[VOLUME 5 I ISSUE 3 I JULY – SEPT 2018] e ISSN 2348 –1269, Print ISSN 2349-5138
http://ijrar.com/ Cosmos Impact Factor 4.236

A STUDY ON VARIOUS DATA COMPRESSION TYPES AND TECHNIQUES

K.Muthuchamy
Assistant Professor, Department of Computer Science, Manonmaniam Sundaranar University College, Sankarankovil, Tirunelveli District – 627 756, Tamil Nadu, India.

Received: June 26, 2018 Accepted: August 10, 2018

ABSTRACT *This article provides the study of various lossy and lossless data compression techniques.Compression is the process of modifying, encoding or converting the bits structure of data in such a way that it consumes less space on storage disk. It enhances reducing the storage size of one or more data instances or elements. This compression is also known as source coding or bit-rate reduction.The data compression is a reduction in the number of bits needed to represent data. The technique of data compression can save storage capacity, speed up file transfer, and decrease costs for storage hardware and network bandwidth. The Compression techniques enables sending a data object or file quickly over a network or the Internet and in optimizing physical storage resources. In this article mainly focus on various, both lossy and lossless data compression techniques.*

Keywords: *Data compression, lossy compression, lossless compression, Bandwidth, Storage*

Introduction

The data compression has wide implementation in computing services and solutions, specifically data communications. The data compression works through several compressing techniques and software solutions that utilize data compression algorithms to reduce the data size. The data may text, audio, video, program, image etc.Data compression technique, also called compaction, the process of reducing the amount of data needed for the storage or transmission of a given piece of information, typically by the use of encoding techniques. Compression predates digital technology, having been used in Morse Code, which assigned the shortest codes to the most common characters, and in telephony, which cuts off high frequencies in voice transmission. Today, data compression is important in storing information digitally on computer disks and in transmitting it over communications networks.

Working principle of compression

The data compression is performed by a program that uses a formula or algorithm to determine how to shrink the size of the data.The algorithm may represent a string of bits -- or 0s and 1s -- with a smaller string of 0s and 1s by using a dictionary for the conversion between them, or the algorithm may insert a reference or pointer to a string of 0s and 1s that the program has already seen.The text compression can remove all unneeded characters, inserting a single repeat character to indicate a string of repeated characters and substituting a smaller bit string for a frequently occurring bit string. The data compression can reduce a text file to 50% or a significantly higher percentage of its original size.For data transmission, compression can be performed on the data content or on the entire transmission unit, including header data. The information is sent or received via the internet, either singly or with others as part of an archive file, may be transmitted in a ZIP, GZIP or other compressed format.

The important of data compression

The data compression can dramatically decrease the amount of storage a file takes up. For instance, in a 2:1 compression ratio, a 20 megabyte (MB) file takes up 10 MB of space. In the result of compression, administrators spend less money and less time on storage.Data compression optimizes backup storage performance and has recently shown up in primary storage data reduction. The compression will be an important method of data reduction as data continues to grow exponentially.Usually any type of file can be compressed, but it's important to follow best practices when choosing which ones to compress. In general, some files may already come compressed, so compressing those files would not have a significant impact.

Data compression types: lossless and lossy compression

Lossless Data Compression

The lossless data compression makes use of data compression algorithms that allows the exact original data to be reconstructed from the compressed data. The lossless concept can be contrasted to lossy data compression, which does not allow the exact original data to be reconstructed from the compressed data.

