A Review of Graph Theoretic and Weightage Techniques in File Carving

SARI ALI SARI1, KAMARUDDIN MALIK MOHAMAD1

1Faculty of Computer Science and Information Technology Universiti Tun Hussein Onn Malaysia, 86400 Parit Raja, Batu Pahat, Johor, Malaysia.

alsarisari46@gmail.com

Abstract. Digital Forensics is a platform that helps in assisting investigation carried out in computer crimes through the recovery of material that is found in digital devices. Material is recovered through a method known as file carving that helps to recover data from storage media. Moreover, it also retrieves the extract hidden, overwritten or deleted files. File carving enables the file recovery without knowledge about contextual information such as file system metadata. Due to recent expansion in the research of information technology, file carving has become an essential technique for general data recovery and digital forensics investigations. Therefore, considering the important role played by file carving the present study conducted a survey of the previous literature to find out various data carving techniques. Results, from the previous literature helped to provide several algorithms for image construction which are based on graph theoretic techniques. Furthermore, the present study classifies the file carving techniques for JPEG file into two categories namely basic carving technique and advanced carving technique. The paper concludes by providing limitations in the present study which could be helpful for future investigations associated with the fragmentation problem of data files.

Keywords: File carving, graph theoretic, weightage techniques, reconstruction, metadata, statistical technique.

1. Introduction

File carving approach used for discovery and reconstruction of file in case the system metadata is corrupted, missing, or otherwise unreliable. This approach involves extracting data from larger data set. It recovers the file from raw data on the basis of specific characteristic of file format provide by the data. The conventional recovery techniques are relatively fast as involving only the file system reading process. The carving approach is mostly used in unallocated space files. This is the area where no information of any metadata referring it is present in the file system.

File carving approach was initiated by the ‘Defense Computer Forensics Lab’ (DCFL) that produced ‘CarvThis’, which is a carving program. Then, Agent Kriss Kendall, following by Agent Jesse Kornblum introduced ‘Foremost’ that is an open source carving tool and afterward extended by Mikus [1] which implemented a module with specific knowledge of ‘Microsoft OLE’. Later, Richard and Roussev [2] introduced another tool named ‘Scalpel’ which is advanced version of foremost in order to enhance performance and decrease the memory usage. ‘LibCarvPath’ and ‘CarvFS’ were developed by Dutch National Police Agency are the virtual file systems that provide zero-storage carving possibilities. Numerous techniques are used in operating systems[3] to efficiently allocate blocks to add data to an existing file or to create a new file. The operating system starts the process by
searching for consecutive blocks, but if the blocks are not available, the file will be stored in two or more locations known as fragmented files or partial files. Fragmentation is caused due to continuous adding more data to a file and low disk space. Hence, the crucial part in file carving is in handling fragmented files.

The conventional recovery techniques were unable to recover a fragmented file especially for those files whose metadata entry is not present. Metadata entry consists of link to blocks that are allocated to the file. Meanwhile, a file carving enables to retrieve files that are fragmented and stored on multiple locations since the techniques of file carving analyses block or a set of blocks based on the characteristics of a specific file format and/or its contents.

2. Overview of JPEG

The JPEG’s term originated from an acronym of the Joint Photographic Experts Group’s committee which formed a standard defined on ISO / IEC 10918 and describes a codec which is responsible for compressing an image into a byte stream and decompressing it into a viewable image. The detail of JPEG is discussed in the next subsection.

2.1. JPEG File structure

JPEG file has two structures known as JFIF and Exif. Generally, JFIF is found in the internet applications whereas Exif is usually found in digital camera images. A comprehensive explanation for these structures are discussed in the next subsection.

2.1.1. JFIF

The “JPEG Interchange Format” (JIF) is a standard that specifies the container used to store the encoded image. Nevertheless, it is rarely used and later developed and extensively used “JPEG File Interchange Format” (JFIF) [4]. JFIF provides more specifics than JIF. Moreover, compared with JIF it easier to implement [5]. JFIF is a format which allows JPEG bitstreams to exchange between a various platforms and applications. Hamilton, E [6], pointed out that the APP0 marker is embedded in the JFIF structure for emerging requirements as well as maintain the JFIF in terms of compatibility with the standard JPEG format interchange. A JPEG file format that applies the JFIF standard consists of a header signature that starts with the SOI and ends with the EOI [7].

2.1.2. Exif

Exchangeable Image Format (EXIF) was developed by Electronics and Association 2016. The specification of the Exif image file requires the image data recording method in the files, and states the following items below:

- Tags used by this standard
- Structure of image data files.
- Definition and management of versions of the format.

