IJIRMET ISSN (Online): 2456-0448
International Journal of Innovative Research in Management, Engineering and Technology
Vol. 6, Issue 4, April 2021

Efficient Design And Implementation Of An Enhanced Lzw Algorithm For Compressing Data

[1] Amal Saroj ,[2] Remya Rajan

[1] Assistant Professor,Mes College ,Marampally
[2] Assistant Professor,Jai Bharath Arts And Science College Arackappady

Abstract In this paper, two improved lossless data compression plans are proposed based on Lempel-Ziv-Welch encoding (LZW) for wireless sensor networks. The schemes pre-process numerical data in different ways of adjusting data characteristics. In to increase the compression rate, the range of values is reduced by calculating the increment between the string sample and the baseline. From time LZW is generally effective for textual data, but does not refer to numeric data, the step is replaced by English letters. And also, with an increasing amount of textual data stored in compressed form, in effect getting information in a packed area has become a major concern. Being the ability to search effectively is highly desirable for accessing randomly compressed data and is required in many applications. The degree of compression can also be improved using a custom LZW algorithm.

1. INTRODUCTION

Sensor nodes are usually powered by small batteries that cannot usually be replaced or recharged; energy is the main limitation in implementing wireless sensor networks [1]. In most cases, wireless nodes are the main cause of energy consumption. Thus, energy saving is usually achieved by limiting radio communication. To extend the life of the network, we prefer to reduce energy consumption by compressing data for transmission. One study shows that the energy consumption for 3000 commands is equal to the energy consumption for transmitting 1 bit over a distance of 100 m via radio [2]. Therefore, data compression before sending is an effective way to effectively use the limited power supply of nodes and maintain a long lifetime of wireless networks [3]. Literature [11] offers a new algorithm for LZW improvement. The range of data values is reduced by calculating the increment between two adjacent data in the sample sequence. The growth is replaced by one character. The difference between adjacent points may reduce the range of sample values, but is not valid for any type of data. For data having different polarities of the adjacent point, increment between adjacent points may enlarge the value span of the sample sequence, for example, positive number minus negative number or vice versa. So we propose two improved schemes to reduce the value span which can adapt to the numerical data with different characters. Aiming at the data characteristics that adjacent two points have polarity changes or the increment between two adjacent data sample is larger than the increment between the sample and their base value,the data value is reduced by calculating increment between the data of sample sequence and the base value, which will lead to smaller value span. Aiming at the data fluctuating in different ranges, baseline data existing cyclical fluctuations, adjusting the baseline is sometimes more simple and effective than adjacent point direct subtraction. But this scheme needs adaptively detection baseline or determining the base value. The value of the data is reduced by calculating the increment between the data in the sampling sequence and the self-adaptive baseline value, which can reduce the data range and increase the compression ratio. The main problems associated with the compression method for search purposes are classified according to their importance:: a) random access and fast (partial) decompression; b) efficient indexing; c) compression ratio

II. IMPROVED LZW ALGORITHM DESIGN

A. Lempel-Ziv-Welch (LZW) Algorithm

LZW - lossless search-based compression algorithm in dictionary [12]. The operating principle of the LZW algorithm is similar to the dictionary. As a result, it generates a row table during encoding. This string table records all the character strings those have already appeared and there is no duplication. Input data flow and character strings in the string table are compared, then the output value is determined and the string table is updated. Since the LZW decompression algorithm can reconstruct the dictionary while processing of the compressed data, it need not transmit the dictionary.

B. The Improvement

LZW - lossless search-based compression algorithm in dictionary [12]. The operating principle of the LZW algorithm is similar to the dictionary. As a result, it generates a row table during encoding. This string table records all the character strings those have already appeared and there is no duplication. Input data flow and character strings in the string table are compared, then the output value is determined and the string table is updated. Since the LZW decompression algorithm can reconstruct the dictionary while processing of the compressed data, it need not transmit the dictionary. There are four main procedures in two improved schemes in this paper. First, the data is pre- processed to reduce the amount of data. Second, the numeric data is converted to a character for compression.In the third step, the pre-partial selective decoding module decodes the compressed text and displays the corresponding part. Finally, the characters are compressed and the code words are displayed.The whole flowchart of the improvement scheme is shown in Figure 1.

