Advanced Text Searching Of Electronic Information Related To Forensic Discovery

Steven L. Haenchen

Submitted to the graduate degree program in
Information Technology and the
Graduate Faculty of the
University of Kansas School of Engineering
in partial fulfillment of the requirements
for the degree of Master of Science

Thesis Committee:

Dr. Hossein Saiedian: Chairperson

Dr. Arvin Agah

Dr. Gunes Ercal-Ozkaya

Date Defended

The Thesis Committee for Steven L. Haenchen certifies
that this is the approved version of the following thesis:

Advanced Text Searching Of Electronic Information Related To Forensic Discovery

Committee:

Dr. Hossein Saiedian: Chairperson

Dr. Arvin Agah

Dr. Gunes Ercal-Ozkaya

Date Approved

Abstract

The Federal Rules of Civil Procedure regarding production of electronic evidence, together with court rulings and penalties, have highlighted the need for timely and accurate production of electronically stored responsive evidence. Key criteria to the legal requirements include costs to produce, identification of responsive information and identification of privileged information within the responsive information. Currently the primary two methods of compliance are manual review of the documents and electronic Boolean text searches.

Text searching technology has been studied for over fifty years generating literally thousands of documents and books for a literature review. The focus of the literature includes accuracy of searching, optimization of searching, and completeness of searching. Some of the literature is based on a specific field of interest such as library cards or patent filings, but most is either generic or relates to either peer-to-peer searching or Internet searching. The documents related to the field of electronic evidence are very limited in number and presented no new search techniques directly.

We identified and classified the search techniques from the literature study after consideration of the applicability to electronic evidence. Using electronic evidence from actual litigation cases, the techniques were implemented to identify the thoroughness of the documents identified in the population and the related costs (time) required to identify such documents. The results from the various techniques were compared along with the costs to identify the "best" text searching method. Based on the results, we recommend implementation of a combination of the techniques to allow responsiveness to different requirements based on the legal circumstances.

Contents

Acceptance Page i

Abstract ii

1 Introduction 1
- 1.1 Justification . . . 2
- 1.2 Problem Statement . . . 4
- 1.3 Methodology . . . 5
  - 1.3.1 Sources of "Electronic Information" . . . 5
  - 1.3.2 Search Capabilities in Leading Forensic Software . . . 7
  - 1.3.3 Tests Conducted . . . 8
- 1.4 Significance . . . 10
- 1.5 Expected Contributions . . . 11
- 1.6 Evaluation Criteria . . . 12
- 1.7 Thesis Organization . . . 12

2 Previous Work/Literature Review 14
- 2.1 Boolean Searching . . . 18
- 2.2 Fuzzy Logic . . . 27
- 2.3 Context Searching . . . 28
- 2.4 Methods Involving Mathematical Probabilities . . . 31

3 Resources Used For Testing 34

4 Tests and Analysis 39
- 4.1 Boolean Searching . . . 42
  - 4.1.1 Boolean Searching - Baseline . . . 43

4.1.2 Boolean Searching - Weighting . . . 45
4.1.3 Boolean Searching - Intersection of Results . . . 46
4.2 FuzzyLogic . . . 47
4.2.1 Fuzzy Logic - Keywords & Tenses . . . 47
4.2.2 Fuzzy Logic - Keywords & Synonyms . . . 48
4.2.3 Fuzzy Logic - Keywords - Extra Character . . . 50
4.2.4 Fuzzy Logic - Keywords - Missing Character . . . 50
4.2.5 Fuzzy Logic - Keywords - Transposition . . . 52
4.2.6 Fuzzy Logic - Keywords - Combo . . . 53
4.3 Context Searching . . . 53
4.4 Methods Involving Mathematical Probabilities . . . 54
4.4.1 Mathematical - Weighting-Number of Keywords . . . 54
4.4.2 Mathematical - Weighting-Value of Keywords . . . 56
4.4.3 Mathematical - Weighting-Value Per Occurrence . . . 57
4.4.4 Mathematical - Weighting-Value & Proximity . . . 57
4.4.5 Mathematical - Text Repetition . . . 57
4.4.6 Mathematical - Baseline - Lingo . . . 58

