International Journal of Recent Technology and Engineering (IJRTE)
ISSN: 2277-3878 (Online), Volume-8 Issue-2, July 2019

Preprocessed Text Compression Method for Malayalam Text Files

Rincy T A, Rajesh R

Abstract: ***The increasing importance of Unicode for text files implies an increase in storage space required for data and the time for the transmission of data, with a corresponding need for compression of data. Conventional compressors fair purely on UTF-8 texts, where each character can span multiple bytes. Malayalam which is one among the four major languages of the Dravidian family, is represented by using Unicode characters. The contribution of this paper is a reversible transformation mapping of the input to reduce the actual size of the input file before a general purpose compression method. After the preprocessing, LZW compression achieves more compression to Malayalam text files containing any characters including ASCII characters. This method can be extended to any native language files containing mostly the characters of only one script.***

Index Terms: Data Compression, Unicode, LZW, UTF-8, Compression Ratio.

I. INTRODUCTION

Compression algorithms lessen the excess or redundancy in the representation of data to reduce the storage space required for that data. Without effective compression, a large number of magnificent innovations, and new advancements to be produced later on, would not be conceivable. The scope of various compression methods is based on its adaptability according to numerous situations which are changing very quickly. This very character of data compression suits its uses in the arenas of satellite imaging, Audio, Video and Media technology, Database Design , Medical imaging techniques etc. Text compression is concerned with the methods for representing the digital data in alternative representations that take less room. It not only helps in reducing the storage space for archival and online data, but also it accelerates data loading for web pages, app data, streamed games and reduces the operational cost of cloud infrastructure. Compression schemes can be named as Lossless and Lossy. A part of original data and quality may be lost in the case of Lossy file compression. It intentionally disposes of a few data in the compression process. In lossless compression, every single unique datum can be recouped when the record is uncompressed. Lossless algorithms can be mainly classified such as statistical, dictionary based and transform based methods[1]. Data compression for different languages of the world including Indian languages are of greatly solicited. Malayalam is a Dravidian language spoken over the Indian territory of Kerala and it is one of the 22 scheduled Indian languages of India. Around 38 million people speaks Malayalam worldwide. It is worth mentioning that Malayalam language is the one having maximum number of letters amongst orthographies of all Indian dialects, with a total of 56 letters in count. As it is represented using Unicode characters, the compression schemes for Unicode characters is of great importance[4]. The Unicode encoding scheme UTF-8 is used by 92.3% of all the websites as per usage statistics available(w3techs 2018). Legacy encodings such as ASCII, represent each character by a single byte. UTF-8 maps characters to sequences of between one to four bytes. The Unicode Standard defines how to encode multilingual text (The Unicode Consortium 2015). Unicode defines three encoding schemes named UTF-8, UTF-16 and UTF-32 which map code points to a sequence of one or more bytes[7]. Here, we exploit the fact that the Unicode code space is divided into blocks of characters belonging to the same script. Most texts use characters from only a handful of blocks. So although the overall Unicode code space is vast, adjacent characters in a text will tend to have nearby code points. In this paper, a new preprocessing technique is described to improve compression of UTF-8 encoded Malayalam language text. This method adjusts the ASCII alphabets in the text file first, then a substitution technique is applied to replace the Malayalam characters with its Unicode. Since the Unicode characters are allocated in blocks of the same script, removal of the frequently used characters before the LZW compression achieves a compression factor of 4 on an average in the case of Malayalam text files.

II. RELATED WORKS

Many compression techniques are available for English language and European languages. The grammatical rule and properties for natural languages are different from English language, so compression should be specially designed. In [4], the Malayalam text compression by variable length encoding is explained, off which the less number of bits is used by the Unicode character. Linkon et al. projected a changed LZW dictionary based index compression technique for Bangle dialect in [5]. Gleave et al. represent a new modified technique of byte-oriented compressors to work straight on Unicode characters[6].

Revised Manuscript Received on 30 July 2019.
***** Correspondence Author
Ms. Rincy T A*, Assistant Professor, Department of Computer Science, Prajyoti Niketan College, Pudukad, Thrissur, Kerala, India.
Dr. Rajesh R, Associate Professor, CHRIST(Deemed to be University), Bengaluru, Karnataka, India

In [8], the system is configured to maintain a set of character tables and a cluster table in memory .
Each Unicode character from the character table is assigned a shortened bit representation. A novel approach to construct a data compression dictionary for Gujarati text is discussed in [9] by Sandip V Maniya and M.J. Sheth. In [10], B. Vijayalakshmi and N. Sasirekha uses a dictionary which contains a collection of Unicode characters indexed with ASCII characters for compression and decompression process. Md. Abu Marjan et al. have proposed a novel approach in which Bengali text is represented efficiently with a better compression ratio[11].

