Evaluation of Different Lempel-Ziv Compression Algorithms

Yair Wiseman
Computer Science Department
Bar-Ilan University
Ramat-Gan 52900
Israel
wiseman@cs.biu.ac.il

ABSTRACT

Lempel-Ziv compression algorithms have several variants. We would like to focus on two prominent versions of Lempel-Ziv algorithms – LZSS and LZW. LZSS was suggested before LZW and usually provides better results; however, there are cases where LZW can outperform LZSS. We aim at finding the best cases of LZSS and the best cases of LZW in order to realize when each algorithm is recommended to be selected.

Keywords: Data Compression Efficiency, Lempel-Ziv Compression

1. INTRODUCTION

Compression techniques are often partitioned into statistical and dictionary techniques (Bell, Cleary, & Witten, 1990). Huffman codes (Huffman, 1952) or Arithmetic codes (Howard & Vitter, 1992) are usually given as an example for statistical techniques, while Lempel-Ziv methods are usually given as an example for dictionary techniques. Usually Lempel-Ziv methods are adaptive (Bell, Witten, & Cleary, 1989); however, a static version of Lempel-Ziv can also be found (Schuegraf & Heaps, 1974).

This paper will concentrate on the adaptive Lempel-Ziv methods, which are also partitioned into two fundamental groups. The first group is based on reference to recurring data in the compressed file. This approach was first introduced as LZ77 (Ziv & Lempel, 1977), and versions of this approach are still commonly used by many commercial software products such as gzip (Free Software Foundation, 1991). The second group creates a dictionary of common phrases in the data. This approach was first introduced as LZ78 (Ziv &Lempel, 1978). Versions of this approach are also very common in use as a separate method such as *"compress"* of UNIX and as a part of more complicated techniques such as GIF (Willard, Lempel, Ziv, & Cohn, 1984).

One of the most popular versions of LZ77 is LZSS (Storer & Szymanski, 1982), while one of the most popular versions of LZ78 is LZW (Welch, 1984). The aim of this paper is to compare the compression efficiency of LZSS and LZW.

2. THE RELATIVE EFFICIENCY OF LZW AND LZSS

In LZSS, the encoded file consists of a sequence of items, each of which is either a single character or a pointer of the form (*offset*, *length*), which replaces a string of *length* characters that appeared as *offset* characters earlier in the file. Decoding of such a file is thus a very simple procedure. For the encoding, however, there is a need to locate the longest recurring strings, for which sophisticated data structures such as hash tables or binary trees have been suggested.

In LZW, the encoded file consists of a sequence of pointers to a dictionary. Each pointer replaces a string of the input file that appeared earlier and has been put into the dictionary. Encoder and decoder must therefore construct identical copies of the dictionary. In LZW, the dictionary is dynamically constructed in a manner that can be precisely redone by the decoder. LZW finds the longest string in the dictionary. If the string $x_1,...,x_n$ is in the dictionary, but the string $x_1,...,x_{n+1}$ is not in the dictionary, LZW will add the string $x_1,...,x_{n+1}$ to the dictionary and will use the string $x_1,...,x_n$ for the current pointing.

In point of fact, both of the methods are based on *pointers*. In LZSS the pointers are the *offset* component, while in LZW, the pointers are the dictionary's pointers. Figure 1 shows how the modification of the pointer size will affect the compression efficiency in both of the methods. The test has been conducted using the King James Bible.

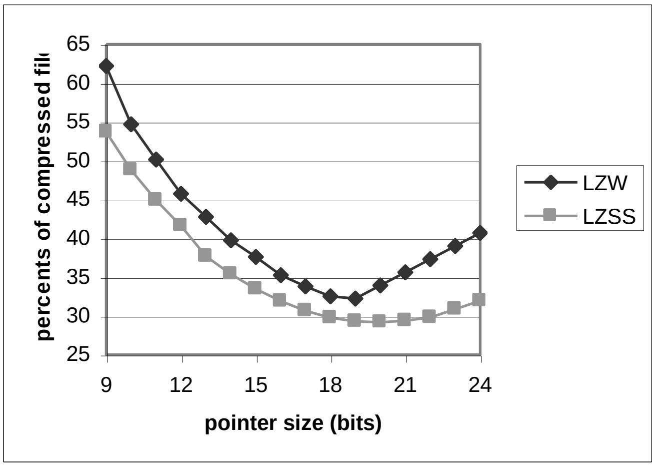

Figure 1

When the pointer size is too small, the possibilities to point are very limited, while a bigger pointer size takes too much space and can harm the compression efficiency.

