QALAAI ZANISTSCIENTIFIC JOURNAL
A Scientific Quarterly Refereed Journal Issued by Lebanese French University – Erbil, Kurdistan, Iraq
Vol. (6), No (3), Summer 2021
ISSN 2518-6566 (Online) - ISSN 2518-6558 (Print)

Features of Application of Data Compression Methods

Noura qusay Ebraheem
Department of Information Technology, College Engineering and Computer Science, Lebanese French, University, Erbil, Kurdistan Region, Iraq
noura.qussy@lfu.edu.krd

Mohammed Sardar Ali
Department of Information Technology, College Engineering and Computer Science, Lebanese French, University, Erbil, Kurdistan Region, Iraq
mo.sardar@lfu.edu.krd

ARTICLE INFO

Article History:
Received: 2/5/2021
Accepted: 8/6/2021
Published: Summer 2021

Keywords: *Data Compression, Data Compression Algorithms, ARJ, ZIP, GZ.*

Doi:
10.25212/lfu.qzj.6.3.34

ABSTRACT

The article describes the known methods of data compression, considers the features of compression statistical and linguistic methods, with and without losses, relatively static and dynamic models. The capabilities of archivers are described and discusses the various data compression techniques, including statistical and linguistic methods, with and without losses, as well as relatively static and dynamic models. Archivers' capabilities are listed. Data compression is used everywhere. Without data compression a 3-minute song would be over 100Mb in size, while a 10-minute video would be over 1Gb in size. Data compression shrinks big files into much smaller ones. It does this by getting rid of unnecessary data while retaining the information in the file. Data compression can be expressed as a decrease in the number of bits required to illustrate data. Compressing data can conserve storage capacity, accelerate file transfer, and minimize costs for hardware storage and network capacity.

1. Introduction

When transmitting and storing information, there is always the problem of message size. The development of computer equipment leads to increased transmission speeds and storage volumes. Perhaps to give the impression that using data compression technologies is inefficient as a result of this growth. But it is a view from

QALAAI ZANISTSCIENTIFIC JOURNAL
A Scientific Quarterly Refereed Journal Issued by Lebanese French University – Erbil, Kurdistan, Iraq
Vol. (6), No (3), Summer 2021
ISSN 2518-6566 (Online) - ISSN 2518-6558 (Print)

only one side. On the other hand, there is an increase in the volume of information itself messages.

Moreover, it is not uncommon for transmission volumes to grow faster than speed transmission. In addition, you must pay for the transfer of information reduction of information transfer fees, another positive aspect of using information compression methods. Therefore, the problem of compression information will always remain relevant (Hashim, Hammood, & Al-azraqe, 2016).

The purpose of compression is to reduce the number of bits required to store or transmit a given bit of information. It allows you to send messages faster and store them more economically and quickly .

The last means what operation receiving given information with Device her storage to pass. It is possible if the data decompression speed is higher than the data read speed from the media information. Compression allows, for example, to write more information to a floppy disk, "increase" the size hard drive, speed up the modem, etc. Working with computers is a widely used data archiving programs in ZIP, GZ, ARJ and other formats. Information compression methods have been developed as a mathematical theory, which was little used in computers in practice for a long time (until the first half of the 80's) (Khalifa, 2005; Nelson & Gailly, 1996; Ziv & Lempel, 1977).

The increase in transmission, still images, sound, and video has led to many methods of lossy compression methods or irreversible coding. To transfer these message types, the following methods may be used. But quite often, it is necessary to transfer documents, database fragments, executable files. The impossibility of full recovery of such files will lead to the loss of information. Therefore, the reverse compression methods are still relevant, which guarantees the full recovery of the original message. Such methods include the simplest compression algorithms developed by the classics of coding theory K. Shannon, R. Fano, D. Huffman (Kavitha, 2016; Khalifa, 2005; Nelson & Gailly, 1996; Zhen & Ren, 2009; Ziv & Lempel, 1977)

QALAAI ZANISTSCIENTIFIC JOURNAL
A Scientific Quarterly Refereed Journal Issued by Lebanese French University – Erbil, Kurdistan, Iraq
Vol. (6), No (3), Summer 2021
ISSN 2518-6566 (Online) - ISSN 2518-6558 (Print)

2. Data Compression

Data compression is a procedure of transcoding data to reduce its volume. Compression is based on eliminating the redundancy of information contained in the input data. Data that does not have redundancy properties (such as a random signal or noise) cannot be compressed.