Lossless data compression is used in many applications. Usually, it is used in the popular ZIP file format and in the Unix tool gzip. This technique is also often used as a component within lossy data compression technologies. The lossless compression is used when it is important that the original and the decompressed data be identical, or when no assumption can be made on whether certain deviation is uncritical. Some examples are executable programs and source code. The few image file formats, notably PNG, use only lossless compression, while others like TIFF and MNG may use either lossless or lossy methods. The GIF uses a lossless compression method, but most GIF implementations are incapable of representing full color, so they quantize the image (often with dithering) to 256 or fewer colors before encoding as GIF. The color quantization is a lossy process, but reconstructing the color image and then re-quantizing it produces no additional loss.
The Lossless compression enables the restoration of a file to its original state, without the loss of a single bit of data, when the file is uncompressed. The lossless compression is the typical approach with executables, as well as text and spreadsheet files, where the loss of words or numbers would change the information.The lossless compression methods may be categorized according to the type of data they are designed to compress. Themain aim of targets for compression algorithms are text, executables, images, and sound. The most of the lossless compression programs use two different kinds of algorithms: one which generates a statistical model for the input data, and another which maps the input data to bit strings using this model in such a way that "probable" data will produce shorter output than "improbable" data.

Lossless compression algorithms

- Thee run-length encoding (also known as RLE)
- The dictionary coders :
    - LZ77 & LZ78
    - LZW
- The prediction by partial matching (also known as PPM)
- The context mixing (also known as CM)
- The entropy encoding :
    - The Huffman coding (simple entropy coding; commonly used as the final stage of compression)
    - A Adaptive Huffman coding
    - arithmetic coding (more advanced)

Run-length encoding
The Run-length encoding (RLE) is a very simple form of data compression in which runs of data are stored as a single data value and count, rather than as the original run. This technique is the most useful on data that contains many such runs: for example, simple graphic images such as icons and line drawings.
Take thisexample; consider a screen containing plain black text on a solid white background. This will be many long runs of white pixels in the blank space, and many short runs of black pixels within the text. For example, take a hypothetical single scan line, with B representing a black pixel and W representingwhite:
WWWWWWWWWWWWBWWWWWWWWWWWWBBBWWWWWWWWWWWWWWWWWWWWWWWWWB
Here we apply a simple run-length code to the above hypothetical scan line, we get the following:
12WB12W3B24WB
Here we interpret this as twelve W's, one B, twelve W's, three B's, etc. The run-length code represents the original 53 characters in only 13. Anyway, the actual format used for the storage of images is generally binary rather than ASCII characters like this, but the principle remains the same. Anyway binary data files can be compressed with this method; file format specifications often dictate repeated bytes in files as padding space. In addition, newer compression methods such as deflation often use LZ77-based algorithms, a generalization of run-length encoding that can take advantage of runs of strings of characters (such as BWWBWWBWWBWW).
The Run-length encoding performs lossless data compression and is well suited to palette-based iconic images. In general, it does not work well at all on continuous-tone images such as photographs, although JPEG uses it quite effectively on the coefficients that remain after transforming and quantizing image blocks.

Dictionary coder
The dictionary coder, also known as a substitution coder, is any of a number of lossless data compression algorithms which operate by searching for matches between the text to be compressed and a set of strings

contained in a data structure (called the 'dictionary') maintained by the encoder. The compressor finds such a match; it substitutes a reference to the string's position in the data structure. A dictionary coders use a 'static dictionary', one whose full set of strings is determined before coding begins and does not change during the coding process. This method is most often used when the message or set of messages to be encoded is fixed and large; for instance, the many software packages that store the contents of the dictionary in the limited storage space of a PDA generally build a static dictionary from a concordance of the text and then use that dictionary to compress the verses.
In general, methods where the dictionary starts in some predetermined state but the contents change during the encoding process, based on the data that has already been encoded. The both LZ77 and LZ78 algorithms work on this principle. The LZ77, a data structure called the "sliding window" is used to hold the last N bytes of data processed; this window serves as the dictionary, effectively storing every substring that has appeared in the past N bytes as dictionary entries. IN this method, instead of a single index identifying a dictionary entry, two values are needed: the length, indicating the length of the matched text, and the offset (also called the distance),this shows that the match is found in the sliding window starting offset bytes before the current text.