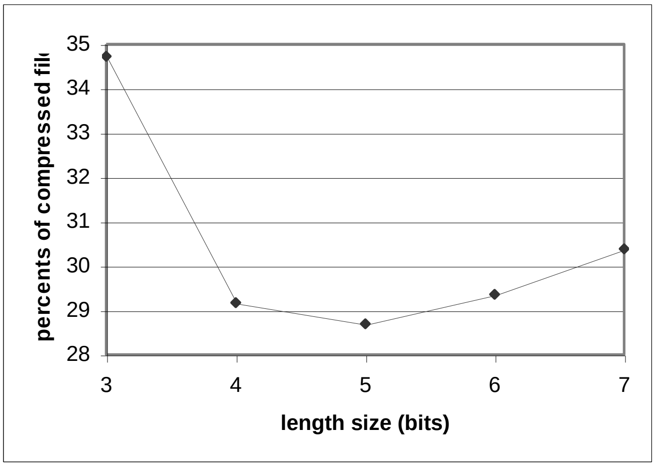

Figure 2

In LZSS there is also another parameter - the size of the *length* component. Figure 2 shows the effect of the *length* component on the LZSS compression efficiency. When the size of the *length* component is too small, only a few characters can be pointed, while a longer *length* component can be a waste because the recurring strings are usually not so long.

In Figure 3 both of the compression methods have been configured to their best parameters according to Figures 1 and 2, i.e. LZSS with 20 bits pointer plus 5 bits for the *length* component and LZW with 19 bits for the pointer. The tested files are from the Calgary Corpus (Bell, Witten, & Cleary, 1988).

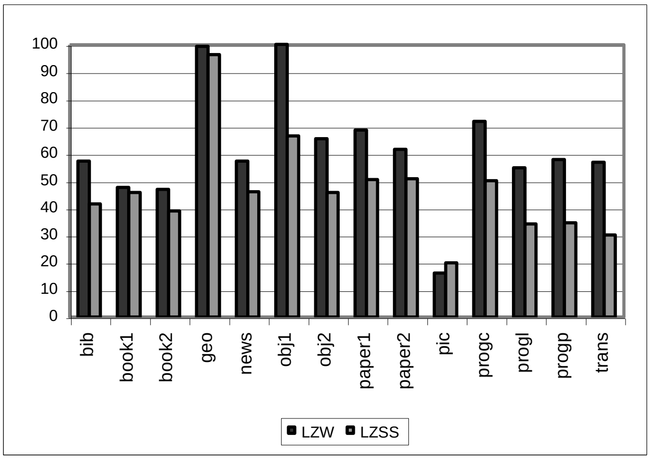

Figure 3

It can be clearly seen in both Figure 1 and Figure 3 that LZSS usually gives better results. However, when compressing the *pic* file, LZW gives better results. The reason for this exception is the uncommon content of this file, which contains a lot of nulls. When the *length* component of LZSS is just 5 bits, LZSS can put a pointer to no more than 32 bits. LZW, however, can theoretically have a pointer to an infinite string. When using LZW, each entry has an old string and a new letter; hence it can save much longer strings.

3. HOW LZW CAN OUTPERFORM LZSS

We would like to determine how good LZW can be. As shown above, LZW compresses the *pic* file better; however, LZSS can easily do better. If we significantly enlarge the *length* component, so a longer string can be pointed, LZSS can compress much better than LZW, e.g. X(1,1000) can be recursively interpreted as a long string of 1001 Xes. Therefore, our aim is to find a case that the adjustment of the *length* component size cannot help out.

Let us look at LZW with a pointer of 9 bits and build a file that contains the 256 possible characters of ASCII. This will put the pairs (0,1) (1,2) .. (254,255) into the dictionary of LZW, according to the algorithm of LZW (Horspool, 1991) (Tao & Mukherjee, 2004). If we put a 0 after the last 255, the pair (255,0) will be added, and these numbers will use up the entire dictionary. The dictionary has 512 entries. The first 256 entries are of the single characters, while the other 256 entries are of the pairs. LZW can handle these pairs better because LZSS saves one bit for character/pointer flag, several bits for the pointer, and several bits for *length* component. Usually LZSS does not bother to replace a pair of characters by a pointer because of the high price, but even if LZSS replaces the pair, the gain will be small, while LZW can save more bits when pointing just to a pair of characters.