The structure of Exif is sown in Figure 1.
2.2. JPEG Marker

Basically, JPEG files consist of two-byte segmentation codes known as markers representing multiple structural sections in the compressed data formats [8]. The start marker of most marker segments contains related group of parameters and numerous markers are stand-alone markers. The examples of stand-alone markers are Start of Image (SOI), Restart Interval Termination (RST) and End of Image (EOI), whereas the examples of related group of parameters markers including Defining Huffman Coding Table (DHT), Defining Arithmetic Coding Table (DAC), Defining Quantization Table (DQT), Defining Hierarchical Progression (DHP), Start of Frame (SOF), Application Segment (APP), Expand Reference Component (ERC) which is reserved for JPEG extensions (JPG) and Comment (COM). More specific on these markers is tabulated in Table 1.

![Figure 1. Basic structure of Exif File](image)

| Code Assignment | Symbol | Description                           |
|-----------------|--------|---------------------------------------|
| **Start of Frame markers, non-differential, Huffman coding** |
| xFFC0           | SOF0   | Baseline DCT                          |
| xFFC1           | SOF1   | Extended sequential DCT               |
| xFFC2           | SOF2   | Progressive DCT                       |
| xFFC3           | SOF3   | Lossless (sequential)                 |
| **Start of Frame markers, differential, Huffman coding** |
| xFFC4           | SOF5   | Differential sequential DCT           |
| xFFC6           | SOF6   | Differential progressive DCT          |
| xFFC7           | SOF7   | Differential lossless (sequential)    |
| **Start of Frame markers, non-differential, arithmetic coding** |
| xFFC8           | JPG    | Reserved for JPEG extensions          |
| xFFC9           | SOF9   | Extended sequential DCT               |
| xFFCA           | SOF10  | Progressive DCT                       |
| xFFCB           | SOF11  | Lossless (sequential)                 |
| **Start of Frame markers, differential, arithmetic coding** |
| xFFCD           | SOF13  | Differential sequential DCT           |
| xFFCE           | SOF14  | Differential progressive DCT          |
| xFFCF           | SOF15  | Differential lossless (Sequential)    |
3. File Structure

Two types of file structures are available in the dataset known as non-fragmented and fragmented file structures. The file structure is said to be non-fragmented when all part of data for the file is in the same or adjacent cluster. Meanwhile, for fragmented file structure is formed when part of data for the file is separate into non-adjacent and different cluster from the other part. These two file structures types are further discussed in detail in the following subsection.

3.1. Non-Fragmented File

The primary focus of file carving is to recover contiguous files [2]. During scanning disk images, it easy to identify when the file has different headers (start of file) and footers (end of file). In the contiguous file case, the available file content between the header and the footer could be extracted as a new file. Nevertheless, the files distributions which arranged in the disk becomes more complex when the files are added, modified and deleted which tend to be fragmented as well. Consequently, contiguous file carving is found to be less efficient since it may result in many false positives become the recovered file is supposed to be intact but actually it does not contain valid data [9].

The example of contiguous file is illustrated in Figure 2. From Figure 2, File1 utilize two clusters and both of the clusters are adjacent to each other. Similar case with File3 where three clusters are contiguous.

![Figure 2. Contiguous files](image-url)
3.2. Fragmented File

Several file carving tools are available however, when it comes to recover fragmented few tools are present. For those files that are contiguous, normal signature-based carving techniques. These techniques work well if the file use headers and footers.

In trying to recover a file if a cluster from header to footer is extracted and consider it as a single file, but in fact it is fragmented file, it will result in incorrect reconstruction of that file [10, 11]. Fragmentation is found to be rare at present file systems, hence, carving fragmented file is important which has not been addressed by current carving tools. The reason of fragmentation to be found relatively less addressed is due to that fact that the investigations that are carried out in the files of interest in digital forensic are mostly fragmented. According to Garfinkel [11], Since, current operating systems are attempting to write files without fragmentation, as they are read and write very quickly. A file must be written in two or more fragments by the operating system. These conditions are listed below.

- The complete file usually split into two or more fragments because if a drive in use is full and there are files that are added or deleted in this drive in less random order over a period of time will have no contiguous region to hold file without fragmentation.
- If the existing file has appended data and the media may not has enough space to place the data in the same location with the file, then the data and the file are directly have to be placed elsewhere.
- Several file systems itself not support writing files in the media in contiguous manner since it depends on the size of file.

The file is said to be fragmented when the complete file is not stored in the correct sequence on consequence blocks in the disk. It means, if a file is fragmented, the arrangement of the blocks from the start to the end of file will lead the file being incorrectly reconstructed [12]. There are two types of fragmented file known as bi-fragmented and multiple fragmented. Bi-fragmented file is a condition of splitting a file into two parts as shown in Figure 3. Meanwhile, as shown in Figure 4, multiple fragmented is a condition in which a file is split more than two parts.