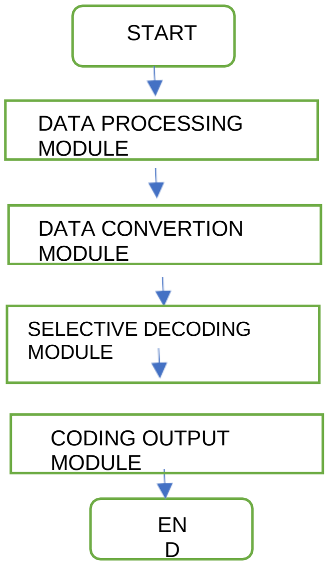


Figure. 1. Whole Flow cjart

a . Data preprocessing module

According to the different characteristics of the numerical data,we design different improvement schemes. The differences lie in the procedure of preprocessing. When the data fluctuate in a certain range, the united base value is given.When the data fluctuate in different ranges, the adaptive base value is given. The data preprocessing module for scheme I and scheme II are illustrated respectively as follows

[i] Data preprocessing module of scheme I

Aiming at the data fluctuating in a certain range, the data value is reduced by calculating increment between the data of sample sequence and the base value. The base value can be obtained by calculating the average value of the first peak value and trough value. Then increments between the data of sample sequence and the base value will be calculated and stored into an array.

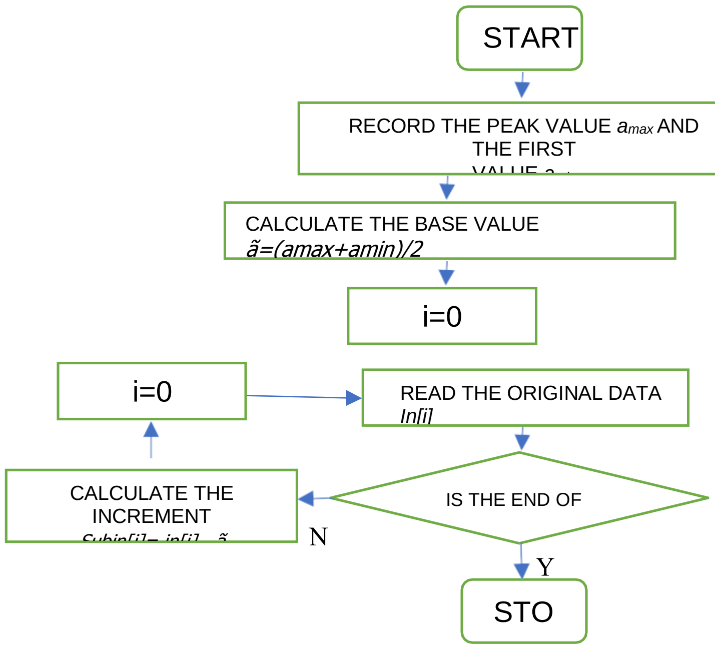

START
RECORD THE PEAK VALUE amax AND THE FIRST
CALCULATE THE BASE VALUE ã=(amax+amin)/2
i=0
i=0
READ THE ORIGINAL DATA In[i]
CALCULATE THE INCREMENT
IS THE END OF
N
Y
STO

Figure. 2. Data Preprocessing Module of scheme I

Suppose that In[i] is a sample sequence, amax is the first peak value within the sample sequence, and amin is the first trough value within the sample sequence. ã is the base value which can be calculated through the formula:

ã = (amax +amin)/2. Suppose that SubIn[i] is a increment sequence which saves increment between the original sample data and the base value, that is Subin[i]= in[i] - ã. The flowchart of the data preprocessing module of scheme I is shown in Figure 2.

