A new combination method for Data Compression Optimization

Msc.Farah R. Shareef
Sabahaliraq2014@gmail.com

Abstract

Now days the amount of data increases, so we need a good compression method in order to reduce the storage space required.Burrows wheeler transform method (BWT) and MoveTo Front method (MTF) are used for preprocessing the input data before compression. In this paper, we proposed three combinations algorithms and comparing between them to get an optimize combination for best compression ratio.

Keywords: BWT, MTF, JBE, GZIP, Zero RLE.

طريقة دمج جديدة لتحسين ضغط البيانات

المستخلص:

في هذه الايام كمية البيانات في زيادة لذا نحتاج الى طريقة ضغط جديدة لتقليل مساحة الخزن المطلوبة. BWT وMTF يستخدمان لتهيئة البيانات المدخلة قبل الضغط. في هذا البحث نقترح ثلاثة خوارزميات مدموجة والمقارنة بينهم للحصول على الدمج المثالي لافضل نسبة الضغط.

1- Introduction

Compression is an encoding process for data to be fewer bits than the original representation so that it getsminimize storage space and less consuming time when communicating over a network [1]. There are two kinds of compression, lossy and lossless.
In Lossy data compression, some information are losing. When the compression file is decompression, the output is not identical to the original data file. It is compatible for Movies, Sounds and Images. While in lossless, it is used with text files [2].

In compression the data, we can used for processing technique in-network so as to save energy because it decrease the size of data so as todecrease data transmitted and/or reduce transfer time because the amount of data is decreased [3].

In this paper we will take a surveyon different data compression algorithms which are used in combination of our proposed algorithms. These algorithms can be categories into preprocessing (transformation) and compression algorithms. Transformation algorithm does not compress data but preprocessingdata to optimize the next sequence ofthe input for transformation or compression algorithm [4].

2- Burrows wheeler transform(BWT)

Burrows wheeler transform(BWT) is widely used for lossless data compression. BWT is a transformation technique first introduced in 1994. The fundamental concept behind this technique is that when a text file or a character string is transformed, the size of the string does not change. The transformation only permutes the string into n permutations, where *k* is the total number of characters in the string[5].

3- Move to Front transform(MTF)

MTF is a preprocessing (transformation) algorithm. It is not for data compression, but can assist to minimize redundancy occasionally. The outcome from this technique is a code of symbol position [6].

4- JBE

J-bit encoding (JBE) task ishandling bits of data to decrease the amount and optimize input for other algorithms. The basic idea is to separate input data into two data where the first data takes the original nonzero byte and the second data takes bit value demonstrating position of nonzero and zero bytes. Both data then can be compress alone with other data compression algorithm to performhigh compression ratio[7].

5- Zero RLE

The Zero Run-Length Coding is a compression method, easy, simple andcoded sequences of zeroes to be shorter strings.

We can use Zero Run-Length Coding for large input, because for example, we can coded 7zeroes with only 3 characters [8].

6- GZIP

It is a general-purpose compression benefit, widespread and is used in manycommercial implementations. The advantages of using such aninstrument would be [9]:

- ✓ It is ready in both open-source and commercial implementations
- ✓ Has a better compression rate (40-50%) and freedom from patented algorithms
- ✓ It is built into http and web-servers as a standard feature.

So the purpose of GZIP Compression is to compressed pages before being transmitted from server to browser, in order to get small transmitted data, then to be quickin delivery. This is useful for servers running Windows operating system [10].

So, gzip is aformatof file and a software application used for compression and decompression the data in the file. Jean-Loup Gailly and Mark Adlerare created the program as an alternativefree software for the compress program utilized in system like Unix, and prepared for use by the GNU Project (the "g" is from "GNU")[11].

7 -Proposed algorithms

Now we have three proposed algorithms to be compared with each other, they are:

A. Proposed algorithm 1:

Depend on: (BWT +MTF +GZIP):

Input: plain text.

Output: compression text.

Algorithm1:

```
1- While Not EOF Do.
2- Read characters of plain text.
3- Call and apply BWT transformation.
4- Call and apply MTF transformation.
5- Call and apply GZIP compression.
6- End while.
```

B. Proposed algorithm 2:

Depend on: (BWT +MTF +Zero RLE + GZIP):
Input: plain text.
Output: compression text.
Algorithm1:

```
1- While Not EOF Do.
2- Read characters of plain text.
3- Call and apply BWT transformation.
4- Call and apply MTF transformation.
5- Call and apply Zero RLE compression.
6- Call and apply GZIP compression.
7- End while.
```

C. Proposed algorithm 3:

BWT transformation algorithm
Input: string of characters(Text).
Output: Transformed text(string L) and Index I.

```
1. While Not Eof Do
2. Read string of characters.
3. Create an N x N matrix M by using the input string S as the first row and rotating
   (cyclic shifting) the string N-1 times and each time adding the new string as a new
   row to the existing ones.
4. Sort the Matrix M lexicographically by rows. At least one of the rows of the newly
   created M' contains the original string S. We name the index of the first such row I.
5. Final step is to take the last characters of each row (from top to bottom-means L-
   column) and write it in a separate string L.
6. End of while.
```

Depend on: (BWT +MTF + JBE +GZIP):
Input: plain text.
Output: compression text.