![Figure 3. Bi-Fragmented files](image)

![Figure 4. Multiple fragmented files](image)

Pal and Memon [13] highlighted basic definitions on some terms regarding the fragmentation as listed in table 2.

| File1a | File2 | File1b |
|-------|-------|-------|
| Cluster1 | Cluster2 | Cluster3 |

| File1a | File2 | File1b | File1c |
|-------|-------|-------|-------|
| Cluster1 | Cluster2 | Cluster3 | Cluster4 |

Table 2. The terms in fragmentation.
Kloet [14] stated that the fragmented file has two categories, namely linear and non-linear fragmentation:

- Linear fragmentation occurs when a file is segmented into two or more parts and the parts are arranged in their original order in the dataset. For example, there are two files shown in Figure 5, which splits F1 into two fragments (F1(1/2) and F1(2/2)).
- Non-linear fragmentation occurs when the different parts of file in the dataset are different order compared to the original file as illustrated in figure 6.

There are some issues need to be considered based on linear and non-linear fragmentation.

- JPEG files is fragmented with other file types such as PDF, Excel and Word [15]. An example of this issue is shown in Figure 7 (a) for linear fragmentation and Figure 7 (b) for non-linear fragmentation.

- Bi-fragmentation JPEG file is fragmented with multi-fragmentation JPEG file [16]. The example of this issue is shown in Figure 8 (a) for Linear fragmentation and Non-Linear fragmentation respectively.
Figure 8 (a). Linear fragmentation

Figure 8 (b). Non-Linear fragmentation

- JPEG files intertwined with another JPEG files [7]. Figure 9(a) and Figure 9(b) provides an example of linear fragmentation and non-linear fragmentation respectively for this issue

Figure 9(a). Linear fragmentation

Figure 9(b). Non-Linear fragmentation

- JPEG file fragmented with missing fragmentation [13]. An example of this is given in Figure 10 for (a) linear fragmentation and Figure 10 for (b) non-linear fragmentation.

Figure 10(a). Linear fragmentation

Figure 10(b). Non-Linear fragmentation

- JPEG file fragmented with a missing header [17]. An example of this is shown in Figure 11 (a) for linear fragmentation and Figure 11 (b) for non-linear fragmentation.

Figure 11(a). Linear fragmentation

Figure 11(b). Non-Linear fragmentation

- JPEG file fragmented with a gap between fragments [11]. An example for this issue is provided in Figure 12 for (a) linear fragmentation and Figure 12 (b) for non-linear fragmentation.

Figure 12(a). Linear fragmentation

Figure 12(b). Non-Linear fragmentation

4. File Carving Categories
Over the years, a lot of file carving techniques that have been evolved. In this research, we classified them into two categories, namely basic carving technique and advanced carving technique reviewed in the following subsection [13].

4.1. Basic Carving Technique
Generally, file carving does not directly access files using file system information in order to recover files. Conversely, basic carving techniques are based on knowledge of the structure of the files. The following sub-section reviewed the most common basic carving technique.

4.1.1. Signature-based carving
The signature-based carving consists of two types including header-footer and the header- “maximum file size” carving. The header-footer carving is a very well-known carving technique and the most frequently used as it is an extremely basic file carving technique. This technique recovers files based on searching for the unique byte sequence for header and footer. The header generally found at the beginning of the file and then searching the footer for the first occurrence in the file. The header and footer locations which are found in the data during searching process are then marked. Afterwards, store all data blocks within these two locations in an output file [14]. This technique assumes that the value of header and footer are not tempered or damaged. It also assumes that the file is not fragmented which means that the file is stored in contiguous data blocks [18, 19].

In the case of file types with only have header information without footer information is called header- “maximum file size” carving. Basically, the carving process for this technique is similar with the header- footer carving. During searching process, the start of the file is determined based on the location of the header which is found in the data and subsequently, end of the file is determined by calculation of the file size. The size value of several file formats is identified in the file header, otherwise an intelligent file size guess is created based on general observations of a given file format. The data blocks between the locations of start of the file and end of the file are stored in an output file after start and end of the file location have been identified. Moreover, it is very important to choose an appropriate size for header- “maximum file size” carving, if it is not provided in the header. On the other hand, the recovered file may consist of either too many unrelated information appended at the end or might lose related information as well. This type of signature-based carving was developed for file formats such as JPEG file type which any unrelated information appended at the end are not taken into account.