[ii] Data preprocessing module of scheme II

When the data fluctuate in different ranges, we put forward scheme II for data preprocessing module. As baseline data excising cyclical fluctuations, the base value of scheme II is not a constant value. The scheme adaptively detect baseline and determine the base value. The first base value ã1 can be obtained by calculating the average value of the first peak value and trough value, ã1 = (amax1 + amin1)/2.The data processing method of scheme II tends to be operated according to the steps bellow:

1) Compare the increment SubIn[k] that each data value in In[i] minuses the base value

ã1 with the desirable sample space;

2) Count the number η that represents the times of the increment continuously exceed the range;

3) If the number of the counter value η continuously exceeds the scope of desirable sample space for ηmax times, build up a data block of the former data that have been count, and then output the start flag of the data block and the primary base value ã1;

4) Recalculate the base value ãj. Calculate the average value of the first peak amaxj and trough aminj in the next part and set the average value as the base value ãj

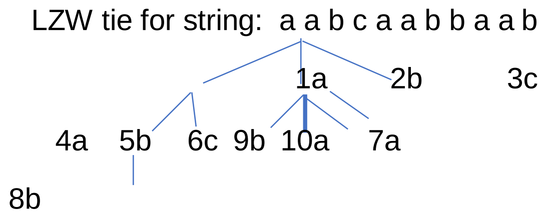
5) Count the number η again, if η > ηmax, partition the data block, output the flag and change the base value ãj.

b . Data conversion module

Traditional LZW algorithm has been proved as an effective method for text data. Since all the data be preprocessed is still numeric, the algorithm can not satisfy numeric data properly. The preprocessed data could not be directly compressed by LZW algorithm. The numeric data should be converted into character before compressing. During processing of converting, the increment is replaced by a single character. 1 ~ 25 is respectively replaced by A ~ Y, -1 ~ -25 by a ~ y, 0 by Z, and all the data out of the span is replaced by character z. z also can be involved in encoding. All increments that are represented by single character z are saved into Over Flow[t] orderly. Suppose that Cha[k] is a sequence prepared for encoding after character conversion.

c . Selective Decoding Module

LZW refurbished algorithms that support random random access to compressed text.Instead of completely unpacking the text and generating the results selectively, we enable random access and partial decoding of compressed text and display the appropriate part. The compression ratio can also be improved using a modified LZW algorithm. Preliminary results regarding storage time and performance are given. This static dictionary is trying to compress all the files in the database. In a networked environment, an encoder and a decoder store the same copy of a public medium. A public assignment needs to be transmitted only once. Compressed files are sent separately without much effort. The decoder will refer to a public attempt to decode the compressed file. As the dictionary gets statistics for a large set of training showing the source property, the overall compression rate is expected to improve, although file sharing may have a worse compression rate.



LZW compression: a a b c a a b b a a b
1 1 2 3 4 2 2 8
Off line LZW : a a b c a a b b a a b
compression 8 7 5 10 5

Figure3.Selective decoding module

Figure 3 shows an example that the series is built, and the template can be within the path from the root to the node or consist of paths of more than one node. Enter “aabcaabbaab” if we are compressing text with the current LZW output code: “11234228”. Code becomes smarter during the coding process. Take, for example, the replacement “aab”; it appears three times in the text and is encoded in the codes "112", "42" and "8". The above example shows that if we can “train” the stream in actual compression, we could get better compression. Thus, two compression bypass schemes have been proposed and tested. The first step builds the entire sample by scanning the text without actually compressing it. Another way is to compress the text from the very beginning with a ready-made trial version.

d.Coding Output Module

In this part, we take some measures to increase compression ratio and decrease storage space. Several improved approach with distinguish advantages are given in this module: At the beginning step, all single characters need not be put into the dictionary. Secondly, the length

of substring max in the dictionary is limited. Thirdly, select appropriate dictionary capacity. In the case of single letter words, the output is the letter’s ASCII. When the word is constituted of several letters, the output is its code word.

1) Replace the initial window (only for a system with adaptive basic values) and basic values. Then send OverFlow [t] organized.

2) Read the first letter and rate it in the string prefix.

3) Read the next character γi from Cha[k], if it's the end, then quit. Otherwise combine ω with γi and judge whether ψ, which is the length of ωγi, is longer than ψmax that have been set in initialization. If it's longer, output its code word, else judge whether ωγi is in the dictionary. If not exist, perform (4), else, ω=ωγi, repeat (3).

4) Put ωγi into the dictionary and correspondingly generate a new code word, then judge whether ω is consisted of single character or multi characters. For the former case, output its ASCII. For the latter case, output the code word, ω=γi, perform (3).