The simplest compression algorithms, also called optimal coding algorithms, are statistical and based on considering the probability distribution of the elements of the incoming message (text, image, file). In practice, the frequencies of occurrence of elements in the incoming message are used to approximate the probabilities. Probability is an abstract mathematical concept related to infinite experimental data samples, and the frequency of occurrence is a quantity that can be calculated for finite data sets. With a sufficiently large number of elements in the set of experimental data, we can say that the frequency of the element is close (with some accuracy) to its probability (Dhawan, 2011; Sharma, 2010; Soltz, 2011).

If the mentioned probabilities are different, then the most probable elements (common ones) can compare shorter code words and, conversely, for unlikely elements to compare longer code words. In this way, you can reduce the average length of the codeword. The optimal encoding algorithm does this so that the average length of the codeword is minimal; like with a shorter encoding length, it becomes irreversible. Such algorithms exist. They are known as Shannon Fano and Huffman. The disadvantage of both methods is that they cannot encode the message more sparingly than one bit per element of the message (letter) (Deshlahra, 2013; Kavitha, 2016; Mulla, Gunjikar, & Naik, 2013; Sai Virali Tummala, 2017; Saroya & Kaur, 2014).

Therefore, the goal was to invent a coding scheme that would encode some elements with less than one bit. In 1977, J. Raissanen proposed one of the best methods for this called "arithmetic coding" and patented this development for IBM. Also known in this group of methods is the group coding algorithm RLE (Run Length Encoding). Assume that the size of the alphabet is N = 256, and we compress a plain text file (ASCII). Most likely, we will not find all N characters of our alphabet in such a file. Then

QALAAI ZANISTSCIENTIFIC JOURNAL
A Scientific Quarterly Refereed Journal Issued by Lebanese French University – Erbil, Kurdistan, Iraq
Vol. (6), No (3), Summer 2021
ISSN 2518-6566 (Online) - ISSN 2518-6558 (Print)

put the length of the code of the missing characters equal to zero. In this case, the list of code lengths will be quite large the number of zeros (code lengths of missing characters) grouped (Din, Mahmuddin, Qasim, & Technology, 2019; QASSIM & SUDHAKAR, 2015; Roshidi Din, 2018). Each such group can be compressed using group coding RLE. This algorithm is extremely simple. Instead of a sequence of M identical elements in a row, the store the first element of this sequence and the number of its repetitions, like (M-1), for example:

RLE algorithm ("AAAABBBCDDDDDDD") = A3 B2 C0 D6.

So far, we have considered a statistical approach to compressing files of unknown format. There is a second fundamental approach, dictionary, where the next character is placed at each step of the compression algorithm as it is (with a special no compression flag), or the boundaries of the word from the previous text that matches the next characters of the file.

Unzipping files compressed in this way is very fast, so these algorithms are used to create self-extracting programs. Vocabulary algorithms are less mathematically sound but more practical.

Among the dictionaries, the LZ77 algorithm was first developed by Israeli mathematicians Jacob Ziv and Abraham Lempel and published in 1977. Many information compression programs use one or another modification of the LZ77. The basic idea of the LZ77 is that the second and subsequent occurrences of a string of characters in the message are replaced by references to its first entry (Kida, Takeda, Shinohara, Miyazaki, & Arikawa, 1998; Knieser, Wolff, Papachristou, Weyer, & McIntyre, 2003; Taleb, Musafa, Khtoom, & Gharaybih, 2010).