5 Evaluation of Results **59**
5.1 Boolean Searching . . . 59
5.2 Fuzzy Logic . . . 61
5.2.1 Fuzzy Logic - Keywords, Tenses and Synonyms . . . 61
5.2.2 Fuzzy Logic - Extra, Missing & Transposition . . . 62
5.3 Context Searching . . . 64
5.4 Methods Involving Mathematical Probabilities . . . 65

6 Conclusion, Contributions, and Future Work **67**
6.1 Conclusion . . . 67
6.2 Recommendations . . . 69
6.3 Summary of Contributions . . . 70
6.4 Future Work . . . 71

A Background Of the FRCP **72**

Bibliography **76**

List of Figures

1.1 Primary Data Tables 8

5.1 Keywords/Tenses/Synonyms - Processing Times vs Hits 62
5.2 Keywords/Transpositions/Extra/Missing - Processing Times vs Hits 63

List of Tables

4.1 Unique Words found for Synonym/Tense Variants 41
4.2 Keyword Weights Used in Tests 42
4.3 Keywords 43
4.4 Boolean Text Search, basic 43
4.5 Boolean Text Search, expanded 44
4.6 Boolean Text Search, Intersection of Results 46
4.7 Tenses 48
4.8 Fuzzy Logic - Keywords & Tenses - Weighted by Word 48
4.9 Fuzzy Logic - Keywords & Tenses - Weighted by Occurrence . . . 48
4.10 Hits by Tense 49
4.11 Synonyms 49
4.12 Fuzzy Logic - Keywords & Synonyms - Weighted by Word 49
4.13 Fuzzy Logic - Keywords & Synonyms - Weighted by Occurrence . 50
4.14 Hits by Synonym 51
4.15 Fuzzy Logic - Keywords & Extra Characters 51
4.16 Fuzzy Logic - Keywords & Missing Characters 52
4.17 Fuzzy Logic - Keywords & Transpositions 52
4.18 Fuzzy Logic - Keywords & Variants 53
4.19 WordNet Phrases 55
4.20 Mathematical - Weighting-Number of Keywords 56
4.21 Mathematical - Weighting-Value of Keywords 56

Chapter 1

Introduction

Electronic evidence, commonly referred to as e-discovery by the legal profession, is growing rapidly and creating challenges to the legal profession. Court decisions in the 1990s raised awareness of a large number of issues related to e-discovery and the extreme amount of costs involved in both compliance and non-compliance of requests. Local and state court procedural rules enacted in the late 1990's also showed large discrepancies between jurisdictions.

Significant changes were made to the Federal Rules of Court Procedure effective December 1, 2006 with the intent of standardizing procedures (where practical) and reducing costs. As part of the cost reduction, parties are to meet early in the discovery process to determine what electronic evidence is to be included and how it is to be delivered. A primary means of determining what is to be included in e-discovery is the use of keywords for Boolean searches. These searches, depending on the circumstances, have been proven costly and ineffective. Couple the "costly and ineffective" with Forester Research estimates that e-discovery business was $1.5 billion in 2007 growing to $4.8 billion by 2011 [20], and we find plenty of reason to reduce costs. An illustration of the lack of technological assistance in

discovery is a Verizon Communications attorney telling a federal judiciary panel in January 2007 that Verizon Communications had spent $14 million on 225 lawyers to search 2.4 million documents to determine if they were privileged or not [20].

Significant research has occurred in other fields on text searching to improve thoroughness, accuracy, and speed. If such research could be applied to the searching of electronic evidence, significant savings and better communication between the litigation parties could be achieved. As we examined the text searching methods, we identified three method classifications in addition to Boolean searching: (1) fuzzy logic to capture variations of words, (2) context searching, and (3) methods involving mathematical probabilities. We identified the interrelated benefits of using these alternative methods: (1) lower costs of finding relevant documents and (2) increased availability of documents because the cost of retrieval is less.

1.1 Justification

Attorneys and other professionals working in the legal profession need to identify and review all the relevant documents for their cases. The problem they face in most instances is two-fold: (1) an enormous number of documents that might be relevant and (2) the large expense of manually looking at every document to determine its relevancy.

