LEMPEL-ZIV-WELCH COMPRESSION USING MULTIPLE DICTIONARIES

Keerthy A.S[1] and Dr. S. Manju Priya[2]

ABSTRACT

Compression of data has been interesting area of analysis in the previous decade as well as the current. Compression algorithms have been implemented and analyzed for various categories of data. When it comes to preserving the originality of compressed data, lossless compression algorithms have priority. One of the sought after lossless compression methodology is the LZW compression algorithm. Many variants of the original LZW algorithm have been proposed and implemented over years. This paper analyses the usage of Multiple Dictionaries in LZW compression.

Keywords : Lossless compression, LZW, Multiple dictionary, Parallel dictionary

I. INTRODUCTION

Lossless compression has been in demand from the early days of data compression. Source codes, text documents, executable programs, DNA sequence data, etc are some examples of data that demands lossless compression[13]. The most prominent methodology for lossless compression was the Dictionary methods that date back to 1980s. Lempel and Ziv proposed the lossless compression algorithm using Dictionaries which was further enhanced by Welch. To improvise the existing dictionary methods, usage of multiple dictionaries have been proposed. This paper analyses the various fields of compression where multiple dictionaries are used and how it improves the usage of single dictionary for compression.

II. EXISTING ALGORITHMS

LZW (Lempel-Ziv-Welch) Encoding Algorithm

There is no requirement of prior knowledge of input files. The statistical properties of the input files can be learnt during encoding. It is particularly suited when the input file has high frequency in redundant characters, frequent patterns, etc. The algorithm builds a dictionary containing list of strings and each entry is assigned a unique code. Every character in the input string is concatenated with a prefix string and compared with entries in the dictionary. If it is present in the dictionary, output will be the code already assigned in the dictionary; otherwise new string will be added to the dictionary. The frequently occurring strings may be stored as symbols which needs less memory than storing original strings. The decoding algorithm uses the same dictionary built by the encoder [1].

LZW uses fixed-word-width dictionaries which consumes time in searching the entire dictionary. An alternative method suggested for overcoming the extensive search in single dictionary is initializing the dictionary with combinations of different characters or using a hierarchical dictionary of variable-word-width [2].

[1] Research Scholar in Computer Science, Karpagam University,

[2] Associate Professor, Department of Computer Science, Karpagam University

Dynamic LZW (DLZW) initializes the dictionary with different combinations of characters organized in hierarchical string tables. It uses least recently used (LRU) policy to keep the frequently used code words in short dictionary. Word based DLZW (WDLZW) identifies words in the input file as a basic unit. The word is searched for in the string table and if found a copy code corresponding to the address of matching string is output. Otherwise the word is added to string table and a literal code is assigned to it [5].

Parallel or Multiple Dictionaries in LZW :

A parallel VLSI architecture implementation replaces the currently existing single dictionary concept by multiple dictionaries of small variable-word-width dictionaries. The resulting scenario consists of multiple dictionaries with different word length which reduces the search time as well as operates in parallel. The authors claim a reduced hardware cost as well as ease of implementation[2].The detailed architectural design of Parallel Dictionary based LZW (PDLZW) is demonstrated in [3]. The dictionary is initiated with all single characters forming the character set of the input file. The input string is mapped to fixed-length code words based on dictionary set. The current substring and next character in the input stream is inserted as new entry into the dictionary [3].

Ming-Bo Ling, et al. has further proposed to combine the features of PDLZW with Adaptive Huffman Algorithm, as two stage data compression architecture. Adaptive Huffman algorithm with dynamic-block exchange (AHDB) takes the output code words from PDLZW and encodes them. The algorithm swaps the most frequently used symbol to the top of the ordered list and the index is encoded into canonical Huffman code word. The authors claim that the method reduces hardware cost and can easily be implemented on to VLSI architecture [4].

Another implementation of PDLZW suggested by Perapong V, et. al, uses a barrel shifter to loading a new input string onto the shift register. Also the dictionary updating is performed using Windowed Second Chance Updating Technique (WSC) that partitions dictionary into windows as k-size phrases and a three bit flag associated with each phrase. Test results exhibit compression ratio better than that of conventional PDLZW and also PDLZW with WSC technique [5].

Static Text Compression Algorithm (STECA) is a language dependent method that uses multiple static dictionaries for compressing text files. The dictionary is pre-constructed and used for all suitable situations. The method is appropriate only when prior knowledge of source is available. Compression is performed using most frequently occurring digrams and trigrams. Indexes of digrams and trigrams are encoded using hash tables. The high memory requirement for has tables is considered as the main drawback of STECA [6].

The analysis of Lempel-Ziv compression in parallel and distributed systems conducted by Agostino suggests that parallel compression is possible with LZW only with the usage of FREEZE deletion heuristic. The method after constructing the dictionary freeze it and the frozen dictionary will be used for parallel compression [7].

Combining PDLZW with wavelet transform on effective compression of EGC data is proposed by Sukesh and Jayashree [8] The ECG signals are wavelet transformed and then given to PDLZW for compression. The compressed output has relevance in storage and transmission of ECG data.

Nirali and Malay [9] combine the PDLZW with arithmetic coding and compare the performance with Deflate. From the input file, PDLZW takes out all the characters and forms initial dictionary. The output of PDLZW is given to the arithmetic encoder as input. Arithmetic encoder computes the frequency of digits, using it calculates the boundary values and generates the compressed bit file. The decoder follows the reverse order of calculations to obtain original string from bit file. The comparative analysis with Deflate shows better performance for proposed algorithm in the case of large files [10].

