ISSN No:-2456-2165

Macronumerical Summing Code

Mahadevan Subramanian
Faculty of Landscape Architecture
Asian School of Architecture & Design Innovations
Kochi, Kerala, India

Abstract:- There are many methods to minimise or shrink binary codes or data compression. Making data shrinkage is better for faster communications and effective storage. Macronumerical summing code is a relatively new technique to shrink binary code structures and you can easily decode it to the original size for without data loss. At present compression is possible only once with the given binary sequence, but using macronumerical summing code you can go on reducing the data to the desired level by repeating the process which is not possible with current available tehnology. In addition there is the possibility to segregate these data easily at the binary code level itself to make a database sorting.

Keywords:- *Macronumerical, Summing, Scan, Matrix, compression,decompression.*

- *Abbreviations:*
  MSC (macronumerical summing code)

I. INTRODUCTION

There are many methods for data compression. Huffmans algorithms, arithmatic coding, sequential data coding and the researches are going on in a rapid pace.

Run length encoding is a kind of coding used in bitmaps and the repetition of data item in succession will be replaced with single code thus the length gets reduced.

Burrows-Wheeler transform work with data blocks as these blocks are compressed bits represented by symbols and be replaced wherever possible Lempel-Ziv algorithm used the text as a dictionary replacing later ocuurance of a string by numbers indicating its occurrence before and its length. Zip and gzip is the other version of this algorithm

Arithmetic coding is one of the best available coding and give a better data compression ratio But till now in all the above methods the compression ratios has a limit. While adopting Macronumerical summing code, you can primarily code a sequence and achieve a number, combine the result with other coded numbers and make a stack of the codes (another matrix and make it optimal by making matrices of 99*99, 999*999, 9999*9999 etc to get maximum benefits of data compression with maximum limited digits) which gives a fresh code and the process can be continued till you get the optimum size ie. the level at which you find the balance of decoding process and data volume to come to the desirable level as the multiple coding will drive you to more data processing. Another major advantage of MSC is its ability to do lossless data compression.

II. EXPLANATION OF MACRONUMERICAL SUMMING CODE

As a primary step we will make a matrix of size 5*5 (refer Figure 1) and fill it with “0” or “X (as red dots)” where “0” is the void cell and “X” represents “1”. Now we check the number of filled cells in X axis, Y axis , Diagonal from left top to right bottom (A diagonal) and diagonal from right top to left bottom (B diagonal). Thus each matrix is scanned for filled cells in four different directions and arrive a number sequence . Refer Figure 1 “12231222210112211100121220 10 (combining X axis, Y axis, X diagonal and Y diagonal in that order) . You can create the same matrix pattern when reversing the reading process i.e. Make a matrix that reads exactly the same sequence we have availed. While scanning a 5*5 matrix we will get 5 digits each in X and Y axis and 9 digits each in both A & B diagonal scanning. Thus a total of 28 digits. When assigning values for each cells in the order from top left to bottom right as 1,2,4,8,16,32,64,128…..sequence we will get a number that equals to 5*5/3.33 digits = 7 (we take only the integer part) as the maximum digits. While we need 28 digits to express the 7 digit number (here we have no advantages in summarising data)

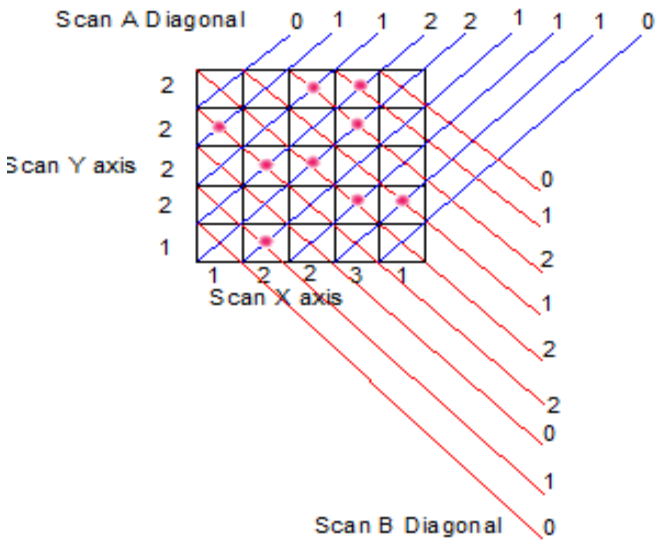

Fig 1:- Reading available on scanning a 5*5 matrix in four directions

ISSN No:-2456-2165

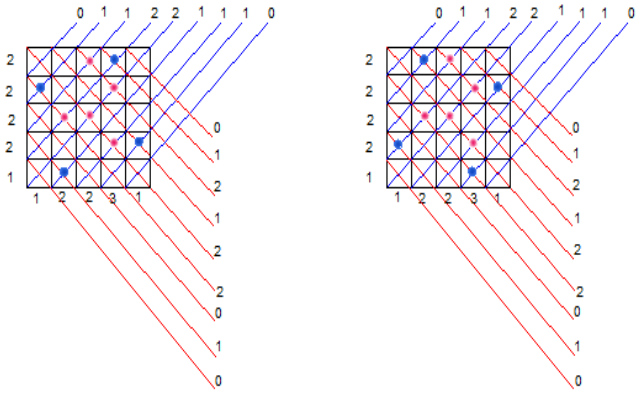

Fig 2:- Reading two 5*5 matrices which has similar reading though different orientation

There are certain orientations as above which give the same reading for different cellular orientation. In the above picture the blue dots have different cellular arrangements, yet give the same reading and this pattern is called isomeric. There can be numerous such isomeric patterns in a larger matrix that give isomeric reading for different orientations and these isomeric patterns make their own permutation and combinations to make isomeric multiple readings. We have made a method to segregate such isomeric designs and give an additional reading then we end up the issue related to the isomeric properties.

