An Algorithm for Faster Keyword Detection on a Forensic Image

Katie Millar
Cybersecurity and Forensics
De Montfort University

Dinesh Mothi, PhD
Course Leader
Ravensbourne University

ABSTRACT
An algorithm that has been developed for keyword searching on forensic images is something that isn't widely dispersed within the forensic community, due to minimal research and literature being carried out and applied within this topic area. This research aims to build upon this by developing and enhancing the efficiency of keyword search through the creation of an algorithm.This has been tested against forensic image formats with the implementation of a scientific method. With the approach demonstrating the efficiency of an algorithm which uses parallel keyword searching to locate a keyword, this has been implemented within the algorithm that has been have created. The results collected are then cross compared against the timings of open source software tools that are developed for the keyword searching of a forensic image.

Keywords
Keyword detection, forensic image, algorithms.

1. INTRODUCTION
Recent studies have been conducted within the last three years examining a variety of algorithms that have been comprised to allow for effective keyword searches on a variety of different platforms, these focusing extensively on two key topic areas with one being Cloud Computing. The research undertaken by Song et al (2017) enabled the keyword searching over encrypted data, using a database structure known as a bloom filter-based tree index to show the relationship between the query and encrypted documents. The combination of these two variables will allow for users to search over encrypted cloud data. However, an additional study conducted during the same year (Miao et al, 2017) produced a cryptographic primitive algorithm using ciphertext-policy and attribute-based encryption (CP-ABE) techniques. Where the authors argued that a single keyword search will result in various unwanted results and unauthorised users will still be able to have access to data with outdated keys. Therefore, instead of grouping just one keyword, two improvement schemes were implemented to facilitate multiple keyword searches and revocation of users. Within a journal article written by Yan et al (2015), multiple algorithms were used to allow for the keyword searching over a Q system, this was achieved through a variety of clustering techniques to narrow down the results of the query inputted. Not only resulting in this process being time consuming, but lengthy if applied to multiple evidence files. Their technique however allowed for both the searching of keywords executed by the user, alongside the generation of feedback for query answers outputted. This proposed strategy was designed to produce keywords of "*high relevance",* and *"answers which are highly informative",* in turn making the relationship between the search query and the keyword critique being more beneficial to the user. The proposed methodology concentrates on the clustering of keywords within joint trees, to allow the user to only see the highest ranked words within their inputted query. The "*top-k"* results which are presented to the user are among the selected few that provide feedback to their inputted keyword, this is turn lowering the amount of keyword attributes outputted and feedback received.

2. RELATED WORK
The forensic field can be disseminated into multiple topic areas, where journals and other forms of publications have tried to experiment with multiple methodologies to provide a reliable outcome to the user. However, key elements within their research fail to tailor to the forensic field, due to their application revolving around databases, clustering, and systems and cloud computing. One research method that provided a minor insight into the keyword searching times within different file formats was the blog post written by DME Forensics (2014), where it was concluded that a .dd forensic image searched for the keyword 42% faster than .E01. This methodology used no algorithm for their research process and started the keyword search from the beginning of the first sector. The compression setting '6' was applied to both images, however segmented image files were not taken into consideration.

The research that involved the implementation of algorithms used a variety of different steps within their research to achieve the output. However, this was partly beneficial for achieving a reliable and sometimes accurate result for what the user inputted, but time constraints weren't taken into consideration when developing and testing this method. As if this were to be applied to the forensic field large quantities of data would need to be searched, and the forensic examiner may not be able to implement all the steps needed (example shown in Figure 1.1) to provide this result due to this being an extensive process.

Other algorithms implemented only allow for these to be used on specific file types, for example encrypted files. Therefore, if some of these algorithms were applied within a digital forensic field these algorithms might not be able to support the additional file types e.g. non-encrypted files and might not be compatible. As well as this certain proposal's only allow for one keyword per search, when dealing with multiple digital source's this cannot be applied. As a case may involve multiple data sources for example both a 60GB and 160GB hard drive, and the use of this algorithm will only allow for one keyword to be searched per time. This can increase the time taken to analyse the data, as multiple keywords cannot be searched per time lowering the performance.

It is evident that there is a gap within the research which has been conducted previously, due to minimal application within the forensic field. This is turn showing that its application to different topic areas other than databases, Cloud Computing and specified systems wouldn't be foreseeable. This is down to the numerous process requirements that enable the user to receive their desired output, and this in turn when applied to the forensic field, particularly within the keyword searching of large quantities of data will not perform as effectively and

efficiently as the research suggests. This is the major gap which has been demonstrated through authors research, due to the algorithms only being applied involving numerous steps and multiple algorithms to achieve the result. Additionally, most of the search seem to incorporate similar algorithms involving the clustering and categorizing of keyword. Yet minimal development has been done to incorporate a solution which can be used on multiple sources and involves minimal step-by-step application by the user.