Both of those signature-based carving types work only if the deleted files are not fragmented and their signatures itself are not damaged or corrupted. Although those carving techniques are quite fast, however they still produced many false positives. This is because if the header and footer values found in the data are too short although when they actually not represent a file, they can appear in the data set [14]. For example, only two bytes occupied in the header and footer value of JPEG filetypes and the chances of these values being repeated are high. Therefore, it may lead to found many similar files in the output that are not really belongs to JPEG filetypes[20].

4.1.2. Content-based Carving
Generally, the main concept of content-based carving is to read every single block in the dataset and then analyze its content in order to determine whether it belongs to a certain file. If necessary, these blocks are then blocks are then rearranged in order to construct the original file. There are two principles in content-based carvings. The first principle is a block can contain only a single file and the second principle is the fragmentation might be occurred at block boundaries [14]. Further step is analyze every block and employ calculation in determining which file belongs to this block. One of the techniques is to calculate the entropy of every block. The blocks belonging to a specific file will have a certain range of entropy value [14]. After that, these blocks need to be rearranged in order to construct original file[20].

4.1.3. Structure-based carving
Structure based carving utilized more internal file structure information in order to reduce the file positives. The process is start by determining the particular level of the file format information. In order to recognize a file, this information is then matched in the raw data set. Since a lot of file format information are being employed, the structure-based carving is significantly reduce the false positives.
However, similar with the signature based carving, it cannot handle fragmented files [14]. This structure based carving also known as “deep carving” or “semantic carving” [20].

4.2. Advanced carving techniques

Meanwhile, for more advanced carving techniques, it not just use the files structure knowledge yet it also uses the individual files contents to recover the data. The advanced carving techniques provided in the next subsection.

4.2.1. Graph theoretic-based Carving

Basically, graph theoretic based carving has been introduced for text-based media and digital image by Shanmugasundram[21] and pal[22] respectively. The approach proposed by Shanmugasundram[21] for text-based media is assign probabilities of candidate for their adjacency in order to recover the fragmentation files. For text-documents, these probabilities of candidate can be determined by utilizing the sliding-window algorithm that evaluates the statistics on the use of symbol in a language or, for generic data, is on the basis of statistical models utilized for data compression. The possibility that assigned to the fragments are utilized in determining the permutation which maximizes the sum of candidate probabilities of fragments neighborhood. This mathematical problem is equivalent in finding a maximum weight in a whole graph of Hamiltonian path [23]. In order to provide the best solution since this problem become intractable, heuristics approach have been introduced. The approach that has been proposed by Pal [22] assigns probabilities candidate to the file fragments of digital images. Then, put all together by utilizing different graph theoretical algorithm, for example an adapted version Shortest Path First (SPF) algorithm that produced the best results for seven dataset of images[24]. More details regarding the approaches in graph theoretic are discussed in the next subsection.

A. Hamiltonian Path Problem

Considered unallocated clusters set as \( b_0, b_1, \ldots, b_n \) is belongs to document D, then, calculate a permutation \( \prod \) of this set which indicates the original structure of document. The next step is determining the order of correct cluster, by identify the pairs of fragment which are adjacent in the original document. Therefore, between clusters \( b_x \) and \( b_y \), the candidate weights \( W_{x,y} \) is assigned which indicates that the cluster \( b_y \) following cluster \( b_x \). After assigned these weights, the clusters permutation which leads to correct reassembly, between all possible permutations, is likely to maximize the sum of adjacent cluster candidate weights. Hence, this finding provides a technique in identifying the correct reassembly with high probability. Thus, the sum of ordering of candidate’s weights is maximize over each possible permutation \( \prod \) of degree \( n \).

This permutation is most likely to result in the document being properly reconstructed. This can be abstracted as a graph problem as well if assumes that the set of all candidate weights \( W \) to form in an adjacency matrix manner of a complete graph of \( n \) vertices, where vertex \( x \) indicates as cluster \( b_x \). While the edge weight \( W_{x,y} \) indicates as the likelihood of cluster \( b_x \) following cluster \( b_y \). In this graph, the correct sequence \( \prod \) is a path which traverse each vertices and maximizes the sum of candidates weight through that path. Therefore, finding this path is equivalent with finding a maximum weight of Hamiltonian path problem in a complete graph [13].

B. K-Vertex Disjoint Path Problem

The main issue with the Hamiltonian path problem is that in the real systems, multiple files are fragmented together have been ignored and the effectiveness of file recovery can be improved when the statistic of multiple files are being considered. This issue is then refined by Pal et al. with K-vertex disjoint path problem [22]. Consider that in the unallocated space, each cluster as vertex in a graph and the edge between vertices to be weighted based on the likelihood that each cluster follows another cluster, then the reassembly problem is equivalent with finding K-vertex disjoint path, where \( K \) represents as the number of files that are identified by their headers. It is considered as disjoint path problem because every cluster in the disk is belong to one and only one file. Moreover, the K-vertex
disjoint path problem is known for being nondeterministic polynomial-time (NP) hard problem.

