

A New Technique of Lossless Image Compression using PPM-Tree

Shams Mahmood Imam, S. M. Rezaul Hoque, Mohammad Kabir Hossain, William Perrizo

Department of Computer Science and Engineering, North South University, Dhaka-1213, Bangladesh.

Department of Computer Science and Engineering, North South University, Dhaka-1213, Bangladesh.

Department of Computer Science and Engineering, North South University, Dhaka-1213, Bangladesh.

Department of Computer Science, North Dakota State University, Fargo, ND, USA.

shams_imam@hotmail.com, rezaok@hotmail.com, mkhossain@northsouth.edu, william.perrizo@mdsu.nodak.edu

Abstract

Digital Signal Processing in recent times is used to produce and process a large number of images. These require a large amount of space to be stored. Hence, image compression techniques are in high demand to reduce the amount of space taken up by images. The basis for image compression is to remove redundant or unimportant data. Of particular interest are the lossless image compression techniques that retain the original information in compact form and do not introduce any errors when the image files are decompressed. In this paper, we discuss such a lossless technique using a data structure that we name "Peano Pattern Mask Tree". It is an improvement over a previously discussed Lossless Image compression technique that uses the data structure – Peano Mask Tree.

Keywords: Coding Redundancy, Lossless Image Compression, Peano Pattern Mask Tree, Peano Tree.

I. INTRODUCTION

Image compression is important for many applications that involve data storage transmission and retrieval such as for multimedia, Internet teleconference, documents, medical imaging etc [1],[2]. The objective of image compression is to reduce redundancy of the image data in order to be able to store or transmit data in an efficient form. This results in the reduction of file size and allows more images to be stored in a given amount of disk or memory space.

General-purpose compression programs can be used to compress images, but the result is less than optimal. This is because images have certain statistical properties that can be exploited by algorithms designed to manipulate these features. These features include different types of redundancy; the difference between the shortest way one could convey a piece of information (i.e., the information content itself) and the data used to represent it. Two such types of redundancy are Coding Redundancy and Inter-pixel Redundancy. Coding Redundancy occurs when data used to represent an image are not utilized in an optimal manner. Inter-pixel Redundancy relates to the fact that adjacent pixel values tend to be highly correlated.

II. DIFFERENT TYPES OF IMAGE COMPRESSION

There are basically two types of Image Compression methods: *Lossy* and *Lossless* [3]. Lossy compression involves the loss of some information. Lossy schemes are capable of achieving much higher compression at the cost of accuracy. When the information from the compressed data using lossy technique is reconstructed, it introduces compression artifacts. In lossless compression schemes, the reconstructed image, after compression, is numerically identical to the original image. A perfect reproduction of the original image can be achieved using it. It is the preferred method for high value content such as medical images, legal papers or image scans made for archival purposes where any sort of degradation cannot be tolerated.

Our proposed technique discusses a variation of a P-Tree that is a quadrant-based lossless tree representation of the original spatial data.

III. INTRODUCTION TO PEANO TREES

A. The Peano Count Tree¹

P-trees are a lossless, compressed, and data-mining-ready data structure. This data structure has been successfully applied in data mining applications ranging from Classification and Clustering with K-Nearest Neighbor, to Classification with Decision Tree Induction, to Association Rule Mining [4]-[7].

The Peano Count Tree (P-tree) is a quadrant-based lossless tree representation of the original spatial data [8],[9]. The general concepts of P-trees are to recursively divide the entire spatial data into quadrants and record the count of 1-bits for each quadrant, thus forming a quadrant count tree. Using P-tree structure, all the count information can be calculated quickly. This facilitates efficient ways for data mining.

For example, a P-Tree for the given 8 x 8 image of single bits is explained below.

¹ Patents are pending for Peano Count Tree (P-tree)

0	0	0	0	1	0	1	0
0	0	0	0	1	0	1	0
0	0	1	1	1	0	1	0
0	0	1	1	1	0	1	0
1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	0
1	1	1	1	0	1	1	0

Fig. 1 8-bit by 8-bit image.

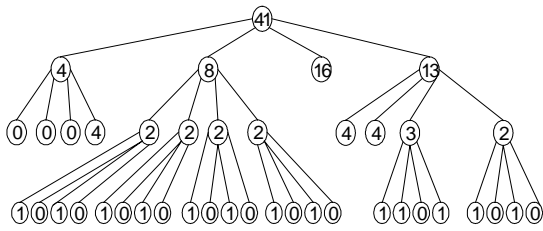


Fig. 2 P-Tree for data in Fig. 1.

