

# AUTOMATED REASSEMBLY OF FRAGMENTED IMAGES

*Anandabrata Pal, Kulesh Shanmugasundaram, Nasir Memon*

Computer Science Department,  
Polytechnic University,  
Brooklyn, NY 110201.

## ABSTRACT

In this paper we address the problem of reassembly of images from a collection of their fragments. The image reassembly problem is formulated as a combinatorial optimization problem and image assembly is then done by finding an optimal ordering of fragments. We present implementation results showing that images can be reconstructed with high accuracy even when there are thousands of fragments and multiple images involved.

## 1. INTRODUCTION

Reassembly of objects from a collection of randomly mixed fragments is a problem that arises in several applied disciplines, such as forensics, archaeology, and failure analysis. This problem is well studied in these disciplines and several tools have been developed to automate the tedious reassembly process [5]. The digital forensic equivalent of the problem, which we call *reassembling fragmented documents*, however, has received little attention. In previous work, some of the authors of this paper, examined the reassembly of text files and binary executables from fragments [9]. In this paper we look at the problem of reassembling a collection of images from a scattered set of fragments.

Digital evidence in general and images in particular are easily scattered and a forensic analyst may come across scattered fragments of images in a variety of situations. Perhaps the most common situation is when analyzing a storage disk from a crime scene, a forensic analyst finds disk segments that correspond to fragments of previously deleted images. Although most file systems provide continuity of data on disk, in order to reduce file fragmentation, some older file systems (such as FAT), and highly active file systems, like that of a busy database server, will often fragment files into discontinuous blocks. Without adequate file table information it is difficult to put the fragments back together in their original order.

---

This work was supported by AFOSR Grant F49620-01-1-0243. Author email addresses are apal01@utopia.poly.edu, kulesh@cis.poly.edu, and memon@poly.edu, respectively.

Another example where image fragments may be found is the system swap file, which is one of the critical areas where lot of useful forensic information can be gathered. However, swap file state and addressing information is maintained in page-tables stored only in volatile memory. Without addressing information from the page-table it is difficult to rebuild contents off a swap file.

Image fragments could also be explicitly hidden in slack spaces of a filesystem. Criminals can modify a file hiding program to choose the blocks on which files are hidden based on a sequence of numbers generated using a password. Knowing the password they can reconstruct the original document, whereas a forensic analyst is left with randomly mixed fragments of a document which will need to be reassembled.

Finally, ubiquitous networking and growing adoption of peer-to-peer systems give anyone easy access to computers around the world. There are many peer-to-peer systems which enable users to store data on a network of computers for easy, reliable access anytime, anywhere. *Freenet*[2], *Gnutella*[3] and *M-o-o-t*[8] are some of the better known systems used by millions of users around the world. These systems are designed to provide reliable, distributed, and sometimes anonymous storage networks. A criminal can use these very systems to hide software tools, documents, and images that might be useful for his prosecution. Most peer-to-peer systems associate a unique key, either assigned by the user or generated automatically, with each document they store. Hence, a person can split a document into fragments and store each fragment in a peer-to-peer system using a sequence of secret phrases as keys, such that he can easily splice the fragments together knowing the proper sequence of secret phrases.

Typically, a document reassembly process will comprise of the following three steps :

1. **Preprocessing:** Encrypting or compressing digital evidence removes structural details that can assist an analyst in reassembling the evidence. During preprocessing, evidence has to be cryptanalyzed and transformed to its original form. For example, some cryp-

tographic schemes derive their keys based on user passwords. Since users tend to choose dictionary based passwords it is quite feasible to attack the password and obtain the key regardless of the size of the key. Besides, brute force attacks on cryptographic algorithms, such as DES, are shown to be feasible[1]. Note that a forensic analyst may not be too constrained on time, making cryptanalysis a feasible if not mandatory process.