Upon reflection of the information found within the literature review a conclusion can be formed on the limitations stated by the authors from the research they've conducted. It is highlighted throughout that there are many strengths for the implementation of algorithms to narrow down the scope of keywords and categorise these into relevant sections. However, a downside to this is that multiple algorithms will need to be incorporated to show this outcome. This is something the authors themselves believe to be a downside to the application of this methodology, due to the various procedures and guidelines needing to be followed within it, and some only allowing for certain text files to be examined.

Although the literature was minimal with regards to the application of this on forensic images, one technique that will be considered on the creation of the algorithm is the parallel processing of the forensic image. It was concluded in in a recent journal written by Golov (2017) that the implementation of a parallel process increases the amount of data that can be analysed in a shorter amount of time. Therefore, the application of this to a forensic image will increase its performance significantly based upon the results found within the research.

On a whole, the forensic aspect is something which is poorly addressed, and is highly overlooked, as the creation of an algorithm that incorporates both time efficiency and effectiveness would be highly valuable within this field and could be applied within other associated disciplines.

Table 1.2 Findings gathered from related works

Forensic tools/Technique	**Findings**	**Has a study been conducting comparing the keyword searching times using these tools?**
Encase	Logical and Physical keyword searches	No
FTK Imager	DT Search	Yes- DME forensics
Autopsy	Keyword search ingest mode	No
Grep	GNU extensions e.g. Nondeterministic Reverse Grep Pattern Matches	No
SearchBin	Plain text string and smaller binary file searches Wild Cards to present multiple outcomes to a keyword search	No
Forensic tools/Technique	**Findings**	**Purpose of Use**
Text Clustering	**Text mining** frameworks and patters using text clusterin Document **Clustering** Algorithms Scalability issues Multiple Sampling Techniques Boolean Operators Statistical-Computer assisted structure	Cloud Computing Textual documents (in one format) Q Systems
Comparison of imaging timings	DME Keyword search comparison between .E01 and .DD using a keyword search	Keyword Searching
Tree Structures	Grouping of Keywords together **Clustering** **Text Mining**	Databases/Keyword Searching
BANK systems	Rooted trees **Clustering**	Keyword Searching/Databases

3. METHOD

The experiment will involve a Windows 10 Operating System and an Intel(R) Core i7-2670QM CPU@2.20GHz processor**,** this processor is manually setup to process using 4 cores and 8 threads. The tool that will be used to examine the data will be Superdetective, coded by my supervisor Dinesh Mothi, the code that will be executed is written incorporating my algorithm. The data will then be inputted and analysed within a Microsoft Excel Spreadsheet using a scatter graph and table structure to form conclusions.

The experimental group will be executed using Superdetective and data will be quantitatively gathered on the dependent variable. The number of threads will then be inputted from 1-8, where the independent variable will include the examination on 2 experimental groups. These consisting of 4 Cores 4 threads, and 4 Cores 8 threads, to enable the cross examining of keyword searching times on the different number of threads. The controlled variable consists of the keyword "Verisimilitude", and will be searched for parallelly on each thread. The results for both 4 cores 4 threads and 4 cores 8 threads will be inputted and analysed within a graph on Microsoft Excel.

The controlled group will not involve the execution of Superdetective, but instead will involve Searchbin.py, Autopsy and dtsearch. These are all keyword searching software, and will be used in order to see whether Superdetective can locate the keyword at the end of the sector in a quicker time than the keyword searching tools. This will involve the searching of the 64GB USB image from the starting sector to the end sector. The results of this will then also be analysed using Microsoft Excel.

Before finalising this hypothesis, we had additional hypothesis' which were altered to fit the conclusion of our analysis, the reader may refer to Appendix 3 in order to form an understanding of the initial starting point of the investigation and how we came to formulate this new hypothesis: Initially as the number of threads increases the time taken to detect the keyword decreases, and reaches a minimum value. Once the minimum value is reached, the time taken to detect the keyword increases with the increase in the number of threads (directly proportional).

The aim is to determine the underlying relationship between the number of threads (independent) and time taken (dependent) to search for a keyword (control) on a forensic image.