The LZ77 uses the already viewed part of the message as a dictionary. To achieve compression, it tries to replace the next part of the message with a pointer to the dictionary's contents. To do this, use the window shifted by the message, divided into two unequal parts. The first, large in size, includes the already viewed part of the message. The second, much smaller, is a buffer that still contains uncoded input

QALAAI ZANISTSCIENTIFIC JOURNAL
A Scientific Quarterly Refereed Journal Issued by Lebanese French University – Erbil, Kurdistan, Iraq
Vol. (6), No (3), Summer 2021
ISSN 2518-6566 (Online) - ISSN 2518-6558 (Print)

stream characters (Awasthi, Sharma, & Husein, 2018). Usually, window size is a few kilobytes and the size of the buffer - no more than one hundred bytes. The algorithm looks for a fragment that matches the contents of the buffer in the dictionary (which takes up the majority of the window. Simplified example: the size of the window is 20 characters, the dictionary is 12 characters, and the buffer is 8.

The message "®_MICROSOFT_PRODUCTS_PRODUCTS_" is encoded. Let the dictionary already full. Then it contains the line "®_SOFTWARE_" and the buffer - the line "PRODUCTS". Looking through the dictionary, the algorithm will find that the matching substring will be "PRO", in the dictionary, it is located with an offset of 2 (count starting from zero) and has a length of 3 characters. The next character in the buffer is "D".

Thus, the source code will be a trinity (2,3, 'D'). After that, the algorithm shifts to the left the entire contents of the window to the length of the matching substring +1 and simultaneously read the same number of characters from the input stream to the buffer (Din, Mahmuddin, et al., 2019; Din, Qasim, & Informatics, 2019; Qasim, Din, Alyousuf, & Informatics, 2020).

In the dictionary, the line "OGRAMNI_PROD" in the buffer - "UKTI_FIRMY". In this situation, the matching substring cannot be detected, and the algorithm will output a code (0.0, 'B'), after which it will shift the window by one character. Then the dictionary will contain "GRAMNI_PRODU" and the buffer - "KTI_FIRM". And so on.

Vocabulary algorithms have been intensively improved. In 1978, the same authors improved their algorithm called LZ78. In 1982 Storer (Storer) and Szymanski (Szimanski), based on LZ77 has developed the LZSS algorithm, which differs from LZ77 by the codes produced, and in 1984 Welch (Welch) the LZW algorithm was created by modifying the LZ78.

3. Lossless compression

Compression is lossless (when it is possible to recover input without distortion) or with loss (recovery is possible with minor distortion). Lossless compression is used to

QALAAI ZANISTSCIENTIFIC JOURNAL
A Scientific Quarterly Refereed Journal Issued by Lebanese French University – Erbil, Kurdistan, Iraq
Vol. (6), No (3), Summer 2021
ISSN 2518-6566 (Online) - ISSN 2518-6558 (Print)

process computer programs and data. Lossless compression is usually used to reduce the amount of audio, photo, and video information. It is much more effective than lossless compression.

Lossless data compression (LDC) is a method of information compression in which encoded information can be recovered to the nearest bit. At the same time, original the data is completely recovered from the compressed state. This type of compression is radically different from lossy data compression. Each type of digital information usually has its algorithms lossless compression (Kavitha, 2016; Ni et al., 2004; Park, 2003).

Lossless data compression is used in many applications. For example, it is used in the popular ZIP file format and Unix utility Gzip. It is too used as a component in lossy compression.

Lossless compression is used when the identity of the compressed data of the original is important. A common example is executable files and source code. Some graphic file formats, such as PNG or GIF, use only lossless compression, while others (TIFF, MNG) can use compression both with losses and without (Syah, et al., 2021). Lossless compression methods can be distributed according to the type of data for which they were created.

The three main types of data for a compression algorithm are text, image, and sound. In principle, any lossless multi-purpose data compression algorithm (multi-purpose means that it can handle any binary data type) can be used for any data type, but most of them ineffective for each major type. Audio data, for example, cannot be a well-compressed text compression algorithm.

4. Comparative analysis of compression methods

4.1 Comparison of Huffman and Shannon-Fano algorithms

Shannon-Fano encoding is a fairly old method of compression, and to date, it is not of particular practical interest. In most cases, the length is compressed by this method sequence is equal to the length of the compressed sequence using Huffman encoding.

