International Journal of Innovative Technology and Exploring Engineering (IJITEE)
ISSN: 2278-3075 (Online), Volume-9 Issue-3, January 2020

A Classification of Lossless and Lossy Data Compression Schemes

Lee Chin Kho, S. S. Ngu, A. Joseph, D. A. A. Mat, K. Kuryati

Abstract*: *Data compression is a promising scheme to increase memory system capacity, performance and energy advantages. The compression performance could affect the overall network performance when compression scheme is implemented in a communication field. Many data compression schemes have been introduced. Most of other researchers choose very limited parameters to analyze the performance of the selected data compression scheme. This paper classifies the major data compression schemes according to nine different perspectives, such as homogeneity, purpose, accuracy, structuring of the data, repetition distance, structure sharing, number of passes, sampling frequency, and sample size ratio. Various data compression schemes are examined and classified according to the parameters mentioned above. The classification will provide researchers with the in-depth insight on the potential role of compression schemes in memory components and network performance of future extreme-scale systems.

Keywords : Data Compression, Lossless, Homogeneity, Accuracy.

I. INTRODUCTION

Data compression is the process of converting data into another format that requires less storage (more efficient) than the original format with some satisfactory accuracy. It is considered as one of the information encoding techniques. Sometimes it is also referred to as bit-rate reduction when applied in networking. In 1848, Morse Code was introduced, which is considered to be the first modern data compression [1-3]. Data compression theory is an extension of the basic information theory, same as any other encoding. Data compression is mainly focused on statistical inference information theory. Compression can be either lossless or lossy.

Revised Manuscript Received on January 30, 2020.
* Correspondence Author

Lee Chin Kho*, Department of Electrical Engineering, University Of Adelaide, Adelaide, Australia.

Sze Song Ngu, Department of Electrical Engineering, University Of Adelaide, Adelaide, Australia.

Annie Joseph, Senior Lecturer, Department of Electronic, University **Malaysia**

Dayang Azra Binti Awang Mat, Senior Lecturer in Faculty of Engineering, University Malaysia.

Kuryati Kipli, Department of Electrical and Electronics, Faculty of Engineering, UNIMAS Malaysia.

Compressed data from lossless compression must be decompressed to exactly its original value.

Compression belongs to algorithmic information theory category for lossless compression and rate-distortion information theory for lossy compression. Both areas were established by Claude Shannon, in the late 1940s and early 1950s as the base for all communication, signaling and data handling [4, 5].

Shannon Fano (SF) coding was the first compression scheme built based on information theory.

When a data unit or sequence of units from the source is com-pressed, the resulting compressed representation will be referred to as the representing code within this study. The representing code should achieve the desired target, either faster transmission, lesser storage or energy consumption, and the degree of compression based on unit by unit individually, which could vary and rarely considered by itself. The aggregate overall compression degree is more significant for the whole data together with the overall performance especially when compressing data from different sources.

Designers of data compression schemes have to handle tight trade-off between the conflicting targets. Those targets are the degree of compression, and the computational resources required (time, temporary storage and energy). Lossy compression faces an additional target that is the amount of distortion introduced, which is highly dependent on the degree of compression. The suitable position in the trade-off limited space is usually decided during design or implementation to target the application of the compression scheme according to a specific situation [3, 6]. When data compression is applied in computer networks, compression is traditionally activated manually by the user at the sender end devices before transferring. When the receiver end devices receive the compressed data, the user again activates decompression manually. Sometimes data compression function is embedded in computer applications or lower layers of end devices. Either way, compression has been largely restricted to end-to-end use. The compression performance could affect the overall network performance.

The objective of this study is to classify the major data compression schemes based on different perspectives. There are 11 common compression schemes such as Joint Photographic Experts Group (JPEG), Run Length Encoding (RLE), Huffman, Arithmetic, Lempel Ziv (LZ), Van Jacobson's Header (VJHC), IP Header Compression (IPHC), Compression Real-time Transport Protocol (CRTP), RObust Header Compression (ROHC), Adaptive Compression-based Technique (ACT), and Lightweight DeCompression (LDC) are considered here.

The rest of this review is organized as follows. Section 2 discusses the related work and motivation.
Section 3 explains the perspectives that use to classifying of compression schemes and Section 4 discusses the listed compression schemes. The classification results are illustrated in Section 5 and conclusion in Section 6.