Afterwards, there has been extensive researches on the use of greedy heuristic algorithm in order to overcome the edge and vertex disjoint problem [25] [26] Nevertheless, this paper provide several algorithms which unnecessarily caused disjoint paths. The algorithms demonstrated that create vertex disjoint path are known as unique path (UP) algorithms. For example, every fragment in the disk is assigned to one and only one image. However, there is issue in UP algorithms that is wrongly assigned a fragment to image A which supposed to be belongs to image B. This will effect the image A and B being incorrectly reconstructed[22].

i. Shortest Path First algorithm
Shortest first path (SPF) algorithm assumes that the lowest average path cost is the best recoveries. The average path cost is derived from the sum of the weights between the recovered file clusters divided the numbers of clusters. The SPF algorithm reconstructs every image one at a time. In addition, the average path costs are only calculated after an image is reconstructed and the clusters which belongs to the image are not deleted. All clusters in the image reconstruction still available for the reconstruction of the rest of the images. This process is being repeated until average path costs for the entire image are calculated. Besides the image which has the lowest path cost is considered as the best recovery and the clusters which belongs to its reconstructions are deleted. Every remaining image used in the deleted clusters need to redo their reassemblies process with the remaining clusters and produced their new average path cost. After complete this process for the remaining images the one which has the lowest average path cost is removed, and this process is stop when entire images in the disk are recovered [13]. This algorithm is then modified by Memon and Pal which focused on the greedy heuristic and named if as greedy SPF UP[22], then they improved this algorithm namely enhanced greedy SPF UP [22].

ii. Non-Unique Path algorithm
Memon and Pal, stated that the Non-unique path (NUP) is a reassemble algorithm which can be used in graph-based file carvers[22]. This algorithm searching the beast match to a chunk and immediately append it. The advantage of this algorithm is that there is no error propagation exist due to wrongly assign a fragment to an image. On the other hand, the disadvantage of this algorithm is that if the fragment is used more than once in images reconstruction process, there will be an error. Pal and Memon then proposed the sequence of this algorithm using greedy heuristic and named it as greedy NUP[22]. Based on NUP algorithm, any fragment, excluded header fragment, which was selected in the image reconstruction will be available for selection in another image reassembly. This can prevent the error from being propagate. But as stated earlier, it is unnecessary lead to disjoint path. Subsequently, another sequence of this algorithm was proposed known as enhanced greedy NUP[22].

iii. Sequential Unique Path algorithm
The Sequential Unique Path (SUP) algorithm is similar with the NUP algorithm, but the difference is in one file only can has a chunk. This shows that the reconstruction is based on which file has been previously reconstructed.

In 2006, Memon and Pal improved this algorithm by utilized greedy heuristic and become greedy SUP[22]. After a fragment is assigned by the algorithm to an image reconstruction, this fragment is inaccessible for being selected in other image reconstruction. Even though this will produce disjoint paths of vertex, but the issue is the paths depend on the arrangement of processed images. Afterward is the detail of SUP algorithm.

First, any arrangement of processing images is randomly selected. But then, any modification on the arrangement of processing images may affected the reassembly result. Let \( P_i \) indicates as the reconstruction path of image \( i \) and \( h_i \) indicates as header fragment for image \( i \). In the beginning of the process, the header \( h_i \) is selected in the reconstruction path as the first fragment (for example, \( P_i = h_i \)). Then, the current fragment is set to be equal to the header, \( s = h_i \). After that, the best \( s \) need to be find in the greedy match \( i \). The best available match is \( s \)'s best \( i \) which not been utilized in other image.
reassembly. Next, $i$ is placed in the reconstruction path $P_i = P_i t$ and set it as current fragment $s = t$ for processing purpose. Once its best match has been found, the process is repeated until image is complete reconstructed. When the image has been reconstructed, move to the next image and the process are being repeated until all $k$ images are reassembled. Then they improved this algorithm namely enhanced greedy SUP [22].

iv. Parallel Unique Path (PUP)
The parallel unique path (PUP) algorithm is a variant of Dijkstra’s algorithm which finds the shortest path from a single source [27]. This algorithm is used to recover the images simultaneously. First, the process start with the headers of image, the algorithm selects the best match cluster for every header within available clusters. The edge weights determine the best match. The best cluster $b_x$ is selected from the best match which has been listed and assigned to the header $b_h k$ of the image. Then, this cluster is deleted from the set of available clusters. The best match of the cluster $b_x$ is then identified and being compared with the best matches of other remaining images. This process is repeated until every image are recovered [13]. This algorithm was improved in 2006 by Memon and Pal based on the greedy heuristic known as greedy PUP [22]. Greedy PUP forms reconstructions of the UP without having reconstructions which depend on the arrangement of the images that being reconstructed. Later they proposed another improvement on this algorithm called enhanced greedy PUP [22].