The enormous number of documents that might be relevant only continues to grow with the large and inexpensive amount of electronic storage today. Emails are sent to numerous recipients creating multiple copies; drafts of budgets are maintained for version control creating near duplicate copies; backup tapes are created and never reused or destroyed leaving many documents discoverable that have long been erased from the active computer system. Record retention policies

have not been updated for the electronic world or, where the policies have been updated, they are not being followed by everyone with access to the storage media.

In a first pass to save costs, attorneys are now requesting discovery documents to be provided electronically whenever possible. Electronic production has the immediate benefit over lots of boxes of paper by being faster and cheaper to deliver. Upon receipt, however, some attorneys immediately print the electronic discovery to review it forfeiting the cost benefit; others view the documents online. In both cases, every document is manually looked at by a professional who makes a judgment as to its relevancy to the case. The documents are tagged, manually or electronically, to facilitate relocation. It is the professional's time that is the most expensive part of the discovery production cost.

The second pass to save costs involves using technology to locate the relevant documents instead of the manual review. Document images are stored in document management software along with text (often obtained by optical character resolution). More advanced document management software stores other file types as well including Excel, Lotus, Word, WordPerfect, text, etc. The documents can have electronic tags added from a manual review, but the real savings comes from keyword searches. Keyword searches, similar to those used with Google, Yahoo, WestLaw and Lexis Nexus, allow the reviewers to focus on just those documents that contain one or more specific words.

Although keyword searches have significantly reduced the cost of finding relevant documents, concern exists that not all relevant documents are identified. Also, because many documents exist that are duplicates or near duplicates, many documents are included multiple times when only one is necessary. Finally, keyword searches alone do not rank the likely relevancy of documents so all documents

containing the keyword must be manually reviewed.

To address the above issues, we have looked at advancements in text searching in other fields of study. We have categorized the relevant search techniques as (1) Boolean searches, (2) fuzzy logic, (3) context searching, and (4) methods involving mathematical probabilities. In this research, we explain briefly the theory of each search technique and show a sample application of the text search technique using actual litigation evidence. The results of each search technique (the number of documents identified and percentage of documents identified that were actually relevant to the subject), together with the costs (time) it required for processing, are compared to the results and costs of the other techniques to determine the "best" technique to use for electronic evidence.

1.2 Problem Statement

The Federal Rules of Civil Procedure ("FRCP") dictate the procedural rules that must be followed on all Federal civil cases. These rules are generated by advisory committees using public drafts and public comments with ultimate adoption and enforcement by the United States Supreme Court. A short history of the FRCP is included as Appendix A.

State court rules are generally more relaxed than the FRCP; however, following the FRCP will almost always guarantee compliance with all the state court rules. The court rules, together with historical court rulings, fines, and penalties have defined the parameters of the electronic discovery process including the costs of non-compliance.

In summary, the over-arching objectives of the FRCP are to (1) minimize costs while (2) allowing access to the broadest amount of relevant information. These

two objectives are inversely related in that more information generally means more cost.

1.3 Methodology

We have implemented numerous text searching techniques from the four classifications of techniques: (1) Boolean searches, (2) fuzzy logic, (3) context searching, and (4) methods involving mathematical probabilities. We compared and contrasted the costs of obtaining the resulting set of electronic documents as well as the benefits of the particular result set. In this section, we discuss the tools used to obtain the test data and process the tests.

1.3.1 Sources of "Electronic Information"

The legal profession has a need for identifying the relevant documents from the universe of all documents in at least two different and very distinct situations. The first is general discovery requests in which one party to a lawsuit requests documents from another party which can be either the opposing party or a third party with documents relevant to the lawsuit. The second is the capture of information on entire electronic devices such as hard drives, PDAs, diskettes and backup tapes. Both situations result in "documents" (used herein as any file or file fragment obtained), but the latter type will typically result in a much larger portion of irrelevant documents being obtained. The search techniques discussed in this research apply equally to the documents from both situations, but are more beneficial in the second situation as more "junk" must be filtered out before manual review occurs. In this research, we focus on an example from the second situation.