II. RELATED WORKS AND MOTIVATION

There are a number of studies that have addressed the compression techniques in different kind of application. The most com-mon applications are data compression, image compression, voice compression, etc. M. A. Laham al. et. had presented a comparative study between various algorithms of data compression techniques in 2007 [2]. In the paper, the compression techniques such as RLE, Huffman, LZ 77, and LZW are briefly dis-cussed. Then, the author compared the compression ratio of LZW and Huffman for the file of .DOC, .BMP, .JPG, and .GIF. Based on the results, LZW performed badly in image data compression, especially .GIF and .JPG file. LZW enlarges the file size to maximum 40% of the input file size, whereas Huffman enlarge 5% of the input file size. Moreover, LZW performed as well as Huffman in text format of .DOC file. Both techniques achieve 80% of compress ratio in text file. Unlikely, the results of compression ratio for both techniques are inconsistent. Huff-man sometime performs better than LZW or the opposite way. We improve the above paper by determining the percentage of average maximum redundancy in a file before compress, so that the relationship of redundancy and compression ratio for compression techniques of RLE, Huffman, arithmetic and LZW can be identified.

In addition, S. Shanmugasundaram al. et. in [3] had provided a survey of text compression algorithms based on statistical and dictionary. The author used a parameter metric of Bit Per Character (BPC) to compare the performance of compression algorithms in twelve different text files. The results showed that the average BPC for RLE was around 8 bits, whereas Shannon Fano coding, Huffman coding, Adaptive Huffman coding and arithmetic coding of BPC is between 5 to 6 bits. Although this parameter metric showed the bits reduction per character among the different compression techniques, it cannot present the over-all performance of the compression in term of complexity, time, efficiency, etc.

III. CLASSIFICATION OF DATA COMPRESSION SCHEMES

All previous works on data compression classification is done by selectively choosing very limited parameters. By doing this, the classification may provide good reference in one aspect but miss out on the others. This paper proposes a methodology that includes all nine different perspectives to classify the different compression schemes. Selection of data compression scheme can be done after more thorough classification. Generally, the classifications divide the compression schemes into two or more categories which have many-to-many relations between all categories from all the other classifications. The resulting overall categories form a hypercube of degree nine.

A. Heterogeneity

The first classification divides compression schemes into mixed (heterogeneous) versus homogeneous data. The classification depends on whether the data compressed originated from different sources or the same source. Heterogeneity also refers to the structure of the data itself, whether homogeneous having the same structure or mixed structures. Homogeneous data usually exhibit much better compression performance and degree compared to mixed data. Since mixing is usually done randomly and out of order without any synchronization, mixed data usually results are around those of the worst part of the mix with lowest correlation with respect to both its compression performance and degree. Thus mixed data are usually handled by part-wise (blocking) compression to reduce the effect of lack of correlation caused by the mixing itself. Almost all known commercial compression tools utilize part-wise compression to handle general data. Whatever the way the different data from different sources are mixed, the resulting mix is totally unpredictable ranging from much higher to much lower correlation that each data source separately. Compressing the data from the different sources separately gives much more predictable performance. For example, the two data sources abab originally having repetition 2 × ab and cdcd having 2 × cd, if mixed could give acbdcadb with no repetition at all.

B. Purpose of Compression

The two main purposes are either storage or communication. Storage compression schemes target smaller final storage while sacrificing computational costs of compression; compression time, compression computational energy and temporary storage. This class of compression schemes are usually used for archiving. Both categories handle either homogeneous or mixed data. Online archiving overcomes the long delay encountered by utilizing heavy caching mechanisms. On the other hand, communication compression schemes target much faster compression time as well as lower compression computational energy. Usually communication compression schemes sacrifice compression degree to achieve the strict requirements. Most network coding utilizes power versus speed trade-off to choose the suitable bit per symbol rate.

C. Required Accuracy

Compression schemes can be generally classified into lossless and lossy. Lossy compression is used when some information loss can be accepted depending on the requirements of usage. Lossy schemes are used for communication purposes more than for storage purposes. For instance, when compressing still images, the human eye is more sensitive to subtle variations in luminance than variation in colour. Thus JPEG image compression works by “rounding off” less important information to coarser quantization levels. Lossless compression is used when the data decompressed must match exactly the original data before compression. Lossy compression is also widely used in the lower layer basic network coding compression schemes, which is al-ready highly tolerant to noise i.e., accuracy loss. Lossless compression is used for critical control data among other usages, such as IP header compression.

