AN APPROACH TOWARDS DEVELOPMENT OF EFFICIENT DATA COMPRESSION ALGORITHMS AND CORRECTION TECHNIQUES

By
JYOTSNA KUMAR MANDAL
Dean, Faculty of Engineering, Technology & Management
Professor,
Department of Computer Science and Engineering
University of Kalyani, Kalyani,
Nadia, West Bengal, INDIA

A thesis submitted to ***Jadavpur University***
for the degree of
DOCTOR OF PHILOSOPHY(*Engineering*)

1999

The Schemes

The present work is divided into two specific problems, one is to reduce the redundancy of error-controlled coding for cost and time effectiveness and other is source independent coding for utilization of storage space and to reduce transmission overhead.

In chapter 2, two universal data compression techniques are proposed. These are - Pivotal Difference Method (PDM) and Running Difference Method (RDM). Repeated use (cascading) of these thechniques over same file/input stream is also proposed to achieve better compression ratio. Chapter 3 deals with proposed Nibble Huffman (NH) Method. A cascading of PDM and RDM with Nibble Huffman Method (NH) is also proposed in this chapter. In chapter 4 well know LZ method is described. Cascading of PDM and RDM with LZ method is also proposed in chapter 4 which shows that the cascaded compression is better than that obtained by LZ alone. A comparison of cascading among PDM-NH, RDM-NH, PDM-LZ and RDM-LZ are also performed in chapter 4. Chapter 5 deals with implementation effect of PDM and RDM on image file. A cascading of both PDM and RDM separately with LZ are also done here and the performances are studied. A comparison with JPEG Lossy compression is also performed. A lossy compression technique using PDM and RDM is developed in chapter 6. Cascading of these techniques with LZ method and repeated use of PDM and RDM are also done and performances are compared. Chapter 7 proposed a scheme of Error Correction using Paired Parity Method (PPM) and its advantage in comparison with Parity Error Correction is discussed. A multilevel cascaded cascaded scheme for cost-effective and reliable transmission using PDM or RDM cascaded with LZ and PPO is also proposed. Discussions are given in chapter 8 and references are drawn in chapter 9.

So, in the present thesis some algorithms are deviced consisting of two aspects of communication - one is compression and other one is error correction. In compression aspects of coding, some methods for space effective storing of information is considered through which time efficient and cost effective transfer of information is possible.

Two data compression methods have been proposed in chapter 2; one is just variant of other. Both come under the category of reversible technique. They are lossless and universal in nature, i.e., both of them find their use in different categories of files i.e.,

text, data and executable file, as well as image file. Another advantage of these techniques is that we don't need to know the probabilities of the occurrence of bytes in the sequence, hence, need only single pass to encode the message. The compression techniques are named as,

1. Pivotal Difference Method (PDM)
2. Running Difference Method (RDM)

In Pivotal Difference Method (PDM), the scheme considers an input file as a sequence of 8 bit bytes. In the output file produced by the compression technique, the first byte is taken as it is, the subsequent bytes are read from the input file and the difference between the binary values of these bytes with respect to the first byte is added to the output file provided the difference is within -30 to +31 i.e., the byte is compressed to six bits. So, we may expect 25% compression.

The problem arises when the difference comes outside the range. This crisis is handled by terminating the input string when the difference is out of range and then restart the process from that byte by adding it as it is in the output string and by calculating the difference again from that new byte as the next pivot, and thus the name of the technique.

One thing should be noted here that, in the output compressed string some input bytes appeared as 8 bits and the others as 6 bits. To discriminate this variation we need to add a marker as the terminator of a string and the beginning of a new pivot which is used in this case is 111111(six ones). Thus actual compression is less than 25% and around 23% to 24%.

Running Difference Method is basically a variant of the previous technique where, instead of using a pivot, the difference of two consecutive bytes is sent to the output string as long as the difference is between -30 to +31.