PPM compression algorithm
Prediction by partial mapping (PPM) is an adaptive statistical data compression technique based on context modeling and prediction. The name stands for Prediction by Partial Matching. Prediction by partial mapping models uses a set of previous symbols in the uncompressed symbol stream to predict the next symbol in the stream.
Predictions are usually reduced to symbol rankings. The n (number of previous symbols), determines the order of the PPM model which is denoted as PPM(n). The Unbounded variants where the context has no length limitations also exist and are denoted as PPM*. Here, if no prediction can be made based on all n context symbols a prediction is attempted with just n-1 symbols. The above process is repeated until a match is found or no more symbols remain in context. At that point a fixed prediction is made. This technique is the inverse of that followed by DMC compression algorithms (Dynamic Markov Chain) which build up from a zero-order model. The most of the work in optimizing a PPM model is handling inputs that have not already occurred in the input stream the obvious way to handle them is to create a never-seen symbol which triggers the escape sequence. This is called the zero-frequency problem. The one variant assigns the "never-seen" symbol a fixed pseudo-hit count of one. The variant called PPM-D increments the pseudo-hit count of the "never-seen" symbol every time the "never-seen" symbol is used. We also say that, PPM-D estimates the probability of a new symbol as the ratio of the number of unique symbols to the total number of symbols observed.
Context mixing
The context mixing is a type of data compression algorithm in which the next-symbol predictions of two or more statistical models are combined to yield a prediction that is often more accurate than any of the individual predictions. In this example, one simple method (not necessarily the best) is to average the probabilities assigned by each model.
Entropy encoding
The entropy encoding is a coding scheme that assigns codes to symbols so as to match code lengths with the probabilities of the symbols. In general, entropy encoders are used to compress data by replacing symbols represented by equal-length codes with symbols represented by codes where the length of each codeword is proportional to the negative logarithm of the probability. That is, the most common symbols use the shortest codes. The two of the most common entropy encoding techniques are Huffman coding and arithmetic coding.
Huffman coding
This coding is an entropy encoding algorithm used for lossless data compression. This technique refers to the use of a variable length code table for encoding a source symbol, such as a character in a file where the variable-length code has been derived in a particular way based on the estimated probability of occurrence for each possible value of the source symbol. It was developed by David A. Huffman, and published in the 1952 paper "A Method for the Construction of Minimum-Redundancy Codes". Therefore, Huffman coding is optimal for a symbol-by-symbol coding with a known input probability distribution, its optimality can sometimes accidentally be over-stated. The arithmetic coding and LZW coding often have better compression capability. The above two methods can combine an arbitrary number of symbols for more

efficient coding, and generally adapt to the actual input statistics, the latter of which is useful when input probabilities are not precisely known.

Adaptive Huffman coding

The Adaptive Huffman coding is an adaptive coding technique based on Huffman coding, building the code as the symbols are being transmitted, having no initial knowledge of source distribution, that allows one-phase encoding and adaptation to changing conditions in data. The benefit of one-pass procedure is that the source can be encoded realtime, though it becomes more sensitive to transmission errors, since just a single loss ruins the whole code.

Arithmetic coding

The Arithmetic coding is a method for lossless data compression. This is a form of entropy encoding, but where other entropy encoding techniques separate the input message into its component symbols and replace each symbol with a code word, the arithmetic coding encodes the entire message into a single number, a fraction n where (0.0 = n < 1.0).

Lossy Data Compression

Thelossy data compression method is one where compressing data and then decompressing it retrieves data that may well be different from the original, but is "close enough" to be useful in some way. The Lossy data compression is used frequently on the Internet and especially in streaming media and telephony applications. In general, mostlossy data compression formats suffer from generation loss: repeatedly compressing and decompressing the file will cause it to progressively lose quality. This is in contrast with lossless data compression. The Lossy compression techniques attempt to eliminate unnecessary or redundant information, focusing more on saving space over preserving the accuracy of the data. Usually, the loss is either minimal or undetectable by human observations. The Lossy compression techniques are used for pictures and music files that can be trimmed at the edges. Apart from text files and processing files, pictures and music do not require reconstruction to be identical to the original, especially if the data dropped is insignificant or undetectable.