For larger matrices (over 37 * 37) the digits originated out of scan will be lesser than the digits availed through summing the value of filled cells. Here the digits required for scanning will be 2(37+37+37+37+36+36)-36=404 (we will get two digits while scanning each rows and there will be one row missing in diagonal scans thus the number 36 and we need to represent only one digit in the diagonal pattern where it is below 10 cells rows and an advantage of four nine digits = 36 which we deducts from total number of digits) We get 411 digits while summing values of the filled cells to the maximum. Thus in this process we will get a digital advantage of 7 (411-404).

While the matrix size increases this advantage in digits increases. For example while scanning a 99*99 (maximum matrix for 2digit number) matrix we need to spend 2(99+99+99+99+98+98)- 36 =1148 digits while you get 2943 digits out of reading the 1148 digits thus an advantage of 1795 digits which is a quite crushing of data to 39 % ! (61 % advantage)

But is it possible to equalise a smaller number to a bigger number. Not at all possible, but what happens here is the isomeric patterns or orientation replicates the number so that different isomeric patterns give the same reading. Here we take the advantage as the required data has a specific meaning and if the reading does not give the ideal result, we can neglect that pattern and consider it as a NON SENSE CODE. While we get the desired result from a specific pattern we consider it as a SENSE CODE. This selection and elimination of certain patterns gives us the advantage of data summarisation and we make the data coding or summing ultimately! (Applicable for small matrices as processing takes time and irrelevant for larger data)

To be more precise and incurring less data processing, we can divide the matrix into 4 parts (in 99*99 matrix let it be a 49*49 matrix and leave the central one row or column which does not interfere in the isomeric pattern) and mark the cells in the isomeric pattern only in the first quarter(left hand top set of matrix). Make a reading in decimal code thus we will have an additional 721 digits which makes the total digits required for coding to 1869. Though the advantage of micronisation of numbers become less, still it is advantageous and reduces data processing time. In this case we will get an advantage of 1074 digits. Though this is a smaller or 37 % digital reduction still beneficial while stacking a very large data! As the matrix size increase the compression ration increases proportionately.

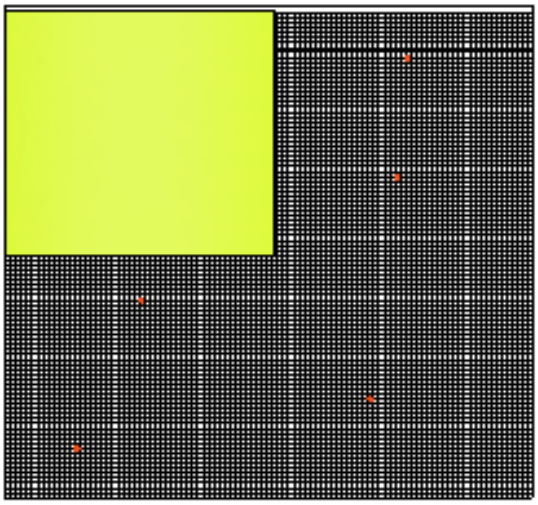

Fig 3:- 99*99 matrix where the yellow region is the separated matrix of 49*49

We can repeat the process of MSC and reach to a minimum required size. But multiple reduction using the available compressed data will increase processing procedures. Ultimately we will come to a balanced or equilibrium state of compression so that the data compressing and processing become optimal for our usage.

III. CONCLUSION

You can compress any larger binary code to smaller binary code using this technique and ultimately you can stack the resultant number to further by repeating the process and ultimately reaching a code size that is far reasonable to transfer and store.

REFERENCES

[1]. Amandeep Singh Sidhu, Er.Meenakshi Garg, Research Paper on Text Data Compression Algorithm using Hybrid Approach.IJCSMC,Vol.3,Issue 12,December 2014,Pg.01-10
[2]. Bruno carpentieri. 2018. Efficient Compression and Encryption for Digital Data Transmission, Hindawi, Volume 2018(Article ID 9591768)
[3]. David Hemmendinger. Professor Emeritus, Department of Computer Science, Union college,Schenectady New York, Co editor of Encyclopedia of Computer Science, 4th ed.(2000)
[4]. I Made Agus Dwi Suarjaya 2012. A New Algorithm for Data Cpmpression Optimization. International Journal of Advanced Computer science and Applications, Vol.3.No.8,2012
[5]. Storer, J. A., and Szymanski, T. G. 1982. Data Compression via Textual Substitution. *J. ACM 29*, 4 (Oct.), 928-951.
[6]. Tanaka, H. 1987. Data Structure of Huffman Codes and Its Application to Efficient Encoding and Decoding. *IEEE Trans. Inform. Theory 33*, 1 (Jan.), 154-156.
[7]. Witten, I. H., Neal, R. M., and Cleary, J. G. 1987. Arithmetic Coding for Data Compression. *Commun. ACM 30*, 6 (June), 520-540.
[8]. Ziv, J., and Lempel, A. 1977. A Universal Algorithm for Sequential Data Compression. *IEEE Trans. Inform. Theory 23*, 3 (May), 337-343.