ISSN: 0976-3104
SPECIAL ISSUE: Computer Science

ARTICLE

A SIMPLE LOSSLESS COMPRESSION ALGORITHM IN WIRELESS SENSOR NETWORKS: AN APPLICATION OF WIND PLANT DATA

J. Uthayakumar[1*], T Vengattaraman[1], J. Amudhavel[2]

[1]*Department of Computer Science, Pondicherry University, Puducherry, INDIA*
[2]*Department of Computer Science and Engineering, KL University, Andhra Pradesh, INDIA*

ABSTRACT

Background: *Wireless Sensor Network (WSN) consists of numerous sensor nodes and is deeply embedded into the real world for environmental monitoring. In the last decades, a large amount of wind plant data is being generated as the interest in renewable sources is growing day by day. Data Compression (DC) techniques are commonly used to reduce the amount of data transmission. As the lossless compression produces reconstructed data same as original data, it is highly useful in applications where accuracy plays a major role.* ***Methods:*** *In this paper, a lossless compression algorithm is devised to compress the data generated in the wind plant monitoring and operation. Lempel Ziv Markov-chain Algorithm (LZMA) is employed to compress the wind plant data in WSN. LZMA algorithm compresses the wind plant data effectively. The sensor node in WSN runs LZMA and sends the compressed data to Base Station (BS).* ***Results:*** *Extensive experiments were performed using real world wind plant dataset such as Eastern dataset and Western dataset. To ensure the effectiveness of LZMA algorithm, it is compared with 3 well-known compression algorithms namely Burrows Wheeler Transform (BWT), Huffman coding and Arithmetic coding (AC).* ***Conclusions:*** *By comparing the compression performance of LZMA method with existing methods, LZMA achieves significantly better compression with a compression ratio of 0.107 (Eastern dataset) and 0.099 (Western dataset) at the bit rate of 0.855 bpc (Eastern dataset) and 0.798 bpc (Western dataset) respectively.*

KEY WORDS
Data Compression
LZMA algorithm
Wind plant data
Wireless Sensor Network

Received: 5 June 2017
Accepted: 25 July 2017
Published: 18 Sept 2017

***Corresponding Author**
Email:
uthayresearchscholar@gmail.com
Tel.: +91 9677583754

INTRODUCTION

The recent advancement in wireless networks and Micro-Electro-Mechanical-System (MEMS) leads to the development of low cost, compact and smart sensor nodes. WSN is randomly deployed in the sensing field to measure physical parameters such as temperature, humidity, pressure, vibration, etc. WSN is widely used in tracking and data gathering applications include surveillance (indoor and outdoor), healthcare, disaster management, habitat monitoring, etc. A sensor node is built up of four components namely transducer, microcontroller, battery, and transceiver. The sensor nodes are constrained in energy, bandwidth, memory and processing capabilities. As the sensor nodes are battery powered and are usually deployed in the harsh environment, it is not easy to recharge or replace batteries. The lifetime of WSN can be extended in two ways: increasing the battery storage capacity and effectively utilizing the available energy. The way of increasing the battery capacity is not possible in all situations. So, the effective utilization of available energy is considered as an important design issue. Several researchers observed that a large amount of energy is spent for data transmission when compared to sensing and processing operation. This study reveals that the reduction in the amount of data transmission is an effective way to achieve energy efficiency. Data transmission is the most energy consuming task due to the nature of strong temporal correlation in the sensed data. DC is considered as a useful approach to eliminate the redundancy in the sensed data. In WSN, the sensor node runs the compression algorithm and the compressed data will be forwarded to BS.