Using this algorithm, it will aid in implementing a keyword search, the algorithm I have derived below has been implemented within a software called Superdetective.exe which was created by my supervisor Dinesh Mothi. This algorithm allows for the parallel processing of the image whilst doing a keyword search, enabling a certain number of threads to be allocated within each of the 4 cores. I foresee that this should slow down the keyword searching time of a forensic image, due to diving this image into separate chunks allowing for this to be found quicker.

Algorithm implemented:

1) Divide the image into K equal blocks or chunks

   $(C_1, C_2, C_3 \ldots C_n)$ such that $n = G + C_2 + C_3 \ldots + C_n + C_n$

   $N = \sum_{e=1}^{n} Ci$

2) Let $C_1 = (Sa_1, Sa_n)$; $C_2 = (Sb_1, Sb_n)$; $C_3 = (Sc_1, Sc_n)$; $C_4 = (Sd_1, Sd_n)$

   Sa_1 = Starting sector of C_1
   Sa_2 = Starting sector of C_2
   Sa_3 = Starting sector of C_3
   Sa_4 = Starting sector of C_4

3) Say the keyword to be searched for is located in one of the sectors of
   $[C_1, C4]$

   Keyword = (Sd_1, Sd_n)

4) To perform a keyword search at the beginning of the starting sector of each chunk and for the search to stop at the end of the chunk for example,C_1 would start from Sa_1 and end at Sa_n, therefore the keyword search must be carried out in the domain of sectors respective to a given chunk e.g.
   $K_w \Rightarrow (Sa_1, Sa_n)$ on $Sa_1 < K_w < Sa_n$

   K_w = Keyword

5) If the keyword search on every chunk is done as mentioned in step 4, then a keyword search should be detected quickly

   $TS_{kw} = f(S_f, S_i, N_c, F_c)$

   TS_{kw} = Time taken to find keyword
   S_f = Size of the forensic image
   S_i = Sector No. location
   N_c = No. of cores
   F_c = Frequency of the core

6) Let the keyword be present in the y^{th} sector, if a keyword search is performed from the starting sector of the forensic image it will take TS_{kw} minutes to detect the keyword y

n.

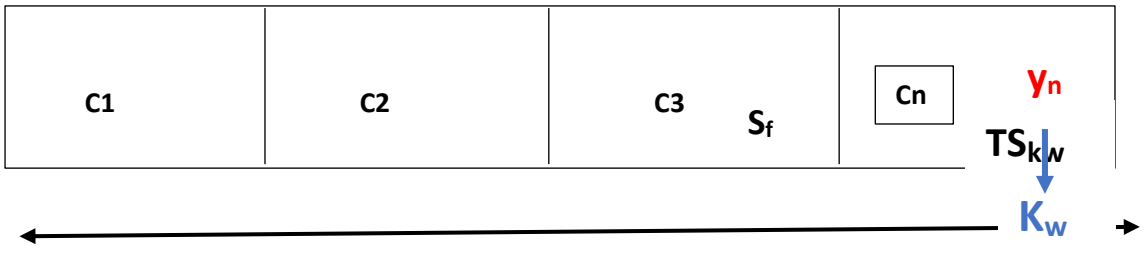

Figure 1: Dividing the regions into location chunk

From figure 1.3, K_w happens to fall in the vicinity of C_4, for example $Sd_1 \leq K_w \leq Sd_n$, and if the keyword search is performed

as mentioned in step 4, then the time taken to detect K_w will be TK_w. Thus, $TS_{kw} < TK_w$. This means that the keyword will be detected quicker than step 6.

7) Time taken to perform a keyword search can be decreased further by increasing the frequency of the processor (F_c). The time will decrease by a factor of $\frac{TSkw}{Fc \pm E}$ **(Where E is the error rate)**

4. RESULTS

Within the data collection proves there were two groups, these included the experimental group and the controlled group. The data collected from the experimental group involved the observation of keyword searching times with the application of the Superdetective tool, which implemented the parallel processing of a digital image within the investigation of the experimental group. This consisted of 4 cores 8 threads (4C8T) , and 4 cores 4 threads(4C4T), where the independent variable was the number of threads within each group. To aid in the testing of the Superdetective tool within the experimental group, the test was repeated five times for each independent variable, and focused upon when the Superdetective tool locates the word "Verisimilitude" at sector number 62461574595. Within tests one to five the number of threads to be searched was entered between one and eight, in order to provide the times it took for the keyword to be found with the addition of one to eight

threads, an example can be shown within appendix. When comparing the data collected against my hypothesis within Test 1 for 4C8T it can be shown within the graphs below that the minimum value occurred within thread 3 at 17 minutes 3 seconds,
(appendix 5.06) whereas it was located within thread 5(appendix 4.01) in 4C4T at 11 minutes 57 seconds.

