INTERNATIONAL JOURNAL OF INNOVATIVE RESEARCH & DEVELOPMENT

ISSN 2278 – 0211 (Online)

A Compression Algorithm Using Variable-to-Fixed Block Arithmetic Codes

Anurag Jain
Professor, Department of Computer Science, R.I.T., Goa, India

Abstract:
In this paper a method for lossless image compression which will produce a better results in terms of overall compression ratio for bi-level images is introduced. This algorithm uses the Block Arithmetic Coding for Image Compression (BACIC) algorithm. For the purpose of compression it uses variable-to-fixed arithmetic coder called Block Arithmetic Coder (BAC)which is a a technique for entropy coding that combines many of the advantages of ordinary stream arithmetic coding with the simplicity of block codes. The code is variable length in to fixed out (V to F), in which the source sequence of bits is partitioned into variable length that are encoded by a fixed length sequences of bits.
Arithmetic coding assumes an explicit probabilistic model of the source. So, this models uses the estimation of probability that serves as an index into a probability table and used to compute p_1 (the probability of a bit equaling one), the probability measure BAC needs to compute a codeword.

1. Introduction

Image compression plays an important role for image storage and transmission. Demand for communication of multimedia data through the telecommunications network and accessing the multimedia data through Internet is growing explosively. Image data comprise a significant portion of the multimedia data and they occupy the large amount of the communication bandwidth for multimedia communication. Storing images in less memory leads to a direct reduction in storage cost and faster data transmissions. These facts justify the efforts, of private companies and universities, on new image compression algorithms. As a result, development of efficient image compression techniques continues to be an important challenge to us, both in academia and in industry. There are two types of compression 1. Loosy Compression and 2. Lossless Compression. This paper presents a technique for Lossless Compression. Lossless compression involves with compressing data which, when decompressed, will be an exact replica of the original data. The bit information of image need to be exactly reproduced when decompressed. Most data contains some amount of redundancy, which can sometimes be removed for storage and replaced for recovery. An approximation of the original image is enough for most purposes, as long as the error between the original and the compressed image is tolerable. The method currently used to send facsimiles is the Group 3 fax algorithm (G3), published in 1980. It uses a modified Huffman code to encode data. To encode an image,G3 raster scans the image according to a static, modified Huffman table given in G3's standard. Because business documents typically have long runs. The overall compression ratio of G3 for business documents is: approximately 7.7. And for halftone images G3's overall compression ratio is unacceptable: approximately 0.50.
For this paper, Overall Compression Ratio O.C.R.) can be defined as the ratio of the total number of bytes in the original bi-level images to the total number of bytes in their compressed files.
If O.C.R. >1 then it indicates that the original image is compressed, but if O.C.R. <1then it indicates that the original image was actually expanded rather than compressed.
Thus, because of G3's inability to compress halftone images, a new bi-level image compression technique is needed.
In 1993, a new standard, the Joint Bi-level Image Experts Group (JBIG) Algorithm was introduced to replace G3.
JBIG uses an arithmetic coder, the IBM QM-coder, which, adapts its coding to each image. As a result, JBIG's O.C.R. is 2.5 times G3's for business documents and is 5.5 times G3's for halftone images. However, JBIG is covered by more than 24 patents, most of which involve the IBM QM-coder. Largely because of these patents, JBIG has not been implemented commercially.
This paper presents Block Arithmetic Coding for Image Compression (BACIC), a new algorithm for lossless bi-level image compression based on the Block Arithmetic Coder (BAC). As with most arithmetic coders, BAC parses its coding interval according to the binary source symbol's probabilities p0 and p1 (the probability of a bit equaling zero and one, respectively). These probability measures together with the size of the codebook, K, uniquely determine a BAC codebook.

2. Estimation of Probability

When coding a bi-level image, an accurate estimate of one of the symbol probabilities, p0 or p1, is needed to maximize the coding efficiency of an arithmetic coder like BAC. Because p0 = (1 - p1) in a bi-level image, an estimate of p1 immediately provides an estimate of p0. To estimate p1, which is usually unknown and typically varies throughout the image.
In business-type and picture-type images, pixel patterns are typically repeated over and over again.