On the other side, wind plant data plays a major role in various data driven operations like integration studies and wind plant operation [1]. The wind plant data set holds many synchronized time sequences of wind power and wind speed in various locations. Generally, the wind plant dataset are larger in size, for example, 24 terabytes of data when archived in netCDF format [2]. The easier way to handle huge amount of data is to compress them. To reduce the amount of data being stored or transmitted, DC techniques has been proposed. It is applicable to compress text, images, audio or video. The data can be alphanumeric characters in a text document or numbers which represent the samples in an audio or image waveforms or series of numbers created by some processes, etc. DC can also be termed as an efficient way of representing the data in its compact manner. The compact form can be achieved by the recognition and utilization of patterns exists in the data. The evolution of DC starts with the Morse code, developed by Samuel Morse in the year 1838 [3]. Morse code is used in telegraph to compress letters. It uses smaller sequences to represent letters which occurs more frequently and thereby minimizing the message size and transmission time. This basic principle is employed in Huffman coding [4]. Nowadays, DC techniques are very essential in most of the real time applications like medical imaging, satellite imaging to Wireless Sensor Networks (WSN) etc. DC became important for utilization of available resources effectively. Without DC, it is very difficult and sometimes impossible to store or communicate huge amount of data files.

ISSN: 0976-3104

Basically, DC techniques falls under two categories: lossless compression and lossy compression. As the name implies, lossless compression refers that no loss of information i.e. the reconstructed data is exactly same as original data. It is used in situations where the loss of information is unacceptable. Example: Text, medical imaging, satellite imaging, etc. Lossless compression is generally used to compress text. As loss of information is intolerable in text compression, compressing text using lossless compression is mandatory. It achieves the compression ratio of 2:1 to 8:1 [5]. In some situations, the reconstructed data need not to be perfectly matched with the original data and the approximation of original data is also acceptable. In those situations, lossy compression is used which involves with some loss of information within the acceptable level. This leads to higher compression ratio when compared to lossless compression. Example: image, audio and video. Lossy compression is commonly used to compress images as the loss of image quality is tolerable. The main principle behind image compression is the correlation of pixels. The neighboring pixels of an image are high correlated. These redundant details are eliminated by computing a less correlated representation of the image. Lossy compression achieves the compression ratio of 100:1 to 200:1 based on the data to be compressed. It achieves higher compression ratio at a cost of loss in quality.

Only few researches have been made on the compression of wind plant dataset. Usually, they are compressed within the process information application with the help of dead-band and swinging door [6]. As they are lossy compression techniques, the same data cannot be obtained after the compression process. It leads to the development of lossless compression of wind plant data in the real world. Once the wind plant data has been compressed, it can be useful in several ways. In WSN, the sensor node executes the compression algorithm to compress the wind plant data. Then, the compressed data will be transmitted to BS. It can be easily shared, consumes less storage with less management cost. The absence of effective lossless compression algorithm for wind plant data motivated us to perform this work.

Contribution of the paper

The contribution of the paper is summarized as follows: (i) A lossless LZMA compression algorithm is used to compress wind plant data in WSN (ii) Two wind plant dataset from NREL (Eastern and Western Dataset) is used, and (iii) LZMA results are compared with 3 well-known compression algorithms namely Huffman coding, AC, and BWT interms of Compression Ratio (CR), Compression Factor (CF) and bits per character (bpc).

Organization of the paper

The rest of the paper is organized as follows: Section 2 explains the different types of classical DC techniques in detail. Section 3 presents the LZMA compression algorithm for wind plant dataset in WSN. Section 4 explains the performance evaluation in both dataset. Section 5 concludes with the highlighted contributions, future work, and recommendations.

RELATED WORK

DC is an efficient way to reduce the amount of data being stored or transmitted. DC compression techniques have been presented in [16]. The popular coding methods are Huffman coding, Arithmetic coding, Lempel Ziv coding, Burrows-Wheeler transform (BWT), RLE, Scalar and vector quantization.

Huffman coding [7] is the most popular coding technique which effectively compresses data in almost all file formats. It is a type of optimal prefix code which is widely employed in lossless DC. It is based on two observations: (1) In an optimum code, the frequent occurrence of symbols is mapped with shorter code words when compared to symbols that appear less frequently. (2) In an optimum code, the least frequent occurrence of two symbols will have the same length. The basic idea is to allow variable length codes to input characters depending upon the frequency of occurrence. The output is the variable length code table for coding a source symbol. It is uniquely decodable and it consists of two components: Constructing Huffman tree from input sequence and traversing the tree to assign codes to characters. Huffman coding is still popular because of its simpler implementation, faster compression and lack of patent coverage. It is commonly used in text compression.