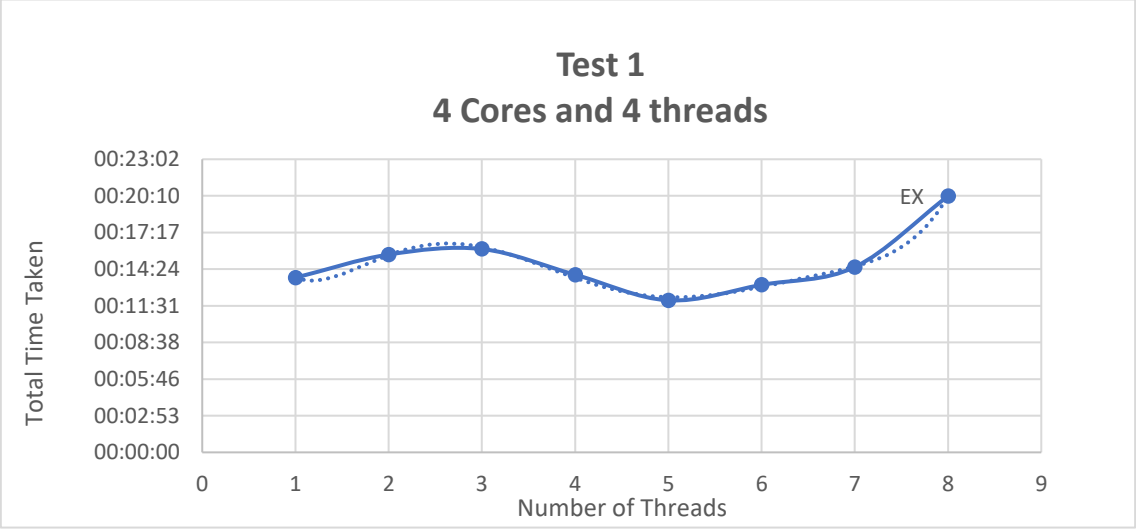

Figure 2: Test 1 4 core 4 threads

As shown above although the minimum value for both threads was located at different points throughout the experiments, although the minimal value for 4C4T was located at thread 5 which is two threads later than 4C8T it still located the keyword in the end sector 5 minutes 46 seconds quicker. Although this may not sound like a large difference in timing when this is compared with large quantities of data it will make a considerable difference, as if multiple sources of digital evidence were being analysed within the one time using different computers this would nevertheless increase the advantages of the tool overall.

When comparing the time taken to find the location of "Verisimilitude" within test one for both 4C4T and 4C8T, it was shown that 4C4T was considerably quicker
(appendix 4.01). This can be found throughout all tests conducted, as test 2 for 4C8T located the keyword in 31 minutes 32 seconds for thread 1 (appendix 5.07), whereas 4C4T located the keyword in 13 minutes 27 seconds for the same thread within test two (appendix 4.02).

