International Research Journal of Engineering and Technology (IRJET) e-ISSN: 2395-0056
Volume: 02 Issue: 02 | May-2015 www.irjet.net p-ISSN: 2395-0072

LOSSLESS DATA COMPRESSION TECHNIQUES AND COMPARISON BETWEEN THE ALGORITHMS

Pooja Singh, Assistant Professor, Vadodara Institute of Engineering, Gujarat, India

---

Abstract:- This research paper provides different data compression methodologies and compare their performance based on the example. Data Compression is a process which reduces the size of data removing excessive information form it. Shorter data size is suitable as it reduces cost. Thus, the main aim of data compression is to remove data redundancy from the store or transmitting data. Data compression is also an important application in field of file storage and distributed system as in distributed system data are to send and receive from all the system. So Speed and performance efficiency are also major factor as in terms of data compression is to be used. Different data compression techniques are thus used for different data formats like text,audio,video and image files. Mainly there are two forms of data compression :- Lossy and Lossless. But in the lossless data compression, the integrity of data is to be preserved.

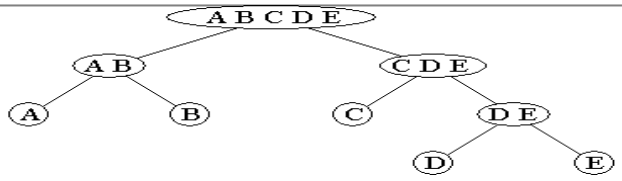
Keywords:- Data Compression, Shannon-Fano Coding, Huffman Coding, Run Length Encoding

I. INTRODUCTION

Data compression is a process that reduces the data size, removing the excessive information and redundancy. Why shorter data sequence is more suitable? the answer is simple it reduces the cost. Data compression is a common requirement for most of the computerized application. Data compression has important application in the area of file storage and distributed system. Data compression is used in multimedia field, text documents, and database table. Data compression methods can be classified in several ways. One of the most important criteria of classification is whether the compression algorithms remove some part of data which cannot be recovered during decompression. The algorithm which removes some part of data is called lossy data compression. And the algorithm that achieve the same what we compressed after decompression is called lossless data compression. The lossy data compression algorithm is usually use when a perfect consistency with the original data is not necessary after decompression. Example of lossy data compression is compression of video or picture data. Lossless data compression is used in text file, database tables and in medical image because law of regulations. Various lossless data compression algorithm have been proposed and used. Some of main techniques are Shannon-Fano, Huffman Coding, Run Length Encoding, and Arithmetic Encoding. In this paper we examine Shannon-Fano, Huffman Coding and Arithmetic Encoding and give compression between them according to their performances.

II. SHANNON-FANO CODING

Claude E. Shannon (MIT) and Robert M. Fano (Bell Laboratories) had developed a coding procedure to generate a binary code tree. The procedure evaluates symbol's probability & assigns code words with a corresponding code length. Compared to other methods the Shannon-Fano coding is easy to implement. In practical operation Shannon-Fano coding is not of larger importance. This is especially caused by the lower code efficiency in comparison to Huffman coding as demonstrated later. Utilization of Shannon-Fano coding makes primarily sense if it is desired to apply a simple algorithm with high performance and minimum requirements for programming. An example is the compression method IMPLODE as specified e.g. in the ZIP format

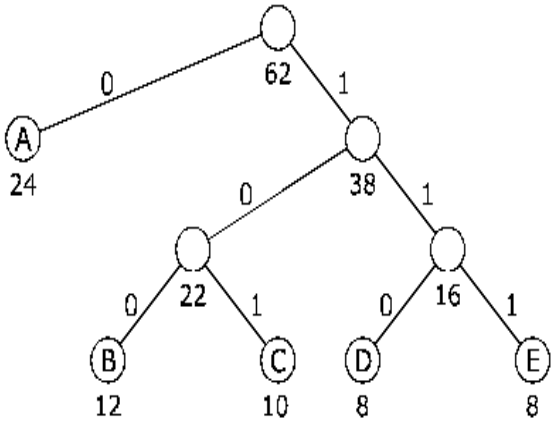


	Step 1			Step 2		Step 3	
Char	*f*	S	C	S	C	S	C
A	24	24	0	24	00	-	-
B	12	36	0	12	01	-	-
C	10	26	1	10	10	-	-
D	8	16	1	16	-	16	110
E	8	8	1	8	-	8	111

Char	*f*	Code	Code length	Total length
A	24	0	1	24
B	12	100	3	36
C	10	101	3	30
D	8	110	3	24
E	8	111	3	24
TOTAL				138 Bit