For output, in order to save space and improve the transmission efficiency, word with single letter, the output of this letter is its ASCII that has 8 bits. The word with several letters, the output of this word is its code word and the code word is described in two bytes that has 16 bits. the maximum dictionary capacity is 32768. If the dictionary overflows then re-encoding so as to meet the requirements of nodes' limited memory space. During decoding, the following two bytes denotes code word when the highest digit equals one. The output with code word and single character are distinguished in this way.

III. PERFORMANCE COMPARISON

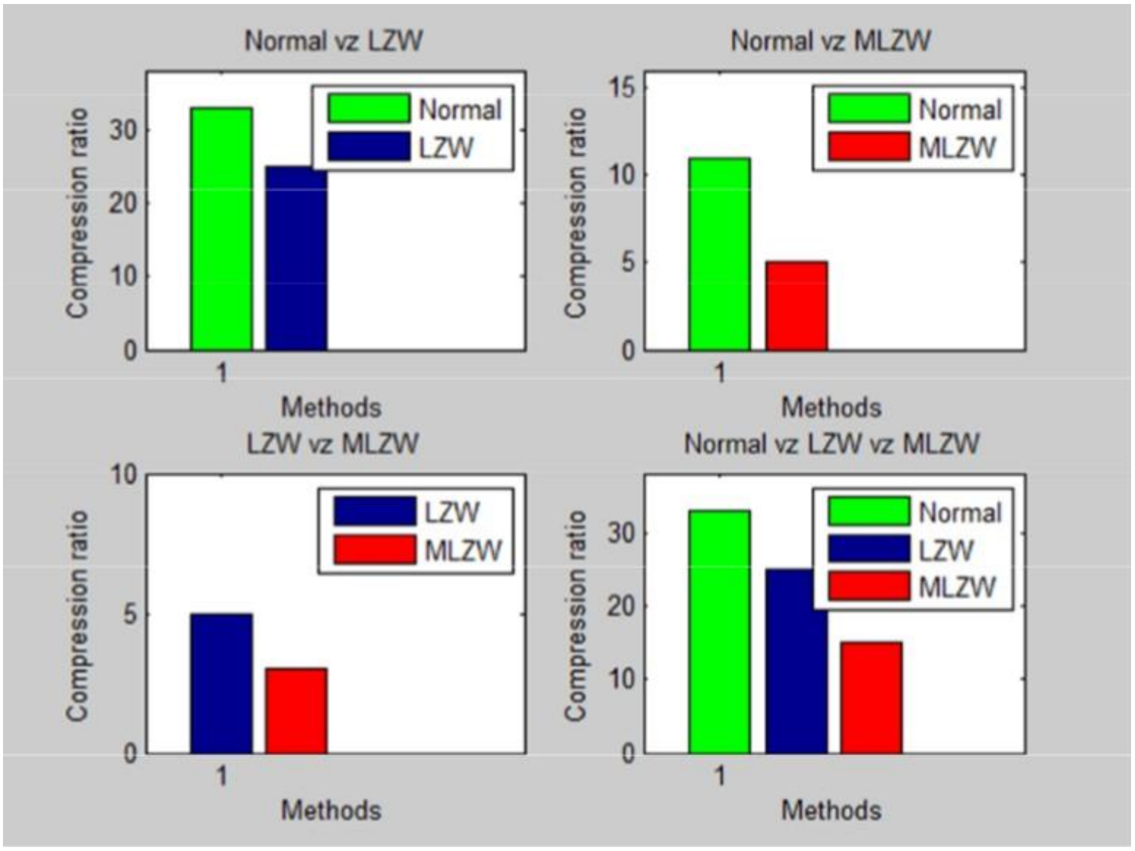


Figure 4: Comparison of compression ratio Between normal and LZW and modified LZW.