III. LEMPEL ZIV WELCH COMPRESSION

Data compression techniques have been done in a widespread manner using LZW compression. Lots of enhancements and improvements have happened in vista of data compression since the invention of LZW. LZW compression technique received vast acceptance as it is giving a solution for files with much repetitive data, specially for text and monochrome images[12]. In short, strings of characters are replaced by single codes in LZW compression. Rather than doing any analysis of incoming text, LZW just adds every new string of character in a table of strings. Commonly 4096 entries are made within the table. In the aforementioned case, 12 bit codes contains the data encoded in this methodology, each implying to one of the entries in the code table. The decompression algorithm is the companion algorithm for compression. The input stream is exactly rebuilt by taking the stream of codes output from the compression algorithm. The data of input stream, can be used to rebuild the table as exactly as it was during compression without passing the string table to the decompression code. A single byte from the input file always delegates codes 0-255 within the code table. During the commencement of encoding of a file, by the LZW program, the code table contains the first 256 entries alone, while the rest of the table being blank. As the encoding proceeds, the LZW algorithm recognises continuous sequences in the data, and adds them to the code table. When the second time a sequence is encountered, compression gets started. The peculiar feature is that no sequence from the input file is appended to the code table, till it has already been placed in the compressed file as individual characters expressing codes 0 to 255[14]. This factor is very commonly found to be a necessity since it lets the decompression program to remake the code table directly from the compressed data, while not allowing the code table to be managed or transmitted separately.
LZW encoding works as follows[14]
Step 1 Initialize dictionary to contain single character string.
Step 2 Read first byte from the data file, store in string x.
Step 3 Read next input character m from the data file.
a) If no such k (input exhausted): output:=code(x):Then EXIT
b) If xm exists in dictionary: x: = xm; Repeat Step 3.
c) Else If xm not in dictionary. Then output :=code (x)
Add the entry xm to Dictionary;
x :=m;
Repeat step 3.
Step 4 End

LZW Decoding Algorithm works as follows[14]:

Step 1 Read first input code, store in OCODE, output the translation of OCODE.
Step 2 Read next input code, store in NCODE.
If no more codes to input: EXIT.
Else:
Step 3 If NCODE exists in Dictionary then x:=translation of NCODE
Else
x:=translation of OCODE
x:=xm
Step 4 Output x.
Step 5 m:=the first character in x;
Step 6 Add entry in Dictionary for OCODE+m
Step 7 OCODE:=NCODE
Step 8 Repeat step 2.
Step 9 End
On verification of algorithm, it is very clear that LZW always attempts to output codes for strings which are previously known. A new string is appended to the string table, always when the output is a new code. This algorithm substitutes repeated occurrences of a string by references to a previous occurrence. As the decoder can recreate the first original version of message with all its exactness, the LZW compression algorithm is referred as “reversible" [12].

IV. PROPOSED COMPRESSION METHOD

The inner redundancy of the source file is very well exploited in this new proposed transform method of lossless text compression. It reduces the size of the original file before the normal compression. Using this pre-processing technique, recommendations have been made for much more efficient data compression for Malayalam text files. In this approach, the ASCII characters in the file are tagged for first stage pre-processing. Respective UNICODE characters are being made by converting all the characters in the second stage of pre-processing and later by eliminating specific redundant characters in Malayalam. It is found that if we perform these two-stages of pre-processing techniques, we can immensely reduce the file size while being fed into a LZW Compressor. While simulating the LZW data compression, both with and without our pre-processing techniques, it is very evident that the pre-processing techniques are of remarkable value. Our experimental results and comparisons with LZW compression shows that the pre-processing techniques achieve a better compression ratio.

A. *Malayalam Documents in UNICODE Format*

Independent of the platform, application or language, the Unicode Standard provides a peculiar number for every character [13]. In the modern world, the Unicode Standard stands as a character coding system well equipped to support all the stages of worldwide interchange of numerous languages including processing and display of the written texts. The Unicode is successful in embracing even historical and classical texts of many written languages[13].

