2014 International Conference on Electronics, Communication and Computational Engineering (ICECCE)

Space Complexity Analysis of Various Sparse Matrix Storage Formats used in Rectangular Segmentation Image Compression Technique

Sumithra Sriram
School of Computer Science and Engineering,
VIT University,
Vellore, India.
Email: sumithrasriram@gmail.com

Saira Banu.J
School of Computer Science and Engineering,
VIT University,
Vellore, India.
Email: jsairabanu@vit.ac.in

Dr. Rajasekhara Babu
School of Computer Science and Engineering,
VIT University,
Vellore, India.
Email: mrajasekharababu@vit.ac.in

***Abstract* -With the increase in the resolution of images, arises the need to compress these images effectively without much loss, for easy storage and transmission. Sparse matrices are matrices that have majority of their elements as zeroes, which brings in the possibility of storing just the non-zero elements in a space efficient manner using various formats. Images, which are essentially matrices, if somehow expressed as sparse matrices, can be similarly stored. The rectangular segmentation is a method that can be used to do so. In this paper, we analyze the space complexity of various storage formats for benchmark matrices and the suitability of these formats to compress images using rectangular segmentation method.**

Keywords—Rectangular segmentation, image compression, sparse matrix, space complexity

I. INTRODUCTION

Sparse matrices have an ever-growing field of application, being used in various scientific and technological fields such as image processing, machine learning and database systems. It is becoming increasingly popular as there are many methods that can be used to store them efficiently. Since most of the entries in a sparse matrix are zeroes, there have been various storage formats that have been proposed to efficiently store only the non-zero elements in the matrix. The space complexity of these formats depends upon the distribution of the non-zero elements in the matrix, and each storage format is known to be effective for a particular form of arrangement.

Image compression is a technique that is used to reduce the size of an image by removing redundant data, for efficient storage and transfer. There are essentially two broad categories of compression techniques- lossy compression, and lossless compression. The lossy compression techniques, that include the famous JPEG compression technique, are those in which a minor loss of accuracy of the pixels is tolerated for considerable reduction in size. These techniques are suitable for natural images. The lossless compression techniques on the other hand, are techniques where no reduction in accuracy is tolerated. Run length coding, entropy coding and arithmetic coding are a few examples of this technique. These are used for medical and satellite images, where pixel level accuracy is necessary.

Many image compression algorithms are based on transformation or decomposition of the images. Quarter-tree decomposition technique is one of the most used techniques, as it is fast, and relatively simple to compute. One drawback of this technique however, is that if the image's size is not a power of 2, then it is segmented into an increasing number of smaller blocks that decreases the compression ratio. This drawback is overcome by the rectangular segmentation technique. To increase the compression ratio further, COO (Coordinate) storage format of spare matrix is combined with this technique. The COO format however occupies a lot of space as compared to other storage formats.

In this paper, we analyze the space complexity of various benchmark matrices when stored in different spare matrix storage methods such as COO, CSR, QCSR and MinQuad. We also analyze the space complexity of small images when compressed with the rectangular segmentation technique with various sparse matrix storage methods. The rest of the paper is organized as follows: Section 2 is about the related work in the fields of discussion; Section 3 addresses the various sparse matrix storage formats available; Section 4 gives a brief overview about the rectangular segmentation technique and our analysis; Section 5 discusses about the experimental analysis and the results obtained; Section 6 gives the conclusion and future works.

II. RELATED WORK

An image is usually represented in a 2-dimensional matrix in the computer memory. Compression of images using sparse matrix storage format is an active research topic. When an image represented in a matrix has more number of zeros, storing the matrix with sparse matrix storage format will effectively increase the compression ratio. Normally, binary images are represented with 0s and 1s. So, compressing binary images using sparse matrix storage method will effectively reduce the storage space and increase the compression ratio. Recent research trend is to use sparse matrix storage format with greyscale and colour images. ZhangTianxu and Zeng Yonghui[1], have worked in exploiting sparse matrix storage method for efficient image compression. TanayaGuha and Rabab K. Ward [2],have done a research work on using sparse

978-1-4799-5748-4/14/$31.00 © 2014 IEEE

matrix storage format in compressing similar images. Yi-Chen Tsai, e tal [3] proposed a quad tree decomposition technique to improve the performance of cartoon images. He Xing-heng and CHEN Hui [5] analyzed quarter-tree decomposition of image compression. It is a simple and fast method of image compression. The main drawback in this method is that any image without the size of 2^n will be segmented with more number of blocks and thereby reduces the compression ratio.Kitty Arora and Manshi Shukla [4], present a comprehensive survey on lossy and lossless image compression techniques. Guha, T. and Ward [14] in their paper have used sparse representation to compute the similarity of images. R.KShengli Chen, et al [6] proposed a new compression algorithm based on rectangular segmentation method with COO sparse matrix storage format. In this paper, we address the same algorithm and analyse the effects of using various sparse matrix storage formats in place of the COO method used.