D. Structuring of Data Compressed

Structured data units usually exhibit repeated (and usually redundant) fields from one data unit to another. That redundancy can be easily totally eliminated by taking advantage of knowing the data structure format. Representing data relatively instead of using absolute reference is the most common methodology to eliminate the redundancy in the repeated data structures.

On the other hand, raw data also known as unstructured data lack any data structure format and cannot be handled using those compression schemes. Unstructured data is merely considered as a stream of similar data unit of unknown internal structure. Most of the lossy compression schemes and network encoding are mainly structure data compression associated with other compression schemes. For unstructured data, compression schemes try to detect repetitions of the whole data unit regardless any internal data structure. IP header compression and data referencing in computer programming are among the well-known areas of using structured data compression. Unstructured compression includes a lot of schemes started from the original Morse code passing through the well-known LZ schemes. Generally, un-structured compression schemes try to remove frequent repetitions of data units by encoding that repetition using some smaller structure.

E. Repetition Distance

The distance between repetitions targeted for compression can be used to distinguish compression schemes. Consecutive compression schemes target repetitions that are consecutive between either single data units or sequences of data units. Distributed encoding structures are used to represent consecutive repetitions more effectively. Consequence compression schemes are better used when some data is available ahead, but still can be used when not available with less efficiency. Non-consecutive compression schemes are used when the distance between the repetitions is longer, other common dictionary like structures are used as look up tables. Methods of building those dictionaries structured are varies a lot from one scheme to another, trees (Huff-man), arrays, tables, lists and so on. When the data size is quite big, a pre-defined size sample of the whole data is scanned to statistically or probabilistically detect the repetitions. The size of data, which that sample is representing, is also different from one scheme to another. Both consequence and non-consequence schemes can be applied for either structured or unstructured data. RLE scheme, as the one implemented in BMP images, is probably the most common consecutive compression scheme. An example of non-consecutive schemes, colour palettes of most image and video data formats are considered a form of common dictionary for a more efficient representation of repeated colours.

F. Encoding/Decoding Structure Sharing

The dictionary like structures generated before or during compression for encoding is sometimes attached with the com-pressed code for either storage or transmission such as colour palettes of image and video compressed data formats as well as most consecutive compression schemes. The dictionary like structures can be either one big structure as the colour palettes in images or distributed smaller structures such as run-length encoding fields preceding compressed runs of data in RLE schemes. When the dictionary structure is attached, it will affect the compression degree since the compressed code size will include the dictionary size. Accordingly, a practical size limit is usually imposed on the dictionary size so that compressed code size does not to exceed the input code size. Other compression schemes do not need to attach the dictionary structures and dis-pose the generated structures; LZ schemes are an example of this category. The decoders of those schemes regenerate the dictionary during decompression without any prior knowledge needed about the dictionary used for compression. When the dictionary is regenerated, additional computational resources are utilized, time, energy and temporary storage. Since the dictionary size does not directly affect the compression degree, the dictionary size can virtually grow infinitely to capture more repetition and achieve better compression degree. In some compression schemes, the dictionary is either fixed or managed independent of data by explicit periodic synchronization mechanisms, Morse code, most lossy schemes and network coding are examples of the former while Compressed Real-time Transport Protocol (CRTP) is an example of the latter. Fixed dictionary cannot adapt to different data contents and can miss lot of repetition resulting in poor compression degree. On the other hand, fixed dictionaries are sorted and optimized for faster access, thus faster encoding/decoding. Periodic dictionary synchronization causes a lot of accesses to the replicated dictionary structures, which adds a lot of additional load to the storage or traffic to the network.

G. Number of Passes Scanning the Input for Compression

Optimal compression schemes require a pre-parse of the input code (at least part of it, sample) to build the dictionary structure that will be used for compression. Additionally, some schemes calculate or predict the final compression degree statistically or probabilistically to decide compression feasibility. Some schemes use more than one pass to perform such pre-processing to prepare the required structures before starting the final real compression pass. LZ Sorter Szymanski (LZSS) decides if the feasibility of the expected compression in a separate pre-pass, likewise arithmetic like schemes builds the dictionary structures in a separate pre-pass. Most literature refer to the former multi-pass compression schemes as statistical compression, since only small sample size ratio are used during the pre-passes. Other compression schemes perform only one pass during which the dictionary structure is built simultaneously with the compression itself, such as most of the LZ schemes. Multi-pass compression schemes usually required longer compression time to achieve better compression degree compared with single compression schemes by adding overhead pass(es). Higher sampling frequencies as well as larger sample size ratios an upscale the effect of this overhead on the compression time. Multi-pass schemes generally achieve better compression degree for heterogeneous data compression to better capture the non-homogeneous mix of data being compressed.