As an example, encoding of some characters of Malayalam language according to the universal principle of Unicode is shown in Table 3.1.

Table 3.1 Unicode Table for Malayalam characters

	0	1	2	3	4	5	6	7
0x0D0			ം	ഃ		അ	ആ	ഇ
0x0D1	ഌ		ഒ	ഓ	ഔ	ക	ഖ	ഗ
0x0D2	ഠ	ഡ	ഢ	ണ	ത	ഥ	ദ	ധ
0x0D3	ര	റ	ല	ള	ഴ	വ	ശ	ഷ
0x0D4	ി	ു	ൂ	ൃ			ൊ	ോ
0x0D5								ൗ
0x0D6	ൟ	ൠ					൦	൧

	8	9	A	B	C	D	E	F
0x0D0	ൈ	ഉ	ഊ	ഋ	ഌ		എ	ഏ
0x0D1	ഘ	ങ	ച	ഛ	ജ	ഝ	ഞ	ട
0x0D2	ന		പ	ഫ	ബ	ഭ	മ	യ
0x0D3	സ	ഹ					ാ	ീ
0x0D4	െ		ൊ	ോ	ൌ	്		
0x0D5								
0x0D6	ൡ	ൢ	ൣ	൤	൥	൩	൪	൫

B. *Architecture of Proposed Compression*

In this section, the architecture of the proposed preprocessing technique is described to improve compression of UTF-8 encoded Malayalam language text.

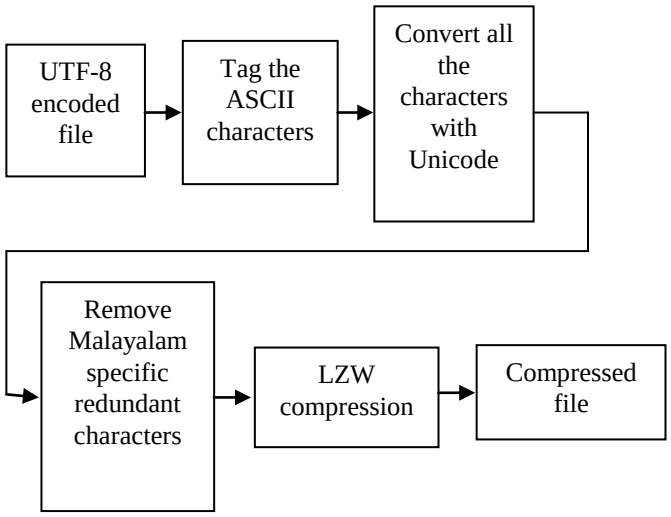

Figure 4.1 Architecture of preprocessed compression

Following steps acquires the improved compression rate:
First is to tag the ASCII characters present on the file. Usually on every file, we use some ASCII characters. Second all the characters in the source file, including the ASCII characters are converted into its corresponding Unicode representation[3]. Nowadays Unicode encoding representation is widely used for representing characters of different languages. 127 Unicode characters are there with Malayalam including vowels, consonants, dependent vowels, chillu letters, digits, signs and fractions. And they are represented from 0D01 to 0D7F in Unicode standard. Fortunately, Unicode characters are allocated in blocks of the same script and most files will use only one or two scripts. And it reduces the code space to a more manageable space.
Taking advantage of the above feature, as a third step, we are removing all redundant characters that represent the Malayalam script as 0x0D and we obtain, a shortened code representation and reduced file size. In the fourth step, a file with a reduced file size is fed into a LZW compressor and it provides a better compression ratio. The original Malayalam file is obtained by doing an LZW decompression initially, which is followed by the substitution and reverse mapping. The decoding process starts with reading a value from the encoded input and generating the corresponding string from the initialized dictionary. In order to rebuild the dictionary, it follows the same way as it was built during encoding. It also obtains the next value from the input and adds it to the dictionary. The concatenation of the current string and the first character of the string obtained by decoding the next input value, or the first character of the string just output if the next value cannot be decoded. The decoder then proceeds to the next input value and repeats the process until there is no more input, at which point the final input value is decoded without any other additions to the dictionary. Much beneficial outcome is very evident when this method is applied to a dataset picked from various sources like online news papers, journals, blogs and articles.