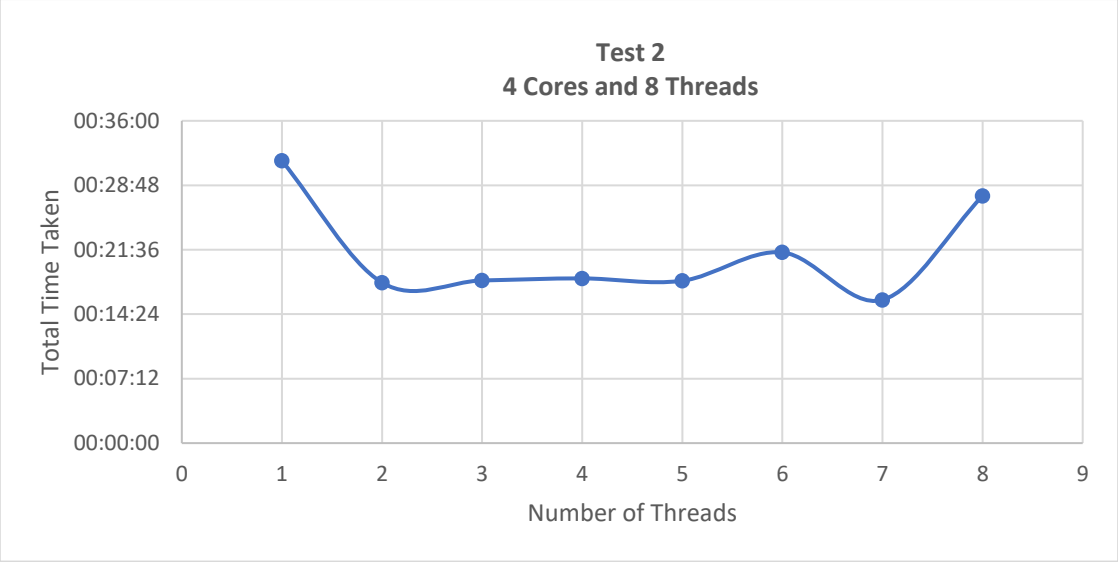

Figure 4: 4 core 8 threads

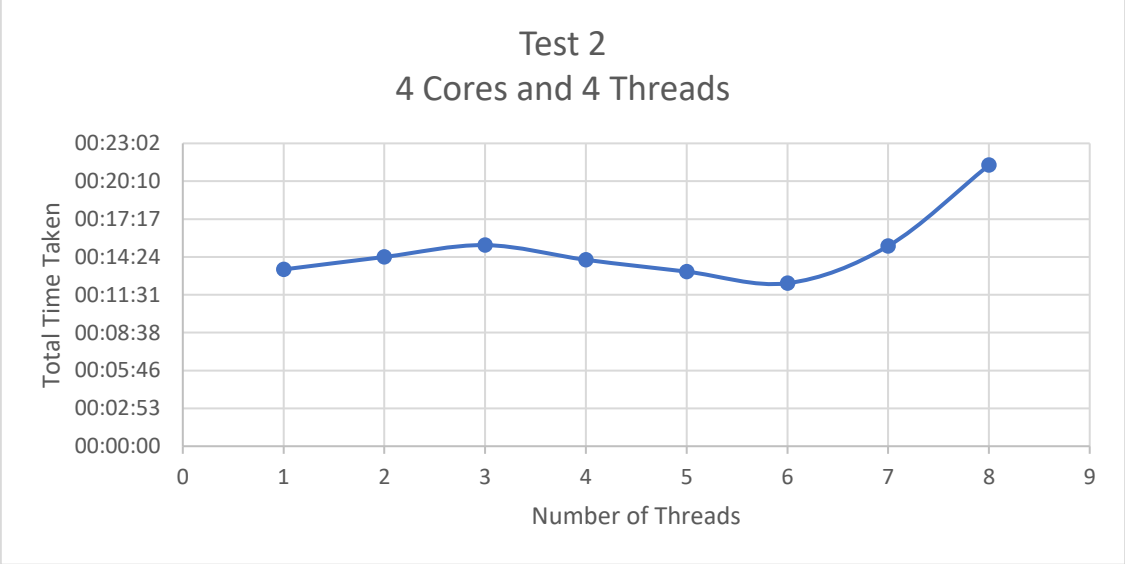

Figure 5: 4 core 4 threads

When comparing the two graphs above with the minimal value at 6 threads it shows a similar trend pattern, which is distinguished through the dip in the graph at thread 6. Within test 2 the minimum value was shown at 12 minutes 24 seconds (appendix 4.02), whereas on test 3 it was shown at 10 minutes 44 seconds (appendix 4.03). As a result, the dip present within both graphs is shallower within test 2 than test 3, due to the timings before the initial minimal value was found varied between 13 mins 17 seconds to 15 minutes 18 seconds. Whereas for test 3 the decline in the graph is more severe due to this varying between 12 minutes 44 seconds to 15 minutes 54 seconds, therefore at the minimal value of 10 minutes 44 seconds the gap is more distinctive.

When concluding the data gathered within the experimental group the graphs shown compatibility to my hypothesis, this being shown more within 4C4T due to the distinctive trendline being shown throughout. Although this is comparable to 4C8T this trend line isn't as apparent due to this differing between test 2 (appendix 4.07) and test 5 (appendix 4.10) which has proven to show the most inconsistency even though the same trend is apparent throughout. This could be due to myself stopping the thread search process as soon as the keyword was present within the last sector. Although this shouldn't have affected the time taken for the word to be located on additional threads, as I was only making notes on the time taken within that once specific location. It may have altered the functionality of the tool, due to me stopping the process before it was finished.