H. Sampling Frequency (Blocking)

When the data is too big, parts of the data are scanned or sampled independent of other data parts to detect repetitions in that part alone.

Morse code like schemes scan or scanned a chosen representative part of the data as a sample to stand for all the data in the world for eternity. On the other hand, LZ like schemes scans every data part to statistically obtain its independent repetition representation. When the data size grows above some limit most schemes start blocking data and handling it part by part (part-wise compression), each block is sampled separately. Different sampling frequencies are usually used in any consecutive and non-consecutive compression schemes whether data is structured or not. The bigger the size of each sample, the lower the frequency. The combination of sampling frequency and sample size ratio for any compression schemes strongly affects its compression degree and performance. The less homogeneous the data is; the higher sampling frequency should be with the same sample size ratio.

I. Sample Size Ratio

The ratio of the size between the scanned sample and the part of the data it is representing. Some data compression schemes scan only a percentage of the whole data (part or block) to find repetitions statistically or probabilistically, the detected repetitions will be used to represent the whole data. Morse code scheme used infinitely growing data size with one time only sample frequency, resulting in almost zero sampling size ratio. Most LZ like schemes used 100% sample size ratio, by scanning the whole data or block to find all the existing repetitions. Human like schemes choose lower percentage for sample size ratio. The less homogeneous the data is, the higher the sample size ratio should be within the same sampling frequency. Choosing the suitable combination of both sampling frequency and sample size ratio for any compression schemes is strongly affected by the information entropy and correlation within the compressed data. Peri-odic dictionary synchronization schemes are usually combined with huge data (blocks). Where fixed dictionary structure schemes are limited for near zero sample size ratios, usually the small sample is good enough to represent the whole data (block).

IV. EXAMPLES OF COMPRESSION SCHEMES

This section presents seven different examples representing a lot of the joint data compression schemes discussed in the previous section. JPEG compression schemes are first discussed to represent both categories of lossy compression as well as structured compression schemes. Consecutive compression is represented afterwards by the RLE compression scheme. Huffman, arithmetic and LZ are explained as general examples in the succeeding sub section. The last sub section introduces an overview of various compression schemes designed for communication purpose.

A. Joint Photographic Experts Group (JPEG) Schemes

JPEG schemes are lossy compression with a few lossless exceptions, which is applied for the generally homogeneous structured data. The pixels colour data structures as well as locality of image areas structure are heavily exploited by structured compression schemes. Still image compress standards of JPEG target storage efficiency, while motion pictures target both storage and communication such video conference streaming applications. Both JPEG standards also utilize consecutive compression schemes.

Distance compression schemes (palette dictionaries) are optionally used and attached to the compressed code when sharing. Although one pass compression can be implemented for most JPEG schemes, it can be quite a complicated project. Most JPEG schemes divides big image data into small times (64 pix-els) to which the main compression is applied with 100% sample size ratio per block.

B. Run-Length Encoding (RLE) Scheme

RLE schemes [7] are lossless compression, which is applied for the generally homogeneous consecutive unstructured data (the internal structure is not utilized). The locality of image areas structures are the only exploited feature to achieve the compression. A lot of network encoding makes use of RLE similar schemes to handle phase ambiguity problems. RLE scheme are also used for efficient storage as in BMP and the now almost history, archive tapes. RLE schemes are quite easily implemented as a single pass coder using distributed attached dictionary like structure. RLE schemes require look-ahead mechanisms that are usually far smaller than to be considered as an extra pass without any real sampling (100% sample size ratio).

C. Huffman Scheme

Huffman schemes [8] are lossless compression, which is applied for the generally homogeneous consecutive unstructured data. Due to the long pre-pass required or enormous tree bookkeeping, Huffman schemes are restricted to storage usage only. Adaptive Huffman schemes with tree regeneration can be done in a single pass unlike the regular static Huffman with the attached dictionary. Huffman schemes must scan the whole source data in the pre-pass (100% sample size ratio), some practical considerations can divide large input into smaller blocks according to the available resources and limitations.