2. **Collating:** Fragments found on a disk could belong to files of different types like text, binary, image and audio. Although, in this paper, we consider reassembling a single type of image (ie 24 bit color images), in reality evidence is usually a collection of mixed fragments of several types. To reassemble the evidence efficiently fragments that belong to a particular type of document could be grouped together. There could be different approaches used to effectively group similar fragments together. We defer a detailed study of such techniques to future work. Here we assume that we have a mixed collection of fragments from different images. However, it should be noted that we do not assume any knowledge of which fragment belongs to which image.
3. **Reassembling:** The final step in the process is to either reassemble a document to its original form or to provide enough information about the original form to reduce the work of a forensic analyst. Ideally, we would like to obtain the proper sequence of fragments that resembles the original document. Even if the process identifies a small number of potential orderings, that in itself would in considerable savings in time and effort to the analyst.

In this paper we focus on the final step and furthermore focus on the case when the underlying fragments are image fragments. That is, we look at reassembling a set of images given preprocessed fragments. The rest of this paper is organized as follows: in the next section we describe the problem formally and introduce a general technique for image reassembly. Section 3 presents initial experimental results and we conclude in section 4 with a discussion on future work.

## 2. THE REASSEMBLY PROBLEM

In this section we formulate the digital object reassembly problem in a more rigorous manner and describe a general approach for a solution to the problem.

### 2.1. Statement of the Problem

Suppose we have a set  $\{A_0, A_1 \dots A_n\}$  of fragments of a document  $A$ . We would like to compute a permutation  $\pi$  such that  $A = A_{\pi(0)} || A_{\pi(1)} || \dots A_{\pi(n)}$ , where  $||$  denotes the concatenation operator. In other words, we would like to determine the order in which fragments  $A_i$  need to be concatenated to yield the original document  $A$ . We assume fragments are recovered without loss of data, that is, concatenation of fragments in the proper order yields the original document intact.

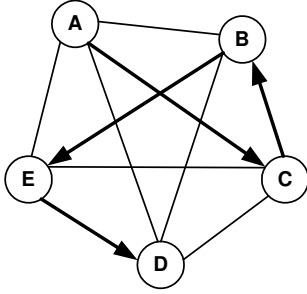
Note that in order to determine the correct fragment re-ordering, we need to identify fragment pairs that are adjacent in the original document. To quantify the likelihood of adjacency one may assign *candidate weights*  $C_{(i,j)}$ , representing the likelihood that fragment  $A_j$  follows  $A_i$ . When dealing with image fragments these weights are computed based on gradient analysis across the boundaries of each pair of fragments. Once these weights are assigned, the permutation of the fragments that leads to correct reassembly, among all possible permutations, is likely to maximize (or minimize) the sum of candidate weights of adjacent fragments. This observation gives us a technique to identify the correct reassembly with high probability. That is, we want to compute the permutation  $\pi$  such that the value

$$\sum_{i=0}^{n-1} C(\pi(i), \pi(i+1)) \quad (1)$$

is maximized (or minimized) over all possible permutations  $\pi$  of degree  $n$ . This permutation is most likely to be the one that leads to correct reconstruction of the document.

The problem of finding a permutation that maximizes the sum in equation (1) can also be abstracted as a graph problem if we take the set of all candidate weights ( $C$ ) to form an adjacency matrix of a complete graph of  $n$  vertices, where vertex  $i$  represents fragment  $i$  and the edge weight  $e_{ij}$  represent the likelihood of fragment  $j$  following fragment  $i$ . The proper sequence  $\pi$  is a path in this graph that traverses all the nodes and maximizes the sum of candidate weights along that path. The problem of finding this path is equivalent to finding a maximum weight Hamiltonian path in a complete graph (See Figure 1) and the optimum solution to the problem turns out to be intractable[4]. However there are many heuristics known in the literature and we employ one such heuristic as discussed in section 2.3.