III. SPARSE MATRIX STORAGE FORMATS

Various sparse matrix storage formats[10][11] that are often used are:

A. *Coordinate (COO) format:*

This is the most general type of storage format that can accommodate any type of sparse matrix. It includes three one-dimension arrays- one for storing the data, one for the column index, and the other for the row index. One disadvantage of this format is that it occupies a lot of space when compared to the other storage formats. The space complexity of this format is given by:

$$S(COO) = 2 \cdot N \cdot S(n)$$

Where N is the total number of non-zeroes and n is the order of the matrix.For example:

$$\begin{pmatrix} 0 & 0 & 0 & 0 \\ 4 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 2 & 0 & 0 \end{pmatrix}$$

Fig. 1: Example matrix

The matrix in Fig. 1 can be represented as:

Data = [4 1 1 2]
Row = [1 1 2 3]
Column = [0 3 0 1]

B. *Compressed Row Storage (CSR) format:*

This is the most widely used type of storage format, as it is both general and is very space efficient. It also includes three one-dimensional arrays- one for storing the data, one for storing the corresponding data's column index and the last one, a pointer array, that is used to store the count of non-zero elements in each row.

The space complexity of this format is given by:

$$S(CSR) = N \cdot S(n) + n \cdot S(N)$$

Where N is the total number of non-zeroes and n is the order of the matrix.For example, the matrix in Fig. 1 can be represented as:

Data = [4 1 1 2]
Column = [0 3 0 1]
Ptr = [0 0 2 3 4]

C. *Quadtree Compressed Row Storage (QCSR) format:*

This is a combination of the Quadtree format and the CSR format. Quadtree format [9] includes recursively dividing the given matrix into four quadrants until each quadrant is of a given size called the density. The quadrants can be classified into three categories:

- Empty node: If all the elements in the quadrant are zeroes.
- Mixed node: If there is a mix of zeroes and non-zero elements.
- Full node: If all the elements in the quadrant are non-zeroes.

For example, the matrix in Fig. 1 can be represented as follows:

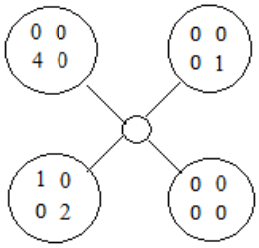

Fig. 2: Quadtree representation of matrix

In the Quadtree CSR format [8], the matrix is recursively divided into quadrants similar to the Quadtree format, and the mixed and the full nodes are stored in the CSR format. This format incurs a space overhead because of the requirement to store additional pointer values for the blocks, but gives very fast computation for sparse matrix vector multiplication. The maximum overhead over the CSR format is given by:

$$S_{oh}\,(QCSR) = (2S_r + S_l)\ xO(4^d\text{-}1)$$

Where, S_r is the space occupied by the index pointer to the region, s_l is the region length and d is the maximum depth of the tree.

D. *Minimal Quadtree format:*

This is a recently proposed storage format [7], which is used for efficient storage. Having recursively divided the matrix into quadrants using the Quadtree technique, we then produce a bit stream. We output a 0 if the quadrant is empty and 0 otherwise. The output bit stream is alone stored. This format can be used only in cases where the arrangement of the non-zero elements needs to be known, as we do not store the value of the non-zero elements themselves.The matrix in Fig. 1 for example will be encoded as 1110.

As the format completely depends upon the arrangement of the non-zero elements, the exact space complexity cannot be estimated. The lower bound is given by

$$S_{low}\ (MinQuad) = (4/3)\ N + \log_4 (n^2/N)$$

And the upper bound is given by:

$$S_{up}\ (MinQuad) = 4N\ (1/3 + \log_4(n^2/N))$$

Where, n is the order of the matrix, and N is the number of non-zero elements.