QALAAI ZANISTSCIENTIFIC JOURNAL
A Scientific Quarterly Refereed Journal Issued by Lebanese French University – Erbil, Kurdistan, Iraq
Vol. (6), No (3), Summer 2021
ISSN 2518-6566 (Online) - ISSN 2518-6558 (Print)

But there are sequences on which the Shannon-Fano method does not give unambiguous coding, so compression Huffman's method is considered more effective. For example (Table 1), consider the sequence containing the following symbols: a - 14, b - 7, c - 5, d - 5, e - 4 (numbers indicate the number of repetitions). Method Huffman compresses it to 77 bits, and Shannon-Fano - to 79 bits.

Table (1): Example of sequence compression

symbol	Huffman algorithm	Shannon-Fano algorithm
a	0	00
b	111	01
c	101	10
d	110	110
e	100	111

Such differences in the degree of compression occur due to the loose definition of the distribution characters' method in the Shannon-Fano method.

How can divide into groups by these steps:

1. The probability of the first group (p1) and the second (p2) is zero.

2.p1 ≤ p2?

- yes: add a character from the beginning of the table to the first group.

- no: add a symbol from the end of the table to the second group.

3.If all the characters are divided into groups, then complete the algorithm. Otherwise, go to step 2.

When constructing a Shannon-Fano code, the division of a set of elements can be performed in several ways. Choosing a partition at level n may worsen the breakdown options in the next levels (n + 1) and lead to suboptimal code. In other words, optimal handling on each step of the path does not guarantee the optimality of the whole set of actions. Therefore, the Shannon-Fano code is not optimal in the general sense,

QALAAI ZANISTSCIENTIFIC JOURNAL
A Scientific Quarterly Refereed Journal Issued by Lebanese French University – Erbil, Kurdistan, Iraq
Vol. (6), No (3), Summer 2021
ISSN 2518-6566 (Online) - ISSN 2518-6558 (Print)

although it gives optimal results for some probability distributions (A. A. J. Altaay, Sahib, & Zamani, 2012; A. A. J. S. Altaay, Shahrin Bin Zamani, Mazdak, 2012; Amarunnishad & Nazeer, 2014).

We can construct several Shannon-Fano codes for the same probability distribution, and they can all give different results. If you build all the possible Shannon-Fano codes for this probability distribution, they will be all the Huffman codes, like the optimal codes. Differences between static and dynamic models Virtually all existing data compression algorithms can be implemented in two ways static and dynamic. Each of them occupies a certain niche, being used in cases where it most convenient. Huffman's algorithm is no exception - used in many archivers and formats. It is sometimes used in static (*JPEG*, *ARJ*) or dynamic (compact utility for *UNIX*, *ICE*) options.

Huffman's algorithm compresses data due to the probabilities of occurrence of characters in the source. To successfully compress and decompress something, you need to know the same probabilities for each character.

Static algorithms handle this task before you start compressing the file. The program runs through the file itself and calculates which character is found how many times. Then, according to the probabilities of occurrence of symbols, the Huffman binary tree from where is extracted is constructed codes of different lengths corresponding to each character. The paragraph is then repeated in the third stage, with each character being replaced by its code from the tree.. Thus, a static algorithm takes two passes to the source file to encode the data (Alyousuf, Din, & Qasim, 2020).

The dynamic algorithm allows to implementation of a one-pass compression model. Not knowing the real one’s probabilities of occurrence of characters in the source file, the program gradually changes the binary tree with each symbol, increasing the frequency of its appearance in the tree and rebuilding the tree itself in this regard. However, it becomes obvious that winning in the number of passes on the source file, the degree of compression is lost. In the static algorithm, the frequencies of occurrence of the characters were known from the very beginning. The lengths of

QALAAI ZANISTSCIENTIFIC JOURNAL
A Scientific Quarterly Refereed Journal Issued by Lebanese French University – Erbil, Kurdistan, Iraq
Vol. (6), No (3), Summer 2021
ISSN 2518-6566 (Online) - ISSN 2518-6558 (Print)

these codes characters were closer to optimal. At the same time, the dynamic model, studying the source, gradually reaches its real frequency characteristics and forms them only after passing the original file.