Here also, the first byte of the input string is taken as it is in the output string, the subsequent bytes are read from the input file and the difference between the binary values of the current byte with respect to the previous byte is added to the output file

provided the difference is within -30 to +31. If the difference goes outside the range the previous string is terminated and the process is restarted by adding the current byte as it is in the output string and then the differences between two consecutive bytes and so on, till the end of file/input string is reached. Here also 111111(six ones) is used as marker as the termination of a string and the beginning of a new string.

The compression ratio in this case also slightly less than 25% but found better than the previous one as it is more unbiased as there is no pivotal concept, the string can extent more than the previous one, thus requiring less marker and more compression.

In a text file the character 'space' is quite frequent which produces the difference out of range. To manage this overhead for space character we suggest to convert the space code from 20_H to $7F_H$ (which is a null character in ASCII). During decoding, this is again converted back accordingly.

Again, since our algorithm considers any file as sequence of bits, this can be repeatedly used for the further compression without suffering any loss. Present thesis considers cascading of upto 4 times with itself, both in case of PDM and RDM and as a result, 55% to 65% compression ratio is achieved for executable files and that for document files such as MS-Word or Word Perfect, a ratio of 60 -70 % is achieved. In case of text files 45% to 60% ratio is obtained. If we increase the number of iterations, ratio of compression may increases further in some cases but, after a few iterations the compression gets saturated. From all of these analyses it is clear that repeated use of RDM and PDM results high degree of compression for all types of files. Even in case of executable files compression ratio is better than standard techniques. If we use normal PDM or RDM on text files there is marginal compression in the first stage of cascading but after subsequent number of repetitions good compression ratio is achieved. For each type of files compression gets saturated after a certain number of iterations and ultimate ratio of compression is high.

A very simple modification of Huffman static compression is also presented in chapter 3 where, bytes are divided into nibbles. These nibbles are taken as a pattern of strings and Huffman's static method is applied on it. A comparison of results obtained in

Pivotal Difference Method (PDM) and Running Difference Method (RDM) with the results of this Nibble-Huffman (NH) is experimented and their relative performances are studied[6,46] . This study shows achievement of high compression ratios in both the cases of cascading of PDM and RDM with Nibble- Huffman (NH). Cascading of RDM with NH gives better results than cascading of PDM with NH. In case of files such as documents under Word Perfect and MS-Words an average compression ratio of 80% is achieved when RDM/PDM cascaded with NH. The ratio in case of executable files in such cascading varies from 75% to 80 %.

PDM and RDM techniques are also cascaded with a universal and most widely used method, LZ[81,100,101] in chapter 4 and the relative performances are studied . This study shows that cascading of PDM or RDM with LZ gives better results in most of the cases particularly in case of executable files. Hence, a sharp increases of compression ratio is observed for PDM and RDM both when cascaded with LZ in comparison with individual LZ for most of the files experimented including executable files but, RDM cascaded with LZ gives much better result than PDM cascaded with LZ in some cases.

From the above analysis of PDM and RDM with NH and LZ methods it is clear that though cascading PDM or RDM with NH or LZ give high compression ratio but cascading of PDM or RDM with LZ give much better results than cascading with NH.

Chapter 5 of the thesis proposed reversible image compression techniques using PDM and RDM. In case of an image matrix / file the difference between two adjacent pixels are very small. Image compression techniques using PDM and RDM use this opportunity to reduce number of bits to store the value of a pixel. As both of PDM and RDM use the difference from a pivoted element or between two adjacent bytes, the techniques may be very much useful in compressing the image matrix to a great extent. Here, in both cases of PDM and RDM, differences are kept as four bit information instead of six bit used in normal PDM and RDM. After keeping the first byte as it is, subsequent bytes are read from the input file /stream and the difference between the binary values of these bytes with respect to the first byte or previous byte is added to the output file provided the difference is within -6 to +7 i.e., the byte is compressed to four bits. In this case a four bit marker, 1111(four ones) is used as the termination of a string and the beginning of a new pivot. Thus, actual compression ratio achieved is less than

50% and is around 48% to 49%. Output of this technique is cascaded with LZ once again, and performances are compared with JPEG lossy compression technique[47, 50, 51, 63] and JPEG compression using arithmetic coding[46, 95] techniques. Observations show that when we compared the results of PDM and RDM cascaded with LZ, RDM give better results than PDM in some cases. But cascading of PDM and RDM with LZ shows better results in comparison to both JPEG and JPEG using Arithmetic coding.