In order to obtain the maximal difference between LZW and LZSS, we put these pairs in the file in different orders. However, we are not able to put only two pairs in adjacent locations in two strings. The second appearance of the adjacent pairs will make LZSS see the two pairs as a quadruplet, hence using just one pointer for the quadruplet. In such a case we say that the second pair is an immediate subsequent. Therefore, we must not allow it to appear again after its first occurrence.

Claim 1

Let *m* and *n* be integers. Let $\{x_1,...,x_n\}$ be a set of sequences of size *m*. There are just three different ways to arrange $\{x_1,...,x_n\}$ if any x_i is not allowed to be an immediate subsequent sequence to x_j in three of the different strings of sequences for every pair of $1 \le i \le n$ and $1 \le j \le n$.

Proof

By induction on the number of sequences in the set.

For n=4, there are 24 possibilities to arrange {x_1,x_2,x_3,x_4} However, when checking all of the possibilities, it can be easily found that there are only three strings of sequences that can fulfill the stipulation that x_i is not allowed to be an immediate subsequent sequence to x_j for every pair of $1 \leq i \leq n$ and $1 \leq j \leq n$. There are some pairs of strings that can be chosen, but the number of the strings will always be not more than three, e.g. {x_1,x_2,x_3,x_4}, {x_3,x_1,x_4,x_2} and {x_3,x_2,x_4,x_1}, so the proof is by checking all of these cases one by one.

Remark: The claim is also true for n=1,n=2,n=3. However, in order to make the proof simpler in the next stage, we used n=4.

Assume the claim is true for n-1, i.e. the set {x_1,...,x_{n-1}} has just three ways to be arranged according to the above stipulation. Let us add the sequence x_n. The sequence x_n can be put in each of the three possible strings just once:

- At the end or at the beginning.
- After one sequence from the end or from the beginning.
- After two or more sequences from the end or from the beginning.

In all of the above cases, at least two sequences at the end or at the beginning will remain adjacent; hence there is just one possibility of adding the new sequence, and the outcome is that {x_1,...,x_n} can be arranged in only three ways. Q.E.D.

Accordingly, a file can be built containing the single characters and three strings of pairs as described above. Such a file will be compressed by LZW with a 9 bits pointer, always in a better manner than by LZSS. The exact ratio is dependent on the pointer size of LZSS. Indeed, a 9 bits pointer size is uncommon. Therefore, a more attractive question is how the file can be enlarged to fit a larger LZW dictionary with an n bits pointer.

If LZW reads pairs, it will add triplets to the dictionary. Because there are three sequences of pairs with each sequence being 256 pairs, LZW will add 768 entries of triplets to the dictionary. As a result, the dictionary will have 1280 entries, which fits an 11 bits dictionary.

Claim 1 asserts there are only three strings of triplets that can be added, i.e. two triplets cannot be adjacent in the same order in more than one string. Moreover, because the claim is true for all the sequences of any size, the file can be enlarged by triplets, quadruplets etc., such that it will have $256 \cdot 3^{m-2}$ items of each sequence in length of m. let us name such a file SPT (stands for Single, Pair, Triplet, etc.). In these cases, even when LZSS uses a pointer, LZW will also be able to use a pointer too. Because LZW pointers are cheaper, the compression efficiency of LZW will be better.

Claim 2

The ratio between the compressed files constructed by LZW and LZSS is the highest in SPT files .i.e., there is no case where LZW can gain a better margin over LZSS.

Proof

Let us look at the number of pointers in both LZSS and LZW. In a file generated in SPT form, the number of the pointers will be equal when the file is compressed by either LZSS or LZW. Let us assume there is a better case. In this better case the number of the pointers of LZW can be less than, greater than, or equal to the number of the pointers of LZSS. Let us analyze each of these cases:

LZW creates fewer pointers: LZW only has pointers by definition, and any string pointed to by LZW can be pointed to by LZSS as well. Therefore, there is no benefit.

LZW creates the same number of pointers as LZSS: This case will yield the same ratio as the file in the above form, so no benefit is gained.

LZW creates more pointers: This case will yield a poorer result for LZW because LZW always puts a pointer, while LZSS uses pointers only in the appropriate cases. If LZSS creates fewer pointers, it will indicate that LZSS has chosen not to put a pointer because it is less expensive. In contrast, LZW puts a pointer because this is its usual behavior, and that pointer is more expensive.

Therefore, LZW outperforms LZSS best in this case. Q.E.D.