In this example, 41 is the number of 1's in the entire image. This root level is labeled level 0. The numbers 4, 8, 16, and 13 found at the next level (level 1) are the 1-bit counts for the four major quadrants in raster order, or Z order (upper left, upper right, lower left, and lower right). This process of storing the count of the number of ones for each of the major quadrants is repeated. When quadrants are composed entirely of 1-bits (called pure-1 quadrants), sub-trees are not needed, and these branches terminate. Similarly, quadrants composed entirely of 0-bits are called pure-0 quadrants, which also cause termination. This pattern is continued recursively using the Peano, or Z-ordering (recursive raster ordering), of the four sub-quadrants at each new level. Eventually, every branch terminates (since, at the "leaf" level, all quadrants are pure).

The P-tree structure can be viewed as a data-mining-ready structure as it facilitates efficient ways for data mining. P-trees have the following features:

- P-trees contain 1-count for every quadrant of every dimension.
- The P-tree for any sub-quadrant at any level is simply the sub-tree rooted at that sub-quadrant.
- A P-tree leaf sequence (depth-first) is a partial run length compressed version of the original bit-band.
- Basic P-trees can be combined to reproduce the original data (P-trees are lossless representations).
- P-trees can be partially combined to produce upper and lower bounds on all quadrant counts.
- P-trees can be used to smooth data by bottom-up quadrant purification (bottom-up replacement of mixed counts with their closest pure counts).

B. The Peano Mask Tree

A variation of the P-tree data structure, the Peano Mask Tree (PM-tree), is a similar structure in which masks, rather than counts, are used. In a PM-tree, we use three-value logic to represent pure-1, pure-0, and mixed quadrants. (A 1 denotes pure-1; 0 denotes pure-0; and m denotes mixed.) The PM-tree for the previous example is also given in Figure 3. We can easily construct the original P-tree from its PM-tree by calculating the counts from the leaves to the root in a bottom-up fashion. A PM-tree is just an alternative implementation for a Peano Count tree [1].

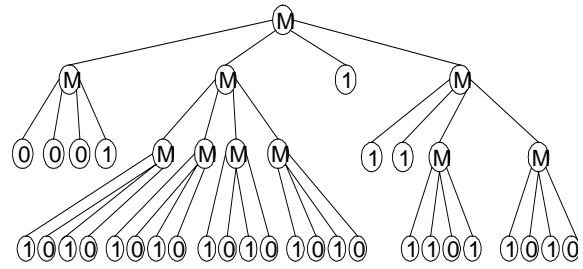


Fig. 3 PM Tree for data in Fig. 1.

C. Opportunities Unexplored in PM Tree

PM Tree was the data structure used for compression in the paper "Lossless Image Compression using P-tree". It is obvious that this method works best when neighborhood pixels share the same bit values with high probability. This is usually the case in higher-order bits for pixels in image data. Neighboring lower order bits have different values due to precision difference. PM Tree does not provide good compression, if any at all, for them. This happens as quadrants that are more mixed are introduced and recursive definition of data quadrants goes on till pure-1 or pure-0 quadrants are one bit long. The above-mentioned paper thus chose to use only the higher four bits to construct the PM Tree. However in instances where these higher order bits lack a high degree of correlation a similar problem to the one just described might occur.

PM Tree has three states; at least two bits are required to save the information for each node of the tree. Two bits are able to provide us with four combinations; hence, one combination remains unexploited in the PM Tree method. We intend to address the above-mentioned lacking in this paper and hence propose a new data structure, The Peano Pattern Mask Tree.

IV. THE PEANO PATTERN MASK TREE

The Peano Pattern Mask Tree (PPM Tree) is another variation of the P Tree. It shares many features of the PM Tree and addresses some unexplored area of the PM Tree. However, instead of using three-value logic, the PPM Tree uses four-value logic. We name this fourth logic as P, for pattern. This pattern is not fixed like pure-1 or pure-0, but rather assumes dynamic values

depending on the tree constructed. In fact, the pattern state is a special case of a mixed state.