AC [8] is an another important coding technique to generate variable length codes. It is superior to Huffman coding in various aspects. It is highly useful in situations where the source contains small alphabets with skewed probabilities. When a string is encoded using arithmetic coding, frequent occurring symbols are coded with lesser bits than rarely occurring symbols. It converts the input data into a floating point number in the range of 0 and 1. The algorithm is implemented by separating 0 to 1 into segments and the length of each segment is based on the probability of each symbol. Then the output data is identified in the respective segments based on the symbol. It is not easier to implement when compared to other methods. There are two

IIOAB JOURNAL ISSN: 0976-3104

versions of arithmetic coding namely Adaptive Arithmetic Coding and Binary Arithmetic Coding. A benefit of arithmetic coding than Huffman coding is the capability to segregate the modeling and coding features of the compression approach. It is used in image, audio and video compression.

Dictionary based coding approaches find useful in situations where the original data to be compressed involves repeated patterns. It maintains a dictionary of frequently occurring patterns. When the pattern comes in the input sequence, they are coded with an index to the dictionary. When the pattern is not available in the dictionary, it is coded with any less efficient approaches. The Lempel-Ziv algorithm (LZ) is a dictionary-based coding algorithm commonly used in lossless file compression. This is widely used because of its adaptability to various file formats. It looks for frequently occurring patterns and replaces them by a single symbol. It maintains a dictionary of these patterns and the length of the dictionary is set to a particular value. This method is much effective for larger files and less effective for smaller files. For smaller files, the length of the dictionary will be larger than the original file. The two main versions of LZ were developed by Ziv and Lempel in two individual papers in 1977 and 1978, and they are named as LZ77 [9] and LZ78 [10]. These algorithms vary significantly in means of searching and finding matches. The LZ77 algorithm basically uses sliding window concept and searches for matches in a window within a predetermined distance back from the present position. Gzip, ZIP, and V.42bits use LZ77. The LZ78 algorithm follows a more conservative approach of appending strings to the dictionary.

LZW is an enhanced version of LZ77 and LZ78 which is developed by Terry Welch in 1984 [11]. The encoder constructs an adaptive dictionary to characterize the variable-length strings with no prior knowledge of the input. The decoder also constructs the similar dictionary as encoder based on the received code dynamically. The frequent occurrence of some symbols will be high in text data. The encoder saves these symbols and maps it to one code. Typically, an LZW code is 12-bits length (4096 codes). The starting 256 (0-255) entries represent ASCII codes, to indicate individual character. The remaining 3840 (256-4095) codes are defined by an encoder to indicate variable-length strings. UNIX compress, GIF images, PNG images and others file formats use LZW coding.

BWT [12] is also known as block sorting compression which rearranges the character string into runs of identical characters. It uses two techniques to compress data: move-to-front transform and RLE. It compresses data easily to compress in situations where the string consists of runs of repeated characters. The most important feature of BWT is the reversibility which is fully reversible and it does not require any extra bits. BWT is a "free" method to improve the efficiency of text compression algorithms, with some additional computation. It is s used in bzip2. A simpler form of lossless DC coding technique is RLE. It represents the sequence of symbols as runs and others are termed as non-runs. The run consists of two parts: data value and count instead of original run. It is effective for data with high redundancy.

A simpler form of lossless DC coding technique is Run Length Encoding (RLE). It represents the sequence of symbols as runs and others are termed as non-runs. The run consists of two parts: data value and count instead of original run. It is effective for data with high redundancy. For example, “ABCABBBBBC” is the input sequence, then the starting 4 letters are assumed as non-run with length 4, and the next 4 letters are assumed as a run with length 5 while B is repeated 5 times. RLE identifies the run of input file, saves the symbol and length of each run. These runs are used to compress the input file and non-runs are kept uncompressed. It is not effective in cases where the redundancy is low and leads to increase in original file size. It is ineffective for less redundant data and leads to increase in compressed file size greater than original file size. Input: YYBCCECCDESSEEEEER and its Output: 2Y1B2C1E2C1D1E2S5E1R. Though RLE is simpler to implement and faster, it fails to achieve better compression when compared to other algorithms. It is used in line drawing, animation and graphic icons, fax, palette-based images like textures.