A typical "computer forensics" case involves imaging and analyzing a computer hard drive. Depending on the situation, the computer forensic personnel must obtain an "exact copy" of the hard drive as evidence. The simplest software to achieve this result is a Linux-based boot disk using the `dd` command. The `dd` command will copy the drive sector-by-sector to a drive of equal or larger size. The resulting image is referred to as a "raw image" as it is an exact duplicate: nothing more and nothing less.

Forensic software specifically used to create "raw images" is available. An example of this specialized software used by law enforcement is HardCopy. Another commonly used imaging software is Norton Ghost. Norton Ghost is easy to use but has the disadvantage of Norton Ghost writing a very small identifier to the hard drive prior to imaging. Therefore, the image is not an "exact" copy, even if the difference can be explained.

The next group of imaging software is classified as "computer forensics software." Three common software packages in this category are Encase, Pro Discover and Forensics Tool Kit ("FTK"). Encase is the market leader and the most proprietary of the three. All three software packages allow you to image hard drives or to import a raw image. The actual use of each software package is unique and complex requiring practice.

For the testing performed in this research, Encase version 6.10.2 was used for imaging. The use of Pro Discover or FTK should not result in any significant differences for the text searching algorithms used for this research since those algorithms are implemented outside the Encase software. Now that we have discussed acquiring the documents we will address the keyword searching capabilities available in Encase which are more advanced than those in Pro Discover and FTK.

1.3.2 Search Capabilities in Leading Forensic Software

Encase 6 has a basic keyword search capability allowing the user to enter a set of keywords, run a keyword search process, and obtain a results set indicating every file occurrence that matches a keyword. A slight enhancement to this search capability is the ability to enter optional characters in the keyword string such as "[A-E]" to indicate the character can be A, B, C, D, or E to match the keyword. Additional options exist for the possible length of digits and possible inclusion of dashes in strings such as phone numbers and tax identification numbers.

What is not provided in Encase 6 include proximity terms such as "near" and "next to" and the ability to weight keywords in a result to identify the most likely document to view first.

Allowing some enhanced text searching capability, Encase has a proprietary language built in and Encase allows for indexing of all the documents to make processing more efficient. The proprietary language, EnScript, is similar to C. EnScript allows for more in depth text searching for the advanced programmer.

The advanced text search capabilities we are describing in this research, namely (1) Boolean searches, (2) fuzzy logic, (3) context searching, and (4) methods involving mathematical probabilities, cannot feasibly be tested within Encase. Therefore, the advanced text search capabilities are tested by extracting the documents from Encase to folders and storing relevant file information in a Microsoft SQL Server database. The tests are written in Visual Basic as absolute processing speed is not of concern to our decisions - only proportional speed.

1.3.3 Tests Conducted

We obtained the test data using a forensically sound procedure with Encase software. We extracted the the data from the electronic evidence and converted it to "words" storing the information in the Microsoft SQL database. The information was stored in the database using relational tables. The three primary tables used are illustrated in Figure 1.1.

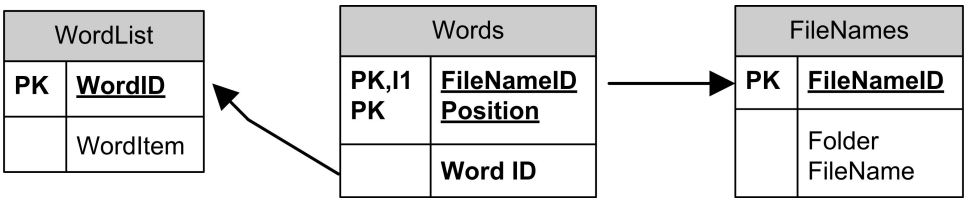


Figure 1.1. Primary Data Tables

The `FileNames` table contains a listing of every file in the electronic evidence including a reference to the evidence descriptor and location within the evidence (i.e. the folder). The `WordList` table contains a listing of every unique word in the electronic evidence (with a `word` being as described in Section 3). The relationship table `Words` identifies every word in the electronic evidence with reference to the file containing the word and the position of the word in the file (by word count). The word position is used in routines that utilize word proximity to other words within the same file.

