ISSN: 2278 – 1323
International Journal of Advanced Research in Computer Engineering & Technology (IJARCET)
Volume 1, Issue 8, October 2012

Data mining approach to evaluate the data lossless Data Compression algorithms

Nishad P.M.
Ph.D Scholar,
Department of Computer Science NGM College,
Pollachi, Coimbatore India

Dr. R.Manicka chezian
Associate professor,
Department of computer science NGM College
Pollachi, Coimbatore, India

Abstract: This paper presents a study of various lossless compression algorithms; to test the performance and the ability of compression of each algorithm based on ten different parameters. For evaluation the compression ratios of each algorithm on different parameters are processed. To classify the algorithms based on the compression ratio, rule base is constructed to mine with frequent bit pattern to analyze the variations in various compression algorithms. Also, enhanced K- Medoid clustering is used to cluster the various data compression algorithms based on various parameters. The cluster falls dissentingly high to low after the enhancement. The framed rule base consists of 1,296 rules, which is used to evaluate the compression algorithm. Hundred and eighty four Compression algorithms are used for this study. The experimental result shows only few algorithm satisfies the range "**High**" for more number of parameters.

Keywords: *Lossless compression, parameters, compression ratio, rule mining, frequent bit pattern, K–Medoid, clustering*

I Introduction

Data compression is a method of encoding rules that allows substantial reduction in the total number of bits to store or transmit a file. Two basic classes of data compression are applied in different areas currently that are lossy and lossless compression [l, 2]. This paper evaluates the performance of all possible lossless compression algorithms using popular data mining technique. Rule mining is used with four ranges for ten parameters and to evaluate the performance the frequent bit pattern and clustered using the modified Sorted K- Medoid [3, 4] algorithm is used. Meteorological data mining with finding the hidden pattern is proposed in this paper [5][6][7]. Lossless compression is evaluated using frequent bit pattern and Enhanced K Medoid algorithm in [10]. The main objective of this paper is to evaluate the performance of various lossless data compression algorithms based on various four. To test the compression ability and survey hundred and eighty four algorithms are used. The parameters considered are Compression Ratio, Compression time, Decompression time and Code efficiency. Initially 1,296 are constructed for evaluating the compression ability and performance of algorithms based on the range values and with sorted K-Medoid algorithm the rules are constructed based on the individual parameters. The entire algorithms average ratio is collected based-on all four parameter, and the minimum and maximum of each parameter is collected to construct the new range for the individual parameter. The minimum represent the lowest for the parameter and the maximum shows the peak of algorithms on various parameters, for example 55.37is the minimum and the 77.919998 is the maximum for the parameter compression ratio shown in table-1

Table-1 Maximum and minimum compression ratio of parameters

Parameter	Minimum	Maximum
Compression Ratio	55.37	77.919998
Compression time	2.7	946.999998
Decompression Time	3.6	951
Code Efficiently	289	78621

To evaluate the performance and generate the rule six ranges are used that is VERY LOW, LOW, BELOW AVERAGE, AVERAGE, MIDDLE and HIGH. The individual range value is calculated for each parameter by uniformly splitting the minimum and maximum compression ratio that is shown in the table -2. The LOW range for the compression ratio is 55.37to 59.128333. After generating the range values the rules are generated. The rules are generated by matching the dataset of each algorithm with the range value if the value falls in any of the range. Then that range is set to construct

the rule. If same rule is processed of many algorithms then the rule is treated as unique and only the algorithm is updated. For example in table-3 S no 7 the same rule is generated for the STUFFIT-14 and WINZIP -14 so the rule is uniquely created and the algorithm column alone is updated the process is repeated up to the end of the dataset. So for two hundred and eleven algorithms 46 rules are triaged shown in table-3 and table-4. From the rules the performance of algorithms can be easily evaluated.

Table -2: Range values for the Parameters.