Quantization is an easiest way to represent larger set of values to a smaller set. The quantizer input can be either scalar or vector. When the input is scalar, then it is called scalar quantizer. When the input is vector, then it is called vector quantizer. Scalar quantization is the simpler and commonly used lossy compression. It is a process of mapping an input value x to a number of output values. Vector quantization (VQ) is a different type of quantization, which is typically implemented by choosing a set of representatives from the input space, and then mapping all other points in the space to the closest representative. Initially, the source output is grouped into blocks or vectors. These vectors are used as the input to quantizer. In both sides of encoder and decoder, a codebook (set of L-dimensional vectors) is available. The vectors in the codebook are termed as code-vectors and each code vector is assigned to an index value. The elements of this code vector are the quantized values of the source output. In order to report the decoder about which code vector was found to be the closest to the input vector, the index value of the code-vector is transmitted. As the decoder also has same codebook, the code vector can be retrieved from its binary index.

ISSN: 0976-3104

Lossless compression techniques are widely used in text compression as loss of information is not tolerable while compressing text. A novel data compression technique is proposed to compress general data using a logical truth table (Mahmud, 2012). Here, the two bits of the data are represented by only one bit. Using this method, the data bits can be represented by its half bits only. i.e. 128 bits can be represented by 64 bits; 64 bits can be represented by 32 bits; 1GB data can be represented by 512MB and so on. The logic truth table like two inputs and four combinations are given below.
IF A=0 and B=0 then Z=0
IF A=0 and B=1 then Z=Ô
IF A=1 and B=1 then Z=1
IF A=1 and B=0 then Z=î

First, input data is tested to identify whether it is odd or even. If it is odd, extra bit is added at the last (0 is added when the last bit is 0 and vice versa). Then the data is split into two bits and truth table is applied to compress the data. Traditionally, there are two levels to represent digital data but proposed method uses four levels. The proposed method compresses the 20 bit data into 10 bits. This method uses Compression Ratio and Compression Factor as performance metrics. The author fails to compare the proposed method with existing method. It can be employed in wired and wireless scenarios.

Although some of common compression techniques can be applicable to some preferred applications, the concentration has been on the technique rather than on the application. However, there are certain techniques for which it is impossible to separate the technique from the application. This is because that several techniques depend on the properties or characteristics of the application. Therefore, some compression techniques were developed with focus on particular applications.

A new database compression method is proposed in large databases using association rule mining. Association rule mining from a database can identify frequent item sets in the database relations. The extracted association rules represent the repeated item sets in the database and they are compressed to decrease the file size. There are five steps in database compression: i. An effective association rule mining method using a tree called Class Inheritance Tree (CIT) is employed, ii. Collection of compression rules from strong association rules, iii. Compressed space for each compression rule is calculated, iv. A heuristic method is employed to resolve the conflicts of compression rules and v. Compression ratio is computed.

To reduce the Inter Track Interference (ITI) read latency of Shingled Magnetic Recording, two lossless compression mechanisms are used. It is a known fact that most of the files stored in disks are lossless compressible. When a sector of user data can be losslessly compressed to a particular level, free storage space will become available to store ECC redundancy. This extra space can be used for storing a stronger ECC than normal ECC for the current sector. This leads to decrease the probability of reading one or multiple neighboring tracks for explicit ITI compensation. Lempel-Ziv-Stores-Symanski (LZSS) algorithm is used for compressing several file types stored in hard disks. This method uses intra-sector data compression method to minimize the read latency is shingled recording. Here, lossless data compression technique is employed on multiple physically consecutive sectors on the same track to increase the compression efficiency. It results to reduction in read latency overhead by enabling a stronger ECC. This is termed as virtual sector compression. Because of overlapping writing in shingled recording, update-in-place feature fails to work. This makes the virtual sector compression practically possible. Two scenarios (fixed size and context aware virtual sector) are used to investigate the results of virtual sector compression. Intra-sector lossless compression results to the decrease in disk read latency.