3. Generalized Block Arithmetic Coding

Block Arithmetic Coding (B.A.C.) of a binary source is a fast, efficient method of source coding that maps variable-length inputs to fixed-length codeword. There are two parameters that can determine a binary block arithmetic coding tree, those are p0 and K. where p0 indicates the probability of a bit equaling zero**,** and K indicates the number of code words in the BAC codebook. It relies on one equation

K0 = [p0.K] (1)

Where K0 is one of the two child nodes of K in (1); the other child node is K1 and this K1= K - K0.
The brackets [ ] in (1) indicates that the product of p0.K is rounded to the nearest integer, unless [p0.K]=K or [p0.K]=0.
If [p0.K] = K then K0 = K-1 ;
And if [p0.K] = 0 then K0 = 1.
The estimate of p0 that BACIC uses is based on the estimate of p1.
And K is an integer constant predetermined by the programmer; typically, is as large as possible to maximize BAC's coding efficiency.
In this section, I explain how BAC encodes a bi-level image based on K and p0. But first I examine how BAC encodes a sequence of image information binary input symbols where suppose p0=0.80 and K=16.
That means the BAC codebook for this binary input sequence contains code words numbered 0–15 (0000–1111, in binary).
To build this BAC coding tree and codebook, BAC starts with K = 16 and uses K0 = [p0.K] = 13 at the first branch. That means the lower thirteen code words, code words 0–12
(0000–1100), of the original 16 have input strings whose first bit is "0". The remaining three code words, 13–15 (1101–1111), have input strings whose first bit is "1."
To continue the tree, BAC moves, in turn, to both of the child nodes K0 = 13 and K1=3. Moving to the first child node K0=13, BAC continues the branching by letting K = 13 and applying equation (1). Now K0 = 10. (K0 = 10 is the lower branch coming from K = 13 in Fig.) Similarly, the first two bits of binary code words 1010–1100 are "01."
Each of the two child nodes of K= 13 (K0 = 10 and K1 = 3) in turn becomes K as well. Each branch is followed until a leaf that means a child node equal to one is encountered; thus, each codeword is assured of having its own unique branch from root to leaf. Finally, once every leaf is found, the coding tree and codebook are complete.

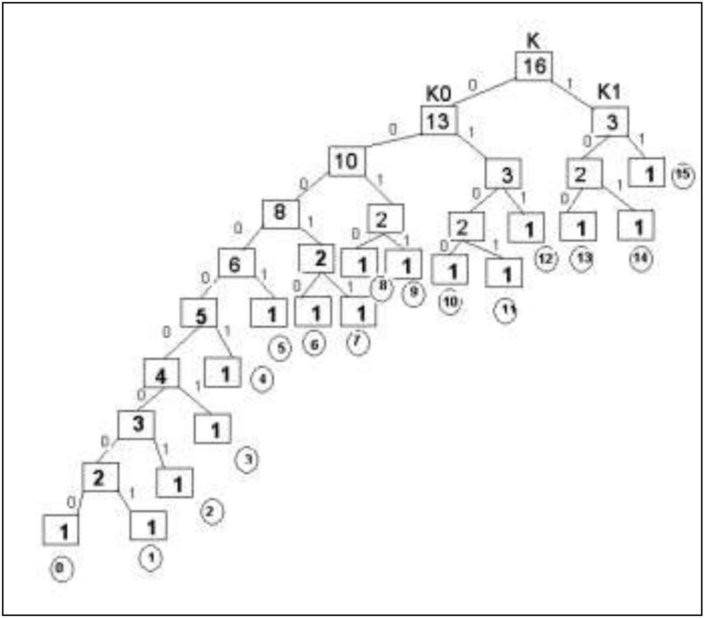

Figure 1: BAC Coding Tree for p0=0.8 and K=16

Input	Codeword	Input	Codeword

String		String	
000000000	0000	0010	1000
000000001	0001	0011	1001
00000001	0010	0100	1010
0000001	0011	0101	1011
000001	0100	011	1100
00001	0101	100	1101
00010	0110	101	1110
00011	0111	11	1111

Table 1

BAC CODEBOOK WITH K = 16, p = 0.80

4. Results: Compression Ratios for Group 3, JBIG, and BACIC

In this section, I present the results for the five business-type documents and five halftone images. I also discuss how the method of half toning a grayscale image affects the halftone image's compressibility.

To test G3's, JBIG's, and BACIC's ability to compress business documents, I used the five test images. As shown in Table III, both JBIG and BACIC have overall compression ratios that are approximately 3 times that of G3, although JBIG's overall compression ratio is about 3.1% greater than BACIC's for these images.