IV. CONCLUSION

The LZW algorithm was considered as a simple and practical method to meet the requirements of WSN. Two improved LZW-based systems are proposed to increase compaction efficiency. The choice of Scheme I or Scheme II is closely related to the nature of the measured data. In order for the compressed data to fluctuate within a certain range, Scheme I achieves a maximum compression rate. But, for the compressed data to fluctuate in different areas, Schedule II reaches the maximum compression rate. The simulation results and analysis show that one algorithm has good performance only for specific data.And also We have shown that random access and partial decoding can be done efficiently on compressed text with a few modifications to the LZW algorithm. We obtain the ability to randomly access any part of the text given the location index. Moreover, our method brings little or no change in the compression ratio. We have shown that if a trained trie is used to compress the whole text file, the compression ratio can actually improve (Figure 4). Hence, choosing the proper improved scheme for LZW algorithm will save more energy and prolong wireless sensor network lifetime.

REFERENCES

[1] Remya Rajan ,Design and Realization of Improved LZW Algorithm forEfficient Compressed Data Retrieval
[2] L. M. Sun, J. Z. Li. Wireless Sensor Networks[M]. Beijing: Tsinghua University Press, 2005: 3-9. (in Chinese)
[3] M. J. Handy, M. Haase, D. Timmermann. Low Energy Adaptive Clustering Hierarchy with Deterministic Cluster-Head Selection[C], In: Proceedings of IEEE International workshop on Mobile and Wireless Communications Network, 2002: 368-372.
[4] N. Kimura, S. Latifi. A Survey on Data Compression in Wireless Sensor Networks[C], In: Proceedings of the International Conference on Information Technology: Coding and Computing, 2005, 2: 8-13.
[5] A. Ciancio, A. Ortega. A distributed wavelet compression algorithm for wireless multihop sensor networks using lifting[C]. In: Proceedings of the Thirtieth International Conference on Acoustics, Speech, and Signal Processing. 2005: 825828.
[6] A. Ciancio, A. Ortega. A dynamic programming approach to distortion-energy optimization for distributed wavelet compression with applications to data gathering in wireless sensor networks[C]. In:Proceedings of the Thirty-First International Conference on Acoustics,Speech, and Signal Processing. 2006: 14-19.
[7] A. Ciancio, S. Pattern, A. Ortega, et al. Energy-efficient data representation and routing for wireless sensor networks based on a distributed wavelet compression algorithm[C]. In: Proceedings of the Fifth International Conference on Information Processing in Sensor Networks. Nashville, 2006, 309-316.
[8] F. Marcelloni, M. Vecchio. A simple algorithm for data compression in wireless sensor networks[J]. IEEECommunications Letters, 2008,12(6): 411-413.
[9] H. G. Deng, J. X. Wang, L. L. Gao, et al. Research on new lossless data compression algorithm for WSNs[J]. Transducer and Micro system Technologies. 2008, 27(11), 60-62, 65
[10] J. L. Zhann, Q. G. Zhou, S. W. Bai, et al. BD-LZW picture compression algorithm for WSN system[C]. In: Proceedings of 3rd International Conference on Pervasive Computing and Applications. 2008, 1:146-150.
[11] M. R. Nelson. LZW Data Compression[J]. Dr. Dobb's Journal of Software Tools. 1989: 14(10): 86-87.

[12] Y. L. Zhou, X. P. Fan, S. Q. Liu, et al. Improved LZW Algorithm of Lossless Data Compression for WSN[C]. In: Proceedings of IEEE International Conference on Computer Science and Information Technology. 2010, 4: 523-527.
[13] G. L. Yang, G. N. Zhang. The Algorithm and Realization of LZW Loseless Compression[J]. Journal of Capital Normal University(Natural Science Edition). 2004, 25:12-
14. (in Chinese)
[14] Nan Zhang, Tao Tao, Ravi Vijaya Satya, and Amar Mukherjee1. Modified LZW Algorithm for Efficient Compressed Text Retrieval. School of Computer Science, University of Central Florida
[15] J. H. Shen. Series of MSP430 16-bit Ultra-low Power Microcontroller Principles and Applications[M]. Tsinghua University Press, 2004. 57 ～98. (in Chinese)
[16] Huan Zhang, Xiao-ping Fan, Shao-qiang Liu, Zhi Zhong. Design and Realization of Improved LZW Algorithm for Wireless Sensor Networks. International Conference on Information Science and Technology March 26-28, 2011 Nanjing, Jiangsu, China