Phasor measurement units (PMU) are used to monitor and regulate the power grid where the devices records globally synchronized measurements of voltage and current signals. It provides the respective phasor magnitude and angles at typical rate of 30 frames/s. To compress the phasor angle data, a new pre-processor is proposed. An adaptive high performance entropy encoder called golomb rice coding. This method involves two stages: In the first stage, preprocessing of data takes place. A new preprocessor based on frequency compensated differential encoding is proposed. In the second stage, golomb rice encoding is used. It is a lossless compression technique where the data transmission from space vessels takes place and higher throughput is achieved with limited memory and computational resources. Golomb encoding is designed to encode non-negative, integer valued signals in which probability value n decreases exponentially. It is very effective than existing coding techniques like Huffman coding and Arithmetic coding. Golomb rice coding is simpler to implement in hardware because it uses only few operators namely bitwise left, bitwise right, bitwise AND, OR, ++ and --. Next, mapping of values to GR codes does not require any lookup tables or trees/ GR compress 8 million phasor angles/second shows that it can be very useful for applications requires less delay. Data rate, compression rate and Error Propagation Rate (EPR) are used to evaluate the performance of this technique.

ISSN: 0976-3104

LZMA COMPRESSION ALGORITHM ON WIND PLANT DATA

LZMA compression is used to compress real time data generated rapidly. The overall operation is shown in [Fig. 1]. The LZMA algorithm compresses the wind power and speed values to reduce the size of the data. LMZA is the modified version of Lempel-Ziv algorithm to achieve higher CR [13]. It is a lossless DC algorithm based on the principle of dictionary based encoding scheme. LZMA utilizes the complex data structure to encode one bit at a time. It uses a variable length dictionary (maximum size of 4 GB) and is mainly used to encode an unknown data stream. It is capable of compressing the data generated at a rate of 10-20 Mbps in a real-time environment. Though it uses larger size dictionary, it still achieves the same decompression speed like other compression algorithms.