IV. RECTANGULAR SEGMENTATION

Rectangular segmentation technique overcomes the disadvantage of the quarter tree segmentation technique by allowing the sizes of the similar blocks to be of any size. It is a lossy image compression technique. In this technique, each pixel is taken one by one and is classified into image blocks, depending on the consistency condition. The consistency condition is a threshold condition that the pixel should satisfy, which is the maximum absolute value of the difference between each pixel and the average value. All the satisfying pixels are then put into one block.

Further compression can be obtained by converting this segmented image into a sparse matrix by keeping only the right-bottom pixel value as such, and replacing all the other elements in the blocks with zeroes. This will however lead to a certain amount of distortion in the image. Shengli Chen, et al [6] in the paper in which they have proposed this technique, have used the COO format for storing the sparse matrix obtained. We have implemented the same, but have used various other sparse matrix storage formats to store the segmented sparse image.

Taking the same 8x8 sample that they have considered in their paper [6], and storing them in various sparse matrix storage formats, we got the results shown in Table 1.

TABLE I. COMPARISON OF FILE SIZES WHEN SAMPLE BLOCK IS STORED USING VARIOUS STORAGE FORMATS

Matrix storage type	File size (Bytes)
Original	139
COO	87
CSR	71
QCSR	80
MinQuad	52

Clearly, we can obtain further compression by using other storage formats to store the final segmented sparse image.

V. EXPERIMENTAL ANALYSIS

We tested the space complexity of a set of benchmark sparse matrices obtained from the University of Florida's collection [12][13], when using various storage formats. Table 2 gives an overview about the matrices taken for analysis, and Table 3 gives the results obtained.

TABLE II. BENCHMARK MATRIX DETAILS

Benchmark Matrices	Number of rows	Number of columns	Number of non-zero elements
bcsstk04	420	420	4140
adder_dcop_08	1813	1813	11242
model6	2096	5289	27628
Ex12	3973	3973	42092
mhd4800b	4800	4800	16160
gemat11	4929	4929	33185
SiNa	5743	5743	102265
Na5	5832	5832	155731
Meg4	5860	5860	26324
Cell1_b	7055	7055	34855

TABLE III. SPACE COMPLEXITY OF SPARSE MATRICES WHEN STORED USING VARIOUS FORMATS

Benchmark Matrices	Original	COO	CSR	QCSR	MinQuad
bcsstk04	391 KB	97 KB	76 KB	118 KB	394 b
adder_dcop_08	6.35 MB	326 KB	176 KB	265 KB	999 b
model6	21.3 MB	456 KB	428 KB	922 KB	787 b
Ex12	30.4 MB	1.13 MB	635 KB	845 KB	349 b
mhd4800b	44 MB	436 KB	263 KB	416 KB	810 b
gemat11	46.5 MB	648 KB	506 KB	1.2 MB	367 b
SiNa	63.6 MB	2.95 MB	1.52 MB	3.19 MB	46 b
Na5	65.9 MB	4.54 MB	2.3 MB	4.25 MB	384 b
Meg4	65.7 MB	425 KB	386 KB	854 KB	739 b
Cell1_b	95.2 MB	587 KB	386 KB	854 KB	739 b

From the table above, it is evident that the Min-QCSR and the CSR formats have the lowest space complexity. Next, we took sample blocks of size 8x8 from images and compressed them using the rectangular segmentation technique, and then converted the resulting matrix into a sparse matrix which we stored using various storage formats. Table 4 shows the compression ratio of images when using sparse matrix storage techniques.

The MinQuad format again, gives the maximum compression ratio. This format however only stores the pattern of arrangement of the non-zero elements, and there is no way in which we can reconstruct the image using this storage model. Hence, the next efficient format, the CSR format, is the most suitable for storing segmented images when represented as sparse matrices. The QCSR format here shows higher compression ratios as the image block taken is small. As seen in the benchmark matrices, the space overhead increases when the size of the matrix increases.

The results also show that the higher the threshold, the greater the compression, as the number of blocks decreases, leading to lesser non-zero elements. But at the same time, the distortion increases as can be seen from the decreasing PSNR values. The PSNR values fall within an agreeable range, meaning there is not much distortion of the image. Hence the Rectangular Segmentation image compression technique, coupled with the CSR sparse matrix storage format with minimum values like 2,5 gives the best compression.