Nishad and Manika Chezian propose a variant of LZW compression by using multiple dictionaries, each of which is sorted. Since the dictionaries have entries in sorted order, the presence of patterns can be identified with simple binary search. The encoding algorithm works in two phases [11] of (1) switching and coding and (2) searching and updating. The initial dictionary contains all characters in the input file in the ascending order. Subsequent dictionaries are constructed in the process and whenever a string is to be searched, the corresponding dictionary alone is scanned. In phase 2 the pattern search takes place in the chosen dictionary using binary search. If pattern is not found in the dictionary, the position to which insertion has to be done is identified and shifts are done for enabling insertion. The method reduces the number of comparisons made and the number of shifts made. Hence it assures better time complexity as well as compression ratio [12].

III. Comparison of Existing Algorithms

Sl no:	Algorithm	Advantages	Disadvantages
1	LZW[2]	Fixed word width dictionary.	Time complexity of search is high.
2	DLZW[5]	Hierarchical string tables	Time complexity of search is high.
3	Parallel VLSI[3]	Multiple dictionary of variable word width. Parallel operation with reduced search time.	Memory intensive
4	PDLZW + Adaptive Huffman [4]	Opcodes are encoded. Reduced hardware cost and easy implementation	Dictionary search is costly
5	PDLZW + barrel shifter[5]	Dictionary is split into k-sized phrases. Faster search.	Freezy policy does not allow adding new insertions into dictionary
6	STECA [6]	Multiple static dictionaries	Need prior knowledge of data, high memory requirement
7	PDLZW +wavelet Transform [8]	Compress ECG data	Data specific
8	PDLZW + arithmetic encoder [10]	Frequency analysis of dictionary output is done and compressed bit file is output.	Low performance for large files
9	MDLZW	Sorted dictionaries with binary search	Dictionary search is time consuming

IV. Conclusion

Efficiency of LZW in data compression has been widely accepted. The performance of the compressors is further enhanced with the usage of multiple dictionaries. Many researchers have further enhanced the compression ratio by combining arithmetic and statistical methods along with MDLZW. The methods have been proven successful in different types of data. The MDLZW with binary search in sorted dictionary increases the efficiency of compression by reducing the number of shifts and comparison needed.

References

1. 1 Kotze, H. C., and G. J. Kuhn. "An evaluation of the Lempel-Ziv-Welch data compression algorithm." In *Communications and Signal Processing, 1989. COMSIG 1989. Proceedings, Southern African Conference on*, pp. 65-69. IEEE, 1989.
2. Ming-Bo Lin, "A parallel VLSI Architecture for the LZW Data Compression Algorithm," *International Symposium on VLSI Technology, Systems and Applications*, June 3-5, 1997, Taiwan, pp 98-101.
3. Lin, Ming-Bo. "A hardware architecture for the LZW compression and decompression algorithms based on parallel dictionaries." *Journal of VLSI signal processing systems for signal, image and video technology* 26, no. 3 (2000): 369-381.
4. MLin, Ming-Bo, Jang-Feng Lee, and Gene Eu Jan. "A lossless data compression and decompression algorithm and its hardware architecture." *IEEE TRANSACTIONS on very large scale integration (vlsi) systems* 14, no. 9 (2006): 925-936.
5. Vichitkraivin, Perapong, and Orachat Chitsobhuk. "An Improvement of PDLZW implementation with a Modified WSC Updating Technique on FPGA." *World Academy of Science, Engineering and Technology* 36 (2009): 611-615.
6. Carus, A., and A. Mesut. "Fast text compression using multiple static dictionaries." *Information Technology Journal* 9, no. 5 (2010): 1013-1021.
7. De Agostino, Sergio. "Lempel–Ziv data compression on parallel and distributed systems." *Algorithms* 4, no. 3 (2011): 183-199.
8. Jayashree, M. J., and A. Sukesh Kumar. "Resourceful scheme of ECG compression using different Wavelet transforms and PDLZW method." In *Electronics Computer Technology (ICECT), 2011 3rd International Conference on*, vol. 3, pp. 99-102. IEEE, 2011.
9. Bhatt, Malay. "Cascading of the PDLZW compression algorithm with arithmetic coding."

In *International Journal of Computer Applications*. 2012.

10. Thakkar, Nirali, and Malay Bhatt. "Two-stage algorithm for data compression." In *Proc. of the Intl Conf. on Advances in Computer, Electronics and Electrical Engineering*, pp. 350-354. 2012.

11. Nishad P M, R. Manicka Chezian, "Enhanced LZW (Lempel-Ziv-Welch) Algorithm by Binary search with multiple Dictionary to reduce time complexity for Dictionary creation in encoding and Decoding" ***In International Journal of Advanced Research in Computer Science and Software Engineering (IJARCSSE)*, pp. 192-198, 2012.**

12. Nishad P M, "A novel approach to reduce computational complexity of multiple dictionary lempel ziv welch mdlzw using indexed k-nearest twin neighbor iktn clustering and binary insertion sort algorithm", Thesis submitted to Bharathiyar university, pp. 22-32, 2014

13. Sebastian Deorowicz, Szymon Grabowski, "Compression of DNA sequence reads in FASTQ format", *Bioinformatics*, pp 860-862, 2011.

Author's Biography

Keerthy A S is currently pursuing Ph D in Karpagam University under the guidance of Dr. S Manju Priya. She has completed MPhil in Bioinformatics from Kerala University and MCA from Calicut University. Her area of interests include Data Mining, Graph Theory, Aritificial Intelligence and Bioinformatics.

Dr.S. Manju Priya has received the Ph.D Degree in Computer Science from Karpagam University in 2014. She is working as Associate Professor in Department of Computer Science, Karpagam University. She has published more than 15 papers in various National/International Journals. Her research interest includes Wireless Sensor Network, Network Security, and Data mining.