Certain portions of testing take a significant amount of time and are used by more than one test. In particular, determining the `words` that contain a keyword or its variant is accomplished using the SQL function `PATINDEX()`. `PATINDEX()` recognizes that "EMPLOY" is included in the words "EMPLOY", "EMPLOYMENT", and "UNEMPLOYED". `PATINDEX()` works by doing a string comparison

of the keyword or its variant on each of the 23,168,285 words in the `WordList` table, a realatively slow process. We therefore created tables to store the `words` applicable to each keyword or its variant and used those tables in lieu of reprocessing the `PATINDEX()` routines each time.

Most of the text searching routines analyzed in this thesis involve identifying documents that contain certain keyworks and variants of keywords. Additionally, many of the text searching routines weight the findings based on criteria specific to the routine. As we tested the text searching routines, a general practice was to write the function to:

- Use the tables for each keyword/variants to identify all applicable `words` for which the related document should be included in the results.
- Identify all the documents that contain the `word` (a "hit").
- Determine the appropriate weight to assign to the "hit" specific to the routine (the "score").
- Store the information regarding the hit and score in the `FileNames` table.
- For some routines, store information about the hit and score in the table of the variant (i.e. tense, synonym, and WordNet) for further analysis of which variants generated additional hits and scores.
- Generate a report of the hits, scores, and processing times involved.

The results of the tests were then analyzed as documented in the Results sections of this thesis (Chapters 4 and 5). In particular, processing time, number of hits, scores, and the number of unique documents identified were compared between the text searching routines.

1.4 Significance

The two most significant issues related to examining any evidence is the cost of the examination and the thoroughness of finding all relevant information.

The cost of examination is always important as everyone wants to spend as little as possible while accomplishing the specific objective (in this case, finding all the relevant evidence to "win" their litigation). Cost is primarily the time of the attorneys, their support staff and expert witnesses (together, "attorney cost"). Cost can also include the charges of third parties to retrieve information from backup tapes, extract E-mails from servers, copy paper documents and even to review all such evidence for exclusionary material.

The attorney cost includes the time to analyze every relevant document and the time to review every non-relevant document provided to the attorney as potentially relevant. Therefore, efficiency in attorney cost requires not only that every relevant document be provided the attorney, but also that only relevant documents be provided the attorney.

Although it is unlikely that we can ever achieve a method of searching electronic evidence that precisely meets both requirements (i.e., every relevant document is provided and only relevant documents are provided), improved search techniques can improve on both the requirements by finding more potentially relevant documents and then identifying the exclusionary (privileged) documents from amongst the potentially relevant documents. In particular, we believe that implementing additional search techniques other than the Boolean keyword search commonly employed today will greatly enhance the ability to find relevant documents and, at the same time, exclude the privileged documents that should not be provided. This improvement will reduce costs of litigation allowing more people

to participate in the judicial system and for settlements to become fairer for the parties involved.

1.5 Expected Contributions

Our research identifies how text searching is being implemented in other fields and how those same techniques could be used in the field of computer forensic discovery with electronic evidence. We discuss the benefits of the techniques in terms of the type of electronic evidence noting that some methods are more applicable to known data versus unknown data thereby eliminating some techniques based on the searcher's specific situation.

Using actual data from litigation cases, we conducted performance tests on the relevant techniques. The results of the tests include processing time, number of documents identified, and percentage of relevant documents identified to total documents identified. The comparable results are presented in this research. The results show how the implementation of the non-Boolean search techniques produce more potentially relevant documents and a higher percentage of relevant documents. The results also show the additional processing time required to process with each of the search techniques, i.e., the additional cost involved in obtaining more and better information.

Our results show how the benefits of implementing these non-Boolean search techniques can save money in discovery while also producing more thorough and more accurate results.