4. CONCLUSION

Usually LZSS can compress data in a better manner than LZW. If we have a long sequence of the same character, LZSS can compress it in a constant few bytes assuming the *length* component is long enough to grip the number of the characters. LZW, however, has to construct the pointers step by step, and it will have pointers to two bytes, three bytes etc. Indeed, if the file length is *m*, we can find the number of the pointers – *n*, by solving the equation $m=n(n+1)/2$, i.e. $O(\sqrt{n})$ pointers. On the other hand, the pointer of LZSS is more expensive; hence in other cases when the number of the pointers is almost equal, LZW will compress better. This paper shows the extreme case where the number of the pointers is exactly equal; hence the efficiency of LZW is significantly better.

5. REFERENCES

[1] Bell, T.C., Cleary, J.G., &. Witten, I.H. (1990)”Text Compression,” Englewood Cliffs, NJ: Prentice-Hall.

[2] Huffman, D.A. (1952) “A Method for the Construction of Minimum-Redundancy Codes,” Proc. IRE, pp. 1098-1101.

[3] Howard, P.G., Vitter, & J. S. (1992) “Analysis of Arithmetic Coding for Data Compression,” invited paper in special issue on data compression for images and texts in Information Processing and Management, 28(6), 749-763.

[4] Bell, T.C, Witten, I.H, & Cleary, J.G. (1989) “Modeling for Text Compression,” ACM Computing Surveys, 33(4), 557-591.

[5] Schuegraf, E.J., & Heaps, H. S. (1974) “A Comparison of Algorithms for Data Base Compression by Use of Fragments as Language Elements,” Information Storage and Retrieval 10, 309-319.

[6] Ziv, J., & Lempel, A. (1977) “A Universal Algorithm for Sequential Data Compression,” IEEE Trans. on Inform. Theory, IT-23(3), 337-349.

[7] gzip, Free Software Foundation, Inc. (1991) 675 Mass Ave, Cambridge, MA, USA.

[8] Ziv, J., & Lempel, A. (1978) “Compression of individual sequences via variable-rate coding,” IEEE Trans. Inform. Theory, 24(5), 530-536.