Another advantage of the dynamic algorithm can compensate for this disadvantage. Because dynamic binary the tree is constantly modified by new symbols, there is no need to remember their frequencies in advance. When unzipped, the program, having received the character code from the archive, will restore in the same way tree, as it did when compressed and will increase by one the frequency of its occurrence. Moreover, no you need to remember which characters are not found in the binary tree. All the characters that will be added to the tree and are what we will need to restore the original data. In static, the algorithm with this case is a bit more complicated - at the beginning of the compressed file, you need to save information about the characters found in the source file and their frequencies. Before the beginning of unpacking, you need to know which characters will occur and their code.

5. Archivers

Suppose the codes of algorithms of type LZ are transmitted for encoding the Huffman algorithm or arithmetic. In that case, the obtained two-step (pipeline, not two-pass) algorithm will give compression results like well-known programs: GZIP, ARJ, PKZIP. The best compression ratio is given by two-pass algorithms that compress the initial data. Consecutively twice, but they run up to twice as slow as a single pass at low magnification—degree of compression (Miran & Kadir, 2019). Most archiving programs compress each file individually, but some compress files in total flow, which increases the degree of compression and complicates working with the received archive. For example, replacing a file with a newer version in such an archive may require it recoding of the entire archive. An example of a program that can compress files in a shared stream is RAR. Unix archivers (gzip, bzip2) compress files in the general stream almost always. Some of the representatives of this category are WinZip and WinRAR.

QALAAI ZANISTSCIENTIFIC JOURNAL
A Scientific Quarterly Refereed Journal Issued by Lebanese French University – Erbil, Kurdistan, Iraq
Vol. (6), No (3), Summer 2021
ISSN 2518-6566 (Online) - ISSN 2518-6558 (Print)

WinZip is a powerful program for working with both Zip files and archives of other formats. In WinZip, built-in support for CAB files and popular formats like TAR, gzip, BinHex, ARJ, LZH, and ARC. Support is provided through external programs. It has improved the compression technology that creates smaller archive files. Improved file management - now available, use "drill down" to view only those documents in folders that are necessary for work, as well add files directly to existing or new folders. WinZip JobMaster is a new tool in WinZip 11.0 Pro, which allows you to pre-backup specific data, setting up and scheduling compression tasks and FTP upload functionality. There is no need to create archive files first on the hard drive - you can immediately create them on CD or DVD. Large ZIP files can be automatically placed on multiple CDs or DVDs.

WinRAR is a popular RAR archiver and archive manager. In addition to its basic format (rar), it can work with zip, cab, ace, arj and other archives. The degree of file compression with this program is much higher than in WinZip. With WinRAR, you can recover damaged files (repair option). The probability of file recovery will be higher if, creating an archive in the program settings was the put recovery record option is specified. Among other WinRAR settings, you can note the ability to create self-extracting archives with the indication of the unpacking path, and establish'the degree of compression from the best (but slowest) to the fastest (but lower quality). In addition, the composition of the archive can be done invisibly.

Since WinRAR, there is an opportunity to compress files by 8-15% better and faster passes great savings in disk space, data loss and, most importantly, working time. WinRAR is great for creating media files. It is also possible to divide the archives into separate parts for recording them on various media. It is ideal for transmitting confidential data over the Internet and other unsecured channels, with 128-bit cryptographic protection and electronic signature archives.

A filename extension can usually identify the file format containing the data you want to unpack before using the archiving program. In table. 2, some typical extensions, corresponding archiving programs and data compression methods are given.

QALAAI ZANISTSCIENTIFIC JOURNAL
A Scientific Quarterly Refereed Journal Issued by Lebanese French University – Erbil, Kurdistan, Iraq
Vol. (6), No (3), Summer 2021
ISSN 2518-6566 (Online) - ISSN 2518-6558 (Print)

Table (2): Correspondence of standard extensions, archiving programs and data compression methods.