1.6 Evaluation Criteria

The most commonly employed text search technique for electronic evidence today is the Boolean keyword search. We use the implementation of the Boolean keyword search technique as the baseline for our evaluation with the assumption that this is the search technique currently in use by attorneys. For each additional search technique we test, we compare the results to the baseline for (1) number of potentially relevant documents identified, (2) percentage of relevant documents to potentially relevant documents identified, and (3) the time it took to process the search technique. A comparison of these results for each of the search techniques is a measure of (1) thoroughness in finding all potentially relevant documents, (2) accuracy in identifying only truly relevant documents, and (3) the additional processing cost for (1) and (2). We draw a conclusion as to overall potential cost savings by comparing the results to the costs of discovery noted in [20].

1.7 Thesis Organization

The thesis will be organized into the following chapters:

- **Chapter 1:** Introduction and Background - The background of the problem, significance and a justification of a solution.
- **Chapter 2:** Previous Work/Literature Review - A review of published literature on text searching and a description of the key text searching techniques to be tested and evaluated.
- **Chapter 3:** Resources Used for Testing - A description of the source of electronic information used in the testing and the processing of the data performed prior to testing.

- **Chapter 4:** Tests and Analysis - The experiment tests, results, as well as explanations and analysis related to the number of documents found.
- **Chapter 5:** Evaluation of Results - The costs (time) to generate the results for each test as well as consideration of the value of the documents identified with each technique.
- **Chapter 6:** Conclusions and Future Work - The conclusions and future research in the field of text searching of electronic evidence.

Chapter 2

Previous Work/Literature Review

Much of the literature in the last couple of years related to "e-discovery", the term commonly used by attorneys for electronic evidence in a litigation case, is focused on either penalties for failure to produce or costs of actual production. Several high profile cases in recent years have heightened the awareness of the cost of non-compliance with multi-million dollar fines, instructions to jurists unfavorable to a party that destroyed evidence, and even the fining of the attorneys themselves for not preventing spoliation by their clients.

The literature related to costs of compliance considers the enormous amount of data retained by clients, record retention policies often ignored, difficulties in determining what data should be considered relevant to a discovery request, reviewing documentation for privilege, methods of delivering documents (including fines for improper delivery) and alternative locations of relevant data such as backup tapes and PDAs.

At first consideration, these two problem areas addressed in the literature

appear to be opposite each other, but the solution appears to be the same for both. An abstraction of the problem is actually a lack of knowledge and/or ability to identify and filter large amounts of documents to comply with requests for production, legal or not. More documents are produced in an organization than is feasible for any person or reasonable size group of persons to be able to manually read every document and tag with all possible subject matters. Because the large volume cannot be managed properly, parties to a lawsuit delete relevant data, fail to provide relevant data, provide irrelevant data and otherwise frustrate opposing counsel and judges. Compliance periods get delayed and costs soar while the primary business of the company gets neglected.

The primary means of compliance today is a combination of manual review and keyword searching. Courts have been using keyword searching and generally accepted it since judges and lawyers routinely use similar technology for Web searches (Google) and legal research (Lexis Nexus, Westlaw). An open issue with keyword searching is the determination of what the keywords will be. The requesting party generally does not want to commit to an inclusive list whereas the complying party expressly wants an inclusive list. The contentious issue may eventually be resolved by the judge.

The next issue involves what the documentation population is. We will not address the issue in this research extensively other than to note that the FRCP and case law have changed the documentation population from "existence", to "used in the ordinary course of business", to its current "reasonably accessible" standard. We believe that implementation of the technologies in this paper will expand the "reasonably accessible" population by making it more cost-effective to find relevant discovery on media such as backup tapes.

Once the population has been filtered to just those documents containing the keywords, a manual review process is often applied to assure no privileged communications are contained in the results. Once privileged documents are removed or redacted, the discovery documents are supplied to the requesting party.

Some additional technological practices in place include expanded keyword searches with Boolean terms such as "NEAR" and "NEXT TO". Additionally, keyword searches have been used with some success in identifying privileged documents (although most attorneys continue to do a manual review of all those documents not flagged as privilege by the keyword search). Finally, some courts have used sampling to determine the value of documents on media not reasonably accessible, e.g. sampling five of 300 backup tapes to see how many relevant documents the backup tapes actually possessed.