D. Arithmetic Schemes

Arithmetic compression [8] is mainly lossless compression un-less fraction accuracy loss can be accepted. Arithmetic compression is applied for the generally homogeneous consecutive un-structured data. Due to the long pre-pass required, arithmetic schemes are restricted to storage usage only. The attached dictionary represents a significant percentage of the compressed code size. The whole source data must be scanned first in the pre-pass (100% sample size ratio), again for some practical considerations it can divide large input into smaller blocks according to the available resources and limitations.

E. Lempel Ziv (LZ) Schemes

LZ [9] is mainly designed for lossless compression but can be easily modified to for lossy applications. LZ is applied for the generally homogeneous consecutive unstructured data. LZ is used in regular storage purposes; it is virtually the standard of practical lossless data compression. LZ is also suitable for communication application owing to the facts that its schemes (fast LZ and ultra-fast-LZ) are probably the fastest existing compression schemes and also its dictionary regeneration capability. With no pre-pass needed in most of LZ schemes the input source data is parsed only one time and the dictionary keeps including all the inputs scanned resulting 100% sample size ratio with once per input data block sampling.

F. Compression Schemes for Communication

In the transport layer, source data (packets) to be compressed is considered mixed structured data consisting of; header (signaling information) and user information which also called as data or payload, except for Molecular Sequence Reduction (MSR) scheme where packets are considered as homogeneous unstructured data. The compression data targeted can be categorized into three; header (homogeneous), data (homogeneous) and both (mixed). Header compression must be lossless, due to the importance of its contents for control and signaling. For example, the Internet protocol (IP) header consists of information for routing the data to its destination. If some information is lost or changed, the packets will fail to reach its destination. In data compression, the information is compressed according on the user requirements. For instance, in the case of lossy compression, the images quality is reduced by permanently eliminating certain information. To compress both information fields, loss-less schemes are commonly used. Some examples of header compression technique used in networks are described below. MultiProtocol Label Switching (MPLS) and Asynchronous Transfer Mode (ATM) are among the state of art network proto-cols with compression scheme concepts as its core design philosophy.

Van Jacobson's Header Compression (VJHC) is the first internet compression scheme that compresses the TCP/IP header in low-speed serial links [10]. It reduces the normal 40 byte TCP/IP packet headers to 3-4 bytes for the average case by sending the differences in the header fields instead. By this way, VJHC can get nearly 50% of compression of the header. The IP Header Compression (IPHC) extends VJHC. It is commonly used for packets over Transport Control Protocol/ Internet Protocol (TCP/IP) and User Datagram Protocol/Internet Protocol (UDP/IP) in low speed links, for TCP streams, IPHC is identical to VJHC [11]. Since UDP streams are connectionless, IPHC introduces the concept of context with some unique CID for each data stream. This context is used to save the data stream header fields that are static or have little change among the continuous packets shared between both the encoder and decoder.

Compression Real-time Transport Protocol (CRTP) was developed to compress streaming multimedia data packets of the Real Time Protocol (RTP). However, CRTP also can compress UDP and IP headers, the 40 bytes of RTP/UDP/IP packet headers can be compressed to only 4 bytes [12]. For multimedia data quality is reduced by lossy compression of the data field of the packets. CRTP wastes a lot of bandwidth when for synchronizing be-tween the encoder and decoder. CRTP can perform well on the small round trip time (RTT) link. In long RTT, the encoder and decoder cannot achieve a good synchronization, which lead to a series of packet loss.

RObust Header Compression (ROHC) is a standardized method to compress the UDP, UDP-Lite, RTP and TCP header of Internet packets. ROHC uses the IPHC concept of context and manages the context identifier (CIDs) reasonably and effectively [13].

Adaptive Compression-based Technique (ACT) for congestion control uses both lossy ADPCM (adaptive pule code modulation) and lossless RLC (run length coding) compression. Discrete wavelet transform is also utilized to categorize data priorities and assign each a different frequency to achieve fairness in wireless sensor networks.

