Volume 4, Issue 3, March 2014 ISSN: 2277 128X
International Journal of Advanced Research in Computer Science and Software Engineering
Research Paper
Available online at: www.ijarcsse.com

Bitwise Data Compression-Encryption Technique

Mohammed Zaid Ul Qamar
Student, Department of E&TC, GHRCEM,
University of Pune, India

Abstract— Here present is an alternate method for lossless data compression where we can represent our data bits in compressed and encrypted form using a single method. Unlike, where we have to use different methods for data compression [1] and data encryption [2]. Bitwise Data Compression-Encryption Technique can be used for both variable and fixed data bits and also eliminates the use of dictionary for data compression [3].

Keywords— BDCET (Bitwise Data Compression-Encryption Technique), Lossless Data Compression, Data Encryption, Variable Length Data, Fixed Length Data, Dictionary.

I. INTRODUCTION

There are many a theorem presenting fixed to variable [4] and variable to fixed [3] length coding, but these theorem requires use of dictionary where it keeps the knowledge of frequently occurring data patterns. If the compressed data [5] produced here is to be encrypted then again different encryption theorems are used such as [1]-[2]. Thus logically a data which is to be compressed and encrypted goes through two different methods. In this method, Bitwise Data Compression-Encryption Technique, we tend to use a single method for both data compression and encryption. If the input data is in fixed form then the output of BDCET will be in fixed form and vice versa unlike [3]-[4]. BDCET generates a key from its own input data which is symmetric in form [6]. BDCET is a lossless data compression technique where bit to bit encryption is done.

II. STATEMENT AND METHOD IMPLEMENTATION

Bitwise Data Compression-Encryption Technique is a method where the use of two different methods for data compression and data encryption is eliminated.

For example consider following data stream '*U*':

U= 000101110010100101

These data bits are now divide into pairs of two bits each i.e. symbols *'S'*.

S= 00 01 01 11 00 10 10 01 01

Now consider a symbol '*A*'. Where A= 11, 00, 01 or 10. Here we will be using A= 01.
Now consider a table as shown below:

TABLE I
COMPARISON BETWEEN SYMBOLS 'U' AND 'A'

S	A	C
00	01	1
01	01	0
10	01	0
11	01	1

Where, S represents data stream U in symbols format.
A represents symbol '01'
C represents subsequent compressed bit.

Considering Table 1, Data stream 'C' can be formed,

S= 00 01 01 11 00 10 10 01 01
↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓ ↓
C= 1 0 0 1 1 0 0 0 0

Data stream 'C' formed here is compressed data form of original input data stream 'U'. If we observe data stream 'C'

we can see that we have represented symbols 00 and 11 with bit 1 and symbols 01 and 10 with bit 0. To overcome this similar symbol representation we will now generate another data stream 'K'. We will call this data stream '*key'*.

To generate key 'K' of data stream 'S' which is symbol wise representation of original data stream 'U' we will consider another table as shown below:

TABLE 2
KEY 'K' GENERATION OF SYMBOL DATA STREAM 'S'

S	K
00	0
01	1
10	0
11	1

Where, S represents data stream U in symbols format.
K represents subsequent key bit.

Considering Table 2, Data stream 'K' can be formed,

S= 00 01 01 11 00 10 10 01 01

K = 0 1 1 1 0 0 0 1 1

Here data stream 'K' produced is used as a symmetric key for data decryption on the receiver side. Combined form of data streams 'C' and 'K' gives us our required compressed and ciphered form of original data stream 'U'. We will denote combined form of data stream 'C' and 'K' as data stream 'Z' which is in ciphered form.

U= 000101110010100101
C=*100110000*
K=**011100011**

Thus data stream 'U' can be represented in its compressed and ciphered form as below

Z= *100110000* **011100011**
C K

Where Z represents compressed and ciphered form of our original input data stream 'U'

III. DECODING DATA STREAM 'Z'

For decoding original data stream 'U', first consider the following table:

TABLE 3
GENERATION OF SYMBOL DATA STREAM 'S'

A	*C*	S1	K	S2	S
01	*1*	00 OR 11	**0**	00 OR 10	00
01	*0*	01 OR 10	**1**	01 OR 11	01
01	*0*	01 OR 10	**0**	00 OR 10	10
01	*1*	00 OR 11	**1**	01 OR 11	11

Where, A represents symbol '01'
C represents subsequent compressed bit.
K represents subsequent key bit.
S1 represents possible data stream U after comparing A and C.
S2 represents possible data stream U after comparing K and C.
S represents data stream U after comparing S1 and S2.

From Table 3, to decode original data signal 'U' first compare symbol '01' with subsequent data bit 'C' from Table 1. After comparing we get two possible outcomes, '00 or 11' and '01 or 10' which are denoted by S1. Now consider bit stream 'K' where bit 1 represents '00 or 10' and bit 0 represents '01 or 11' as we refer Table 2. Mark these possible outcomes after considering bit stream 'K' as symbol S2. Now compare symbol S1 and S2 within each row as obtained in Table 3 and mark the common symbol. This common symbol produced is nothing but our symbol data stream 'S' which is the symbol form of original input data stream 'U'.

IV. RESULT ANALYSIS

BDCET gives designer the liberty to design its own encoder and decoder like designer could design the Table1 and Table2 according to its wish. Data 'C' and Key 'K' could be transmitted in a single stream. Here also designer has much option of placing Data 'C' and Key 'K' in a single stream such as *100***011***110***100***000***011** or **011***100***100***110***011***000* etc. Unlike, [3]-[4], BDCET gives option of variable-variable and fixed-fixed length coding. Use of dictionary for data coding where frequently used patterns are saved has been eliminated. BDCET gives us the option of implementing other data encryption method once the data stream 'Z' is formed to apply more security to our data signal.

V. CONCLUSIONS

BDCET is a compression cum encryption theorem, thus it could very much implemented wherever data security is needed. One major advantage of BDCET is that it eliminates the use of dictionary, thus every encoder and decoder has unique a design. Individual can use BDCET easily without opting for complex encryption algorithms.

REFERENCES

[1] S. Haykin, "Fundamental Limits of Information Theory" in *Communication Systems*, 4th ed., John Wiley & Sons Inc, 2001, pp.575–581.

[2] B.A. Forouzon, "Cryptography" in *Data Communications and Networking*, 4th ed., McGraw Hill, 2007, pp.931–951.

[3] J. Ziv and A. Lampel. (1977, May.). A Universal algorithm for sequential data compression. *IEEETrans. Inform. Theory, vol.* IT-23, pp. 337–343.

[4] S. Haykin, "Fundamental Limits of Information Theory" in *Communication Systems*, 4th ed., John Wiley & Sons Inc, 2001, pp.578–580.

[5] S. Haykin, "Fundamental Limits of Information Theory" in *Communication Systems*, 4th ed., John Wiley & Sons Inc, 2001, pp.580–581.

[6] B.A. Forouzon, "Cryptography" in *Data Communications and Networking*, 4th ed., McGraw Hill, 2007, pp.935–948

