Optimization storage of Frequency table in Universal Combinatorial Coding

Jun Lu
College of Computer Science and Technology, Key Laboratory of Database and Parallel Computing Heilongjiang University, Heilongjiang Province
Harbin, China
lujun111_lily@sina.com

Juan Mo
College of Computer Science and Technology
Heilongjiang University Heilongjiang University
Harbin, China
mokaiyue@yeah.net

Tingting Gao
College of Computer Science and Technology
East University of Heilongjiang
Harbin, China
15695582@qq.com

Zhuo Zhang
College of Computer Science and Technology
Heilongjiang University Heilongjiang University
Harbin, China
owenzhuo37@163.com

Abstract—Universal combinatorial coding is a unique coding. It is based on the principle of permutation and combination. It has the multiple coding features and has a wide variety of applications. The sequence is handled by universal combinatorial coding and the result includes the ordinal, the frequency table, the length of the ordinal and so on. In order to save as little space as possible for the coded data, this thesis focuses on storage of the frequency table. Taking advantage of whole frequency table to forecast the frequency table information of each segment, it determines the frequency table storage rules based on the characteristics of the coding data which content is irregular. It adopts 0-1 labeling method to store frequency table information as bitwise. The effective bit number of character frequency will be stored automatically by collecting statistics of character frequency. The experiment shows that the frequency table storage method advanced in this paper can improve storage efficiency and save the coding information space.

Keywords- universal combinatorial coding; frequency table; compression; Optimization; statistics

I. INTRODUCTION

The current coding technology and application mostly derive from some basic coding methods, such as Arithmetic coding, dictionary encoding and so on [1-2]. With noticeable features, Universal Combinatorial Coding is unique among all these data coding methods. It is a lossless coding based on the principle of permutation and combination, which has the multiple coding features including Arithmetic coding, dictionary encoding and Tree Encoding. Furthermore, it has a great applying potentiality in the fields of data re-compression, data encryption/decryption [3]. Segment handling is the general method in dealing with code data. After the coding process, the coded data of each segment data should include ordinal, frequency table and some supplementary information, such as ordinal length. In order to improve the computing speed, series of optimization and parallel methods are adopted [4-7]. In addition, in order to save as little space as possible for coded data, frequency table optimization is necessary.

In order to restore the segmented data, frequency table contains code element frequency information in segmented data. Each encoded character (code element) and the corresponding frequency value can be decoded through the stored information in segment frequency table. If the actual character frequency values are stored directly, frequency table takes relatively large space. The paper [8] adopted the method that the different value between each character frequency and the fixed value was stored. But in this paper, the different value between adjacent character frequencies is stored and the storage efficiency can be improved.

II. FREQUENCY TABLE COMPRESSION PRINCIPLE

Encoding with universal combinatorial coding technology for each sequence data which length is L, the core algorithm is to solve the ordinal number of the sequence. The encoded group data includes the ordinal number, ordinal length and the frequency table. In paper [9], it has been confirmed that the length of ordinal is less than the length of the corresponding sequence. In order to offset the space occupied by frequency table with the difference between the length of the sequence and the ordinal as possible, it is to say, the encoded data space need occupy as little as possible, the compressing optimization of frequency table is essential. According to the character of the universal combinatorial coding in the paper [10], the differences of the character frequencies in the file with irregular content (such as compression file) are small, so the corresponding ordinal space is bigger. The difference between ordinal length and the sequence length is small. It cannot satisfy the frequency table space. Frequency table storage optimization in this paper is aimed at these kinds of files, such as RAR compression files.

This article firstly analyzed the frequencies of the group data which length is 128k in the compressed file. Then according some rules, characters of each group are

ordered according increment or decrement trends. At last, the difference of the adjacent character frequency is computed. Experiments show that the differences concentrate in certain areas. Fig.1 is a difference scatter plot figure for a compressed file whose length is 128k. Except that some characters which frequency difference is too big or too small, frequency differences mostly concentrate in certain range.