G. Lightweight DeCompression (LDC)

LDC schemes are lossless compression, which is applied for the generally homogeneous consecutive unstructured data. LDC can be used in storage and communication purpose. The repetition distance and sampling frequency are based on the data block size that divided into two types; fixed LDC and variable LDC. The optimal data block size for fixed LDC is 16Kbyte [14]. The whole data block must be scanned first in pre-pass (100% sample size ratio) while generating the dictionary. Detailed submission guidelines can be found on the journal web pages. All authors are responsible for understanding these guidelines before submitting their manuscript.

V. RESULT AND DISCUSSIONS

The following table shows the result of applying the categorization presented earlier on the examples listed in this section.

Table- I: Classification of the Compression Schemes

Classification / Scheme	Homogeneity	Purpose	Accuracy	Structuring of data	Repetition distance	Structure sharing	No. of Passes	Sampling frequency	Sample size ratio
JPEG	Homogeneous + Mixed	Storage & Comm.	Lossy / Lossless	In 2 stages	8X8 ~ 64X64 blocks	Optional (palettes)	3	Blocking	100% of block
RLE	Homogeneous	Storage & Comm.	Lossy & Lossless	None	Practical limit	Optional (palettes)	1	Max. Run-length	100% of block
Huffman	Homogeneous	Storage & Comm.	Lossless	None	Whole stream	Coding tree	2	Whole stream	Small percent
Arithmetic	Homogeneous	Storage & Comm.	Lossless	None	Whole stream	Yes	2	Whole stream	Small percent
LZ	Homogeneous	Comm.	Lossless	None	Practical limit	Mostly No	Mostly 1 (2 or 3)	Practical limit	100% of block
VJHC	Mixed	Comm.	Lossless	Utilized	Practical limit	No	1	N/A	N/A
IPHC	Mixed	Comm.	Lossless	Utilized	Practical limit	No	1	N/A	N/A

CRTP	Mixed	Comm.	Lossy & Lossless	Utilized	Practical limit	No	1	N/A	N/A
ROHC	Mixed	Comm.	Lossless	Utilized	Practical limit	No	1	N/A	N/A
ACT	Mixed	Comm.	Lossy & Lossless	Utilized	Block	No	2	Block	100% of block
LDC	Homogeneous	Storage & Comm.	Lossless	None	Whole data block	Yes	2	Whole data block	100% of block

VI. CONCLUSION

The compression schemes discussed in this paper shows the general tendency of most compression designers to target homogeneous data or structured data. Based on the classification of the compression schemes completed in this paper, most well designed lossy schemes offer much better compression than lossless depending on the acceptable distortion degree. The nature of data especially in mixed non-homogeneous data is more important than the compression scheme used, even if the compression scheme is extremely aggressive. All compression schemes choose only one of the conflicting targets to try to approach as much as possible, compression speed (time), compression degree (ratio), compress resources (temporary memory and energy) or compression complexity. Most storage targeting compression schemes mostly care about the compress degree rather than any other compression performance. While communication targeting compression schemes are usually more concerned with the energy resource efficient utilization within some time restrictions

ACKNOWLEDGMENT

This research is supported by Faculty of Engineering, Universiti Malaysia Sarawak (UNIMAS).

REFERENCES

1. Graham, "Signal coding and processing," 2nd ed. Cambridge University Press, 1994, pp. 34.
2. M. Mahoney, Data compression explained, online book, Dell Inc., 2013.
3. D.A. Lelewer and D.S. Hirschberg, "Data compression," ACM Comput. Surv., vol.19, no.3, pp.261–296, Sept 1987.
4. J.H. Pujar and L.M. Kadlaskar, "A new lossless method of image compression and decompression using huffman coding techniques," J. of Theoretical and Applied Info. Tech., vol.15, no.1, pp.18–23, May 2010.
5. J. Ziv and A. Lempel, "A universal algorithm for sequential data compression," IEEE Trans. on Info. Theory, vol.23, no.3, pp.337–343, May 1977.
6. M.K. Tank, "Implementation of Limpel-Ziv algorithm for loss-less compression using VHDL," in Proc. 1st Int. Conf. on Con-tours of Comput. Tech., Thinkquest, Berlin, 2010 pp.275–283.
7. D. Mathpal, M. Darji, "A research paper on lossless data compression techniques" International Journal for Innovation Research in Science &Technology (IJIRST), vol. 4, no. 1, pp. 190 – 194, 2017
8. M. Senthilkumar, V. Mathivanan, "Analysis of data compression techniques using Huffman coding and arithmetic coding" International Journal of advance research in computer science and software engineering, vol. 6, no. 5, pp 930 – 936, 2016
9. H. N. Dheemanth, "LZW data compression" American Journal of Engineering Research (AJER), vol. 3, no 2, pp 22-26, 2014
10. V. Jacobson, "Compressing TCP/IP header for low-speed serial links," RFC1144, Feb 1990.
11. S. Casner and V. Jacobson, "Compressing IP/UDP/RTP header for low-speed serial links," RFC2508, Feb 1999.
12. M. Degermark, B. Nordgren, and S. Pink, "IP header compression," RFC2507, Feb 1999.
13. L.E. Jonsson, "RObust Header Compression (ROHC) terminology and channel mapping examples," RFC3759, Apr 2004.
14. Kho L. C., Ngu S. S., Tan Y. and Lim A. O., "Applications of data compression technique in congested network," Wireless Commun., Networking, and Application, Lecturer Notes in Electrical Engineering, Springer, vol. 348, pp. 235–249, 2016

