APPLYING BINARY PARTITIONING TO WEIGHTED FINITE AUTOMATA FOR IMAGE COMPRESSION

Yang Kai, Ong Ghim Hwee

School of Computing, National University of Singapore
3 Science Drive 2, Singapore 117543

ABSTRACT

Fractal-based image compression techniques give efficient decoding time with primitive hardware requirements, which favors real-time communication purposes. One such technique, the Weighted Finite Automata (WFA) is studied on grayscale images. An improved image partitioning technique — the binary or bin-tree partitioning — is tested on the WFA encoding method. Experimental results show that binary partitioning consistently gives higher compression ratios than the conventional quad-tree partitioning method. Moreover, the ability to decode images progressively rendering finer and finer details can be used to display the image over a congested and loss-prone network such as the Image Transport Protocol (ITP) for the Internet, as well as to pave way for multi-layered error protection over an often unreliable networking environment such as the UDP.

1. INTRODUCTION

Data compression has been an important issue in relation to transmission and storage of information for a long time. In particular, digital images require a large amount of space in their internal representation often as arrays of pixels, incurring too much storage space and transmission time over a communication channel. Techniques for finding more compact image representations are desired. Also, general-purpose and selectively-reliable transport protocols have been developed to transmit compressed images over the Internet [1].

Many image compression techniques have been developed to suit different applications. Lossy image compression techniques can provide high compression ratios but introduce some annoying artifacts. Fractal-based image compression is one of the lossy data compression techniques that have been developed in the last decade. It makes use of self-similarities existing in an image at different resolutions under a set of affine transformations and exploits this kind of redundancy in order to alleviate the high cost of storage, thus achieving compression.

The WFA method attempts to represent a sub-image as a weighted sum of other sub-images to accomplish compression. WFA starts with an image to be processed. It locates sub-images that are identical or very similar to the entire image or to other sub-images, and subsequently constructs a graph that reflects the relationship between these sub-images *vis-à-vis* the entire image. The various components of the graph are compressed and the final product is a much more compactly represented domain.

An important issue involved in fractal-based image compression is the partitioning of an image into sub-images or blocks for encoding. There are various approaches to the use of different block shapes. One such commonly used approach is based on quad-tree partitioning. It has also been shown that a partitioning scheme based on rectangular blocks is simple yet offers high quality of decomposed images [2]. In this paper, we investigate a simple, new approach to WFA-based image compression with binary partitioning. In particular, we seek to improve the compression ratio by introducing intermediate steps into quad-tree partitioning. We will show that binary partitioning gives higher compression than quad-tree partitioning for images of 512×512 pixels.

2. WEIGHTED FINITE AUTOMATA FOR IMAGE COMPRESSION

First we introduce WFA as a modelling tool for specifying greyscale images, and then we describe inference algorithms in the next sections for their construction.

The WFA accepts grayscale pictures containing $2^m \times 2^m$ ($m \in Z^+$) pixels where pixel intensity ranges from 0 to 255. According to Culik and Kari [3, 4], a WFA specifying an image does not merely search for one of the known states that best matches the sub-state under investigation; but

Support for this work was provided by the NUS ARF, under grant R-252-000-118-112.

instead, it uses a linear combination of (possibly all) known states to arrive at a better approximation.

An m-state WFA A, over alphabet Σ is specified by:

1. A row vector $I^A \in R^{1\times m}$ (initial distribution),
2. A column vector $F^A \in R^{m\times 1}$ (final distribution),
3. Weight matrices $W_a^A \in R^{m\times m}$ for all $a \in \Sigma$.

To display WFA using diagrams, represent the m states by circles; the initial (first value) and final (second value) distribution being shown inside each state. If $(W_a)_{i,j} \neq 0$, place an edge from state i to state j, labeled by $a((W_a)_{i,j})$. For example, suppose that we have the initial distribution $I^A = (1 \quad 0)$, final distribution $F^A = \begin{pmatrix} \frac{1}{2} \\ 1 \end{pmatrix}$; weight matrices: $W_0 = \begin{pmatrix} \frac{1}{2} & 0 \\ 0 & 1 \end{pmatrix}$, $W_1 = \begin{pmatrix} \frac{1}{2} & \frac{1}{4} \\ 0 & 1 \end{pmatrix}$, $W_2 = \begin{pmatrix} \frac{1}{2} & \frac{1}{4} \\ 0 & 1 \end{pmatrix}$, $W_3 = \begin{pmatrix} \frac{1}{2} & \frac{1}{2} \\ 0 & 1 \end{pmatrix}$.