The controlled group focused on other image searching tools already available for the keyword searching on forensic images, and doesn't use the Superdetective tool. This group was incorporated to show the comparison of times when parallel processing was and was not enabled, and how long the indexing process took within these tools before the keyword searching could take place. The tools used to analyse the data consisted of Searchbin.py, Autopsy and dtSearch, from these results I found that the time taken to locate the keyword varied amongst the tools used. Only one test was run on Autopsy, this was done for both experiment 1 (4 cores, 4 threads) and experiment 2 (4 cores, 8 threads). The image that was used for the examination

of both experiments was the Disk Image of the 64GB USB. The start time and finish time focused on the indexing of the image itself before the keyword was searched. This is due to the indexing of the image resulting in the keyword being found within a second due to the image itself already being categorised into sections making for a quicker search. Therefore, it wouldn't provide an accurate result that showed the exact time taken to load the image to begin the examination. Due to it only focusing on the keyword inputted, and in turn would provide a misinformed analysis of the time taken to process and output a keyword search on this software.

The time taken for both experiments to index and enable the searching of a keyword using Autopsy showed 4 cores 4 threads (appendix 5.01) being 23 minutes slower than 4 cores 8 threads (appendix 5.02), which took exactly 40 minutes to index. When comparing this to the experimental group it can be shown that the keyword searching times for both 4 cores 4 threads and 4 cores 8 threads, which the used of the Superdetective tool is significantly quicker when incorporating parallel processing.

4.1 The Law of Hyperdetectability

During the course of experiments, it was observed that if a keyword is present in a certain location or region it would be detected instantaneously as shown in figure 6 below.

```
pointer is at position -1
'Verisimilitude' found at index: 15615393824.000000 on 64gbusb by search3 on Wed Mar 28 14:29:38
 (100982370320)
pointer is at position 4294967295pointer is at position -1
'Verisimilitude' found at index: 46846181440.000000 on 64gbusb by search7 on Wed Mar 28 14:29:38
 (100982526431)
pointer is at position -1
'Verisimilitude' found at index: 31330787568.000000 on 64gbusb by search5 on Wed Mar 28 14:29:42
 (101027528229)
'Verisimilitude' found at index: 99999984.000000 on 64gbusb.001 by search1 on Wed Mar 28 14:29:42
 (101027684485)
'Verisimilitude' found at index: 15715393794.000000 on 64gbusb by search3 on Wed Mar 28 14:29:43
 (101027997003)
'Verisimilitude' found at index: 7907696897.000000 on 64gbusb by search2 on Wed Mar 28 14:29:43
 (101028465767)
'Verisimilitude' found at index: 54753878263.000000 on 64gbusb by search8 on Wed Mar 28 14:29:43
 (101028465767)
'Verisimilitude' found at index: 23523090691.000000 on 64gbusb by search4 on Wed Mar 28 14:29:43
 (101028778275)
'Verisimilitude' found at index: 46946181382.000000 on 64gbusb by search7 on Wed Mar 28 14:29:43
 (101028778275)
'Verisimilitude' found at index: 39138484469.000000 on 64gbusb by search6 on Wed Mar 28 14:29:43
 (101029403304)
```

Figure 6: Depiction of law of hyperdectability

This region is known as the region of hyperdectability(H_{DR}) as shaded in lavender shown in figure 7.

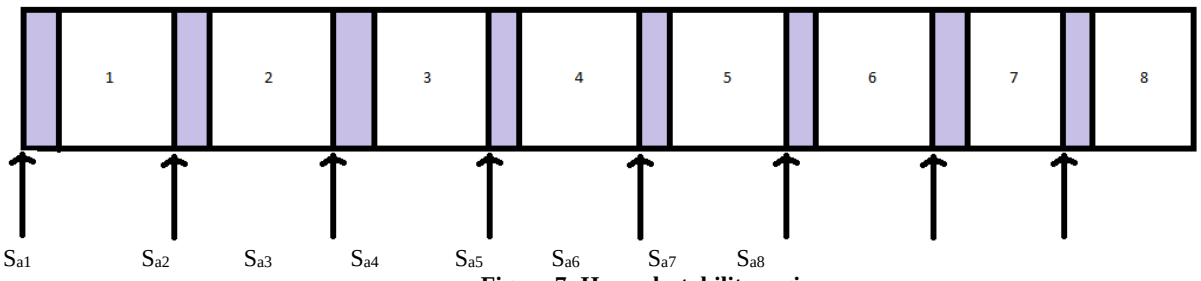

Figure 7: Hyperdectability region

(Sa_1,Sa_2, Sa_3,…, Sa_8) = Sector origins of each chunk

The region of hyper detectability (H_{DR}) is given by:

$H_{DR} = Sa_n + K; (n \epsilon N)$

Sa_n = Sector origins of a chunk

K = Constant

K is defined as the instantaneous rate of change of sector offset for time less than or equal to a second for a given chunk.