[9] Willard, L., Lempel, A., Ziv, J. & Cohn, M. (1984) “Apparatus and method for compressing data signals and restoring the compressed data signals,” US patent - US4464650.
[10] Storer, J.A., & Szymanski, T.G. (1982) “Data Compression via Textual Substitution,” Journal of ACM, 29(4), 928-951.
[11] Welch, T.A. (1984) “A technique for high performance data compression”, IEEE Computer, 17(6), 8-19.
[12] Bell, T.C., Witten, I.H., & Cleary, J.G. (1988) Modeling for Text Compression, Technical Report 1988-327-39, The University of Calgary, Calgary, Alberta Canada.
[13] Horspool, R.N. (1991) "Improving LZW," Proc. Data Compression Conference (DCC 91), Snowbird, Utah, IEEE Computer Society Press, Los Alamitos, CA, pp. 332-341.
[14] Wiseman Y., (2020), "Autonomous Vehicles", Encyclopedia of Information Science and Technology, Fifth Edition, Vol. 1, Chapter 1, pp. 1-11.
[15] Wiseman, Y., (2018), "In an era of autonomous vehicles, rails are obsolete", International Journal of Control and Automation, Vol. 11, No. 2, pp. 151-160.
[16] Wiseman, Y., (2021), "Intelligent Transportation Systems along with the COVID-19 Pandemic will Significantly Change the Transportation Market", The Open Transportation Journal, Vol. 15, No. 1, pp. 11-15.
[17] Wiseman Y., (2018), "Vehicle identification by OCR, RFID and Bluetooth for toll roads", International Journal of Control and Automation, Vol. 11, No. 9, pp. 67-76.
[18] Wiseman Y., (2020), "Conjoint Vehicle License Plate Identification System", The Open Transportation Journal, Vol. 14, No. 1, pp. 164-173.
[19] Wiseman, Y., (2021), "COVID-19 Along with Autonomous Vehicles will Put an End to Rail Systems in Isolated Territories", In IEEE Intelligent Transportation Systems, Vol. 13, No. 3, pp. 6-12, doi: 10.1109/MITS.2021.3049409.
[20] Wiseman, Y. (2019), "Driverless cars will make union stations obsolete", The Open Transportation Journal, Vol. 13, No. 1, pp. 109b 115.
[21] Wiseman Y., (2017, May). "Real-time monitoring of traffic congestions" In proceedings of 2017 IEEE International Conference on Electro Information Technology (EIT-2017, Lincoln, Nebraska, USA, pp. 501-505.
[22] Wiseman, Y., (2017), "Tool for online observing of traffic congestions", International Journal of Control and Automation, Vol. 10, No. 6, pp. 27-34.
[23] Wiseman Y., (2017), "Computerized traffic congestion detection system". International Journal of Transportation and Logistics Management, Vol. 1, No.1, pp. 1-8.
[24] Wiseman Y., (2018)., "Efficient Embedded Computing Component for Anti-Lock Braking System", International Journal of Control and Automation, Vol. 11, No. 12, pp. 1-10.
[25] Wiseman Y., (2018), "Ancillary ultrasonic rangefinder for autonomous vehicles", International Journal of Security and its Applications", Vol. 12, No. 5, pp. 49-58.
[26] Wiseman, Y. (2019), "Driverless cars will make passenger rail obsolete [opinion
[27] Wiseman Y., (2017), "Self-Driving Car - A Computer will Park for You", International Journal of Engineering & Technology for Automobile Security, Vol. 1, No. 1, pp. 9-16.
[28] Wiseman Y., (2017), "Remote Parking for Autonomous Vehicles", International Journal of Hybrid Information Technology, Vol. 10, No. 1, pp. 313-324.
[29] Wiseman Y., (2014), "Device for Detection of Fuselage Defective Parts", Information Journal, Tokyo, Japan, Vol. 17(9(A)), pp. 4189-4194.
[30] Wiseman Y., (2013), "Fuselage Damage Locator System", Advanced Science and Technology Letters, Vol. 37, pp. 1-4.
[31] Wiseman Y., (2010), "Take a Picture of Your Tire!", Proc. IEEE Conference on Vehicular Electronics and Safety (IEEE ICVES-2010) Qingdao, ShanDong, China, pp. 151-156.
[32] Wiseman Y., (2013), "The Effectiveness of JPEG Images Produced By a Standard Digital Camera to Detect Damaged Tyres", World Review of Intermodal Transportation Research, Vol. 4, No. 1, pp. 23-36.
[33] Wiseman Y., (2013), "Camera That Takes Pictures of Aircraft and Ground Vehicle Tires Can Save Lives", Journal of Electronic Imaging, Vol. 22, No. 4, 041104.
[34] Wiseman Y., (2017), "Safety Mechanism for SkyTran Tracks", International Journal of Control and Automation, Vol. 10, No. 7, pp. 51-60.
[35] Wiseman Y., (2017), "Automatic Persistent Inspection of SkyTran Track System", http://u.cs.biu.ac.il/~wiseman/skytran1.pdf .
[36] Grinberg I. and Wiseman Y., (2007), "Scalable Parallel Collision Detection Simulation", In Proceedings of Signal and Image Processing, Honolulu, Hawaii, pp. 380-385.