An example will help us better understand the method. To construct a PPM Tree for any data set, we first need to construct a PM Tree for the same data set. Once the PM Tree is created, we search through the leaf level quadrants to look for the most frequently occurring four-bit combination. In the example for PM Tree shown above, we see that “1010” is the most frequently occurring quadrant at the leaf level. We make this our Pattern and change the original PM Tree into the following:

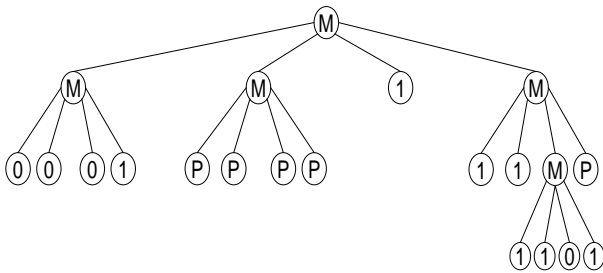


Fig. 4 First Stage of the PPM Tree for the PM Tree in Fig. 3

During the next stage, we search the tree to check whether any of the M nodes have as its children four P nodes. If a node possesses this property, we call it a “Pattern Parent.” We notice that only M nodes can have this property. All M nodes having the Pattern Parent property are transformed into a P node and all children removed. This search is carried out in a bottom-up manner where nodes at the lowest levels are transformed before any node at a higher level is transformed. As a result of this, we finally get the PPM Tree.

It can be seen that this tree has fewer nodes than the corresponding PM Tree and hence will require lesser space when we store it, if space required for each node in both trees is the same.

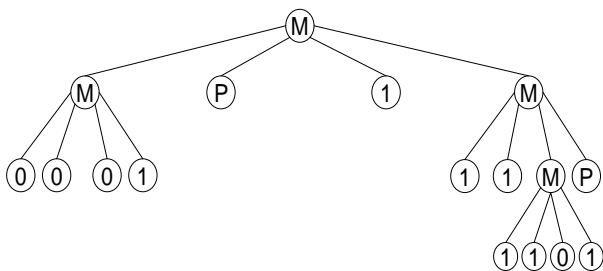


Fig. 5 The final PPM Tree of data in Fig. 1

It can also be noted that the “pattern” can be chosen dynamically, so that we can create the PPM tree with the fewest nodes. The breadth-first representation of the PPM tree above is: MMP1M000111MP1101.

V. PROPOSED SCHEME

Any image can be viewed as a two-dimensional array of pixels, with each pixel having various descriptive attributes. A BMP image contains descriptive attributes of

size three bytes, one each for Red, Blue and Green components. Each of these color components can take an intensity value in the range from 0 to 255. Thus, BMP images require 24 bits per pixel.

The nodes of a tree are stored in breadth-first order using the following encoding scheme:

- For mixed quadrant, store binary value 10
- For patterned quadrant, store binary value 01
- For pure-1 quadrant, store binary value 11
- For pure-2 quadrant, store binary value 00.

Since two bits are required to store the nodes of both PM and PPM trees, for any set of data the PPM tree will require lesser space than the PM tree.

Our initial compression scheme will deal with images that have size as exact powers of 2. Consider an n pixel image. In our proposed scheme, we first construct 24 n x n dimensional arrays using each of the 24 bits of each pixel. We then create a PPM tree for each of these arrays. If storing the PPM tree for the corresponding data set from which it was generated requires lesser space, the tree is stored or else the original data set is stored in raw form. We require a header file to keep track of which sets are stored in raw or compressed form. The header file format is described below:

Table I Proposed File Format

Length of original image as a power of 2 [1 byte]
Bits representing which segment are stored in raw or compressed form.[24 bits = 3 bytes]
Data for Red Component (8 Segments)
Data for Blue Component (8 Segments)
Data for Green Component (8 Segments)

1	0	1	0	1	0	1	0	0	1	1	1	1	0	0	0	0	0	0	0
0	1	1	1	1	1	1	1	0	0	1	1	1	1	1	0	0	1	1	

Fig. 6 Bit Representation of PPM Tree of Fig. 5

While storing the PPM tree, we store the pattern in the first four bits (represented in bold in the Figure 6) and then store the nodes of the tree in breadth-first order. Since we do this, we do not require storing a count of the number of nodes in the tree. While decoding the tree, decoding stops when all nodes are non-mixed nodes.

VI. EXPERIMENTAL RESULTS

A program was written for implementing the image compression scheme described above. It was compared to the compression scheme mentioned in [1]. The images used in this experiment can be found at: <http://www.invisionsoftworx.tk/articles.htm/>

Table II Experimental results

Image Name	Size KB	Size using method [1]	Compression ratio using method [1]	File Size Using Proposed method	Compression Ratio using proposed method
0220m	768	614.6	1.25	548.5	1.40
Aurora	768	411.1	1.87	304.5	2.52
Big35	768	664.3	1.16	614.4	1.25
Chem01	192	191.4	1.00	170.6	1.12
Colors	768	490.4	1.57	457.8	1.68
Game3	768	891.2	0.86	722.5	1.06
Infrared	768	453.6	1.69	293.1	2.62
S97_10540	768	557.5	1.38	530.9	1.45
W20p40K	768	446.2	1.72	296.7	2.59

In Table II we listed the experimental results showing the comparison between our proposed method and the method described in [1]. From Table II it can be noted that our proposed algorithm always performs better than method in [1]. Moreover we see in image 6 method in [1] increases the file size rather than compressing it whereas our proposed method gives a compression ratio of 1.06 for the same image. In fact, in our proposed method such contradictory situations never arise and in the worst case compression ratio is 1 (i.e. no compression).