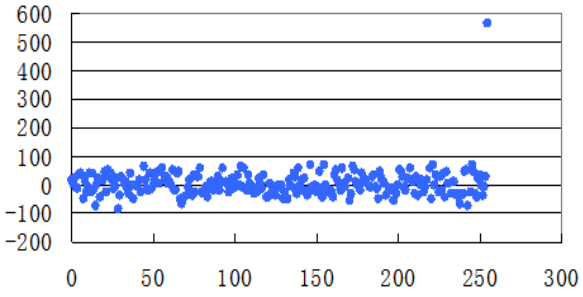

Figure 1. Statistical frequency difference scatter plot

Fig. 1 reflects that the differences of the character frequency concentrate a fixed area, only one frequency value is more than 500, which is the biggest frequency in this group and the value is true. The other data are the differences of the adjacent character frequencies. By the actual statistical analysis, about 95.7% frequency difference values are in [-64, 64]. One more space-saving method, bit-by-bit space saving is found. Each frequency can use 6 bits（2^6=64）or even less to be expressed.

III. FREQUENCY TABLE STORAGE REALIZATION

Suppose to divide a compression file into several group data which length is 64K. If the length of the last data is less than 64K, then the last data is directly added at the end of the encoded data. The paper[3] also mentions that frequency table can be stored bit by bit and then be compressed, but the frequency table of each group data is handled separately and each character frequency of the group data is the difference value between the frequency and a certain fixed data m, e.g. m=1000. The compressing optimization of frequency table in this paper is considered with affiliation between each group character frequency and the global character frequency, and then the space occupied by the file frequency table is further reduced.

The storage rule of the global frequency table is shown as below: First, character frequencies of the whole compression file are computed, then the characters are sorted in descending order according to their frequencies. Only the first character frequency is stored as real frequency, the rest characters storage the difference values between the character and the previous one. It need store 255 character frequencies. The last character frequency needn't store, its value is that the group length subtracts the frequency of the previous 255 characters. Global frequency table actually reflects the probability of the corresponding character frequency in each group.

The bitwise storage of the entire file character frequency is adopted by 01 labeling method. It can be shown as Fig. 2.

In Fig. 2, suppose that there is 6 effective bits behind 0, then 5 effective bits behind 10, the total bits (7 bits) are same; there is 7 effective bits behind 110, and there is 6 effective bits behind 1110, the total bits (10 bits) are same. The rest can be done in the same manner.

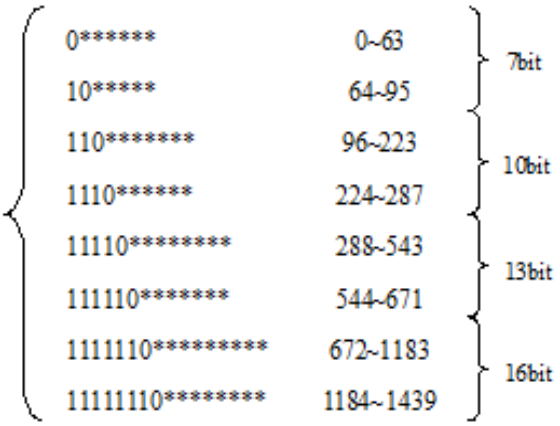

Figure 2. Entire document frequency table storage rules

Storage rules of each group frequency tables: Compute the characters and the frequencies of the first group data which length is 64k, and make the character order same as the entire file character order. At this time, the frequency difference value can be positive or negative. Compute the characters and the frequencies of the second group data which length is 64k, this group's character order is same as the frequency difference value sorted in descending between the entire file character frequency and the first group character frequency. Only the first character frequency is stored really, the rest frequencies are stored as the difference value between the character frequency and the previous one. Each group is stored by this way. In order to save space, the characters and frequencies of the last 64K group need not to be stored, they can be gotten by the way that entire character frequency minus each previous group character frequency.

The bitwise storage of each group character frequency is also adopted by 01 labeling. What different to the globe frequency table is that it needs one bit in front of the labeling bits. As Fig. 3, suppose there is 5 effective bits behind 0, then 4 effective bits behind 10, the total bits (7 bits) are same. The other frequency would be stored as this method.