Range	Compression Ratio	Compression time	Decompression Time	Code Efficiently
VERY LOW	55.37	2.7	3.6	289
	59.128333	160.083333	161.5	13344.33333
LOW	59.128334	160.083334	161.6	13344.33334
	62.886666	317.466666	319.4	26399.66667
BELOW AVERAGE	62.886667	317.466667	319.5	26399.66668
	66.644999	474.849999	477.3	39455
AVERAGE	66.645	474.85	477.4	39455.1
	70.403332	632.233333	635.2	52510.33333
MIDDLE	70.403333	632.233334	635.3	52510.33334
	74.161665	789.616665	793.1	65565.66667
HIGH	74.161666	789.616666	793.2	65565.66668
	77.919998	946.999998	951	78621

The HIGH range represents the good compression ratio, Compression time and Compression time but the VERY LOW range represents only the LOW range represents good code efficiency. VERY LOW RANGE OF the compression ratio, Compression time and Compression time represents the poor performance and the HIGH range of Code Efficiently is also poor. So the algorithms compression ratio falls HIGH for any parameter then the performance of the algorithm is good except for the code efficiency for those parameters. For example QPress 0.38b , QuickLZ 1.40b9 , then LZTurbo 0.95 and Zhuff 0.2 is quite good for the parameters Compression Ratio, Compression time, Decompression Time and it may not give the best result for the parameters Code Efficiently. To simplify the evaluation of algorithm another methodology is used in this paper called frequent bit pattern in this stage the generated rule dataset is processed for finding the frequent bit paten. Initially set the desired values for each parameter. For example Compression Ratio is "HIGH" Compression time is "HIGH" Decompression Time is "HIGH" and Code Efficiently is "VERY LOW" is set to the parameters.

So the frequent bit pattern evaluates what are the algorithms which satisfies the best of good compression ratio for individual or group of parameters. For example only based on the first parameter Compression Ratio two rules are triggered that is.

a. If Compression Ratio is “HIGH‟ And Compression time is „HIGH‟ And Decompression Time is “LOW‟ And Code Efficiently is „ BELOW AVERAGE then LZTurbo 0.95 + Zhuff 0.2

b. If Compression Ratio is “HIGH‟ And Compression time is „HIGH‟ And Decompression Time is “LOW‟ And Code Efficiently is „ MIDDLE then QPress 0.38b + QuickLZ 1.40b9

No algorithm satisfy High Compression ratio for all parameters so the rule triggered is zero. Here, using this evaluation methodology the effectiveness of various algorithms based on single or multiple parameters can be easily evaluated. For four parameters 16 combinations of frequent bit pattern can be evaluated. The table -5 shows the rulers trigged for 1024 combinations of frequent bit pattern. For example in the table 1 row 0 and column 0 gives 46 rules because the bit pattern is performed unconditionally in row 0 column 1 gives two that is based on the first parameter Compression Ratio is “HIGH‟. And in the last row and last column gives Zero because that is evaluated using the all parameters, so easily clarify that no algorithm terns best compression ratio for all parameters.