The Lossy compression permanently eliminates bits of data that are redundant, unimportant or imperceptible. The Lossy compression is useful with graphics, audio, video and images, where the removal of some data bits has little or no discernible effect on the representation of the content.Graphics image compression can be lossy or lossless. In general, Graphic image file formats are typically designed to compress information since the files tend to be large. JPEG is an image file format that supports lossy image compression. The image formats such as GIF and PNG use lossless compression.

JPEG

The Programs using complex graphics are showing up in virtually every area of computing applications including games, education, desktop publishing, graphical design, and most recently the World Wide Web. Even though graphics do a great deal to enhance the usability and visual aesthetics of such applications, they consume prodigious amounts of disk storage.

The image compression research began in the late 1970s, most compression concentrated on using conventional lossless techniques. This such types of compression, which included statistical and dictionary methods of compression, did not tend to perform well on photographic, or *continuous tone* images. The main problem with statistical techniques stemmed from the fact that pixels in photographic images tend to be well spread out over their entire range. If the colors in an image are plotted as a histogram based on frequency, the histogram is not as "spiky" as one would like for statistical compression to be effective.

In the year 1980s, extensive research pushed the development of lossy compression algorithms that take advantage of known limitations of the human eye. This algorithms play on the idea that slight modifications and loss of information during the compression/decompression process often do not affect the quality of the image as perceived by the human user. Finally, the JPEG continuous tone image compression specification was standardized, named after the Joint Photographic Experts Group, the standards group consisting of members from both the CCITT and the ISO that wrote the specification. The JPEG specification includes separate lossy and lossless algorithms; however, the lossless algorithm is rarely used due to its poor compression ratios.

MP3

MPEG audio standard is a high-complexity, high-compression, and high audio quality algorithm. The MPEG (Motion Pictures Experts Group), a group that works on standards for both audio and video systems, saw the benefits of the digital representation of audio data. Its advantages include high immunity to noise,

stability, reproducibility, and the efficient implementation of audio processing functions through a computer.
The MPEG is a working group that was formed from the existing JPEG (Joint Photographic Experts Group) standard for bit reduction of still pictures. The MPEG goal is to devise a suitable encoding scheme for transmitting moving pictures and sound over various broadcast links and recording them in standard digi tal storage media. As moving pictures are often accompanied by sound, MPEG also defined a standard for encoding audio information. In the year 1992, a standard for audio and video encoding was devised known as MPEG-1.
MP3 stands for MPEG-1 Layer 3. Layers represent a family of coding algorithms. The Layer 1 has the lowest complexity compared to the other layers. The Layer 2 requires a more complex encoder and decoder and is directed more towards different applications. Thus, Layer 2 is more able than Layer 1 to remove the redundancy in the signal that takes up space. The Layer 3 is again more complex and further reduces redundancy and relevance from the data. . The layers are backwards-compatible, which means that any software or hardware capable of decoding Layer 3 audio should also be able to decode Layers 1 and 2.MPEG-1 Layer 3 is defined as an open standard, so the description is available to anyone interested in implementing the standard. Since no single company owns the standard, public example source code is available and the format is well defined.

Compression versus data deduplication
The Compression is often compared to data deduplication, but the two techniques operate differently. Deduplication is a type of compression that looks for redundant chunks of data across a storageor file system and then replaces each duplicate chunk with a pointer to the original. The Data compression algorithms reduce the size of the bit strings in a data stream that is far smaller in scope and generally remembers no more than the last megabyte or less of data.The File-level deduplication eliminates redundant files and replaces them with stubs pointing to the original file. The Block-level deduplication identifies duplicate data at the subfile level. This system saves unique instances of each block, uses a hash algorithm to process them and generates a unique identifier to store them in an index. The Deduplication typically looks for larger chunks of duplicate data than compression, and systems can deduplicate using a fixed or variable-sized chunk.
The Deduplication is most effective in environments that have a high degree of redundant data, such as virtual desktop infrastructure or storage backup systems. The Data compression tends to be more effective than deduplication in reducing the size of unique information, such as images, audio,videos, databases and executable files. Many storage systems support both compression and deduplication.