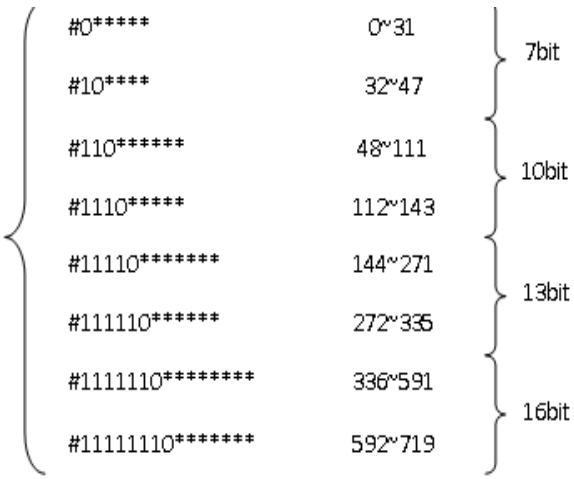

Figure 3. Grouped frequency table storage rules

In bitwise storage method as shown in Fig.1 and Fig 2, the number of effective bit behind 0 is supposed. In fact, the effective bit should be confirmed automatically.

Analyze the characters frequency of each group and find the way to reduce the frequency table space of the group. Sorted the 255 absolute difference value in increasing and take out the 40% position difference value in 255 characters, find out an integer m that is bigger or equal than this value, m is an integer power of 2. At this moment, the effective bit behind 0 is n ($m=2^n$).

In order to find out the best position, the experimental data in the follow 6 figures are gotten from RAR compression file. Extracted character frequency difference values respectively are from different group length (256K, 128K, 64K, 32K, 16K and 8K) in 6 different files under different ratio position (60%, 55%, 50%, 45%, 40%, 38%, 35% and 30%) taken as the standard. Hereby, the standard means the effective bits behind 0 according some certain character frequency value of ratio position.