Table -3 Rules for two hundred and eleven algorithms based on the four ranges

R No	Compression Ratio	Compression time	Decompression Time	Code Efficiently
1	HIGH	VERY LOW	VERY LOW	VERY LOW
2	MIDDLE	VERY LOW	VERY LOW	VERY LOW
3	HIGH	LOW	VERY LOW	VERY LOW
4	AVERAGE	VERY LOW	VERY LOW	VERY LOW
5	HIGH	BELOW AVERAGE	BELOW AVERAGE	VERY LOW
6	MIDDLE	LOW	VERY LOW	VERY LOW
7	BELOW AVERAGE	VERY LOW	VERY LOW	VERY LOW
8	HIGH	LOW	LOW	VERY LOW
9	HIGH	MIDDLE	MIDDLE	VERY LOW
10	MIDDLE	BELOW AVERAGE	VERY LOW	VERY LOW
11	LOW	VERY LOW	VERY LOW	VERY LOW
12	MIDDLE	LOW	LOW	VERY LOW
13	HIGH	AVERAGE	AVERAGE	VERY LOW
14	MIDDLE	MIDDLE	VERY LOW	VERY LOW
15	AVERAGE	LOW	VERY LOW	VERY LOW
16	AVERAGE	BELOW AVERAGE	VERY LOW	VERY LOW
17	BELOW AVERAGE	VERY LOW	VERY LOW	LOW
18	AVERAGE	VERY LOW	VERY LOW	LOW
19	MIDDLE	LOW	LOW	LOW
20	LOW	VERY LOW	VERY LOW	LOW
21	MIDDLE	HIGH	HIGH	LOW
22	MIDDLE	BELOW AVERAGE	BELOW AVERAGE	LOW
23	AVERAGE	BELOW AVERAGE	VERY LOW	LOW
24	VERY LOW	VERY LOW	VERY LOW	LOW
25	MIDDLE	AVERAGE	VERY LOW	LOW
26	AVERAGE	LOW	VERY LOW	LOW
27	AVERAGE	BELOW AVERAGE	BELOW AVERAGE	BELOW AVERAGE
28	BELOW AVERAGE	VERY LOW	VERY LOW	BELOW AVERAGE
29	AVERAGE	BELOW AVERAGE	VERY LOW	BELOW AVERAGE
30	AVERAGE	LOW	LOW	BELOW AVERAGE
31	MIDDLE	MIDDLE	MIDDLE	BELOW AVERAGE
32	LOW	VERY LOW	VERY LOW	BELOW AVERAGE
33	VERY LOW	VERY LOW	VERY LOW	AVERAGE
34	BELOW AVERAGE	VERY LOW	VERY LOW	AVERAGE
35	LOW	VERY LOW	VERY LOW	AVERAGE
36	AVERAGE	BELOW AVERAGE	VERY LOW	AVERAGE
37	BELOW AVERAGE	BELOW AVERAGE	VERY LOW	AVERAGE
38	AVERAGE	MIDDLE	VERY LOW	MIDDLE
39	BELOW AVERAGE	BELOW AVERAGE	VERY LOW	MIDDLE
40	AVERAGE	AVERAGE	AVERAGE	MIDDLE
41	BELOW AVERAGE	VERY LOW	VERY LOW	MIDDLE
42	AVERAGE	MIDDLE	MIDDLE	MIDDLE
43	LOW	VERY LOW	VERY LOW	MIDDLE
44	AVERAGE	MIDDLE	VERY LOW	HIGH
45	BELOW AVERAGE	BELOW AVERAGE	VERY LOW	HIGH
46	LOW	VERY LOW	VERY LOW	HIGH