[37] Grinberg I. and Wiseman Y., (2013), "Scalable Parallel Simulator for Vehicular Collision Detection", International Journal of Vehicle Systems Modelling and Testing, Inderscience Publication, Vol. 8, No. 2, pp. 119-144.
[38] Wiseman Y., K. Schwan and P. Widener, (2004), "Efficient End to End Data Exchange Using Configurable Compression", Proceedings of The 24th IEEE Conference on Distributed Computing Systems (ICDCS 2004), Tokyo, Japan, pp. 228-235.
[39] P. Weisberg and Wiseman Y., (2009), "Using 4KB Page Size for Virtual Memory is Obsolete", Proc. IEEE Conference on Information Reuse and Integration (IEEE IRI-2009), Las Vegas, Nevada, pp. 262-265.
[40] P. Weisberg and Wiseman Y., (2015), "Virtual Memory Systems Should Use Larger Pages rather than the Traditional 4KB Pages", International Journal of Hybrid Information Technology, Vol. 8(8), pp. 57-68.
[41] Wiseman Y., (2017), "Automatic Alert System for Worn Out Pipes in Autonomous Vehicles", International Journal of Advanced Science and Technology, Vol. 107, pp. 73-84.
[42] Wiseman Y. and Grinberg I., (2016), "When an Inescapable Accident of Autonomous Vehicles is Looming", International Journal of Control and Automation, Vol. 9 No. 6, pp. 297-308.
[43] Wiseman Y. and Grinberg I., (2016), "Autonomous Vehicles Should Not Collide Carelessly", Advanced Science and Technology Letters, Vol. 133, pp. 223-228.
[44] Wiseman Y. and Grinberg I., (2016), "Circumspectly Crash of Autonomous Vehicles", Proceedings of IEEE International Conference on Electro Information Technology (EIT 2016), Grand Forks, North Dakota, USA, pp. 382-386.
[45] Y. Wiseman, "Diminution of JPEG Error Effects", The Seventh International Conference on Future Generation Information Technology, Vol. 117, pp. 6-9, (2015).
[46] Y. Wiseman, "Alleviation of JPEG Inaccuracy Appearance", International Journal of Multimedia and Ubiquitous Engineering, Vol. 11(3), pp. 133-142, (2016).
[47] Y. Wiseman, "Enhancement of JPEG compression for GPS images", International Journal of Multimedia and Ubiquitous Engineering, Vol. 10, No. 7, pp. 255-264, (2015).
[48] Y. Wiseman, "Improved JPEG Based GPS Picture Compression", Advanced Science and Technology Letters, (2015).
[49] Y. Wiseman, "The still image lossy compression standard - JPEG", Encyclopedia of Information Science and Technology, Third Edition, Vol. 1, Chapter 28, (2014).
[50] Y. Wiseman, "A Pipeline Chip for Quasi Arithmetic Coding", IEICE Journal - Trans. Fundamentals, Tokyo, Japan, Vol. E84-A No.4, pp. 1034-1041, (2001).
[51] Y. Wiseman, "Burrows-Wheeler Based JPEG", Data Science Journal, Vol. 6, pp. 19-27, (2007).
[52] Y. Wiseman, "Efficient Embedded Images in Portable Document Format (PDF)", International Journal of Advanced Science and Technology, Vol. 124, pp. 129-138, (2019).
[53] Y. Wiseman and E. Fredj, "Contour Extraction of Compressed JPEG Images", ACM - Journal of Graphic Tools, Vol. 6, No. 3, pp. 37-43, (2001).
[54] E. Fredj and Y. Wiseman, "An O(n) Algorithm for Edge Detection in Photos Compressed by JPEG Format", Proc. International Conference on Signal and Image Processing SIP-2001, Honolulu, Hawaii, pp. 304-308, (2001).
[55] Y. Wiseman, "Adjustable and Automatic Flush Toilet", International Journal of Control and Automation, Vol. 13, No. 4, pp. 1-10, (2020).
[56] D. Livshits and Y. Wiseman, "Cache Based Dynamic Memory Management for GPS", Proceedings of IEEE Conference on Industrial Electronics (IEEE ICIT-2011), Auburn, Alabama, pp. 441-446, (2011).
[57] D. Livshits and Y. Wiseman, "The Next Generation GPS Memory Management", International Journal of Vehicle Information and Communication Systems, Vol. 3(1), pp. 58-70, (2013).
[58] R. B. Yehezkael, Y. Wiseman, H. G. Mendelbaum & I. L. Gordin, "Experiments in Separating Computational Algorithm from Program Distribution and Communication", LNCS of Springer Verlag Vol. 1947, pp. 268-278, 2001.
[59] Y. Wiseman, "ARC Based SuperPaging", Operating Systems Review, Vol. 39(2), pp. 74-78, 2005.
[60] Y. Wiseman, "Advanced Non-Distributed Operating Systems Course", ACM - Computer Science Education, Vol. 37(2), pp. 65-69, 2005.
[61] M. Reuven & Y. Wiseman, "Reducing the Thrashing Effect Using Bin Packing", Proc. IASTED Modeling, Simulation, and Optimization Conference, MSO-2005, Oranjestad, Aruba, pp. 5-10, 2005.
[62] M. Reuven & Y. Wiseman, "Medium-Term Scheduler as a Solution for the Thrashing Effect", The Computer Journal, Oxford University Press, Swindon, UK, Vol. 49(3), pp. 297-309, 2006.
[63] Y. Wiseman, "The Relative Efficiency of LZW and LZSS", Data Science Journal, Vol. 6, pp. 1-6, 2007.