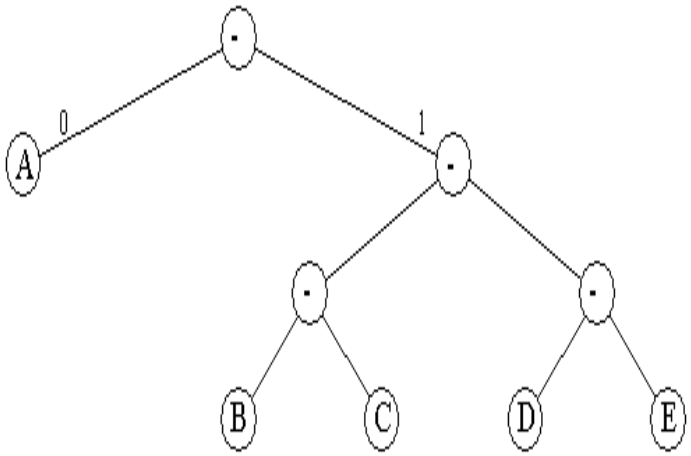
III. HUFFMAN CODING

The Huffman coding is a procedure to generate a binary code tree. The algorithm invented by David Huffman in 1952 ensures that the probability for the occurrence of every symbol results in its code length. Huffman codes are part of several data formats as ZIP, GZIP and JPEG. Normally the coding is preceded by procedures adapted to the particular contents. For example the wide-spread DEFLATE algorithm as used in GZIP or ZIP previously processes the dictionary based LZ77 compression.



IV. SHANNON-FANO CODING VERSUS HUFFMAN CODING.

The point is whether another method would provide better code efficiency. According to the information theory perfect code should offer an average code length of 2.176 bits or 134,882 bit in total. For comparison purposes the former example will be encoded by the Huffman algorithm.



		Shannon- Fano coding			Huffman Coding		
Char	*f*	Code	Len	Sum	Code	Len	Sum
A	24	00	2	48	0	1	24
B	12	01	2	24	100	3	36
C	10	10	2	20	101	3	30
D	8	110	3	24	110	3	24
E	8	111	3	24	111	3	24
Total		140 bits				138 bits	

The Shannon-Fano code does not offer the best code efficiency for the exemplary data structure. This is not necessarily the case for any frequency distribution. But, the Shannon-Fano coding provides a similar result compared with Huffman coding at the best. It will never exceed Huffman coding. The optimum of 134,882 bit will not be matched by both.

V. MEASURING COMPRESSION PERFORMANCES

Performance measure is use to find which technique is good according to some criteria. Depending on the nature of application there are various criteria to measure the performance of compression algorithm. When measuring the performance the main thing to be considered is space efficiency [5]. And the time efficiency is another factor. Since the compression behavior depends on the redundancy of symbols in the source file, it is difficult to measure performance of compression algorithm in general. The performance of data compression depends on the type of data and structure of input source. The compression behavior depends on the category of the compression algorithm: lossy or lossless. Following are some measurements use to calculate the performances of lossless algorithms.

Compression ratio: compression ratio is the ratio between size of compressed file and the size of source file.

$$CR = \frac{Size\ after\ compression}{Size\ before\ compression}$$

Compression factor: compression factor is the inverse of compression ratio. That is the ratio between the size of source file and the size of the compressed file.

$$CF = \frac{Size\ before\ Compression}{Size\ after\ Compression}$$

Saving percentage calculates the shrinkage of the source file as a percentage.

Compressed pattern matching: compressed pattern matching is the process of searching of pattern in compressed data with little or no decompression shown in following table.

$$SP = \frac{size\ before\ compress - size\ after\ compress}{size\ before\ compression}\ \%$$

COMPRESSION METHOD	ARITHMETIC	HUFFMAN
Compression Ratio	Very good	Poor
Compression Speed	Slow	Fast
Decompression Speed	Slow	Fast
Memory Space	Very low	Low
Compressed Pattern Matching	No	Yes
Permits Random Access	No	Yes

VI. REFERENCES

[1] Introduction to Data Compression, Khalid Sayood, Ed Fox (Editor), March 2000.

[2] Burrows M., and Wheeler, D. J. 1994. A Block-Sorting Lossless Data Compression Algorithm. SRC Research Report 124, Digital Systems Research Center.

[3] Ken Huffman. Profile: David A. Huffman, Scientific American, September 1991, pp. 54–58.

[4] Blelloch, E., 2002. Introduction to Data Compression, Computer Science Department, Carnegie Mellon University.

[5] Cormak, V. and S. Horspool, 1987. Data compression using dynamic Markov modeling, Comput. J., 30: 541–550.

[6] Cleary, J., Witten, I., "Data Compression Using Adaptive Coding and Partial String Matching", IEEE Transactions on Communications, Vol. COM-32, No. 4, April 1984, pp 396-402.

[7] Mahoney, M., "Adaptive Weighting of Context Models for Lossless Data Compression", Unknown, 2002.

[8] Capocelli, M., R. Giancarlo and J. Taneja, 1986. Bounds on the redundancy of Huffman codes, IEEE Trans. Inf. Theory, 32: 854–857.

[9] Kaufman, K. and T. Shmuel, 2005. Semi-lossless text compression, Intl. J. Foundations of Computer Sci., 16: 1167-1178.

[10] Pu, I.M., 2006, Fundamental Data Compression, Elsevier, Britain.

[11] D. S. Taubman and M. W. Marcellin, JPEG2000: Image Compression Fundamentals, Practice and Standards, Kluwer Academic Publishers, Massachusetts, 2002.

BIOGRAPHIES

Pooja Singh
Assistant Professor,
Vadodara Institute of Engg.
Kotambi, Vadodara.
Gujarat- 390009

