An Efficient RLE Algorithm for Compressing Image Based upon Tolerance Value

Ms. Neha A. Bhatia1, Prof. Himanshu Arora2, Ms. Anuradha Konidena3
Department of Computer Science, CET-IILM, Greater Noida, INDIA.
Department of CS & IT, ACERC, Kukas, Jaipur, INDIA.
Department of Computer Science, CET-IILM, Greater Noid, INDIA.

ABSTRACT

RLE is very effective data compression algorithm for last many years. In this paper we proposed a modified and enhanced variation of RLE data compression algorithm. This replacement strategy will help to avoid long length strings with frequent different characters.

Index Terms: RLE, Data Compression, Compression Ratio.

I. INTRODUCTION

Data compression is a process through which an input data stream is converted into another data stream that is of smaller size [1]. It is a process to cut down the redundancies in data representation so as to decrease data storage requirements and hence communication costs. Reducing the storage necessity is equivalent to increasing the capacity of the storage medium and hence communication bandwidth. Data can be characters in a text file, numbers that are samples of speech or image waveforms, or sequence of numbers that are generated by other processes.

II. RELATED WORK

DC exploits redundancies in data file and converts into a smaller size output file by reducing the redundancy. With rapid advancement in multimedia and internet applications, DC becomes an important research area. DC helps us to compress data and therefore save space and network bandwidth. A better compression technique should work better on each data. To compare efficiency of DC techniques, different parameters can be checked -

1. Compression Ratio
2. Time to compress/decompress data.
4. Quality of data reconstructed after compression.

Most of the compression algorithms emphasize on better compression ratios. A compression ratio can be defined as a ratio between data after compression and data before compression.

A. Types of Data compression

Based on reconstructed data, compression can be -
1) Lossless
2) Lossy

Lossless compression technique involves no loss of information. Whereas lossy compression technique is used where a little loss of information is acceptable.

Some of the lossy techniques are -
1. Run Length Encoding
2. Huffman Encoding
3. LZW
4. Arithmetic Encoding

In this paper, we discuss Run Length Encoding technique, with some of its problems and proposed a method to overcome the problem of traditional RLE.

Run length encoding (RLE) is a technique which says that-
if a data item i occurs n consecutive times in an input stream, replace that n occurrences with a single pair (n,i). The n consecutive occurrences of data item are known as run length and this approach of data compression is known as RLE. The RLE can be applied to image compression as well. It is a loss less compression technique.

III. PROBLEMS WITH RLE ALGORITHM

When applied to image compression, there are some problems which are faced by RLE.

1. The technique of run length coding exploits the high inter-pixel redundancy that exists in relatively simple images. In RLE, gray levels that repeat along each row of image are observed. A ‘run’ of consecutive pixels whose gray level is identical is replaced with two values - the run length and the pixel value.
length and the gray level of all the pixels in the run. Hence, the sequence (10, 10, 10 and 10) becomes (4, 10). RLE can be applied on row-by-row basis, or an image can be considered as 1-D data stream in which the last pixel in a row is adjacent to the first pixel in the next row. This can lead to slightly higher compression ratio. For the special case of binary images, the value of run is not needed, unless it is the first run of the row. This is because there are only two possible values for a pixel in a binary image. If the first run has one of the values, the second run implicitly has the same value as the first, and so on.

2. If the run length is 1, RLE replaces one value with a pair of values. It is therefore possible for RLE to increase the size of the data set in images where run of length 1 are numerous. This might be case in noisy or highly textured images. For example, consider following situation - 46, 47, 48, 44, 43 then RLE will encode it like (1, 46), (1, 47), (1, 48), (1, 44), (1, 43). This is the main problem with RLE.

IV. PROPOSED WORK

As we can use RLE in image compression also, so we are proposing a scheme which is compressing an image with the help of RLE. We have slightly modified the traditional RLE and enhanced it so that it can be used in situations where traditional RLE does not work well. In an image, if a pixel value is changed a little then it will not affect the image a lot. So, RLE can be enhanced as -

ALGORITHM

Step 1 Set the value of T and L.

Step 2 Set the parent element to value 1 and previous element equal to null.

Step 3 Read the next element from input string.

Step 4 If the element is EOF, then write out the pair- count and parent element.

Step 5 Set the next element as current element.

Step 6 If the difference between current element and next element is in the range of plus/minus L, then set the previous element to current element and go to step 2.

Step 7 Read and count additional elements until a non matching element is found or L is achieved.

Step 8 Write out the pair- count and parent element.

Step 9 Write out the next element as parent element and go to step 2.

Flow Chart

V. CONCLUSION

As compression is very useful and an important part in image processing, by using enhanced RLE algorithm, while compressing image, a good compression ratio is achieved . But since enhanced RLE is slightly lossy compression technique so user have to decide up to which extent quality can be recovered after decompression. If user needs higher compression ratio, value L can be increased but it will decrease the quality of image also. In similar way if L is decreased, high quality image can be recovered after decompression. Same thing can be proved with the value of variable T. Since the algorithm is yet to be implemented so results are not attached. But the algorithm will work as proposed.

REFERENCES