Table: 4 Algorithms or Compressors for the Corresponding Rules

R No	Algorithms Or Compressors
1	CCM 1.30c + CCMx 1.30c + DURILCA 0.5 + DURILCA light 0.5 + FreeARC 0.666 + NanoZip 0.09a + WinRK 3.1.2
2	7-Zip 9.25a + ACE 2.6 + COMPRESSIA 1.0b + CSC 3.2a6 + CTXf 0.75 + DGCA 1.10 + FlashZIP 0.99b8 + FreeARC 0.666 + LZPX(J) 1.2h + NanoZip 0.09a + SBC 0.970 rev3 + SQUEEZ 5.63 + StuffIt 14.0 + WINZIP 14 + WinACE 2.69 + WinRAR 4.1b3 + WinRK 3.1.2
3	NanoZip 0.09a
4	7-Zip 9.25a + BA 1.01 + BALZ 1.15 + BCM 0.12 + BIX 1.00 b7 + BMA 1.35b + BSC 3.0.0 + BSSC 0.95a + BioArc 1.9 + Blizzard 0.24b + CABARC 1.00.0106 + DARK 0.51 + DST 0.91b + DURILCA light 0.5 + FlashZIP 0.99b8 + GRZipII 0.2.4 + JAR 1.02 + LZTurbo 0.95 + M99 2.2.1 + MNZIP + PPMVC 1.2 + PPMd var J rev.1 + RZIP 2.1 + SQUEEZ 5.63 + Tornado 0.4a + UHARC 0.6b + WINIMP 1.21 + WinHKI 1.74 + WinTurtle 1.60 + YBS 0.03f
5	NanoZip 0.09a
6	PIM 2.90 + RZM 0.07h + SQUEEZ 5.63 + UHARC 0.6b
7	AIN 2.32 + BCArchive 1.08.7 + BZIP2 1.0.5 + BZP 0.3 + DZip 2.90 + EAZEL 1.0 + ESP 1.92 + File2Pack 2.0 + GZIP 1.2.4 + LHARK 0.4d + LZTurbo 0.95 + PKZIP 2.50 + QUAD 1.12 + RINGS 1.6 + SLUG X + SZIP 1.12 + THOR 0.96 + Tornado 0.4a + UC II 3.05 + VuZip 1.8 + WINZIP 14 + WinHKI 1.74 + WinXP (Built-in Zip)
8	UHARC 0.6b
9	DURILCA 0.5
10	Ultra7z Opt 0.05
11	ALZip 7.0b1 + Etincelle RC2 + THOR 0.96
12	BIT 0.7 + CMM4 0.2b
13	LPAQ8 + PPMonstr J
14	Quark 0.95r
15	CABARC 1.00.0106
16	UFA 0.04b1
17	ARJ 2.85 + ARJ32 3.15 + CODEC 3.21 + Comprox 0.3.0 + DZip 2.90 + EAZEL 1.0 + GZIP 1.2.4 + LHARK 0.4d + M99 2.2.1 + QazaR 0.0 pre5 + SEMONE 0.6 b2 + TURTLE 0.07 + UC II 3.05 + VuZip 1.8 + YBS 0.03f
18	BruteCM 0.1d + EPC 1.0 + M1 0.3b-1 + PACKET 0.91a + UHBC 1.0 + WinHKI 1.74
19	DST 0.91b
20	Crush 0.01 + DeepFreezer 1.06 + SR3a + WINZIP 14
21	EPM r9
22	Bee 0.7.9 bld 0316 + TC 5.2 dev2
23	LZPM 0.16
24	LZTurbo 0.95 + Zhuff 0.2
25	UFA 0.04b1
26	QC 0.050 + UHBC 1.0
27	PPMN 1.00b1 km
28	ASD 0.2.0 + HOOK 1.4 + MAR + PPMX 0.07
29	DACT 0.8.42 + QLFC 6.6w
30	PPMN 1.00b1 km
31	Bee 0.7.9 bld 0316
32	Chaos Comp 3.0
33	QPress 0.38b + QuickLZ 1.40b9
34	LZ2A
35	ULZ 0.02 + YZX 0.04
36	ABC 2.4 + QazaR 0.0 pre5
37	CHILE 0.3d
38	QLFC 6.6w
39	SZIP 1.12
40	BOA 0.58b
41	LZXQ 0.4
42	BOA 0.58b
43	HYPER 2.5
44	ZZIP 0.36c
45	HA 0.999b
46	SAR 1.0

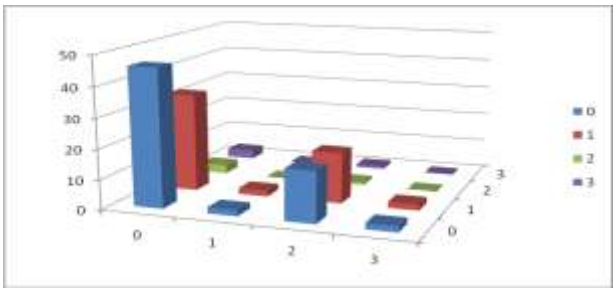
The modified K-Medoid Algorithm is used to evaluate and cluster the algorithms based on the parameters. After finding the Medoid values that are sorted dissentingly. So this leads to the clarity that is the first cluster indicates that compression algorithm gives the best output. Generated rule is fetched and assigned value 1,2,3,4,5,6 to the ranges HIGH, MIDDLE, AVERAGE, BELOW AVERAGE,LOW and VERY LOW respectively so the numeric equivalent is passed to the range and the range data set is processed to the K-Medoid algorithm for each parameter three clusters are generated. That is shown in the table-6 the clusters generated in the ratio (Percentage) is shown in table -7 and graph-1.