Each state represents a portion of the image and the edges attached to a state represent the linear combination of states which, when calculated, produce an approximation of the image portion. The diagram of such a WFA is given in Figure 1 below for illustration.

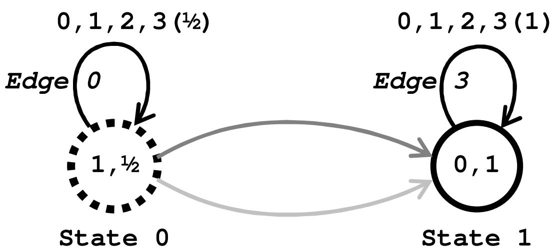

Figure 1. *Diagram representation of the WFA*

Given the WFA A over alphabet Σ shown in Figure 1, the initial distribution $I^A = (1 \quad 0)$ is obtained from the first value inside each state, ordered by the state number; similarly the final distribution $F^A = \begin{pmatrix} \frac{1}{2} \\ 1 \end{pmatrix}$ is obtained from the second value inside each state.

When an edge exists from state i to state j (i and j may be the same state), place the value enclosed in parenthesis in position (i, j) of matrix W_a^A where a is the label associated with the edge. For example, the bottom edge joining state 0 to state 1 has label "3"and a value of ½; in position (0,1) of weight matrix W_3^A we place the value ½. To visualize the actual image that results from this WFA, refer to Figure 2 shown in the following column.

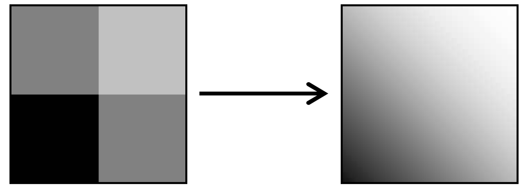

Figure 2. *Actual image being described by this WFA*

3. THE QUAD-TREE APPROACH TO WFA IMAGE ENCODING

A multi-resolution image is specified by assigning the grayscale value 0 to 255 to every node of the infinite quad-tree. If the outgoing edges of each node of the quad-tree are labeled 0, 1, 2, 3 we get a uniquely labeled path to every node; its label is called the address of the node. The address of a node at depth k is a string of length k over the alphabet Σ = {0,1,2,3}. Hence, a grayscale multi-resolution image can be specified as a subset of strings over the alphabet Σ. Regular sets of strings are specified by finite automata; therefore, finite automata can be used to specify regular multi-resolution images.

If a finite automaton A represents an image I, then each state of A must correspond to a sub-square of I, with the initial state corresponding to the entire I. Moreover, if there is a transition from state i to state j labeled by 0 (or 1 or 2 or 3), then the image corresponding to state j is the SW (or NW or SE or NE respectively) quadrant of the image corresponding to state i. An example below in Figure 3 shows the recursive zooming of the quad-tree.

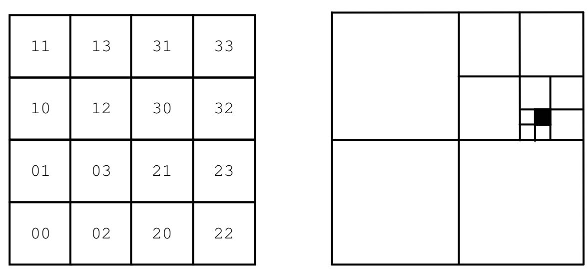

Figure 3*. The addresses of the quadrants of the 4×4 square, and the sub-square specified by 3203*

From the previous section we know that in order to arrive at a good approximation of any state image, a linear combination of some known states is employed. In mathematical terms, the crucial part of the WFA construction algorithm lies in representing a given vector by linear combinations of other known vectors. Since our goal is data compression, we have to approximate a vector with as few others as possible and as accurately as possible. Obviously these two requirements have to be balanced. The approximation with matching pursuit (MP) was introduced by Mallet and Zhang [5]. In this technique, an approximation is constructed step by step using a greedy strategy.

The MP approximation of a vector $\vec{v}$ is constructed by selecting the best matching vector from a given codebook. The component of this best matching vector is then subtracted from $\vec{v}$. The residue is now encoded in the same way as $\vec{v}$ and the process continues until a given abortion criterion is fulfilled.