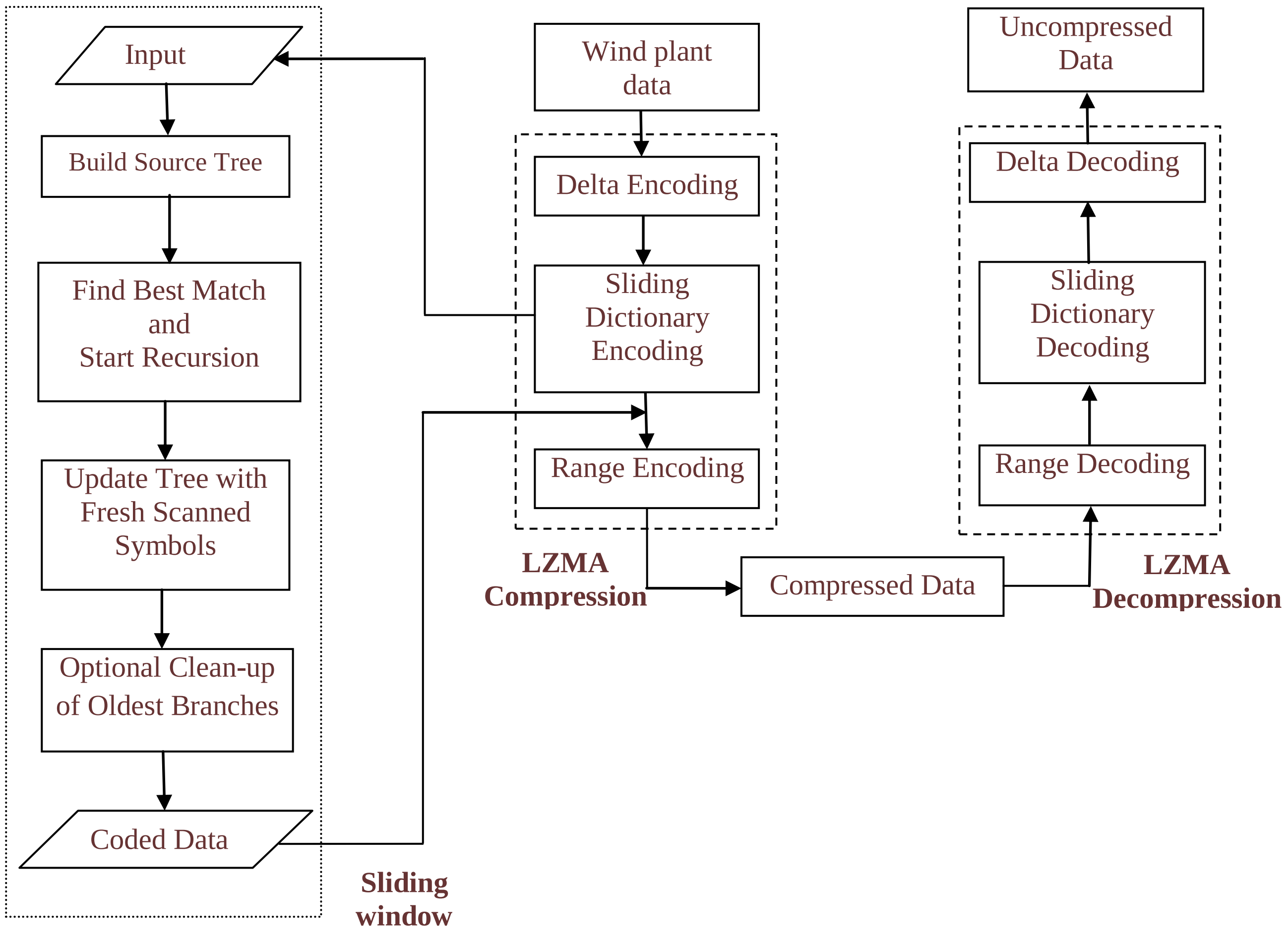

Fig. 1: Overall operation of LZMA on wind plant data.

LZ77 algorithm encodes the byte sequence from existing contents instead of the original data. When no identical byte sequence is available in the existing contents, the address and sequence length is fixed as '0' and the new symbol will be encoded. LZ77 also employs a dynamic dictionary to compress unknown data by the help of sliding window concept.

LZMA extends the LZ77 algorithm by adding a Delta Filter and Range Encoder. The Delta Filter alters the input data stream for effective compression by the sliding window. It stores or transmits data in the form of differences in sequential data instead of complete file. The output of the first-byte delta encoding is the data stream itself. The subsequent bytes are stored as the difference between the current and its previous byte. For continuously varying real time data, delta encoding makes the sliding dictionary more efficient. For example, consider a sample input sequence as 2,3,4,6,7,9,8,7,5,3,4 7. The input sequence is encoded with LZMA technique and the encoded output sequence is 2, 1, 1, 2, 1, 2,-1,-1,-2,-2, 1. So, the number of symbols in the input sequence is 8 and the number of symbols in output sequence is 4.

Static and adaptive dictionary are the commonly used dictionaries. The static dictionary uses the fixed entries and constants based on the application of the text. Adaptive dictionaries take the entries from the text and generate on run time. A search buffer is employed as a dictionary and the buffer size is chosen based on the implementation parameters. Patterns in the text are assumed to occur within range of the search buffer. The offset and length are individually encoded, and a bit-mask is also separately encoded. Usage of an appropriate data structure for the buffer decreases the search time for longest matches. Sliding Dictionary encoding is comparatively tedious than decoding as it requires to identify the longest match. Range encoder encodes all the symbols of the message into a single number to attain better CR. It efficiently deals with probabilities which are not the exact powers of two.

The steps involved in range encoder are listed below.

- Given a large-enough range of integers, and probability estimation for the symbols.
- Divide the primary range into sub-ranges where the sizes are proportional to the probability of the symbol they represent.
- Every symbol of the message is encoded by decreasing the present range down to just that sub-range which corresponds to the successive symbol to be encoded.
- The decoder should have same probability estimation as encoder used, which can either be sent in advance or derived from already transferred data.