A new cascaded bit level image compression technique which is a reversible graphics compression is also proposed in chapter 5 of the thesis[23, 52, 53]. This technique takes input images which may contain upto32 gray levels. A mapping based compression technique is applied to the input images which produces an intermediate image matrix with a compression ratio varies from 37.5% to 87% depending on number of gray levels in the input images. This intermediate image is taken as input to the next stage which uses arithmetic coding scheme. Results obtained are compared with PDM and RDM cascaded with LZ for different images. These results also show a better compression for the cascaded PDM or RDM with LZ in case of gray scaled images.

In case of a black and white image the above algorithm works in a similar way. The only thing is that we strip off seven bits instead of three from MSB and information is kept as a single bit. But for a 256 level image the pixel /byte is divided by 8 to make it 32 levelled and then the algorithm described above is applied. The scheme is proposed in chapter 6. Reverse procedure is followed at the time of decoding.

A lossy compression technique using PDM and RDM is also proposed in this chapter where, the pixel values are quantized while running PDM and RDM, then, output of PDM and RDM is made input to the LZ method and the performance is compared with individual LZ method. These observations show better compression in cascaded LZ with PDM or RDM than individual LZ.

For the purpose of error controlled coding, a universal error correction technique using Paired Parity Method (PPM) is proposed in chapter 7. Paired Parity Operation (PPO) is defined as cumulative XOR operation between two bits of two consecutive code words alternatively, starting from MSB of lower order upto LSB of higher order codeword for a block of bytes. The result of this operation will emerge as a single bit

output for each operation which is defined as "Paired Parity Bit"(PPB) . The value of PPB is distinct for each operation. Only in some special cases like sequence of block of "gray code" or "j-code."[5,10,44,54,55] PPB is constant. The formula to compute Paired Parity Bit (PPB) can be written as,

$$PPB_{n-bit} = \sum_{i=(n-1)}^{0} \oplus (x_i \oplus y_i)$$

where, $\Sigma\oplus$ represents "XOR-SUM" and $X(x_{n-1}x_{n-2}.....x_0)$ and $Y(y_{n-1}y_{n-2}.....y_0)$ are two consecutive bytes of information.

That means, the PPB can be obtained by making an exclusive-OR operation between two consecutive bytes of information's and then obtaining bitwise exclusive-OR operation of the result. Using PPB, single error can be corrected and multiple errors may only be detected. Because, if a single error occurs in a word at the time of transmission, the value of PPB will be changed for two consecutive PPO. For these two paired parity operations, three adjacent words are needed, of which, the middle word is erroneous. PPB for other sequences in the block will remain unchanged. But in case of boundaries, the situation is different. Here, only one PPB will be changed in each case. The situation can be explained explicitly as follows: for first byte of the block, if single error occurs, PPB in the first position will change its value. If in the last byte (nth) of the block, a single error occurs, PPB in (n-1) th position will change its value. If we apply this scheme in rows and columns then, both erroneous rows and columns can be detected. Cross section of erroneous row and column gives the position of error.

Newly proposed method (PPO) is more cost effective than block parity method. As, in block parity method[5,42], for each block of 8 bytes of data 2.1 byte (17 bits) overhead are needed to detect and correct single error. But, in Paired Parity Method (PPM), for each block of 8 bytes of data exact 2 bytes (16 bits) overhead are needed to detect and correct single error. So, for each 8 byte we can save one bit overhead. Thus for an information of 256 KB we can save 1 KB of transmission overhead. Hence, for the proposed scheme, the transmission will be both cost effective as well as space effective. This scheme may be cascaded with PDM, RDM, and Cascaded PDM with LZ or cascaded RDM with LZ for which transmission may be reliable as well as cost effective. A Multilevel Cascaded scheme for cost-effective and reliable transmission using PDM or RDM cascaded with LZ and PPO proposed for the purpose.