Location of Input Bits From Left Hand Side									BAC Code (Decimal)	BAC Code (Binary)
1	2	3	4	5	6	7	8	9		
0	0	0	0	0	0	0	0	0	0	0000
0	0	0	0	0	0	0	0	1	1	0001
0	0	0	0	0	0	0	1		2	0010
0	0	0	0	0	0	1			3	0011
0	0	0	0	0	1				4	0100
0	0	0	0	1					5	0101
0	0	0	1	0					6	0110
0	0	0	1	1					7	0111
0	0	1	0						8	1000
0	0	1	1						9	1001
0	1	0	0						10	1010
0	1	0	1						11	1011
0	1	1							12	1100
1	0	0							13	1101
1	0	1							14	1110
1	1								15	1111

Table 2

Compression Ratio For Five Test Images			
Images	**G3**	**JBIG**	**BACIC**
Test Image # 1	10	40.7	38.9
Test Image # 2	8.17	35.9	33.4
Test Image # 3	13.7	34.8	33.5
Test Image # 4	7.89	23.3	23
Test Image # 5	4.82	9.11	8.79
Overall C.R.	**8.916**	**28.762**	**27.518**

Table 3

I tested five additional business type documents: I chose both handwritten and typed documents to test how the compression ratios of JBIG and BACIC differ for each class of business document.

The remaining eight additional business documents are called the "modified CCITT" documents.

The compression ratios of all 5 documents for G3, JBIG, and BACIC are shown in Table III. Table IV classifies the compression ratios of the additional business-type documents.

As we see in Table IV, BACIC's overall compression ratio is the same as JBIG's and 2.4 times greater than G3's. BACIC yields compression ratios slightly higher than JBIG's for typed documents, while JBIG's compression ratios are higher for handwritten documents.

Compression Ratio For Five Business Type Documents			
Images	**G3**	**JBIG**	**BACIC**
Template	9.76	60.2	61.4
Block Diagram	12.4	53	53.9
Ratio Table	16.5	56.2	61.4
Abstract	4.53	9.54	9.64
Page 5 (From a book)	9.2	23.8	24
Overall C.R.	**10.478**	**40.548**	**42.068**

Table 4

5. Conclusions

In this paper, I established a new method for lossless compression of bi-level images that was published in IEEE on MAY-2001 by Maire D. Reavy and Charles G. Boncelet. This new method, BACIC, uses BAC coder for encoding the image. In section II I discussed how we can implement the probability model to build a BAC tree.

In section III, I explained how BAC encodes bi-level images with the help of two parameters p0 and K by using a simple example to illustrate the procedure. Finally, in Section IV, I presented the results for BACIC and compared them to G3's and JBIG's results.

What is more, BACIC's overall compression ratio is 2.4 times G3's for the business-type documents and 6.1 times G3's for all classes of halftone images. BACIC performs particularly well for halftone images generated by dispersed-dot ordered dither: here, BACIC's overall compression ratio is 12.1 times G3's.

In conclusion, BACIC compresses bi-level images as efficiently as JBIG and outperforms G3 for every class of bi-level image.

6. References

i. IEEE Transations On Image Processing: Maire D. Reavy and Charles G. Boncelet
ii. ISO CCITT Recommend. T.4, Standardization of group 3 facsimile apparatus for document transmission., 1980.
iii. ISO/IEC JTC1 CD 11544: Coded representation of picture and audio information-progressive bi-level image compression, 1993.
iv. T. C. Bell, J. G. Cleary, and I. H. Witten, Text Compression. Englewood Cliffs, NJ: Prentice-Hall, 1990.
v. V. Bhaskaran and K. Konstantinides, Image and Video Compression Standards. Norwell, MA: Kluwer, 1995.
vi. C. G. Boncelet, "Block arithmetic coding for Markov sources," in Proc.
vii. IEEE Int. Symp. Information Theory, 1993, p. 118.
viii. "Block arithmetic coding for source compression," IEEE Trans. Inform. Theory, vol. IT-39, pp. 1546–1554, Sept. 1993.
ix. J. Bradley, XV ANSI C Program: An Interactive Image Display for the X Window System. Philadelphia, PA: Univ. Pennsylvania, 1994.
x. M. K. Kuhn, "JBIG-KIT portable ANSI C library," Univ. Erlangen, Erlangen, Germany, 1996.
xi. W. B. Pennebaker, J. L. Mitchell, G. G. Langdon, and R. B. Arps, "An overview of the basic principles of the q-coder adaptive binary arithmetic coder," IBM J. Res. Develop., vol. 32, pp. 717–726, Nov. 1988.
xii. J. Poskanzer, Pgmtopbm ANSI C Program: Silicon Graphics, Inc., 1989.
xiii. R. Ulichney, Digital Halftoning. Cambridge, MA: MIT Press, 1987.