PERFORMANCE EVALUATION

To ensure the effectiveness of the LZMA algorithm while compressing wind plant data, its lossless compression performance is compared with 3 different, well-known compression algorithms namely Huffman coding, AC, and BWT.

Metrics

In the section, various metrics used to analyze the compression performance are discussed. The performance metrics are listed below: CR, CF, and bpc.

Compression Ratio (CR)

CR is defined as the ratio of the number of bits in the compressed data to the number of bits in the uncompressed data and is given in Eq. (1). A value of CR 0.62 indicates that the data consumes 62% of its original size after compression. The value of CR greater than 1 result to negative compression, i.e. compressed data size is higher than the original data size.

$$CR = \frac{\text{No. of bits in compressed data}}{\text{No. of bits in original data}} \qquad (1)$$

Compression Factor (CF)

CF is a performance metric which is the inverse of compression ratio. A higher value of CF indicates effective compression and lower value of CF indicates expansion.

$$CF = \frac{\text{No. of bits in original data}}{\text{No. of bits in compressed data}} \qquad (2)$$

bits per character (bpc)

bpc is used to calculate the total number of bits required, on average, to compress one character in the input data. It is defined as the ratio of the number of bits in the output sequence to the total number of character in the input sequence.

$$BPC = \frac{\text{No. of bits in compressed data}}{\text{No. character in the original data}} \qquad (3)$$

Dataset description

For experimentation, two publicly available wind pant dataset are used. The data is collected by National Renewable Energy Laboratory (NREL). The two major sources of data is used in this work are NREL Eastern [14] and NREL Western dataset [15]. The data were collected by AWS Truepower and 3-TIER. The Eastern dataset holds time sequences of 1326 hypothetical wind plant of different locations, capacities and wind turbine power curves, the capacity of the wind plant are in the range of 100 to 1435MW with a median capacity of 370MW. These data were archived at an accuracy of 100kW and 0.001m/s. The Western dataset holds time series of 32000 hypothetical wind plant of 1.2 million different locations, capacities and wind turbine power curves, with the same capacity of 30 MW. These data were archived at an accuracy of 1kW and 0.01m/s.

RESULTS AND DISCUSSION

To highlight the good characteristics of LZMA based lossless compression algorithm for wind plant data, it is compared with 3 states of art approaches namely Huffman coding, AC and BWT coding. A direct comparison is made with the results of existing methods using the same set of 2 datasets. Table 1 summarizes the experiment results of compression algorithms based on three compression metrics such as CR, CF and bpc. As evident from [Table 1], the overall compression performance of LZMA algorithm is significantly better than other algorithms on both two dataset. It is observed that LZMA algorithm achieves slightly higher compression on Western dataset than Eastern dataset. It is also noted that Huffman coding and arithmetic coding produces poor results than BWT. The compression performance on 2 different dataset files reveals some interesting facts that the compression algorithms perform extremely different based on the nature of applied dataset. The existing BWT method achieves better compression performance than Huffman coding and AC, but it fails to give performs well than LZMA algorithm. Likewise, Huffman and Arithmetic coding also produce appropriately equal compression performance. This is due to the fact that the efficiency of the arithmetic coding is always better or at least identical to a Huffman code. Similar to Huffman coding, Arithmetic coding also tries to calculate the probability of occurrence of particular certain symbols and to optimize the length of the necessary code. It achieves an optimum code which exactly corresponds to the theoretical specifications of the information theory. A minor degradation result from inaccuracies, because of correction operations for the interval division.