Algorithm1:

```
1- While Not EOF Do.
2- Read characters of plain text.
3- Call and apply BWT transformation.
4- Calland apply MTF transformation.
5- Call and apply JBE compression.
6- Call and apply GZIP compression.
7- End while.
```

Now, calling BWT algorithm as:

The first stage of our proposed algorithms is BWT. It takes the input text, which we used as a string **S** of **N** characters which are chosen from an ordered alphabet of **X** characters [12].

BWT Back transformation algorithm

Example: If the text is (abc),it will be:

Result of BWT

MTF-Transformation algorithm

```
Input: string (from BWT), indexing alphabet.
Output: index of characters.
  1- While Not End of Line do
  2- Ch=read char.
  3-    Index=Get the index of the character based on the indexing alphabet (called global list Y).
  4-    Move the character to the first position of the indexing alphabet.
  5-    End while.
```

MTF-Back Transformationalgorithm

```
Input: codes (index of characters).
Output: input characters (BWT characters).
  1- While Not End of Line do
  2- Read code.
  3- Mappingto the alphabetic of input character.
  4- End while.
```

Alphabeticsortingfor
first columnTake last column

.

c	a	b

BWT Back transformation algorithm

```
Input: string of characters (L-column).
Output: Original input text.
    1- Sort the string L lexicographically and name it F.
    2- Put an index for the characters of the F-column.
    3- Matching between the characters in F-column with the characters of L-column.
    4- Get the character of the index value i from F-column.
    5- Repeat 3 and 4 until we get all the input text(original).
  The result fromthis stage consists of all stored characters.
```

First column		
a	b	c
c	a	b
b	c	a

		Last column
a	b	c
b	c	a
c	a	b

The second call of our proposed algorithms is MTF, the output from BWT, is input to MTF stage.

Example: If input is (c a b)

$Y={}^{0}a\ {}^{1}b\ {}^{2}c$ ==> save index position 2

Y= c a b ==> save index position 1

Y= a c b ==> save index position2

The result of MTF is 212.

Third call of our proposed algorithm1, is GZIP algorithm, the output from MTF, is input to GZIP stage.

In the proposed algorithm2, we need Zero RLE compression:

```
GZIP Compression
Input: MTF or Zero RLE or JBE codes
Output: compression codes.
1- Prepare and determine an output file name.
2- Read bytes from input.
3- Calling compress gzip function.
4- Write the result to output file.
```

```
GZIP Decompression
Input: compression codes.
Output: MTF or Zero RLE or JBE codes
    1. Determine an input file name (output file name).
    2. Read from output compression file as bytes.
    3. Calling decompress gzip function.
    4. Write the result to output file(original).
```

```
Zero RLE Compression
Input: number codes from MTF.
Output: Numbers with zero's coding depending on table1.
    1. Read numbers from MTF output.
    2. If the (num. >= 0 And num. =<8) then ADD 1/ex: 7 will be 8.
    3. If the num. >= 9 then Remain num./ex:9 remain 9.
    4. If the num. is equal to zero or zero's then depend on Table1 for coding.
```

```
Zero RLE Decompression
Input: number codes depending on Table1.
Output: Numbers of MTF output.
    1- Read output numbers with zeros from encoding zero RLE.
    2- If the num. (num. >= 0 And num. =<8) then Decrease 1/ex:7 will be 8.
    3- If the num. >= 9 then Remain num. /ex:9 remain 9.
    4- Return coding to zero's depending on Table2.
```

Zero RLE compression depending on table1 and table2:

No. of zero's	Coding string
1	0
2	1
3	00
4	01
5	10
6	11
7	000
…..	…..
16	1001

Table1(Encoding)

Coding string	No. of zero's
0	0
1	00
00	000
01	0000
10	00000
11	000000
000	0000000
…..	…..
1001	16 zero

Table2(Decoding)

Example: If numbers are (58900001) – 8byte will be:

Index	No. Before adding
1	5
2	8
3	1

+1

Index	No. After adding
1	6
2	9
3	2

Table3Table4

And (0000) will be (01) depending on Table1.So the result is: (699012)-6byte.