AUTHORS PROFILE

Lee Chin Kho, was born in Malaysia in 1980. She received the B. Eng. (Hons) degree in electronics engineering from the Multimedia University (MMU), Cyberjaya, Malaysia, in 2003, the Master degree in electrical engineering from the University of Adelaide, Adelaide, Australia, in 2004, and the Ph.D. degree from the Japan Advanced Institute of Science and Technology (JAIST), Japan in 2015. She has been with the Universiti Malaysia Sarawak, Sarawak, Malaysia since 2005. Her research interests include Internet of Things (IoT), smart home, network congestion control, and network coding.

Sze Song Ngu, was born in Malaysia in 1980. He received the B.S. (Hons) degree in electronics engineering from the Multimedia University, Ayer Keroh, Malaysia, in 2003, the M.S. degree in electrical engineering from the University of Adelaide, Adelaide, Australia, in 2004, and the Ph.D. degree from the University of Glasgow, Glasgow, U.K., in 2013. Since 2005, he has been with the Universiti Malaysia Sarawak, Sarawak, Malaysia. His research interests include renewable energy, electrical generator design, and power electronics.

Annie Joseph, received her B.Eng. degree in Electronic and Electrical Engineering from College University Technology Tun Hussein Onn, Malaysia, in 2005. She received her M.S. degree in 2006 from University Science Malaysia. Then, she join as a lecturer in Universiti Malaysia Sarawak under Department of Electronic, Faculty of Engineering in 2006. She then received Doctor of Engineering in Electrical and Electronic Engineering at Kobe University, Japan and promoted as a senior lecturer in 2014. Her research interests include online learning, concept drift, feature extraction and machine learning.

Dayang Azra binti Awang Mat, is a Senior Lecturer in Faculty of Engineering, University Malaysia Sarawak from 2006 until now. She graduated from Kyushu University Japan, in 2014 with Doctor of Engineering (in RF and millimeter-wave design). She received her MEng. (Computer and Communication) in 2006 at Universiti Kebangsaan Malaysia and degree in BEng. (Hons) Electronic and Telecommunication at Universiti Malaysia Sarawak. She is expert in mobile and wireless communication, antenna and filter design, RF both in microwave and millimeter-wave.

Kuryati Kipli, received her B.Eng. degree in Electronic and Computer Engineering from the Universiti Malaysia Sarawak (UNIMAS), Malaysia, in 2004, the M.Sc. degree in Electronic and Computer Engineering from the University of Birmingham, U.K in 2007 and Ph.D degree Engineering at the School of Engineering, Deakin University, Waurn Ponds, Australia in 2015. From 2004 to 2006, she was a Process Integration Engineer at the 1st Silicon (M'sia) Sdn Bhd, Samajaya Industrial, Sarawak, Malaysia. From 2006 – 2007, she was a Tutor at the Faculty of Engineering, Universiti Malaysia Sarawak. From 2008-2014, she worked in the Department of Electrical and Electronics, Faculty of Engineering, UNIMAS as lecturer. She is now a senior lecturer at the same department. Since 2008, she has authored and coauthored more than 70 publications (including journal publications, conference publications and book chapter). Her current research interests include pattern recognition, biomedical image processing and signal analysis.