TABLE IV. THE COMPRESSION RATIOS OF IMAGES WHEN STORED USING VARIOUS SPARSE MATRIX STORAGE FORMATS WITH DIFFERENT THRESHOLD VALUES (Q), AND THE PSNR

Images	RSSMS with COO		RSSMS with CSR		RSSMS with QCSR		RSSMS with MinQuad		PSNR	
	Q=2	Q=5	Q=2	Q=5	Q=2	Q=5	Q=2	Q=5	Q=2	Q=5
Boat	1.56	2.99	1.82	3.63	2.88	5.34	5.19	7.79	49.5056	43.0044
Lena	1.98	5.21	2.48	5.3	3.53	7.95	6.11	11.35	49.1509	43.046
Goldhill	1.91	5.42	2.43	6.45	3.12	12.46	5.5	18.7	49.599	42.862

VI. CONCLUSION

From the results, it is evident that the MinQuad format has the least space complexity, as it is just a string of bits. For image compression though, it is unsuitable as it is impossible to recreate the image from this format. Hence the next space efficient and general storage format, the CSR format is the most suitable format for storing these images. The PSNR of the original and the reconstructed images is also high, proving there is very less distortion caused. Further, this method can be implemented for bigger low contrast images such as satellite images and medical images, for efficient storage with minimum loss of data. This can also be extended to store similar images.

REFERENCES

[1] Zhang Tianxu, Zeng Yonghui. A effective image compression algorithm based on sparse matrix .Wu Han: journal of Huazhong University of Science and Technology (Natural Science Edition), 34(2), 2006.

[2] Tanaya Guha, and Rabab K. Ward. Image Similarity Using Sparse Representation and Compression Distance, IEEE Transactions on Multiledia, Vol. 16, No. 4, June 2014.

[3] Yi-Chen Tsai; Ming-Sui Lee; Meiyin Shen; Kuo, C.-C.J., "A Quad-Tree Decomposition Approach to Cartoon Image Compression," Multimedia Signal Processing, 2006 IEEE 8th Workshop on , vol., no., pp.456,460, 3-6 Oct. 2006

[4] Kitty Arora; Manshi Shukla. A Comprehensive Review of Image Compression Techniques. (IJCSIT) International Journal of Computer Science and Information Technologies, Vol. 5 (2) , 2014, 1169-1172

[5] He Xing-heng and CHEN Hui.Method based on Quarter-tree method of Decomposition.

[6] Shengli Chen, Xiaoxin Cheng, Jiapin Xu" Research on Image Compression Algorithm based on Rectangle Segmentation and Storage with Sparse Matrix"IEEE,2012

[7] Simecek, I; Langr, D.; Tvrdik, P., "Minimal Quadtree Format for Compression of Sparse Matrices Storage," Symbolic and Numeric Algorithms for Scientific Computing (SYNASC), 2012 14th International Symposium on , vol., no., pp.359,364, 26-29 Sept. 2012

[8] Jilin Zhang, Enyi Liu, Jian Wan, Yongjian Ren, Miao Yue and Jue Wang,(2013). Implementing Sparse Matrix-Vector Multiplication with QCSR on GPU, International Journal on Applied Mathematics & Information Sciences,Volume 7, Mar. 2013, Pages 473-482.

[9] Ivan Simecek, (2009). Sparse Matrix Computations Using the Quadtree Storage Format, SYNASC '09 Proceedings of the 2009 11th International Symposium on Symbolic and Numeric Algorithms for Scientific Computing, Pages 168-173.

[10] Nathan Bell and Michael Garland, (2008). Efficient Sparse Matrix-Vector Multiplication on CUDA, NVIDIA Technical Report NVR-2008-004", December 2008.

[11] Saira Banu J, Vaideswaran , Sumithra S .Performance Analysis on Parallel Sparse Matrix Vector Multiplication Micro-Benchmark Using Dynami Instrumentation Pintool, ARTCOM 2013, Elsevier.

[12] T. Davis, The University of Florida sparse matrix collection[EB/OL], http://www.cise.ufl.edu/research/sparse/matrices,(1997).

[13] T. A. Davis and Y. F. Hu, "The University of Florida Sparse MatrixCollection," ACM Transactions on Mathematical Software, 2011.

[14] Guha, T.; Ward, R.K., "Image Similarity Using Sparse Representation and Compression Distance," Multimedia, IEEE Transactions on , vol.16, no.4, pp.980,987,June 2014 doi: 10.1109/TMM.2014.2306175.