[64] Y. Wiseman & I. Gefner, "Conjugation Based Compression for Hebrew Texts", ACM Transactions on Asian Language Information Processing, Vol .6(1), article no. 4, 2007.
[65] I. Grinberg & Y. Wiseman, "Scalable Parallel Collision Detection Simulation", Proc. Signal and Image Processing (SIP-2007), Honolulu, Hawaii, pp. 380-385, 2007.
[66] Y. Wiseman, "ASOSI: Asymmetric Operating System Infrastructure", Proc. 21st Conference on Parallel and Distributed Computing and Communication Systems, (PDCCS 2008), New Orleans, Louisiana, pp. 193-198, 2008.
[67] Y. Wiseman, J. Isaacson & E. Lubovsky, "Eliminating the Threat of Kernel Stack Overflows", Proc. IEEE Conference on Information Reuse and Integration (IEEE IRI-2008), Las Vegas, Nevada, pp. 116-121, 2008.
[68] M. Itshak & Y. Wiseman, "AMSQM: Adaptive Multiple SuperPage Queue Management", Proc. IEEE Conference on Information Reuse and Integration (IEEE IRI-2008), Las Vegas, Nevada, pp. 52-57, 2008.
[69] R. Ben Yehuda & Y. Wiseman, "The Offline Scheduler for Embedded Transportation Systems", Proc. IEEE Conference on Industrial Electronics (IEEE ICIT-2011), Auburn, Alabama, pp. 449-454, 2011.
[70] Y. Wiseman & P. Weisberg, "Economical Memory Management for Avionics Systems", IEEE/AIAA 31st Digital Avionics Systems Conference (DASC), 2013.
[71] Y. Wiseman & Alon Barkai, "Diminishing Flight Data Recorder Size", IEEE/AIAA 31st Digital Avionics Systems Conference (DASC), 2013.
[72] R. Ben Yehuda & Y. Wiseman, "The Offline Scheduler for Embedded Vehicular Systems", International Journal of Vehicle Information and Communication Systems, Vol. 3(1), pp. 44-57, 2013.
[73] Y. Wiseman & Alon Barkai, "Smaller Flight Data Recorders", Journal of Aviation Technology and Engineering, Vol. 2(2), pp. 45-55, 2013.
[74] P. Weisberg & Y. Wiseman, "Efficient Memory Control for Avionics and Embedded Systems", International Journal of Embedded Systems, Vol. 5(4), pp. 225-238, 2013.
[75] Y. Wiseman, "Steganography Based Seaport Security Communication System", Advanced Science and Technology Letters, Vol. 46, pp. 302-306, 2014.
[76] P. Weisberg, Y. Wiseman & J. Isaacson, "Enhancing Transportation System Networks Reliability by Securer Operating System", Open Journal of Information Security and Applications, Vol. 1(1), pp. 24-33, 2014.
[77] Y. Wiseman, "Noise Abatement at Ben-Gurion International Airport", Advanced Science and Technology Letters, Vol. 67, pp. 84-87, 2014.
[78] Y. Wiseman, "Protecting Seaport Communication System by Steganography Based Procedures", International Journal of Security and Its Applications, Sandy Bay, Tasmania, Australia, Vol. 8(4), pp. 25-36, 2014.
[79] Y. Wiseman, "Noise Abatement Solutions for Ben-Gurion International Airport", International Journal of U- & E-Service, Science & Technology, Vol. 7(6), pp. 265-272, 2014.
[80] P. Weisberg & Y. Wiseman, "Virtual Memory Systems Should use Larger Pages", Advanced Science and Technology Letters, Vol. 106, pp. 1-4, 2015.
[81] Y. Wiseman & Y. Giat, "Red Sea and Mediterranean Sea Land Bridge via Eilat", World Review of Intermodal Transportation Research, Vol. 5(4), pp. 353-368, 2015.
[82] Y. Wiseman, "Can Flight Data Recorder Memory Be Stored on the Cloud?", Journal of Aviation Technology and Engineering, Vol. 6(1), 16-24, 2016.
[83] Y. Wiseman & Y. Giat, "Multi-modal passenger security in Israel", Multimodal Security in Passenger and Freight Transportation: Frameworks and Policy Applications, Edward Elgar Publishing Limited, Chapter 16, pp. 246-260, 2016.
[84] Y. Wiseman, "Traffic Light with Inductive Detector Loops and Diverse Time Periods", Contemporary Research Trend of IT Convergence Technology, Vol. 4, pp. 166-170, 2016.
[85] Y. Wiseman, "Unlimited and Protected Memory for Flight Data Recorders", Aircraft Engineering and Aerospace Technology, Vol. 88(6), pp. 866-872, 2016.
[86] Y. Wiseman, "Conceptual Design of Intelligent Traffic Light Controller", International Journal of Control and Automation, Vol. 9(7), pp. 251-262, 2016.
[87] Y. Wiseman, "Compression Scheme for RFID Equipment", Proc. IEEE International Conference on Electro Information Technology (EIT 2016), Grand Forks, North Dakota, USA, pp. 382-386, 2016.
[88] Y. Wiseman, "Efficient RFID Devices", Proc. The 42nd Annual Conference of IEEE Industrial Electronics Society (IECON 2016), Firenze (Florence), Italy, pp. 4762-4766, 2016.
[89] Y. Wiseman and I. Grinberg, "The Trolley Problem Version of Autonomous Vehicles", The Open Transportation Journal, Vol. 12, pp. 105-113, 2018.