Let $p, n \in N$ and $D = \{g_0, \dots, g_{p-1}\}$ with $g_i \in R^n$ ($i \in \{0, \dots, p-1\}$) be a set of $p \geq n$ non-zero vectors. The set D is often called the dictionary, codebook or domain pool. With span (D) we denote the set of all linear combinations of vectors in D. In order to make the following calculations a bit easier we assume that, without loss of generality, each vector in the dictionary has unit Euclidean norm $\|\vec{v}\|$. If span $(D) = R^n$ then D contains a set of n linearly independent vectors. The matching pursuit algorithm begins by projecting $\vec{v}$ on a dictionary vector g_{i_0} and computing the residue Rv by:

$$\vec{v} = \langle \vec{v}, g_{i_0} \rangle g_{i_0} + Rv$$

Since Rv is orthogonal to g_{i_0} (by virtue of the vector inner product), the following equation holds:

$$\|\vec{v}\|^2 = \left|\langle \vec{v}, g_{i_0} \rangle\right|^2 + \|Rv\|^2$$

Hence, we have to choose g_{i_0} such that $\left|\langle \vec{v}, g_{i_0} \rangle\right|$ is maximized, since residue $\|Rv\|$ has to be minimized. The next iteration continues with Rv instead of $\vec{v}$, and we compute the following iteration:

1. Set $R^0 v = \vec{v}$
2. Subsequent residues are computed by solving simultaneously:

$$\begin{cases} R^m v = \langle R^m v, g_{i_m} \rangle g_{i_m} + R^{m+1} v \\ \|R^m v\|^2 = \left|\langle R^m v, g_{i_m} \rangle\right|^2 + \|R^{m+1} v\|^2 \end{cases}$$

   By choosing g_{i_m} to maximize each of the $\left|\langle R^m v, g_{i_m} \rangle\right|$.
3. Finally, summing up the last equation in m from 0 to a stopping index $M-1$ yields:

$$\vec{v} = \sum_{m=0}^{M-1} \langle R^m v, g_{i_m} \rangle g_{i_m} + R^M v$$

4. THE BIN-TREE APPROACH TO WFA IMAGE ENCODING

The discussion so far has been under the assumption that the WFA is approximated by recursively dividing into 4 state image quadrants, each zooming to a higher level of detail. However, each of the quad-tree partition looks at only 25% effective area of the original state, much of the spatial correlations are lost in the division process.

Seeing in this light, we propose a modification of WFA image approximation using binary partitioning. The idea is to first divide the current (square) state q horizontally into 2 (rectangular) sub-states q'_0 and q'_1, and later divide q'_0 and q'_1 again vertically to produce 4 (square) sub-states q''_{00}, q''_{01}, q''_{10} and q''_{11} just as what a single division of the quad-tree partitioning would have done. This effectively separates the previous quad-tree partitioning into two steps: a horizontal cut followed by a vertical cut. The rational behind this separation is to better utilize the spatial correlations among image segments, by looking at 50% effective area. An example is given below in Figure 4. Instead of mechanically following the horizontal and then vertical cut to every (square) state image, a further enhancement tests both alternatives and picks the better choice. However, this involves some back-tracking.

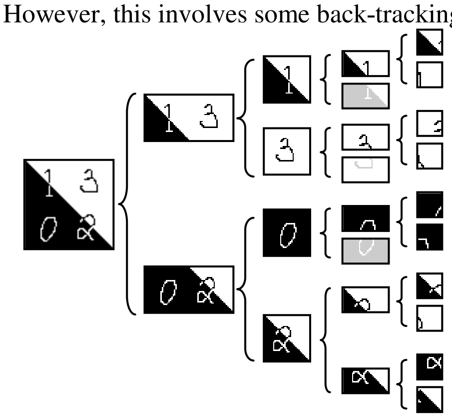

Figure 4. *This image is cut twice: horizontally followed by vertically (redundant sub-images are in pale shades)*

An outline of the WFA construction algorithm is presented below: Suppose that an image has been partitioned into a set of non-overlapping regions. Starting with the complete image, the current range image is recursively inspected if it can be approximated with a linear combination of arbitrary known sub-images. If a linear combination yields only a poor approximation, this range image will be bin-tree divided again and the recursion continues, while a new state image is appended to the WFA under construction. On the other hand, if this approximation already satisfies the given quality threshold, the corresponding transitions are appended to the WFA and the recursion terminates.

The main task of the image regeneration involves calculation of all the range images. This only requires some simple manipulation for each range image:

$$\psi^A = \sum_{i=1}^{M} w_i \cdot W_i^A$$

Where w_i are the (quantized) weights and W_i^A are the weight matrices for automaton *A* in the linear combination of *M* domain images. Every range image must not only be computed at its fixed size in the original image but also at any desired size in the linear combinations. Moreover, domain images that are referred to more than once are cached (stored in an array) to avoid multiple computations of the same image segment.