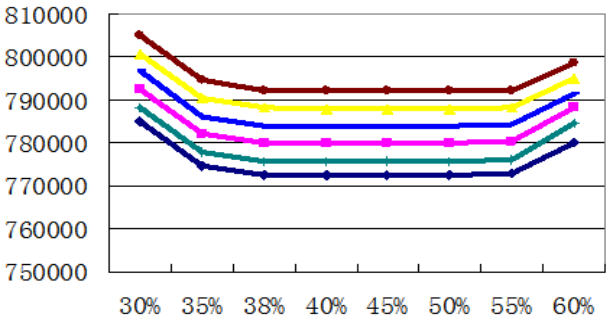

Figure 4. frequency table Spaces of 8K group

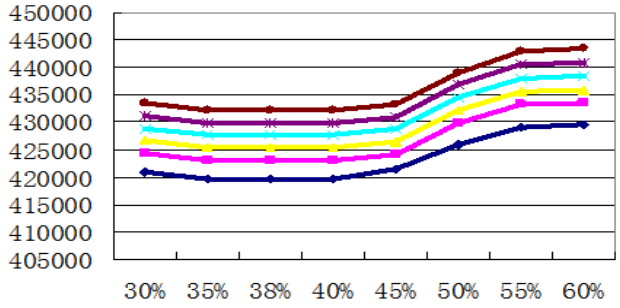

Figure 5. frequency table Spaces of 16K group

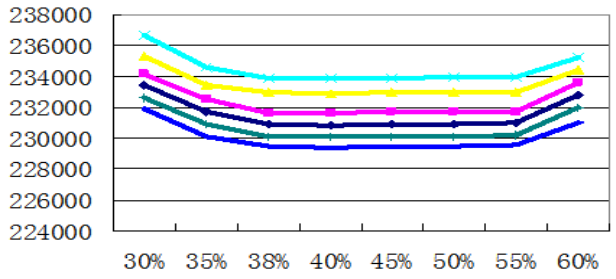

Figure 6. frequency table Spaces of 32K group

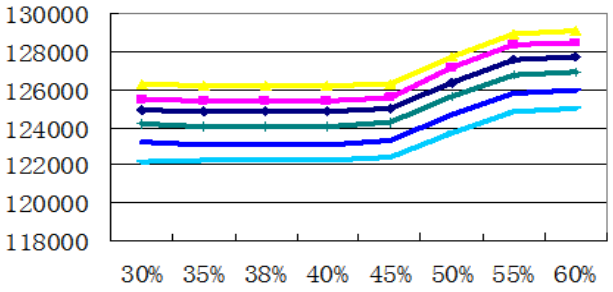

Figure 7. frequency table Spaces of 64K group

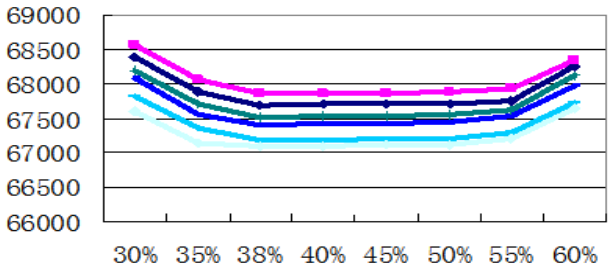

Figure 8. frequency table Spaces of 128K group

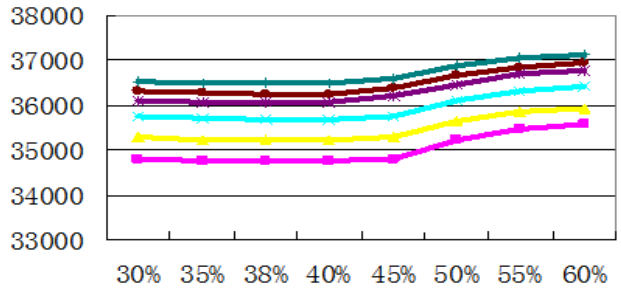

Figure 9. frequency table Spaces of 256K group

It can be known from fig.4, fig.6 and fig.8 that the curves of 8K, 32K and 128K group with different ratio position as the standard to measure the effective bit behind 0 are similar. From Fig.5, fig.7 and fig.9, it shows that the curves of 16K, 64K and 256K group with different ratio position as the standard to measure the effective bit behind 0 are similar. The common is that no matter the group length, the ideal position is in 38%~40%. At this time, the frequency table space is smallest. So in this thesis, according to sorting 255 character frequencies as increasing, the effective bits behind 0 are confirmed by selecting 40% position in the character frequency difference value. Because in most cases, the group frequency table space is minimum.

IV. FREQUENCY TABLE OPTIMIZATION TEST

The frequency table optimizing makes the encoded data space reduced. This section explains the frequency table optimizing efficiency by comparing the frequency table space before optimizing and after optimizing.

In order to see the positive effect caused by frequency table compression to coding, the experiments handled the same file twice according to different group length. One is the frequency table which is compressed, the other is that the frequency table doesn't be compressed. At last, compare and analyze the frequency table space after the

twice combination coding. The experimental data can be shown as Tab.I.

TABLE I. SPACE COST BEFORE AND AFTER OPTIMIZING

group data size	The size without compression	The size of compression
8K	4622592	792080
16K	2311424	433288
32K	1155328	233912
64K	577792	126304
128K	289024	67872
256K	144640	36596

In the Tab. I, the data reflect well that it saved space cost of encoded data after frequency table optimizing in universal combinatorial coding techniques. The smaller each group data is, the better of the frequency table compression is. Fig.10 shows the space cost comparison chart before and after optimizing frequency table of different group and the same file is encoded by universal combinatorial coding techniques. Abscissa represent different plaintext group length, which is 8K, 16K, 32K, 64K, 128K and 256K with the unit of "KB" respectively. Vertical axis represents the space costs of frequency table with unit of"B".

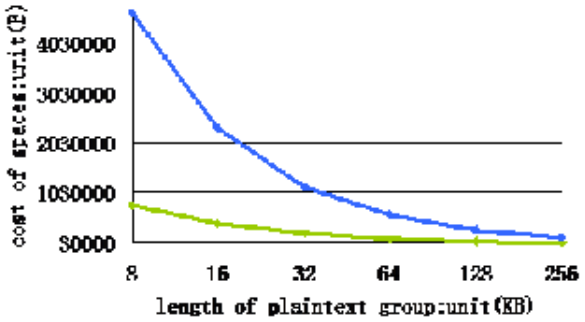

Figure 10. Space cost comparison chart before and after optimizing Frequency table

The shorter the group data is, the larger the encoded frequency table space is. The longer the group data is, the smaller the encoded frequency table space is. The main reason is that the frequency table of each group needs to be stored. It means that storing frequency value of 255 more for one more group (Code element length is 8 bits. Code element number is 2^8). When the packet data length greatly reduced, the number of the group greatly increased and the total length of the frequency table greatly increased too. The figure shows that the slope of non-optimized frequency table space curve is big and the slope of optimized frequency table space curve is small. With the decrease of the grouped data length, non-optimized frequency table space grows fast and optimized frequency table space grows slowly.