JBE Compression:

Input: **codes number from MTF.**

Output: **first-DATA I (original nonzero) and second-DATA II(value of bits for nonzero and zero bytes).**

1- **Read the numbers from MTF output.**
2- **Take byte (8-numbers) from MTF.**
3- **For each num. in byte**
   - A- **If num. is non zero then put num. in Array (DATA I).**
   - B- **If num. is zero then put 0 in temporary array *OR***
     If num. is non zero then put 1 intemporary array.
   - C- **If temporary is filled with 8 numbers, then print the byte value into DATA II.**
4- **Clear temporary Array.**
5- **Iterate steps (1-4) until end file.**
6- **Print DATA I and DATA II.**
7- **We can use a compression method(like GZIP) forDATA I and DATA II thatcan be compressed separately before combined (optional).**

JBE Decompression:

***Input*: JBE output.**

***Output*: codes number of MTF.**

1. **Read the numbers of DATA II output.**
2. **Convert each num. in DATA II to binary.**
3. **Ifbinary value =1 then convert to its value from DATA I .**
4. **If binary value =0 then remain 0.**
5. **Print values.**
6. **Iterate 1-5 until end of DATA II.**

In the proposed algorithm3, we need JBE (j-bit encoding):

Example: If the original numbers are: (25 0 20 14 0 8 0 9).

Encoding is:

original	Data I (non zero)	Temporary
25	25	1
0		0
20	20	1
14	24	1
0		0
8	8	1
0		0
9	9	1

Table5

Temporary will be converted to binary value (10110101)= 181 (Data II) and (DATA I) that we need it for decoding in order to return the original numbers.

8- Performance Analysis

The performance analysis of three combination algorithms are done for five different text files sizes. The compression ratio of these proposed algorithms are compared with each other, so the results are:

Files	Original size(bytes)	BWT+MTF+GZIP (Algorithm1)	BWT+MTF+ Zero RLE+GZIP (Algorithm2)	BWT+MTF+ JBE+GZIP (Algorithm3)
File1	298	204	203	266
Comp.Ratio		68.4	68.1	89.2
File2	452	259	254	314
Comp.Ratio		57.3	56.1	69.4
File3	892	474	443	537
Comp.Ratio		53.1	49.6	60.2
File4	1092	583	549	642
Comp.Ratio		53.3	50.2	58.7
File5	4514	1008	806	977
Comp.Ratio		22.3	17.8	21.6

Table6- The results of the compression of our combination proposed algorithms.

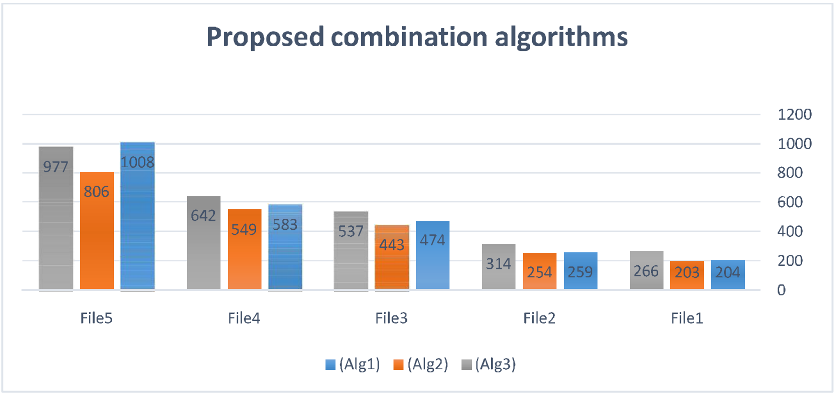

Figure 1-sizes of different filesafter applying three proposed combination algorithms.

9- Conclusions

In this paper, we proposes three combination algorithms that can be used to optimize other algorithm. We applied 5 files of different sizes, 3 combination algorithms have been checked and compared. Thesecond algorithm(BWT+MTF+Zero RLE+GZIP) wins, because it has a better compression ratio by using betweenmove to front transform (MTF) and GZIP compression.

10- References

1) Sidhu ,A., S., Garg ,Er. M.,"Research Paper on Text Data CompressionAlgorithm using Hybrid Approach",2014.
2) Forouzan.B,M.,"Foundations of computer science", Second Editon,2008.
3) Capo-Chichi, E.P.,Guyennet,H., Friedt,J.," K-RLE : A new Data Compression Algorithm forWireless Sensor Network",2009.
4) Suarjaya ,I ," A New Algorithm for Data Compression Optimization", International Journal of Advanced Computer Science and Applications, Vol. 3, No.82012.
5) Bhuyan, M.P., Deka, V. ,Bordoloi, S., "Burrows Wheeler Data Compression And Secure Transmission", International Journal of Research in Engineering and Technology, Vol.02, Dec-2013.
6) Campos, A. S. E. "Move to Front", Available: http://www.arturocampos.com/ac_mtf.html (last accessed July 2012).
7) Ambadekar S., Gandhi, K., Nagaria, J., Shah ,R.,"Advanced Data Compression Using J-bit Algorithm",2013.
8) Daniel Schiller , " The Burrows-Wheeler Algorithm", August 5, 2012.
9) Smitha S. Nair, "XML Compression Techniques: A Survey", University of Iowa, USA.
10) Available at: http://www. Zen cart support, what is GZIP, and why should I use it?
11) Available at: gzip - Wikipedia, the free encyclopedia.html.
12) Jay,Geissberger,"Burrows Wheelr Transformation",Telekommunikation,389.130.