It should be noted that the optimal solution may not necessarily result in reconstruction of the original. However, if candidate weights have been properly assigned, then the optimal solution should have a large number of fragments in or almost in the right place. Hence, it would be perhaps better for an automated image reassembly tool to present to the forensic analyst a small number of most likely reorderings, based on which the correct reordering can be manually



**Fig. 1.** A Complete Graph of Five Fragments & Hamiltonian Path (ACBED)

arrived at. The question that remains is how do we assign candidate weights for pair of fragments being adjacent, in an efficient and meaningful manner? We address this question in the next subsection.

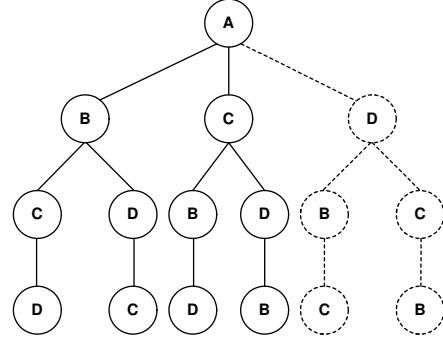
## 2.2. Assigning Adjacency Weights

Note that the header information of the image being re-assembled can be easily determined from the first fragment of every image in the collection, and from this information the width of the image in pixels obtained. Then, the basic approach for assigning candidate weights for a pair of fragments essentially involves examining pixel gradients that straddle the boundary formed when the two fragments are joined together. It is known an image consists mostly of smooth regions and the edges present have a structure that can often be captured by simple linear predictive techniques. Hence one way to assess the likelihood that two image fragments are indeed adjacent in the original image is to compute prediction errors based on some simple linear predictive techniques like one of this used in lossless JPEG or even better, the MED predictor used in JPEG-LS [6] and computing the absolute sum of prediction errors for the pixels along the boundary formed between the two fragments. That is, prediction errors are computed for pixels in the last row of the first fragment and the pixels in the first row of the second fragment. The number of pixels for which a prediction error is computed hence is equal to the width in pixels of the image.

## 2.3. $\alpha - \beta$ Pruning

Since we know the first fragment of each image we can represent all the paths in our weighted graph by a tree (See Figure 3). As we can see, the tree expands exponentially. Therefore we prune the tree to obtain a set of near optimal solutions. Our pruning method is adopted from  $\alpha - \beta$  pruning used in game theory[7].

Finding the optimal solution in this tree simply means examining each path in the tree looking for one that max-



**Fig. 2.** A Tree of Four Fragments with Path (A, D) Pruned

imizes the sum of candidate probabilities along that path. By pruning we try to avoid examining paths that we believe may not contribute enough to our solution. A naive approach to pruning the tree is to choose a node with maximum candidate probability at each level. We call this the *greedy* approach or greedy heuristic. However, this method can be extended to look not only at current level but also at  $\beta$  levels deep and choose a node at current level that maximizes the sum of candidate probabilities. In addition, instead of choosing a single node at each level, which limits our results to a single sequence, we choose  $\alpha$  best matches at each level resulting in  $\alpha$  best sequences.

## 2.4. The complete solution

Now we have all the pieces required to describe our approach to reassembling images given a collection of their fragments. We first examine all fragments and identify those that correspond to image headers having known formats. From these, the number of fragmented images and the width  $w$  of each fragmented image is determined. We then compute weights that represent the likelihood of adjacency for a given pair of fragments by computing the sum of absolute prediction errors across the ending  $w$  pixels of the first fragment to the starting  $w$  pixels of the second fragment. Repeating the process for all fragments results in a complete weighted and directed graph. We then use the  $\alpha - \beta$  pruning solution for computing maximum weight hamiltonian paths to compute a small number of near-optimal reorderings of the fragments. The actual reordering is likely to be contained in this set or at worst can be easily derived from this set by a forensic analyst.