Table 1: Performance comparison of various compression methods on wind plant data

Dataset	Compression techniques	Compression Ratio (CR)	Compression Factor (CF)	bits per character (bpc)
Eastern Dataset	LZMA	0.107	9.356	0.855
	Huffman	0.566	1.766	4.528
	BWT	0.133	7.545	1.060
	Arithmetic	0.569	1.789	4.470
Western Dataset	LZMA	0.099	10.026	0.798
	Huffman	0.665	1.769	4.520
	BWT	0.109	9.151	0.874
	Arithmetic	0.559	1.787	4.476

On the other side, Huffman coding generates rounding errors because of its code length and is limited to a multiples of bit. The variation from the theoretical value is more than the inaccuracy of arithmetic coding. In overall, LZMA results in effective compression than existing methods. Generally, dictionary based coding approaches find useful in situations where the original data to be compressed involves repeated patterns. LZMA extends LZW with the range encoding technique which enables to produce significantly higher compression than LZW. Interestingly, LZMA achieves significantly better compression with the compression ratio of 0.107 (Eastern dataset) and 0.099 (Western dataset) at the bit rate of 0.855 (Eastern dataset) and 0.798 (Western dataset) respectively.

CONCLUSION

This paper employs a Lempel Ziv Markov chain Algorithm (LZMA) lossless compression technique to compress the wind plant data. LZMA is a lossless DC algorithm which is well suited for real time applications. LZMA algorithm compresses the wind plant data effectively. Extensive experiments were performed using the real world wind plant dataset such as Eastern dataset and Western dataset. To ensure the effectiveness of LZMA algorithm, it is compared with 3 well-known compression algorithms namely Burrows Wheeler Transform (BWT), Huffman coding and Arithmetic coding (AC). By comparing the compression performance of LZMA method with existing methods, LZMA achieves significantly better compression with a compression ratio of 0.107 (Eastern dataset) and 0.099 (Western dataset) at the bit rate of 0.855 bpc (Eastern dataset) and 0.798 bpc (Western dataset) respectively.

CONFLICT OF INTEREST

There is no conflict of interest.

ACKNOWLEDGEMENTS

None

FINANCIAL DISCLOSURE

None

REFERENCES

[1] Ummels HHB, Tande JO, Estanqueiro A, et al. [2009] Design and operation of power systems with large amounts of wind power Final Report of IEA Task 25, VTT Research Notes.

[2] Potter CW, Lew D, McCaa J, Cheng S, Eichelberger S, Grimit E. [2008] Creating the dataset for the western wind and solar integration study (U.S.A)," in *Proc. 7th Int. Workshop on Large Scale Integration of Wind Power and on Transmission Networks for Offshore Wind Farms*, Madrid, Spain.

[3] Hamming RW. [1986] Coding and information theory.

[4] Vitter Jeffrey S. [1987]. Design and Analysis of Dynamic Huffman Coding. 34: 825–845.

[5] Drost SW, Bourbakis N. [2001] A Hybrid system for real-time lossless image compression. Microprocess. Microsyst. 25: 19–31. doi:10.1016/S0141-9331(00)00102-2

[6] Srisooksai T, Keamarungsi K, Lamsrichan P, Araki K. [2012] Practical DC in wireless sensor networks: A survey. *J Netw Comput Appl* 35 (1): 37–59.

[7] Huffman DA. [1952] A Method for the Construction of Minimum-Redundancu Codes. 1098–1102.

[8] Witten Ian H, Neal Radford M, Cleary John G. [1987] Arithmetic coding for DC. *Commun. ACM.* 30(6): 520–540.

[9] J Ziv. A Lempel. [1977] A Universal Algorithm for DC. *IEEE Trans. Inf. Theory*, 23(3): 337–343.

[10] Ziv J, Lempel A. [1978] lz78.pdf. IEEE. 530–536.

[11] Welch TA. [1984] A technique for high-Performance DC. *IEEE*. 8–19.

[12] Burrows M, Wheeler D. [1994] A block-sorting lossless DC algorithm. *DC*. 124:18.

[13] Tu Z, Zhang S. [2006] A Novel Implementation of JPEG 2000 Lossless Coding Based on LZMA. in *Proceedings of the Sixth IEEE International Conference Computer and Information Technology*.

[14] NREL, Wind Integration Datasets Aug. [2010] [Online]. Available: http://www.nrel.gov/wind/integrationdatasets/eastern/data.html

[15] NREL, Wind Integration Datasets Aug. [2009] [Online]. Available: http://www.nrel.gov/wind/integrationdatasets/western/data.html