V. EXPERIMENTAL RESULTS

Measuring the performance of a compression scheme is difficult. There is perhaps no better way than simply testing the performance of a compression algorithm by implementing the algorithm and running the programs with sufficiently rich test data. The following observations were made as a result of testing the proposed compression with respect to twelve different types of Malayalam files of varying size, and it is represented in Table 5.1
Performance evaluation of the proposed algorithm is done using two parameters - Compression Factor and Saving Percentage[1].
Compression Factor= Size of the original File / Size of the Compressed File
Saving %= 1-(Size of the Compressed File / Size of the Original File)
From Table 5.1, it is evident that the compression factor achieved by the proposed system is better, which means it results in more savings of storage space.

Table 5.1 Results after proposed compression

Sl. No	File Name(Filename.txt)	Original File Size (in KB)	Compressed file size -normal LZW(in KB)	Compressed File Size - proposed method (in KB)	Compression Factor - proposed method	Saving % - Proposed method
1	File1	43851	12144	10974	3.9959	.749
2	File2	45629	12429	11175	4.0831	.755
3	File3	30108	8570	7794	3.8630	.741
4	File4	46454	12671	11447	4.0582	.753
5	File5	82409	21723	19746	4.1735	.760
6	File6	77948	20858	18842	4.1369	.758
7	File7	99547	26036	23510	4.2342	.763
8	File8	59611	15927	14337	4.1578	.759
9	File9	48677	13127	11835	4.1130	.756
10	File10	40762	11399	10313	3.9525	.747
11	File11	37928	10650	9681	3.9178	.744
12	File12	56671	15266	13796	4.1078	.756

The graph in figure 5.1 is made on the basis of Table 5.1 which shows the comparison of file sizes before and after the proposed compression.

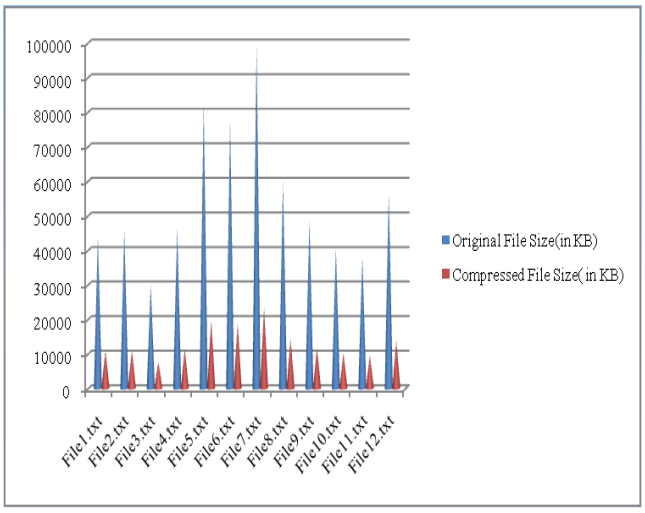

Figure 5.1 Comparison Graph of File Size

In the graph, the vertical axis represents the length of input string in bytes and horizontal axis represents the file name. The graph in figure 5.2 shows the comparison between the normal LZW and the proposed preprocessed method followed by LZW compression.

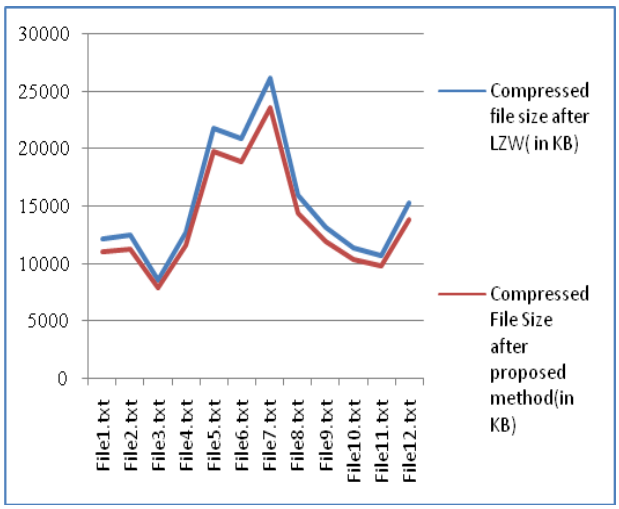

Figure 5.2 Comparison Graph of LZW and Proposed Method

From the above graph, it is clear that the proposed method achieves better compression than the usual LZW compression for the Malayalam text files. And it surely paves a way to store the Malayalam text files with minimum storage.