5. RESULTS AND DISCUSSIONS

In order to align our comparisons along with standard images used in the literature, we choose to benchmark the WFA encoder with three of the widely used grayscale images (512×512 pixels) — Boat, Goldhill and Lena. Rate-distortion curves are given below in Figure 5:

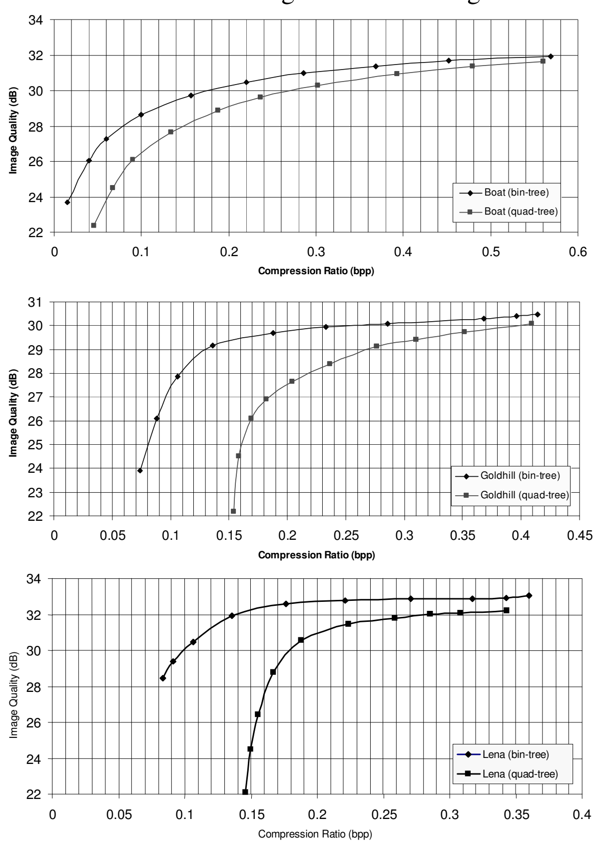

Figure 5. *Rate-distortion curves of image quality verses compression ratio (Boat, Goldhill and Lena)*

The platform used is Pentium III 1.4GHz 512MB memory running Windows XP Professional. The results obtained are comparable to those achieved by Hafner in [6]. We further observe the following for WFA encoding method:

1. Shape of the curves tends to be wide apart at low bits-per-pixel, and converge at high bits-per-pixel. This is evident for WFA since it uses a linear combination of known state images to approximate new ones; the bigger the domain pool, the higher the probability of finding a good approximation; no matter which partitioning method is used.
2. At the same PSNR ratio, bin-tree partitioning outperforms quad-tree partitioning by 10% to 25% reduction in storage; while at the same compression rate, bin-tree partitioning leads quad-tree partitioning by 0.5 dB to 1.5 dB.
3. Encoding time for all the three images is in the order of a few 100's of seconds, whereas the decoding time is in the order of several 100's of milliseconds.

6. CONCLUSION

In this paper, we have presented a binary partitioning method for the WFA-based image compression. A square image is split into rectangular and square blocks with binary partitioning of the image. Experimental results show that binary partitioning gives higher compression results than quad-tree partitioning for large images. The improvement can be as high as about 17% for 512×512 images. From the transmission perspective, the proposed binary partitioning approach to fractal image compression can be designed to decode partially received, out-of-order image data. This scheme can be customized to couple with the Image Transport Protocol for image transmission over loss-prone congested or wireless networks.

7. REFERENCES

[1] S. Raman, H. Balakrishnan, and M. Srinivasan. "An Image Transport Protocol for the Internet", International Conference 2000 on Network Protocols, Japan, November 2000.

[2] N. Ponomarenko et al. "Modified Horizontal Vertical Partition Scheme for Fractal Image Compression", 5th Nordic Signal Processing Symposium, Norway, October 2002.

[3] Karel Culik II and Jarkko Kari. "Image Compression Using Weighted Finite Automata", *Computer and Graphics 17*, Pages 305-313, 1993.

[4] Karel Culik II and Jarkko Kari. "Efficient Inference Algorithm for Weighted Finite Automata", Fractal Image Compression, Springer-Verlag, 1994.

[5] S. G. Mallet and Z. Zhang. "Matching Pursuits with Time-Frequency Dictionaries", *IEEE Signal Processing Vol. 41 No. 12*, December 1993.

[6] Ullrich Hafner, "Refining Image Compression with Weighted Finite Automata", *Data Compression Conference 1996*, Pages 359-368, March 1996.