Data compression and backup
The Compression is often used for data that's not accessed much, as the process can be intensive and slow down systems. The Administrators, though, can seamlessly integrate compression in their backup systems.The backup is a redundant type of workload, as the process captures the same files frequently. In an organization that performs full backups will often have close to the same data from backup to backup.

Technologies and products that use data compression
The Compression is built into a wide range of technologies, including storage systems, databases, operating systems and software applications used by businesses and enterprise organizations. The compressing data is also common in consumer devices, such as laptops, PCs and mobile phones.Here many systems and devices perform compression transparently, but some give users the option to turn compression on or off. It can be performed more than once on the same file or piece of data, but subsequent compressions result in little to no additional compression and may even higher the size of the file to a slight degree, depending on the data compression algorithms.The WinZip is a popular Windows program that compresses files when it packages them in an archive. The Archive file formats that support compression include ZIP and RAR.

Conclusion
The advantages of compression are a reduction in storage hardware, data transmission time and communication bandwidth -- and the resulting cost savings. The compressed file also requires less time for transfer, and it consumes less network bandwidth than an uncompressed file. The important disadvantage of data compression is the performance impact resulting from the use of CPU and memory resources to compress the data and perform decompression. The Lossless compression techniques, as their name implies, involve no loss of information. Even, If data have been losslessly compressed, the original data can be recovered exactly from the compressed data after a compress/expand cycle. The Lossless compression is

generally used for so-called "discrete" data, such as database records, spreadsheets, word-processing files, and even some kinds of image and video information.Text compression is a significant area for lossless compression. It is very important that the reconstruction is identical to the text original, as very small differences can result in statements with very different meanings. For example, consider the sentences "Do not send money" and "Do now send money." A similar argument holds for computer files and for certain types of data such as bank records. But the lossless techniques Run length and Huffman encoding can be used for image compression with some modified approach, this also give lossy image.In essence, lossless compression algorithms are most needed in cases that require compression where we want the reconstruction to be identical to the original.

References

- Senthil Shanmugasundaram, Robert Lourdusamy, A Comparative Study OfText Compression Algorithm, International Journal of Wisdom BasedComputing, Vol.1 (3)
- JayavrindaVrindavanam ,SaravananChandran, Gautam K. Mahanti, “A Survey of Image Compression Methods” International Journal of Computer Applications® (IJCA) 2012
- Mamta Sharma, “Compression Using Huffman Coding”,International Journal of Computer Science and Network Security, Vol.10, No.5,May 2010
- Alarabeyyat, S. Al-Hashemi1, T. Khdour1, M. Hjouj Btoush1,S.Bani-Ahmad1, R. Al-Hashemi “Lossless Image Compression Technique Using Combination Methods ”Journal of Software Engineering andApplications, 2012.
- Ken Huffman. Profile: David A. Huffman, Scientific American, September1991, pp. 54–58.
- S.R. Kodituwakku. U. S.Amarasinghe Comparison Of Lossless DataCompression Algorithms For Text Data
- http://www.ieeeghn.org/wiki/index.php/Historyof Lossless Data CompressionAlgorithms
- Dipperstain M. 1998, RunLength Encoding (RLE): Discussion and Implementation.
- McNaughton J. 2001, Data Compression Theory and Algorithms. **4**: 263-286.
- https://www.techopedia.com/definition/5447/lossy
- https://searchstorage.techtarget.com/definition/compression
- https://www.britannica.com/technology/data-compression
- https://cs.stanford.edu/people/eroberts/courses/soco/projects/data-compression/lossless/lz78/example.htm
- https://cs.stanford.edu/people/eroberts/courses/soco/projects/data-compression/lossy/index.htm