4.2.2. Bifragment Gap Carving
Bifragment Gap Carving (BGC) is introduced by Garfinkle to address fragmentation scenario [11]. The term “bifragment” is referred to a file that is split into two fragments. A technique for fast object validation has been proposed by Garfinkle for bifragmented files recovery where it only performs for files with known header and footer. Furthermore, this technique requires decoding which is the process of converting information of a file into its original format which is very exhausted process as it requires each block to be decoded. This process intended to determine whether the block is belongs to the addressed file. For cases where the two fragments are close against each other, the BGC perform very well. However, according to Pal & Memon BGC fails to scale when the fragmented files have large gaps and also when the files have more than two fragments. Moreover, even the validation/decoding process achieved successful result, it does not guarantee the correct recovery of file. Besides, it performs well with files with possess validated/decoded via their structure. The clusters which corrupted or missing will always produce worst result [13].

![Maximum Possible Gap – Six Clusters](image)

**Figure 13.** The example of BGC algorithm

Figure 13 shows a file named “recovery JPEG” which has two fragments. The base fragment is start from cluster 300 and ending with the cluster 303. While the next fragment, the starting cluster is 306 and ends at 307. BGC would find the correct recovery by utilized the fast object validation when the gap size is two. While for clusters 300 to 303 and clusters 306 to 307 produced successful recovery of file.

4.2.3. Smart Carving
Pal et al. [12] identified the issue inherent with their previous researches into recover of randomly
fragmented images by employing the pre-computed weights value and provide the framework and results for a carver which included the behavior of standard fragmentation in a disk and one that scales to huge disks[13].

The Smart Carver file has three key component as illustrated in Figure 14 which can handle fragmented and non-fragmented data within a various of digital devices and also file systems. The pre-processing is the first phase for this file caver. In this phase, the file system data cluster are using decompressed or decrypted as needed. The second phase is collation phase, where the clusters of data being classified based on a type of file and for the next attempt is optional in determining whether they part of a particular file. The reassembly phase is the final phase for this file carver, where all clusters identified and merged in the collation phase are being pieced together in order to reconstruct the file[13].

![Figure 14](image)

**Figure 14.** Smart carving with three components which are pre-processing, collation and reassembly. The summary of file carving categories is tabulated in Table 3.

**Table 3:** Summarization of file carving Categories
5. Weigatage Technique

The weights value must be assigned between the fragments in order to recreate a path in image reconstruction. This value describes the likelihood how well they fit together. Given that $H_i$ indicates as a header fragments of $i$ and $F_j$ indicates as a non-header fragments $j$, then the weight is computed by employing $W(H_i, F_j)$. The weights calculation of two non-header fragments cannot be perform in the JPEG file format. This is due to the lack of important information for the decoding such as entropy coding tables or quantization tables which lead the data cannot be analysed. Moreover, when a file consists of more than two fragments, $H_i$ specifies the header fragments and the identified fragments are belonging to this header fragment. A comparison is considered success when $H_i$ and $F_j$ belong to each other and the arrangement of all data within those fragments are correctly in place. In addition, this only works for JPEG file type format as not all file types formats are depends on their context like JPEG file type format. Meanwhile, the candidate weight between ant two fragments can be evaluated by using various techniques which are elucidated in the following sub-section[22, 28].

5.1. Histogram intersection

This approach compares the distribution of the image values between two segments of image. The more similar between two images are, the difference is less in the histogram. This make evaluation of candidate weight more easy. The difference supposed to be minimal when the fragments belong to each other. The main issue in this algorithm is when the data is missing in between, the second fragment is switched which has influence to its image colour and positioning. Another issue of this algorithm is when the fragment is very small for example only one cluster. It is impossible to produce pattern matching correctly because of the small amount of data cannot decoded correctly[28].