Table -5 Frequent bit pattern number of rules triggered for the parameters Compression Ratio is "HIGH" And Compression time is "HIGH" And Decompression Time is "LOW" And Code Efficiently is " BELOW AVERAGE"

	0	1	2	3
0	46	2	17	2
1	33	2	17	2
2	3	0	1	0
3	3	0	1	0

II. Result and discussions

The clustered percentage ratio indicates that only the 29.8913037825% algorithm gives best result for Compression Ratio, 4.347826% algorithm gives best result for Compression time, 36.95652% algorithm gives best result for Decompression Time 71.73913% algorithm gives best result for Code Efficiently 6.52173913%. Three clusters is created for each parameter so totally thirty cluster is created for ten parameters is shown in table -5Graph -2 to graph -5 represents the compression ratio of the compression algorithms on various parameters which satisfies the Range "HIGH". Only few compression algorithms satisfy the „HIGH" for more number of parameters. No algorithm falls in HIGH range for all parameters shown in table-4. Graph -6 to graph -9 shows the overall performance of all algorithms based on different parameters and also the number of cluster percentage. Graph -10 to graph-15 shows the range cluster p percentage of all algorithms. The graph-16 shows the overall percentage of cluster



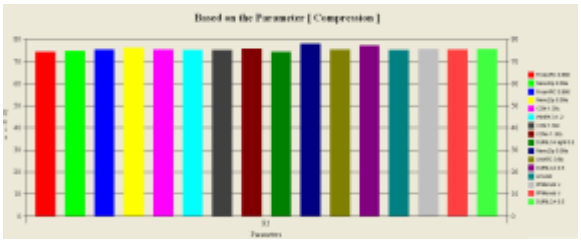
Graph-1 Frequent bit pattern number of rules triggered for the parameters Compression Ratio is "HIGH" and Compression time is "HIGH" And Decompression Time is "LOW" And Code Efficiently is "BELOW AVERAGE"

Table -6 Clustered rules based on K-Medoid Algorithm based on Parameters.

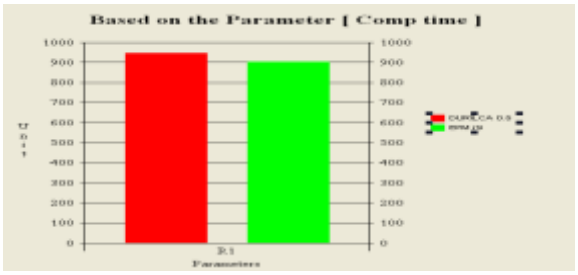
S No	Compression Ratio	Compression time	Decompression Time	Code Efficiently
1	0	5	5	5
2	1	5	5	5
3	0	4	5	5
4	2	5	5	5
5	0	3	3	5
6	1	4	5	5
7	3	5	5	5
8	0	4	4	5
9	0	1	1	5
10	1	3	5	5
11	4	5	5	5
12	1	4	4	5
13	0	2	2	5
14	1	1	5	5
15	2	4	5	5
16	2	3	5	5
17	3	5	5	4
18	2	5	5	4
19	1	4	4	4
20	4	5	5	4
21	1	0	0	4
22	1	3	3	4
23	2	3	5	4
24	5	5	5	4
25	1	2	5	4
26	2	4	5	4
27	2	3	3	3
28	3	5	5	3
29	2	3	5	3
30	2	4	4	3
31	1	1	1	3
32	4	5	5	3
33	5	5	5	2
34	3	5	5	2
35	4	5	5	2
36	2	3	5	2
37	3	3	5	2
38	2	1	5	1
39	3	3	5	1
40	2	2	2	1
41	3	5	5	1
42	2	1	1	1
43	4	5	5	1
44	2	1	5	0
45	3	3	5	0
46	4	5	5	0

Table -7 cluster percentage

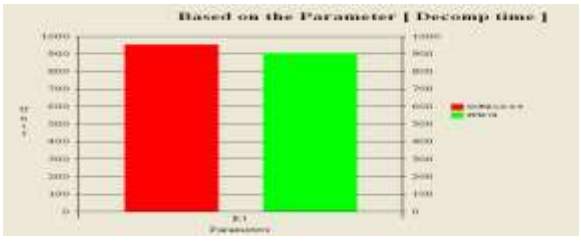
	VERY LOW	LOW	BELOW AVERAGE	AVERAGE	MIDDLE	HIGH
Compression Ratio	13.04347826	21.73913043	30.43478261	17.39130435	13.04347826	4.347826
Compression time	2.173913043	13.04347826	6.52173913	23.91304348	17.39130435	36.95652
Decompression Time	2.173913043	6.52173913	4.347826087	6.52173913	8.695652174	71.73913
Code Efficiently	6.52173913	13.04347826	10.86956522	13.04347826	21.73913043	34.78261