Extension	Programs	Encoding type
.z	compress	LZW
.arc	arc, pkarc	LZW, Huffman
.zip	zip, unzip, pkzip, pkunzip	LZ77, LZW, Huffman, Shannon-Fano
.gz	gzip	LZ77, Huffman
.bz2	bzip2	Huffman, Burroughs-Willer
.arj	arj	LZ77, Huffman
.ice, .lzh	lha, lharc	LZSS, Huffman
.pak	pak	LZW

Virtually all file formats use compression to store graphic information data. The image file format is also typically identified by a file name extension. In the table. 3 shows some typical image file extensions and their corresponding compression methods data.

Table (3): Correspondence of standard extensions of graphic files and methods of data compression

Extension	Encoding type
.gif	LZW
.jpeg,. jpg	LOSSY COMPRESSION, HUFFMAN, OR ARITHMETIC
.bmp, .pcx	RLE
.tiff, .tif	CCITT / 3 FOR FAXES, LZW OR OTHERS

6. Conclusions

The classical methods of data compression of Shannon-Fano and Huffman give way to newer ones: arithmetic from the group of statistical methods and dictionary methods of data compression, which is a rational complement to the statistical. The construction of archivers considers the peculiarities of compression and vocabulary,

and statistical methods, with and without losses, static and dynamic models. As a rule, archivers built on several data compression methods.

References:

Altaay, A. A. J., Sahib, S. B., & Zamani, M. (2012). An introduction to image steganography techniques. Paper presented at the 2012 International Conference on Advanced Computer Science Applications and Technologies (ACSAT).

Altaay, A. A. J. S., Shahrin Bin Zamani, Mazdak. (2012). An introduction to image steganography techniques. Paper presented at the 2012 International Conference on Advanced Computer Science Applications and Technologies (ACSAT).

Alyousuf, F. Q. A., Din, R., & Qasim, A. J. (2020). Analysis review on spatial and transform domain technique in digital steganography. *Bulletin of Electrical Engineering and Informatics*, *9*(2), 573-581.

Amarunnishad, T., & Nazeer, A. (2014). Secured Reversible Data Hiding In Encrypted Images Using Hyper Chaos. International Journal of Image Processing (IJIP), 8(6), 423.

Awasthi, Y. O. G. E. S. H., Sharma, A. S. H. I. S. H., & T. Husein, S. (2018). A Critical Analysis of Internal and External Sorting Algorithms through MATLAB. *Journal of Advanced Research in Dynamical and Control Systems*, *9*, 2789–2799.

Deshlahra, A. (2013). Analysis of Image Compression Methods Based On Transform and Fractal Coding.

Dhawan, S. (2011). A review of image compression and comparison of its algorithms. International Journal of Electronics & Communication Technology, IJECT, 2(1), 22-26.

Din, R., Mahmuddin, M., Qasim, A. J. J. I. J. o. E., & Technology. (2019). Review on steganography methods in multi-media domain. 8(1.7), 288-292.

Din, R., Qasim, A. J. J. B. o. E. E., & Informatics. (2019). Steganography analysis techniques applied to audio and image files. 8(4), 1297–1302.

Hashim, E. W. A., Hammood, M. O., & Al-azraqe, M. T. I. (2016). A Cloud Computing System Based Laborites' Learning Universities: Case Study of Bayan University's Laborites-Erbil. *Book of Proceeding*, 538.

Kavitha, P. (2016). A Survey on Lossless and Lossy Data Compression Methods. International Journal of Computer Science & Engineering Technology, 7(03), 110-114.

Khalifa, O. O. (2005). Wavelet Coding Design for Image Data Compression. Int. Arab J. Inf. Technol., 2(2), 118-127.

Kida, T., Takeda, M., Shinohara, A., Miyazaki, M., & Arikawa, S. (1998). Multiple pattern matching in LZW compressed text. Paper presented at the Proceedings DCC'98 Data Compression Conference (Cat. No. 98TB100225).

Knieser, M. J., Wolff, F. G., Papachristou, C. A., Weyer, D. J., & McIntyre, D. R. (2003). A technique for high ratio LZW compression. Paper presented at the Proceedings of the conference on Design, Automation and Test in Europe-Volume 1.