$$K = \frac{ds_0}{dt} / t \leq 1$$

Or if we calculate the slope of the secant line from a s_0 v$_s$ t graph we get the value of K. Slope of the secant line= $\frac{S_n - S_1}{t_n - t_1}$. Let tn-t1 =T and Sn-S1 =S. This means in T seconds S sectors have been covered. Now, S/T will give the value of constant (K). And, for a s_v v$_s$ t graph the value of K is $\int_0^1 Sv(t).dt$

Whilst calculating the value of H_{DR} for threads 4 and 8, it was found that certain threads had the same H_{DR} values, and found that even and odd threads are reverse harmonics in multiples of 2 and 3 respectively, and they happen to exhibit reverse or inverse harmonics i.e., for example, thread 4 detects a keyword at a faster rate than thread 8 or thread 16. Therefore, constant K would also be known as reverse (inverse) harmonic region

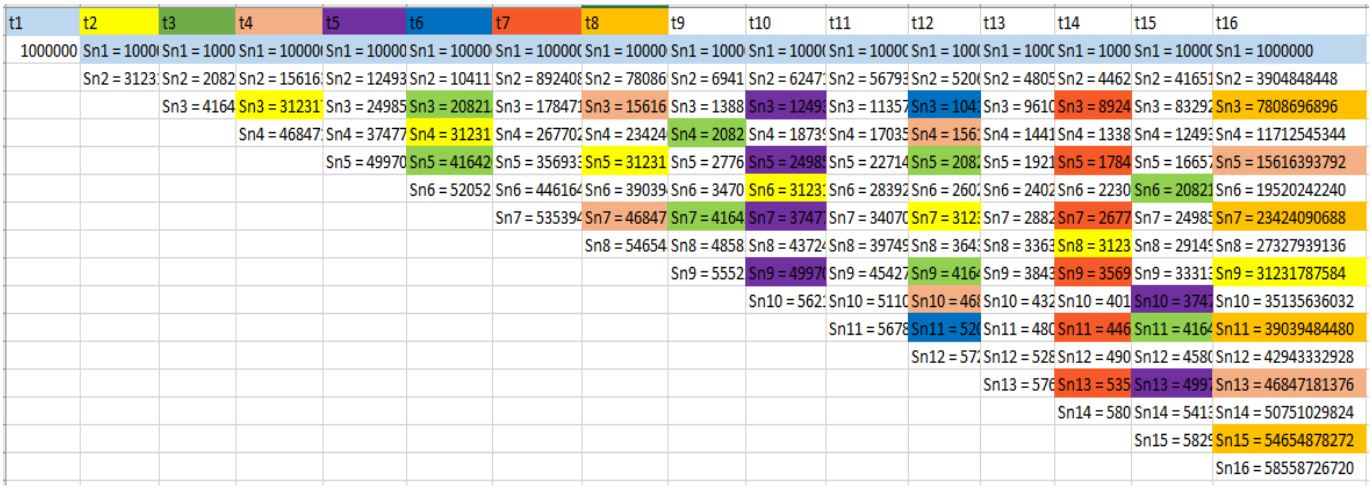

Figure 8: Reverse harmonic regions for threads 1 to 16 for 64GB forensic image.

The law of hyperdetectability or the law of simultaneous reverse harmonic detection states that "Ideally, if a keyword is located within the bounds of reverse harmonic region (KRHR) from its respective chunk's origin then that keyword would be detected within less than or equal to a second." This further implies that the rate of detection is independent of the search velocity (s_v) i.e., $\frac{ds_0}{dt} = 0$, and is also independent of the size of the forensic image or size of the storage medium (Ψ). Also, with number of cores constant, if the number of threads increase, time taken to detect a keyword (T_{kw}) will increase after the minimum value, but it the keyword falls in KRHR then the law will come into play. From the experiments it has been hypothised (which lays the foundation for future work) that T_{kw} is a function of number of cores (C_n) per thread (th_n), and as the number of threads approaches the size of storage medium Ψ, hypervelocities can be achieved (this is another way of saying that the law comes into play again). Hypervelocity is defined as the rate of change of sector offset ($\frac{ds_0}{dt}$) for time t ≈0.

Therefore, $T_{kw} \Rightarrow \lim_{thn \to \Psi} \zeta\left(\frac{Cn}{thn}\right) \to \chi$ (Hypervelocity).