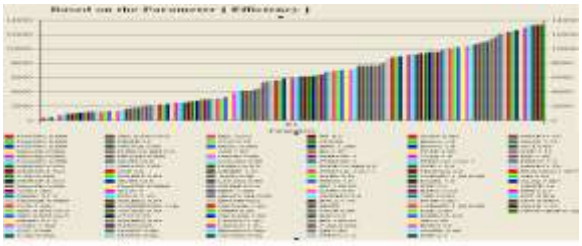
Graph-2 Based On Compression Ratio



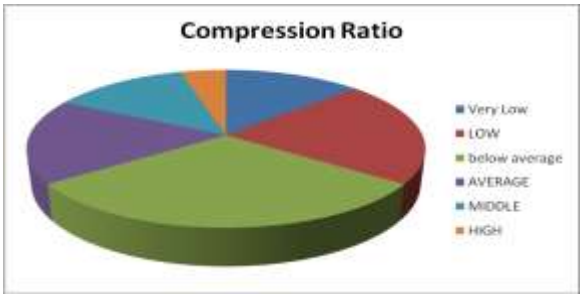
Graph-3 Based on Compression Time



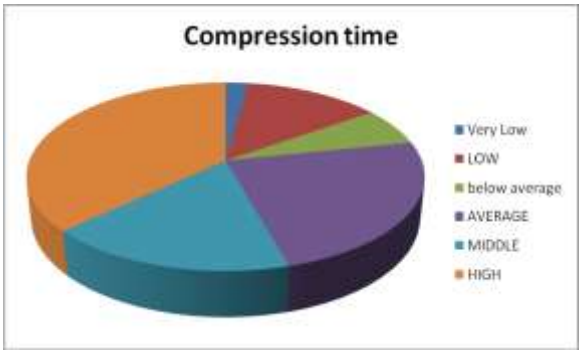
Graph-4 Based on Decompression time



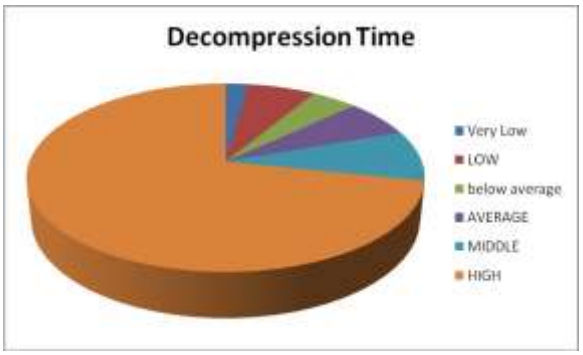
Graph-5 Based on code efficiency



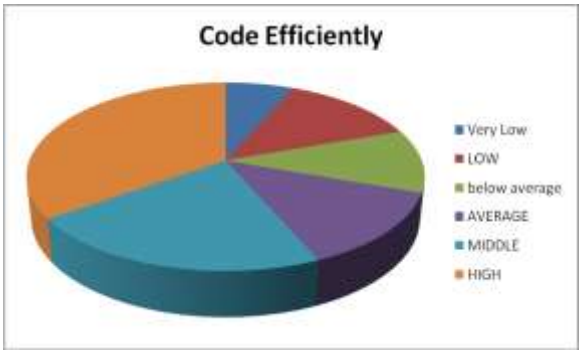
Graph-6 Cluster presentage based on Compression ratio



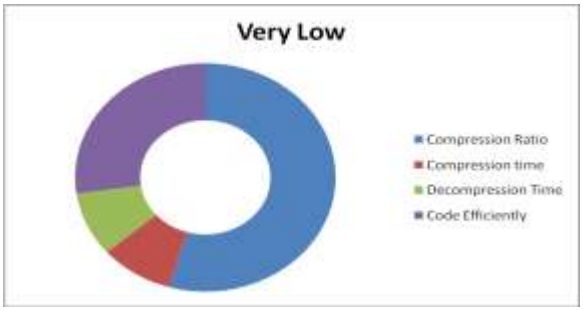
Graph-7 Cluster presentage based on Compression Time