Miran, A., & Kadir, G. (2019). Enhancing AODV routing protocol to support QoS. *International Journal of Advanced Trends in Computer Science and Engineering*, *8*(5), 1824–1830.

Mulla, A., Gunjikar, N., & Naik, R. (2013). Comparison of Different Image Compression Techniques. International Journal of Computer Applications, 70(28).

Nelson, M., & Gailly, J.-L. (1996). The data compression book: M & t Books New York.

Ni, Z., Shi, Y. Q., Ansari, N., Su, W., Sun, Q., & Lin, X. (2004). Robust lossless image data hiding. Paper presented at the Multimedia and Expo, 2004. ICME'04. 2004 IEEE International Conference on.

Park, S.-G. (2003). ADAPTIVE LOSSLESS VIDEO COMPRESSION.

Qasim, A. J., Din, R., Alyousuf, F. Q. A. J. B. o. E. E., & Informatics. (2020). Review on techniques and file formats of image compression. 9(2), 602–610.

QASSIM, A. J., & SUDHAKAR, Y. (2015). Information Security with Image through Reversible Room by using Advanced Encryption Standard and Least Significant Bit Algorithm.

Roshidi Din, O. G., Alaa Jabbar Qasim. (2018). Analytical Review on Graphical Formats Used in Image Steganographic Compression. Indonesian Journal of Electrical Engineering and Computer Science, Vol 12, No 2, pp. 441~446. doi: 10.11591

Sai Virali Tummala, V. M. (2017). Comparison of Image Compression and Enhancement Techniques for Image Quality in Medical Images.

QALAAI ZANISTSCIENTIFIC JOURNAL
A Scientific Quarterly Refereed Journal Issued by Lebanese French University – Erbil, Kurdistan, Iraq
Vol. (6), No (3), Summer 2021
ISSN 2518-6566 (Online) - ISSN 2518-6558 (Print)

Saroya, N., & Kaur, P. (2014). Analysis of image compression algorithm using DCT and DWT transforms. International Journal of Advanced Research in Computer Science and Software Engineering, 4(2).

Sharma, M. (2010). Compression using Huffman coding. IJCSNS International Journal of Computer Science and Network Security, 10(5), 133-141.

Soltz, M. A. (2011). Method and system to determine an optimal tissue compression time to implant a surgical element. In: Google Patents.

Syah, R., Davarpanah, A., Nasution, M. K., Wali, Q., Ramdan, D., Albaqami, M. D., ... & Noori, S. M. (2021). The Effect of Structural Phase Transitions on Electronic and Optical Properties of CsPbI3 Pure Inorganic Perovskites. *Coatings*, *11*(10), 1173.

Taleb, S. A., Musafa, H. M., Khtoom, A., & Gharaybih, K. (2010). Improving LZW image compression. European Journal of Scientific Research, 44(3), 502-509.

Zhen, C., & Ren, B. (2009). Design and realization of data compression in real-time database. Paper presented at the 2009 International Conference on Computational Intelligence and Software Engineering.

Ziv, J., & Lempel, A. (1977). A universal algorithm for sequential data compression. IEEE Transactions on information theory, 23(3), 337-343.

تایبەتمەندیەکانی جێبەجێ کردنی ڕێگاکانی پەستانی داتا (زانیاری)

پوختە:

ئەم ووتارە باسی ڕیگاکانی پەستانی داتاکان دەکات کە زانراوو ناسراون ، وە باسی لایەنە باشەکانی ڕێگاکانی و ئاماریی و زمانەوانی دەکات ، بە لەدەستدان وە بەبێ لەدەستدانی داتاکان ، باسی تواناکانی ئەمبارکردنی(ئەرشیف کارەکان) دەکات و گفتوگۆ دەکات دەربارەی جۆرەها تەکینیکەکانی (ڕیگاکانی) پەستانی داتاکان، لەگەڵ ڕێگاو شێوازەکانی ئاماریی و زمانەوانیدا، بە لەدەستدان وە بەبێ لەدەستدان داتاکان، سەرەوایی باسی کردنی نموونەکان چەسپاو و کاریگەر بە شێوەیەکی ڕێژەیی ، وە باسی تواناکانی ئەمبارکارەکان(ئەرشیف کارەکان) دەکات . پەستانی داتاکان بەکاردێت لە گشت