5. CONCLUSION

The first research objective focuses extensively on previous algorithms which have been implemented on forensic images in other research works or within the forensic tools, the opinions that have formed the basis of our conclusions for this objective is from knowledge gained within the literature review. Algorithms that have been created to be implemented within a specific filed or workplace environment mainly consisted of Q Systems (Yan et al, 2015), Databases and cloud computing (Song et al, 2017). Where the algorithms used combine multiple techniques or a step by step approach (Babu and Sumathi, 2014), which includes both text clustering and text mining. This approach for locating a keyword is not applicable within the forensic field however, as some of these methodologies incorporate specific steps or file formats to be used within each step. Therefore, these approaches lack the flexibility to be applied to other areas other than the ones they were designed for.

6. REFERENCES

[1] 27037:2012, I. (2017) *ISO/IEC 27037:2012 - Information technology -- Security techniques -- Guidelines for identification, collection, acquisition and preservation of digital evidence*. [Online] Available from: https://www.iso.org/standard/44381.html [Accessed 13/11/2017]

[2] ABOU-ASSALEH, T. and AI, W. (2004) *Survey of Global Regular Expression Print (GREP) Tools*.

[3] ANDERSON, D. et al. (2016) *An introduction to Management science*. 15th ed. Boston: Cengage

[4] ARY, D. et al. (2018) *Introduction to Research in Education*. 10th ed. Cengage

[5] Association of Chief Police Officers *(ACPO)* (2013) *ACPO Guidelines on Computer Based Electronic Evidence*. [Online] Available from: http://www.digital-detective.net/digital-forensics-documents/ACPO_Good_Practice_Guide_for_Digital_Evidence_v5.pdf [Accessed 13/11/2017]

[6] BABU, J. and SUMATHI, K. (2014) *An Approach to Improve Computer Forensic Analysis via Document Clustering Alogrithms*. 2nd ed. International Journal of Innovative Research in Computer and Communication Engineering

[7] BASIAS, N. and POLLALIS, Y. (2018) Quantitative and Qualitative Research in Business & Technology: Justifying a Suitable Research Methodology. Available from http://sibresearch.org/uploads/3/4/0/9/34097180/riber_7-s1_sp_h17-083_91-105.pdf

[8] BEM, D. et al. (2008) Computer Forensics- Past, Present and Future. *Journal of Information Science and Technology*p.44

[9] CARRIER, B. (2017) *Autopsy*. [Online] Available from: https://www.sleuthkit.org/autopsy/ [Accessed 9/2/2018]

[10] CASEY, E. and BRENNER, S. (2011) *Digital evidence and computer crime: Forensic Science, Computers, and the Internet by Eoghan Casey*. 3rd ed. Waltham, MA: Academic Press

[11] COHEN, L., MANION, L. and MORRISON, K. (2013) *Research methods in education*. 7th ed. Routledge

[12] COHEN, M., GARFINKEL, S. and SCHATZ, B. (2009) Extending the advanced forensic format to accommodate multiple data sources, logical evidence, arbitrary information and forensic workflow. *Digital Investigation*Vol. 6 , pp.S57-S68

[13] COONS, P. (2014) *3 Methods of Forensic Imaging*. [Online] Available from:

http://www.d4discovery.com/discover-more/3-methods-of-forensic-imaging#sthash.PWvsjHdS.dpbs [Accessed 5/4/2018]

[14] CRAWFORD and STUCKI. (1990) Peer review and the changing research record. *Journal of the American Society for Information Science*Vol. 41 (3), pp.223-228

[15] DECHERCHI, S. et al. (2010) *Text Clustering for Digital Forensic Analysis*.

[16] DECHERCHI, S. et al. (2009) Text Clustering for Digital Forensics Analysis. *Advances in Intelligent and Soft Computing*pp.29-36

[17] DME Forensics (2014) Forensic Images for DVR Analysis - E01 or DD. [weblog] *DME Forensics*. Available from: http://info.dmeforensics.com/blog/forensic-images-for-dvr-analysis-e01-or-dd/ [Accessed 10/3/2018]

[18] *dtSearch (2018)– Text Retrieval / Full Text Search Engine*. [Online] Available from: https://www.dtsearch.com/ [Accessed 9/2/2018]

[19] ELLINGWOOD, J. (2013) *Using Grep & Regular Expressions to Search for Text Patterns in Linux | DigitalOcean*. [Online] Available from: https://www.digitalocean.com/community/tutorials/using-grep-regular-expressions-to-search-for-text-patterns-in-linux [Accessed 13/3/2018]

[20] Github (2018) *Sepero/SearchBin*. [Online] Available from: https://github.com/Sepero/SearchBin/blob/master/searchbin.py [Accessed 13/3/2018]

[21] Golov, N. and Rönnbäck, L., 2015, October. Big data normalization for massively parallel processing databases. In *International Conference on Conceptual Modeling* (pp. 154-163). Springer, Cham.