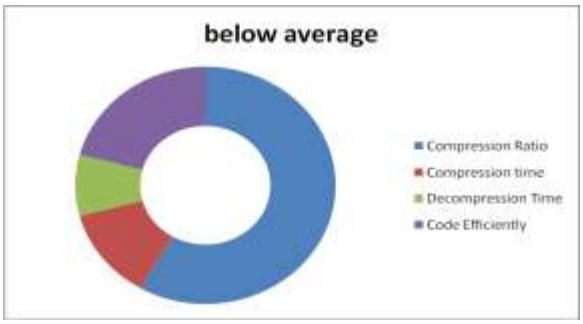
Graph-8 Cluster presentage based on Decompression rTime



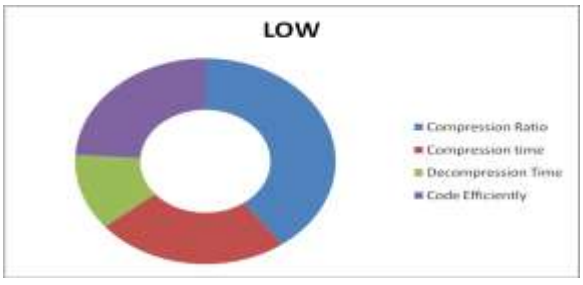
Graph-9 Cluster presentage based on code Efficienty



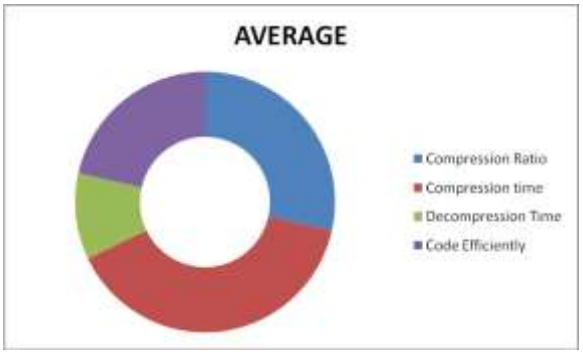
Graph-10 Cluster Range presentage very low



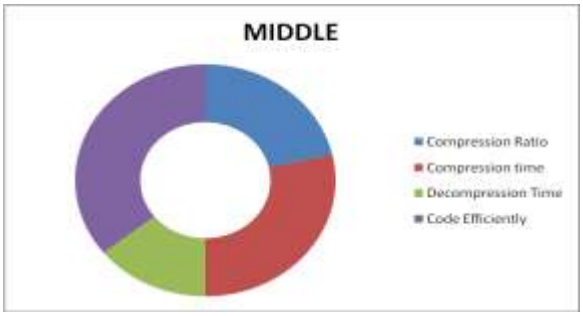
Graph-11 Cluster Range presentage Below average



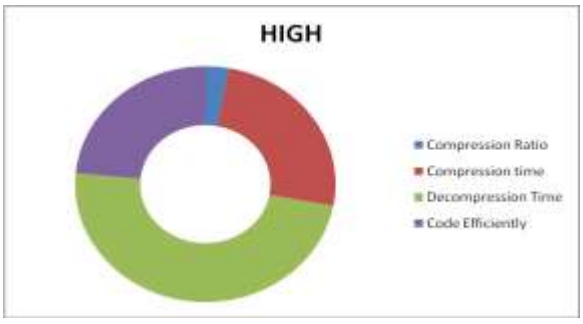
Graph-12 Cluster Range presentage low



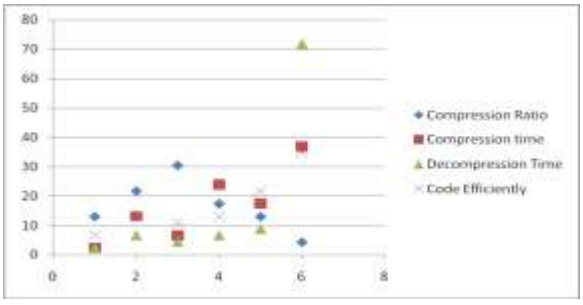
Graph-13 Cluster Range presentage Average



Graph-14 Cluster Range presentage middle



Graph-15 Cluster Range presentage high



Graph-16 Cluster presentage based on all parameters

III. CONCLUSION AND FUTURE ENHANCEMENT

In this study, how efficiently and effectively rule mining can be applied to evaluate the performance of various data compression algorithm on different types of files is shown. The major contribution of this work is 1,296 number of rules are framed and the implemented using VB script. The most important focus of this paper is how pattern bit can be applied in the rule mining. The primary aim of reducing time in searching of rules in a large rule-base with 1,296 rules is achieved hundred percentages. Also K-Medoid clustering algorithm is applied to group the rules based on the performance of lossless compression algorithms. This clustering helps to reduce the time ratio of rules triggering. This work can be further extended using any other algorithm.