[90] Y. Wiseman, "Compaction of RFID Devices using Data Compression", IEEE Journal of Radio Frequency Identification, Vol. 1(3), pp. 202-207, 2018.
[91] Y. Wiseman, "High Occupancy Vehicle Lanes are an Expected Failure", International Journal of Control and Automation, Vol. 12(11), pp. 21-32, 2019.
[92] Y. Wiseman, "Israel Complementary International Airport", International Journal of Control and Automation, Vol. 12(7), pp. 1-10, 2019.
[93] Y. Wiseman, "Adjusted JPEG Quantization Tables in Support of GPS Maps", Journal of Mobile Multimedia, Vol. 17(4), pp. 637-656, 2021.
[94] Y. Wiseman, "Blaumilch Canal on Ayalon Highway", Daaton, 2015, Available online at: http://www.daaton.co.il/Article.aspx?id=3290
[95] Y. Wiseman, "Revisiting the Anti-Lock Braking System", Technical Report, 2021.
[96] Y. Wiseman, "Isolated Territories and Infrastructure development: A case for land transportation investment in Madagascar", Interdisciplinary Approaches to the Future of Africa and Policy Development, IGI Global Publishing, Chapter 5, pp. 78-97, 2022.
[97] Y. Wiseman, J. Isaacson, "Safer Operating System for Vehicle Telematics", technical report, 2010.
[98] Y. Wiseman, J. Isaacson, E. Lubovsky and P. Weisberg, "Kernel Stack Overflows Elimination", Advanced Operating Systems and Kernel Applications: Techniques and Technologies, pp. 1-14, IGI Global, 2010.
[99] Y. Wiseman "Airport in Dothan Valley is Ideal", Technical Report, 2020.
[100] M. Itshak and Y. Wiseman, "Enhancing the Efficiency of Memory Management in a Super-Paging Environment by AMSQM", Advanced Operating Systems and Kernel Applications: Techniques and Technologies, pp. 276-293, IGI Global, 2010.
[101] Y. Wiseman, "Conjoint Reliable Vehicle License Plate Identification System", Technical Report, 2020.
[102] M. Reuven and Y. Wiseman, "Alleviating the Thrashing by Adding Medium-Term Scheduler", Advanced Operating Systems and Kernel Applications: Techniques and Technologies, pp. 118-136, IGI Global, 2010.
[103] Y. Wiseman, "Controlling Dynamic Traffic by Road Expansion Can be Accomplished by Double Decker Roads - Case Study", Technical Report, 2022.
[104] Y. Wiseman, "Rail in Islands is an Expected Failure", Technical Report, 2020.
[105] Y. Wiseman, "Road Planners should not Look Just Right and Left But Rather Should Also Look Up", Technical Report, 2022.
[106] Y. Wiseman "Can a Flight Data Recorder be Situated in a Cloud?", Technical Report, 2016.
[107] Y. Wiseman, "JPEG Quantization Tables for GPS Maps", Automatic Control and Computer Sciences, Vol. 55(6), 2021.
[108] Y. Wiseman, "Intelligent Transportation Systems along with the COVID-19 Guidelines will Significantly Change the Transportation Market", Techical Report, 2021.
[109] Y. Wiseman, "Cracked Pipes Alert System for Autonomous Vehicles", Technical Report, 2017.
[110] Y. Wiseman, "EPC Compression", Technical Report, 2016.
[111] Y. Wiseman, "Warning System for Cracked Pipes in Autonomous Vehicles", Advances of Machine Learning in Clean Energy and Transportation Industry, Chapter 9, pp. 261-276, 2021.
[112] Y. Wiseman, "Madagascar had better invest in a single land transportation infrastructure", Interdisciplinary Approaches to the Future of Africa and Policy Development, IGI Global Publishing, 2021.
[113] Y. Wiseman, "Teaching Research of Operating Systems", Technical Report, 2022.
[114] Y. Wiseman, "Improving the Super-Paging Swapping Algorithm", Technical Report, 2022.
[115] M. Dreyfuss and Y. Giat, "Optimal spares allocation to an exchangeable-item repair system with tolerable wait", European Journal of Operational Research 261 (2), pp. 584-594, 2017.
[116] Y. Giat, "The effects of output growth on preventive investment policy", American Journal of Operations Research 3 (06), pp. 474-486, 2013.