| No. | Techniques | Base Carving Technique Category | Comments |
|-----|------------|----------------------------------|----------|
| 1   | Signature-based carving | header/footers | Both types of signature-based carving only works when the deleted files are not fragmented and their signatures are not corrupted. |
| 2   | Content-based carving | a. Rate of change (ROC) | It is not always precise enough to give an automated carving tool. Finding the calculations which can distinguish a block of data that belongs in a file and one that does not. |
|     |             | b. Entropy | |
|     |             | c. Byte frequency cross-correlation (BFC) | |
| 3   | Structure-based carving | It is only use internal file structure information in order to reduce the false positive. Can not handle fragmented files. |

| Advance Carving Technique Category | | |
|-----------------------------------|---|---|
| 1 | Graph theoretic-based carving | A. Hamiltonian Path Problem | It does not take into account that in real systems multiple files are fragmented together and that recovery can be improved if the statistics of multiple files are taken into account Pal et al[8] |
|     | B. K-vertex disjoint path problem | |
|     | i. Shortest Path Fast (SPF) | Afer an image is reconstructed the clusters assigned to the image are not removed, only the average path cost is calculated |
|     | ii. Non-Unique Path algorithm (NUP) | These will be an error of a fragment is used more than once in the reconstruction of images. |
|     | iii. Sequential Unique Path algorithm (SUP) | When the algorithm assigns a fragment to an image reconstruction, the fragment will be unavailable for selection in the reconstruction of any other images. Though this creates vertex disjoint paths, the problem is that the paths depend on the order of images being processed. |
|     | iv. parallel unique path algorithm (PUP) | The best fragments being matched with the current set of fragments may be better matches for fragments that have not been processed as yet, thus leading to error propagation again. |
| 2 | Bifragment Gap Carving | It does not scale for files fragmented with large gaps and for more than two fragments. The successful of decoding/validating process does not guarantee the correct recovery of files. It only works with files that have a structure that can be validated/decoded. Missing or corrupted clusters will result in the worst case often. |
| 3 | Smart Carving | As soon as the files are fragmented, the forensic results of these applications become unsatisfactory. |
5.2. Huffman decoding
The Huffman decoding cannot be employed to evaluate a weight, but can be helpful in rejecting fragments quickly. This Huffman decoding is usually compressed the stored image data. In order to produce a correct decompresion, the correct Huffman table is required which is stored in the header fragment. Considering that the JPEG files consist of different Huffman tables that are incompatible with each other, but they can be used in data validation for the second fragment. The Huffman symbols in the second fragment should be comply with the stored Huffman table in first fragment. Moreover, a Huffman decoder needs to be adapted or implemented in this approach [28].

5.3. Pixel matching (PM)
The Pixel Matching (PM) is the most simple approach where the total values of pixel matching within the edges of size width, w for two fragments are summed. From Figure 15, the value of width w is equal to 5 and the PM will compare the every numbered pixel in fragment i with every numbered pixel in fragment j either matched or not. The weight value of is increased by one when the pixel matched. For PM approach, if the value of weight is higher, the assumed match is better[22].

![Figure 15](image)

**Figure 15.** The values of pixel are compared to calculate the candidate weights between two fragments.

1) Median Edge Detection (MED)
Every single pixel of fragment is predicted based on the value of the pixel above, to the left and left diagonals to it[29]. By utilizing Median Edge Detection (MED), the absolute value of the difference between the output of predicted value within fragment and the actual value is summed. For MED, the assumed match is better when the weight value is lower[22].

2) Sum of Differences (SOD)
The Sum of Differences (SOD) is computed within the RGB pixel values of the edge from each fragment with another each fragment. Based on the Figure 5.1, the SOD will sum up the absolute value of the difference between every single numbered pixel within fragment i over the same numbered pixel within fragment j. For this approach, the better the assumed match when it produce lower value of weight[22].
3) Euclidean Distance (ED)

Euclidean Distance (ED) is similarly measured across the boundary from the pixels RGB values. The lower value of ED means similarity score is higher. ED focuses on the similarity of color between the adjacent boundary pixels instead of the smoothness and the integrity of the entire image.

5.4. Pattern matching

Pattern matching analyzes similar with the histogram intersection which the data of image fragments is decoded. The distinct structures are identified which spread over the comparing fragments. The algorithm determines either the comparative fragment complies or not to the identified geometrical constraints in the second fragment. The graphical analysis of image fragments is assumed to produce in a high computational effort[28].

6. Conclusion

The current study has highlighted the importance of file carving in digital forensic to carry out investigation using different techniques and approaches to recover files. Used metadata and statistical techniques. Furthermore, the paper highlighted popular carving techniques and reconstruction algorithms to examine the reconstruction effectiveness to recovery various file types formats. Results of the survey revealed that previous studies used graph theoretic approaches especially weightage technique in handling huge amount of data systematically for multiple fragmentation file.

References

[1] Mikus, N., An analysis of disc carving techniques. 2005, NAVAL POSTGRADUATE SCHOOL MONTEREY CA DEPT OF COMPUTER SCIENCE.

[2] Richard III, G.G. and V. Roussev. Scalpel: A Frugal, High Performance File Carver. in DFRWS. 2005.

