedical equipment information. The file was acquired from a nearby hospital. The image file is from a Vax 11/750**.** It was captured using Vax Tips. It has been observed that non adaptive Huffman coding produced 59.1377% compression ratio compared to the 54.9974% produced by the adaptive technique for the text file. The results were even more significant for the image file. The non-adaptive technique produced a compression ratio of 55.0303% where as the adaptive technique for 2 characters produced a compression ratio of 40.9873%[1].

3. PROPOSED ALGORITHM

To improve the compression ratio for web page, a compression algorithm is designed which is specialized for application protocol, instead of general purpose compression algorithms. The basic idea of the specialized compression algorithm is the introduction of static dictionary including the control messages of the application protocol and a specific encoding scheme which is actually an improved version of Huffman encoding scheme for all redundant words in a web page.

Huffman Encoding, an algorithm for the lossless compression of file is based on the frequency occurrence of a symbol in a file that is to be compressed. This Algorithm is a variable – length coding system that assigns smaller codes for more frequently used characters and larger codes for less frequently used characters in order to reduce the size of files being compressed & transferred, and opened at the client side will execute a small java script program, which will decompress the contents on page and the same will be displayed to the user . It has been observed that compression ratio and the time to compress a web page increases proportionately with the time. The compression /Decompression technique presented in this paper can be used for faster web page transfer to client machine.

4. METHODOLOGY

The technique of compressing the web page works on the basis of frequency of occurrence of letters, using frequency analysis and encoding scheme. A java-based program, being used to compress the web page, is using Huffman encoding scheme to encode and compress the web page contents. Java Program identifies & stores the statistics of each of the word in a vector/array. The cost benefit analysis is done between lengthy

words with lesser frequency and comparatively smaller words with higher frequency.

The first step of compression with Huffman Encoding includes the building of a table of bit encoding, the table gives a sequence of bits that is used to encode characters or may it can be expressed as Huffman coding tree[5]. The second step is to read the web page to be compressed and process one character at a time. To process each character find the bit sequence that encodes the characters using the table built in step one and write this bit sequence to compressed file. The third step is loading of page on the web server, so that client can fetch them.

When the compressed page is fetched by the user through a web browser, the compressed page is transferred to the client side in lesser time using lesser bandwidth because of the smaller size. As the browser starts rendering the web page, a small java script is executed automatically, which decodes the web page contents by replacing the encoded characters with actual words? Because the major chunk of the time is consumed in data transmission, rather than decoding the web page, the page is displayed in lesser time.

5. RESULT ANALYSIS

The results of compressing Web pages with the Huffman encoding are shown below in a Figure1 using MATLAB . It is clear that the amount of compression increases with the increase in size of the web page. For smaller pages which are less than 50KB, the size of compressed file becomes bigger .So for all practical purposes the lower limit for compressing the web page was set to be 50 KB.

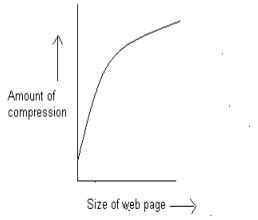

Fig1: Compression Graph

It was also observed that the C.P.U time consumption in compressing a web page using the developed tool. .,increases linearly with the increase in the size of the web page as shown in Figure 2.

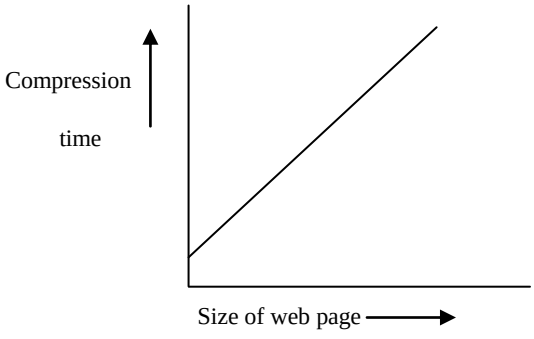

Fig2: Compression time graph

The compression technique introduced can be beneficial in achieving larger browser compatibility. Most of the browsers including the most popular one: Mozila and Microsoft Internet explorer, are capable of executing the java script. The same scripting language is used in the compressed web page which leads to compatibility. It will also provide security to the web pages if they are compressed and then transmitted using secure http, so that it becomes a bit difficult to decode the page.

6. CONCLUSION

The proposed web page compression technique has been found to be very effective, in compressing the web page up to 70-80%,in some cases. But still there is a scope of improvement in order to best utilize the C.P.U time. The existing Apache and IIS module also support web page compression at the server end . They give effective compression ratio on web, but both have a limitation. Both of the servers compress the files at server end but before transferring the page to client machine the page is uncompressed. So efforts can be made related to compression of pages on client side.An effort will be made to implement proposed scheme of sending compressed pages in existing Apache and IIS web servers.

7. REFERENCES

[1] Ahmed Desoky and Mark Gregory, Compression of text and binary files using Adaptive Huffman Coding Techniques.In Proceedings of IEEE conference, Aug, 2002, U.S.A.

[2] D.A. Huffman 1952,A method for the construction of minimum redundancy codes, IRE 40, 9, (Sept 1952), 1089- 1101.

[3] David A. Clunie, 2000, Lossless Compression of Grayscale Medical Images- Traditional and State of the Art (Approaches,www.dclunie.com/papers//spie_mi_2000_compression).

[4] H.Kruse and a. Mukherjee, Data compression using Text encryption, Proceedings of the Data compression Conference, 1997,IEEE computer Society Press,pp.447.

[5] Owen L.Astrachan,2004,Huffman coding:A CS2 assignment.

[6] F.Awan, and A. Mukherjee ,LIPT: A lossless Text Transform to improve compression, Proceedings of International conference on information and Theory :Coding and Computing, IEEE Computer Society ,Las Vagas Nevada ,April 2001.

[7] Manber U,1994 A Text Compression scheme that allows fast searching directly in the compressed file ,pro.of ombinatorial pattern matching (lecture Notes in computer science V807), Springer: London ,113-124.

[8] T.Welch,1984 “A Technique for High-Performance data Compression”,IEEE Computer,Vol 17.

[9] M.Burrows and D.J. Wheeler. Nov2011, A Block-sorting Lossless Data Compression Algorithm,SRC Research Report 124,Digital Systems Research centre.

[10] J.Ziv and A.Lempel, 1977,A universal algorithm for sequential data compression, IEEE Trans.inform.Theory, vol.IT-23, no.3, pp.337-343.