III ACKNOWLEDGMENT

Late Dr. N. Nalayini. Associate Professor, Department of computer science, NGM College, Pollachi

IV. REFERENCES

[1]. T. C. Bell, J. G. Cleary, and I. H. Witten, "Text Compression English word" C1iffs:N. J. Prentice-Hall, 1990.

[2]. Rafael C. Gonzalez and Richard E. Woods, "Digital Image Processing, Reading, Massachusetts": Addison-Welsley Publishing Company, 1992.

[3]. Velmurugan and T. Santhanam "A commiserative analysis between K-medoid methods and Fuzzy C-means clustering algorithms for statistically distributed data points" JTAIT 2011

[4]. Park. Hae-sang; Lee. Jong-seok; Jun. Chi-hyuck. "A K-means-like Algorithm for K medoids Clustering and Its Performance ". Proceedings of the 36th CIE Conference on Computers & Industrial Engineering.. Taipei. Taiwan. p.1222-1231 Jun. 20-23 (2006).

[5]. Sarah N. Kohail and Alaa M. El-Halees "Implementation of Data Mining Techniques for Meteorological Data Analysis" International Journal of Information and Communication Technology Research. 2010\

[6]. Bartok J., Habala O., Bednar P., Gazak M., and Hluch L., "Data mining and integration for predicting significant meteorological phenomena" Procedia Computer Science, p.37 – 46. 2010

[7]. Berkhin P., "Survey of clustering data mining techniques, Accrue Software, San Jose", CA, Tech. Rep., 2002

[8]. Artiles, J.; Gonzalo, J. and Sekine, S. WePS 2 Evaluation Campaign: "overview of the Web People Search Clustering Task". 2nd Web People Search Evaluation Workshop (WePS 2009), 18th WWW Conference. 2009.

[9]. Park. Hae-sang; Lee. Jong-seok; Jun. Chi-hyuck. "A K-means-like Algorithm for Kmedoids Clustering and Its Performance". Proceedings of the 36th CIE Conference on Computers & Industrial Engineering.. Taipei. Taiwan. p.1222-1231 Jun. 20-23 (2006)

[10]. Nishad P.M. and Dr. N. Nalayini "Rule Base with Frequent Bit Pattern and Enhanced k-Medoid Algorithm for the Evaluation of Lossless Data Compression". International Journal of Advanced Research in Computer Science. Volume 3, No. 1, Jan-Feb 2012 pp 305-311

Biography:

Nishad PM M.Sc., M.Phil. Seven months Worked as a project trainee in Wipro in 2005,five years experience in teaching, one and half year in JNASC and Three and half year in MES Mampad College. He has published thirteen papers national level/international conference and journals. He has presented three seminars at national Level. Now he is pursuing Ph.D Computer Science in Dr. Mahalingam center for research and development atNGM College Pollachi.

Dr. R.Manicka chezian received his M.Sc., degree in Applied Science from P.S.G College of Technology, Coimbatore, India in 1987. He completed his M.S. degree in Software Systems from Birla Institute of Technology and Science, Pilani, Rajasthan, India and Ph D degree in Computer Science from School of Computer Science and Engineering, Bharathiar University, Coimbatore, India. He served as a Faculty of Maths and Computer Applications at P.S.G College of Technology, Coimbatore from 1987 to 1989. Presently, he has been working as an Associate Professor of Computer Science in N G M College (Autonomous), Pollachi under Bharathiar University, Coimbatore, India since 1989. He has published fifty five papers in international/national journal and conferences: He is a recipient of many awards like Desha Mithra Award and Best Paper Award. His research focuses on Network Databases, Data Mining, Distributed Computing, Data Compression, Mobile Computing, Real Time Systems and Bio-Informatics.