[3] Maqsood, M., S.A. Lashari, M.A. Saare, S.A. Sari, Y.M. Hussein, and H.O. Hatem. Minimization Response Time Task scheduling Algorithm. in IOP Conference Series: Materials Science and Engineering. 2019. IOP Publishing.

[4] Hamilton, E., JPEG file interchange format. 2004.

[5] Mullan, P., C. Riess, and F.J.D.I. Freiling. Forensic source identification using JPEG image headers: The case of smartphones. 2019. 28: p. S68-S76.

[6] Hamilton, E., JPEG File Interchange Format. 1992.

[7] Abdullah, N.A., K.M. Mohamad, R. Ibrahim, M.M.I.J.o.D.C. Deris, and Forensics, X_myKarve: Non-Contiguous JPEG File Carver. 2016. 8(3): p. 63-84.

[8] Telegraph, I. and T.C. Committee, CCITT Recommendation T.81: Terminal Equipment and Protocols for Telematic Services : Information Technology - Digital Compression and Coding of Continuous-tone Still Images - Requirements and Guidelines. 1993: International Telecommunication Union.

[9] Shi, K., M. Xu, H. Jin, T. Qiao, X. Yang, N. Zheng, J. Xu, and K.-K.R.J.D.I. Choo, A novel file carving algorithm for National Marine Electronics Association (NMEA) logs in GPS forensics. 2017. 23: p. 11-21.

[10] Ravi, A., T.R. Kumar, and A.R. Mathew. A method for carving fragmented document and image files. in 2016 International Conference on Advances in Human Machine Interaction (HMI). 2016. IEEE.

[11] Garfinkel, S.L.J.d.i., Carving contiguous and fragmented files with fast object validation. 2007. 4: p. 2-12.

[12] Pal, A., H.T. Sencar, and N.J.d.i. Memon, Detecting file fragmentation point using sequential hypothesis testing. 2008. 5: p. S2-S13.

[13] Pal, A. and N.J.I.s.p.m. Memon, The evolution of file carving. 2009. 26(2): p. 59-71.

[14] Kloet, S.J.A., Niederlane: Eindhoven University of Technology, Measuring and improving the quality of file carving methods. 2007: p. 4-79.
[15] Mohamad, K.M., A. Patel, T. Herawan, and M.M.J.o.D.F.P. Deris, myKarve: JPEG image and thumbnail carver. 2010. 3(2-4): p. 74-97.
[16] Pal, A., H.T. Sencar, and N. Memon, Detecting file fragmentation point using sequential hypothesis testing, digital investigation, 2008. 5: p. S2-S13.
[17] Uzun, E. and H.s.T. Sencar, Carving orphaned JPEG file fragments. IEEE Transactions on Information Forensics and Security, 2015. 10(8): p. 1549-1563.
[18] Beek, C.J.W.p.M., Introduction to file carving. 2011.
[19] Merola, A.J.S.I., Data carving concepts. 2008. 4.
[20] Nadeem Ashraf, M., Forensic Multimedia File Carving. 2013.
[21] Shanmugasundaram, K. and N. Memon. Automatic reassembly of document fragments via data compression. in 2nd Digital Forensics Research Workshop, Syracuse. 2002.
[22] Memon, N. and A.J.t.o.i.p. Pal, Automated reassembly of file fragmented images using greedy algorithms. 2006. 15(2): p. 385-393.
[23] Thing, V.L., T.-W. Chua, and M.-L. Cheong. Design of a digital forensics evidence reconstruction system for complex and obscure fragmented file carving. in 2011 Seventh International Conference on Computational Intelligence and Security. 2011. IEEE.
[24] Deshmukh, A., M.J.J.o.S. Desai, and Research, A Survey on Corrupted Video Recovery Using CODEC Specifications. 2014. 3(12).
[25] Carmi, P., T. Erlebach, and Y. Okamoto. Greedy edge-disjoint paths in complete graphs. in International Workshop on Graph-Theoretic Concepts in Computer Science. 2003. Springer.
[26] Kolliopoulos, S.G. and C. Stein. Approximating disjoint-path problems using greedy algorithms and packing integer programs. in International Conference on Integer Programming and Combinatorial Optimization. 1998. Springer.
[27] Dijkstra, E.W.J.N.m., A note on two problems in connexion with graphs. 1959. 1(1): p. 269-271.
[28] Schildendorfer, B., Carving fragmented JPEG images Content-based file carving of non-contiguously fragmented JPEG images. 2012, Department for Information Security at the University of Applied Science St. Poelten.
[29] Martucci, S.A. Reversible compression of HDTV images using median adaptive prediction and arithmetic coding. in IEEE international symposium on circuits and systems. 1990. IEEE.