Comparison Graph

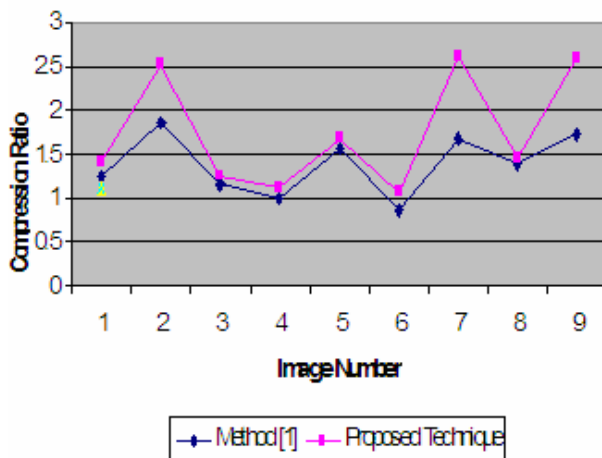


Fig 7 Comparison graph using data of Table II

VII. PROSPECTS OF THE COMPRESSION SCHEME

Basic p-trees can be constructed from the compressed file trivially after reconstruction of the PPM Tree. Once the basic p-trees have been created, we get data mining ready structure that facilitates efficient data mining tasks. The p-tree based decision tree induction method is significantly faster than existing classification methods [7]. P-tree data structure allows computing the Bayesian probability values efficiently. Bayesian classification with P-trees has been used successfully on remotely sensed image data to predict yield in precision agriculture [4]. Experimental results showed that using p-tree techniques in an efficient association rule-mining algorithm, P-ARM has significant improvement compared to FP-growth and A-priori algorithms. [4].

VIII. CONCLUSION

We are knowledgeable of the fact that we have presented here a scheme for compression of images that have dimensions as powers of two. But this limitation can be overcome using slight modification to the proposed scheme and file header structure. The proposed scheme is an improvement of compression using p-trees; hence, they can be used with slight modification in all applications that use p-trees. In addition, the PPM tree structure is also a data mining ready structure providing it an upper hand over other compression techniques when used in data mining applications.

IX. ACKNOWLEDGEMENT

Authors of this paper want to acknowledge Dr. William Perrizo, professor, Department of Computer Science, North Dakota State University, Fargo, ND, USA for his continuous support and help to accomplish this work.

REFERENCES

- [1] Fazle Rabbi, Mohammad Hossain, et. al. "Lossless Image Compression using P-tree," Proceedings of ICCIT 2003, Dhaka, Bangladesh.
- [2] Erickson B J, Manduca A, Persons K R, et. al. "Evaluation of irreversible compression of digitized posterior-anterior chest radiographs," *J Digit Imaging* 1997; 10(3): 97-102.
- [3] B. C. Vemuri, S. Sahni, F. Chen, C. Kapoor, C. Leonard, and J. Fitzsimmons "Lossless image compression," Available at <http://citeseer.nj.nec.com/559352.html>
- [4] Mohammad Hossain, et. al. "Bayesian Classification for Spatial Data Using P-tree," Proceedings of *IEEE INMIC 2004*, December 24-26, 2004 in Lahore, Pakistan.

- [5] M. K. Hossain and W. Perrizo, "Automatic fingerprint identification system using p-tree," Proceedings of *ICCIT 2002*, Dhaka, Bangladesh.
- [6] William Perrizo, William Jockheck, Amal Perera, "Multimedia Data Mining using P-Trees," Available at http://www-staff.it.uts.edu.au/~simeon/mdm_kdd2002/abstracts/14.html
- [7] Quang Ding, Qin Ding and William Perrizo. "Decision tree classification of spatial data streams using peano count trees," Proceedings of *ACM Symposium on Applied Computing (SAC '02)*, Madrid, Spain, March 2002, pp. 413-417.
- [8] H. Samet, "The Quadtree and related hierarchical data structure," *ACM Computing Survey*, 16, 2, 1984.
- [9] W. Perrizo, Peano count tree lab notes, Technical report NDSU-CSOR-TR-01-1, Fargo, ND, 2001.