It should also be noted that keyword searches are not a practical means of determining relevant parts of a database system. Reasons include normalization resulting in a single occurrence of text referenced thousands of times [58]. Such electronic data is not discussed further in this research.

We performed a literature review for text searching and noted that most text searching can be categorized as: (1) Boolean searches, (2) fuzzy logic, (3) context searching, and (4) methods involving mathematical probabilities. There is overlap between the categories and each has its own strengths and weaknesses.

Jason R. Baron works for the National Archives and Records Administration in College Park, Maryland. He received a request to produce relevant documents that was 1,726 paragraphs long and indicated it was applicable to any document created in the last 50 years. The request required searching through approximately 20 million emails for relevancy. Baron recognized that a manual review of 20

million emails was not feasible and instead relied on Boolean searches to identify the relevant documents and comply with the request. In examining the experience, Baron stated: "were a similar tsunami wave of litigation to wash over the agency in the future, I would be recommending using more sophisticated and alternative ways of searching for evidence, including methods drawn from notions of fuzzy logic, concept searching, and statistical techniques; third, I found that there was little in the way of present-day research showing what search and information retrieval methods were objectively better to use in a legal context" [7].

The Boolean search capabilities Baron used were keyword searches that identified if a document contained one or more keywords. Keyword searches can be performed using set logic [22]. Minor advancements occur with weighting of results to identify which documents are more likely to be relevant than others.

Fuzzy logic expands on Boolean searches by identifying slightly misspelled words, different tenses, considering both singular and plural forms and using thesaurus words as replacements. Fuzzy logic also attempts to emulate natural language by excluding minor words. For example, "all birds that live in Africa" is searched using the expression "bird* + liv* + Africa" [6].

Context searching deals with inferring a subject from the words. When one hears "John hit the ball with the bat," one can infer the subject is baseball. One possible solution we consider in this research uses the public database WordNet being developed at Princeton to infer subject [25]. Use of WordNet alone returns a large set of synonyms for the keyword, well beyond the list a thesaurus would return. The researcher must either manually select which words to use as additional keywords or include them all.

Yi discusses an implementation of using WordNet with searching for patents.

His implementation involves building on ontology database. He uses clustering to minimize the manual intervention needed, but still notes the need for expert's supervison [56]. Ma also relies on building an ontology database and searching for hashes in the database instead of for keywords in text. Like any text searching algorithm that requires the building of a history or database, its usage with electronic data will be limited to searches of data used in the ordinary course of business, but most likely not applicable to searches of hard disk images, etc. [37].

For the final category, methods involving mathematical probabilities, we identified a number of proposed solutions including approximate patterns [45], uninformed and informed similarity matching [27] and concept-driven clustering [41]. In the litigation field, this final category appears to be most applicable when hiding of information is involved, such as a fraud investigation. As our sample data used is not applicable to the hiding of information situation, this particular advanced text searching method was not tested. Instead, it is included in the suggestions for additional research.

2.1 Boolean Searching

Boolean searches are the most common method employed to identify the relevant documents for discovery from the population of documents. The method is accepted by the courts because of its general acceptance in the community (Web searches and legal searches both employ keywords with Boolean operators) and previous usage without significant findings of error.

The original arguments against keyword searches were related to error rates, particularly the rate of responsive documents that would fail to be included in the response (error of omission). This argument fails to get much of the court's

attention following a study by Blair and Moran in 1985 [6] that determined manual reviews of a large amount of documents by attorneys only identify 20 to 25% of the responsive documents. Keyword searching has proven to have better results although such studies are questioned based on the vagueness of determination of what is or isn't a responsive document. In the retesting performed by a group for the Sedona Conference wherein documents determined responsive or not were reexamined by another, significant disagreement was found. On a scale of -1 being complete disagreement and +1 being complete agreement, the retest result was +0.49 [7]. With this degree of subjectivity involved coupled with the adversary nature of litigation, some disputes will never be resolvable.

Once the documents in the population matching the keywords have been extracted (the "responsive documents"), the size of the response must be considered. An inappropriate keyword(s) that caused a huge volume of documents to be considered responsive must be reconsidered. The responding party generally does not want to disclose any more information than necessary and the requesting party does not want to be burdened by lots of irrelevant documents it must manually consider. Variants of keywords must be considered and preferably agreed to between the parties. Variants include plurals, abbreviations, and tenses of words. Operators such as "NEAR" and "NEXT TO" are instrumental to limiting results but must be used appropriately.