VI. CONCLUSION

Due to the unprecedented explosion of Malayalam textual information through the use of the Internet, Malayalam text compression is very essential now a days, for the optimum use of resources. In this work, a novel preprocessing technique is described to improve compression of UTF-8 encoded Malayalam language text. ASCII to Unicode conversion followed by the removal of certain code words results in the reduction of file size. A LZW compression with the reduced file size improves the compression significantly. Compression factor is calculated for various Malayalam files. On an average, we get the saving percentage of Malayalam documents as 75%. During the implementation process, we found that this method is efficient, and produces the best codes for individual data symbols, and thus better data compression.

REFERENCES

1. David Salomon, "*Data Compression: The Complete Reference*", 4th Edition, Springer, 2007.
2. C. E. Shannon, A mathematical theory of communication, *Bell Syst. Techn. J.*, 1948, Vol. 27, pp. 379–423, 623–656.
3. Svend Juul and Morten Frydenberg. "UNICODE2ASCII: Stata modules to translate between Unicode and ASCII", Available at: https://ideas.repec.org/c/boc/bocode/s458080.html, Accessed on 2016.
4. Sajilal Divakaran, Biji C. L., Anjali C., Achuthsankar S. Nair , "Malayalam Text Compression", International Journal of Information Systems and Engineering, Vol 1, No.1, pp 7-11, April 2013
5. Linkon Barua et al., "Bangla Text Compression based on Modified Lempel-Ziv-Welch Algorithm", *Proceedings of IEEE International Conference on Electrical, Computer and Communication Engineering*, pp. 113-118, 2017.
6. Adam Gleave and Christian Steinruecken, "Making Compression Algorithms for Unicode Text", *Proceedings of Data Compression Conference*, pp. 22-25, 2017

7. Shihjong Kuo, “Processors, Methods, Systems, and Instructions to Transcode Variable Length Code Points of Unicode Characters”, U.S. Patent, 2017.
8. Mahesh Dattatray Kulkarni et al., “System and Method for Compression and Decompression of Text Data”, U.S. Patent, 2017.
9. Sandip V Maniya and M.J. Sheth, “Compression Technique based on Dictionary Approach for Gujarati Text”, *International Journal of Engineering Research and Development*, Vol. 4, No. 8, pp. 101-108, 2012.
10. B. Vijayalakshmi and N. Sasirekha, “Lossless Text Compression For UNICODE Tamil Documents”, ICTACT Journal On Soft Computing, January 2018, Volume: 08, Issue: 02, ISSN: 2229-6956 (ONLINE)
11. Md. Abu Marjan, M Palash Uddin, Masud Ibn Afjal and Md. Dulal Haque, "Developing an efficient algorithm for representation and compression of Large Bengali Text", The 9th International forum of Strategic technology(IFOST), October 21-23, 2014, Cox’s Bazar, Bangladesh, pp 1-9
12. J. Weimin, “Application Research of the LZW Algorithm in Data Communications,” Computer Engineering & Science Journal, vol. 26, no.5, pp. 46-48, 2004
13. www.unicode.org/standard/standard.html
14. Nishad PM, R. Manika Chezian,"Behavioral Study of Data Structures on Lempel Ziv Welch (LZW) Data Compression Algorithm and its Computational Complexity", 2014 International Conference on Intelligent Computing Applications, 6-7 March 2014, pp 268-269

AUTHORS PROFILE

Ms. Rincy T A, received the M. Phil degree in computer science from Bharathiar University of Coimbatore, Tamil Nadu, India in 2009. She is currently an assistant professor in the Department of Computer Science at Prajyoti Niketan College, Pudukad, Thrissur, Kerala. Her main interests include Data Structures, Algorithms and Compression Techniques. rincyanto@gmail.com

Dr. Rajesh R is working as Associate Professor in the Department of Computer Science, CHRIST (Deemed to be University) Bangalore, India. Dr.Rajesh research interests are in the areas of Data Structures and Analysis of Algorithms. He has published 35 papers in various Journals and Conferences. Dr. Rajesh is also serving as Managing Editor, Lead Guest Editor, Associate Editor, Editorial board member and Technical Committee member of various National and International Conferences and Journals. He has received Veenus International Foundation’s Outstanding Faculty award and Dewang Mehta Education Leadership Award to his credit.
r.rajesh@christuniversity.in