In universal combinatorial coding, the data encoded of each group contains frequency table, ordinal length and ordinal of this group. The length of the group data before encoding is L, which value can be 8K, 16K, 32K, 64K, 128K and 256K. When ordinal number of each group is calculated, the ordinal number of the group can be stored by an array which size is Big_long. Big_long is L/4. Each element in an array occupies 4 bytes, it means that the data is processed by 2^{32} system. The space of ordinal length can be neglected. The average space of each group ordinal, the remaining space and the average frequency table space corresponding to each group data are shown in table II. Four columns in the table use array element as a unit(4 bytes).

TABLE II. ORDINAL SPACE、FREE SPACE AND AVERAGE FREQUENECY TABLE SPACE

Group size	Ordinal space	Free space	frequenecy table space
2048	2008	40	43.87
4096	4051	45	47.99
8192	8144	48	51.84
16384	16331	53	55.98
32768	32708	60	60.17
65536	65463	73	64.88

The first three columns can be obtained from running combination coding algorithm. The last column can be obtained by the method that the last column of Tab.1 is divided by 4 * number of groups. It can be shown from the Tab.2 that when group data length is 256k, encoded group data length is less than the group data before encoding. But the longer the packet data length is, the slower the combination coding speed is. For a RAR compression file, universal combinatorial coding can be adopted to further improve the efficiency of data storage through optimizing the frequency table.

V. CONCLUSION

Optimization storage of frequency table of universal combinatorial coding is researched in this thesis. It firstly adopts 01 label method to storage the globe frequency table, which actually reflects each corresponding character frequency probability in each group data. According to the sequence and storage rules, frequency table of each group can be deduced one by one by the affiliation between global characters and group characters. In addition, according to the statistics information of each character frequency, the effective bits of character frequency will be confirmed automatically and then the storage efficiency of encoded data is increased.

ACKNOWLEDGMENT

This research is supported by Scientific Research Fund of Heilongjiang Provincial Education Department of China (No. 12521395)

REFERENCES

[1] J.Rissanen, G.Langdon, "Compression of Black - White Image with Arithmetic Coding", *IEEE Trans On Comm*. vol.29, no. 6, 1981, pp.858-867.

[2] J. Ziv , A. Lempel, "Compression of Individual Sequences via Variable Rate Coding", *IEEE Transactions on Information Theory*. vol.24, no. 5, 1978, pp530-536.

[3] Jun Lu, Zhuo Zhang, and Juan Mo. "Research on Universal Combinatorial Coding". The Scientific World Journal Volume 2014, Article ID 414613,8 pages http://dx.doi.org/10.1155/2014/414613

[4] Lu Jun, Liu DaXin,"Optimization on Computing Base-data in Constant Grade Compression," 2009 International Forum on Information Technology and Applications (IFITA 2009), ChengDu, P. R. China, May, 2009, pp. 68–71.

[5] Lu Jun, Liu Da-Xin,"Research on Parallel Technology within Section in Combinatorics Coding," The 2010 International Conference on Computer Application and System Modeling (ICCASM 2010), Taiyuan, P. R. China, October,2010, pp.252-255.

[6] Zhuo Zhang, Jun Lu, Ting Ting Gao, Juan Mo, Yu Liu. Research on Max Ordinal in Universal Combinatorics Coding Based on GPU Parallel Computing. 2nd International Conference on Measurement, Information and Control (ICMIC 2013). Harbin, China, August 16-18. 2013, Vol.1, pp 457-461.

[7] Juan Mo, Jun Lu, Nan Wang, Zhuo Zhang, Yu Liu. The GPU Parallel Algorithm of Whole Ordinal in Universal Combinatorics Coding. 2nd International Conference on Measurement, Information and Control (ICMIC 2013). Harbin, China, August 16-18. 2013, Vol.1, pp 467-471.

[8] Lu Jun, Liu DaXin,"Optimization of frequency table storage in Constant Grade Compression," 2009 International Forum on Information Technology and Applications (IFITA 2009), ChengDu, P. R. China, May, 2009, pp. 72–74.

[9] Lu Jun, Wang Tong, Liu Da-xin. "Research on Ordinal Properties in Combinatorics Coding Method". Journal of Computers. vol. 6, Jan. 2011, pp. 51-58.

[10] Lu Jun, Wang Tong, Li Yibing, "Study on Optimization Technology in Computing Ordinal Number," Journal of Computers, vol. 5, Feb. 2010, pp. 210-217,.