QALAAI ZANISTSCIENTIFIC JOURNAL
A Scientific Quarterly Refereed Journal Issued by Lebanese French University – Erbil, Kurdistan, Iraq
Vol. (6), No (3), Summer 2021
ISSN 2518-6566 (Online) - ISSN 2518-6558 (Print)

شوێنێك ، و ژمارێكی زۆری فایلەكانی هەمەجۆری داتای پەستێندراو بەكاردهێنێت ، بەبێ پەستاندنی داتاكان. كە قەبارە گۆرانیەك ماوەكەی 3 خولەكە لەسەرەوەی 100 میگابایتە زیاترە لە قەبارەدا ، لە كاتێكدا كە قەبارەی ڤیدیۆیەك ماوەكەی 10 خولەكی لە سەرەوەی 1میگابایتە، پەستانی داتاكان دەبێتە مایەی بجووك كردنەوەی فایلە گەورەكان و گۆرینی بۆ فایلێكی گەلێك بچووك تر ، ئەوەش ئەنجام دەدات لە رێگەی رزگار بوون لەو داتایانەی كە پێویست نین ، لەگەل پاراستن و مانەوەی ئەو داتایانەی كە لە نێو فایلەكادا هەیە ، دەتوانرێت پەستانی داتاكات بەوە دەربڕین: كە ئەو رێگایەیە بۆ كەم كردنەوەی ژمارەی بایت كە پێویست نین بۆ رووونكردنەوەی داتاكان. پەستانی داتاكان دەتوانێت ببێتە مایەی پاراستنی فراوانی توانای ئامرازی ئەمبارکردن(تخزین) ، و خێراكردنی گواستنەوەی فلیەكانە ، وە كەم كردنەوەی تێچوونی ئەمبارکردنی ئامێرەكان وە توانای تۆرەك.

ووشە سەرەكیەكان:پەستانی داتاكان، و خوارزمەیەكانی پەستانی داتاكان (ARJ، ZIP، GZ)

ميزات تطبيق أساليب ضغط البيانات

الملخص

تتناول المقالة الطرق المعروفة لضغط البيانات ، وتدرس ميزات الطرق الإحصائية واللغوية لها ، مع/و بدون خسائر، و نماذج ثابتة و فعالة نسبيًا. وصف قدرات المحفوظات (الارشيف) وتناقش تقنيات ضغط البيانات المتنوعة، بما في ذلك الأساليب الإحصائية واللغوية ، مع/و بدون خسائر ، وكذلك النماذج الثابتة والفعالة نسبيًا. وتم سرد قدرات المحفوظات. وان ضغط البيانات تستخدم في كل مكان. و تستخدم العديد من أنواع الملفات المختلفة ليانات مضغوطة. بدون ضغط البيانات ، ستكون حجم الأغنية التي تبلغ مدتها 3 دقائق أكثر من 100 ميجابايت حجما ، في حين أن حجم الفيديو الذي تبلغ مدته 10 دقائق يزيد عن 1 جيجابايت. حجما يؤدي ضغط البيانات إلى تقليص الملفات الكبيرة إلى ملفات أصغرمنها بكثير. يقوم بذلك عن طريق التخلص من البيانات غير الضرورية مع الاحتفاظ بالمعلومات الموجودة في الملف. يمكن التعبير عن ضغط البيانات على أنه تقليص عدد البتات المطلوبة لتوضيح البيانات. أن ضغط البيانات تستطيع الحفاظ على سعة التخزين ، وتسريع نقل الملفات ، وتقليل تكاليف تخزين الأجهزة وسعة الشبكة.

الكلمات الرئيسية: ضغط البيانات ، خوارزميات ضغط البيانات ، ARJ، ZIP، GZ