## 3. IMPLEMENTATION & EXPERIMENTS

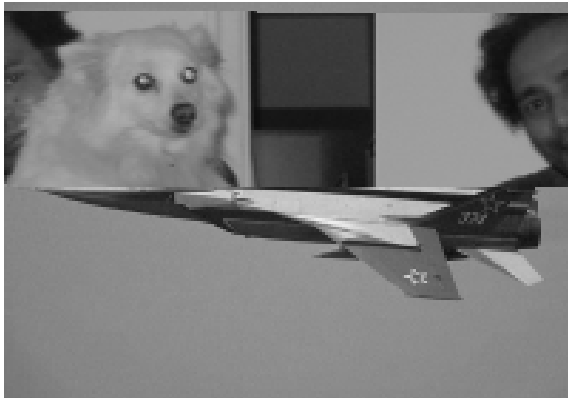
This section presents experimental results and discussion of the results. We used 24-bit color Windows bitmaps as the images in our experiments. Three datasets were chosen for experiments. All pictures used were saved or converted into

24-bit color bitmaps. The pictures within a dataset were then randomly fragmented together into 4K (4096 Byte) sizes. 4K sizes were used because it is the size of FAT32 clusters. Simple header checking code was able to determine the headers for each image in a dataset. Our experiments assumed that all image fragments were present and, therefore, that all images could be reconstructed in their entirety if the proper ordering was found.

The first data set used was a collection of 7 relatively non-related images. There were 4 images of fighter planes mixed in with 1 image of a woman, 1 image of a face and an image with a dog. All images but 1 of a fighter plane were reconstructed perfectly with our greedy algorithm. If however, we iterate and throw out the fragments that we were able to use to build an image successfully, then we were able to reconstruct the last fighter plane.

The second data set was a collection of 8 facial images in colour, and black and white that were roughly of the same dimensions. All the color images reconstructed perfectly but all three black and white images were garbled. One of the images reconstructed showed part of the face, and another image showed part of the same face from the previously mentioned image.

The third set of images were 4 images of picturesque nature background. All 4 pictures were reconstructed perfectly with the greedy algorithm.



(a) Improperly Reassembled Image with Greedy Heuristic



(b) Original Image - Dog. (c) Original Image - Fighter Plane

**Fig. 3.** Example of improper reassembly by greedy heuristic which can be rectified with  $\alpha - \beta$  pruning or by iterations.

#### 4. CONCLUSION

We have introduced and discussed a general procedure for automated reassembly of scattered image evidence. Experimental results show that even by using a simple greedy algorithm where the best candidate probabilities are used results in most images being reconstructed in their entirety even with a simple greedy heuristic. Even those images that are not reconstructed in their entirety tend to have a large number of fragments that are in the correct order. This is helpful because, if an analyst can identify proper subsequences in these candidate reorderings, they can combine these subsequences to form unit fragments and iterate the process to eventually converge on the proper reordering with much less effort than if they were to perform the task manually.

In future work we will implement the  $\alpha - \beta$  pruning heuristic and report results at the conference. We shall also investigate methods to collate fragments of documents from mixed fragments of several documents.

#### 5. REFERENCES

- [1] P. Cho and I. Hamer. DES cracking on the transmogripher 2a. *Cryptographic Hardware and Embedded Systems, LNCS 1717, Springer-Verlag.*, pages 13–24, 1999.
- [2] Freenet. <http://freenetproject.org/>.
- [3] Gnutella. <http://gnutella.wego.com/>.
- [4] C. E. Leiserson et. al. Introduction to algorithms. *MIT Press*, 2001.
- [5] Stolfi J. Leitao. A multi-scale method for the reassembly of fragmented objects. *Proc. British Machine Vision Conference - BMVC 2000*, 2:705–714, 2000.
- [6] N. Memon and R. Ansari. The JPEG Lossless Compression Standards. *Handbook of Image and Video Processing*. A. Bovik, Editor, Academic Press, 2000.
- [7] R. Moore and D. Knuth. An analysis of alpha-beta pruning. *Artificial Intelligence*, pages 293–326, 1975.
- [8] M o-o t. <http://www.m-o-o-t.org/>.
- [9] K. Shanmugasundaram and N. Memon. Automatic Reassembly of Document Fragments via Data Compression. Presented at the *2nd Digital Forensics Research Workshop*, Syracuse, July 2002.