The responsive documents are generally reviewed manually for privilege, but significant savings and accuracy can be achieved by using keyword searches first to identify privilege documents. Recent case law is favorable for the responding party that accidentally provides privileged documents in e-discovery in allowing such documents to be subsequently quashed (the opposing party cannot use the

information in court, however, they are still aware of the information).

The delivery of the responsive documents to the requester must be in a generally acceptable format. Prior practices of eliminating meta-data, changing file names to random numbers, placing all files in a single folder, etc. are no longer allowed since several large sanctions have occurred for such practices. Sanctions in one case occurred for providing large documents as multi-page TIF files instead of single-page TIF files as requested. Many deliveries today must include a load file for standard legal document library programs such as Concordance. (Remember, delivery and receipt of discovery between two businesses is generally reciprocal).

String Searching Methods generally involve text of what you are searching for (the "pattern") and text of what you are searching in (the "text"). The answer initially sought is "Where is the first occurrence of pattern in text?" with expansions for what is sought addressed later in this research. The answer with pattern "AME" in text "I AM AMERICAN" is six (i.e. "AME" begins with the sixth character of "I AM AMERICAN").

```
     AME
I AM AMERICAN
123456789
```

There are many published algorithms for text searching including:

```
Naïve, or Brute Force, Algorithm
Knuth-Morris-Pratt (KMP)
Boyer-Moore (BM)
Simplified Boyer-Moore
Boyer-Moore-Horspool (BMH)
Fischer-Paterson
Shift-OR Algorithm
Karp-Rabin (KR)
Aho-Corasick (AC).
Commentz-Walter (CW)
```

The various algorithms all produce results with 100% reliability. The differences addressed are primarily speed and amount of memory needed to operate.

Baase describes three of these algorithms noting the key differences [4].

First is the Naïve, or Brute Force, algorithm. This method is how most people would perform the searching manually.

They compare the first character of the pattern with the first character of the text. “A” does not match “I”.

If they do not match, the pointer to the text is incremented to the next character and step (1) is repeated based on the new pointers. “A” does not match “ ” (blank space). Increment and repeat again. “A” matches “A”.

When the characters match, the next character of both the pattern and text are compared. “M” in “AME” matches “M” in “AM”. If they match, (3) is repeated until the entire pattern is matched to the text at which time the algorithm ends returning the starting position of pattern in text. In this case, the next characters “E” in “AME” does not match the “ ” (blank space) in the text.

When the pattern and text fail to match, the algorithm must “back up” to the first character in the pattern and the last starting position for comparison in text (from step (1)) plus one for the next character thereafter. Step (1) is repeated based on the new pointers.

Because the Naïve algorithm tests every character in text and potentially every character in pattern for every character in text, the possible cost of operations is expressed as O=(m*n) where:

```
m is the length of text, and
n is the length of pattern.
```

This cost is considered the highest of the algorithms and is the result of the “back up” noted in step (5).

The second method Baase addresses reduces the cost of the “back up” [4]. The Knuth-Morris-Pratt (KMP) algorithm uses a finite automata methodology

to read the text only once. In its simplest of terms, when the comparison results in a mismatch, instead of backing up in the text, the pattern pointer is adjusted to the next possible match knowing the history to that point in text. The cost result is a potential complete read of text and a complete read of pattern or O=(m+n), much less than (m*n).

The third method Baase addresses is Boyer-Moore (BM) [4]. BM differs in that it does some "pre-processing" of the text and pattern at a slight cost. This cost is recovered by implementing the theories that (1) more information is derived by looking at the last character of the pattern instead of the first character of the pattern and (2) based on what is learned in (1), multiple characters can be skipped.

In simplified terms, BM will start by comparing the third character of "AME" to the third character of "I AM AMERICAN" in our example since the pattern is three characters in length. It will "pre-process" the "A" in "I A" and identify that it exists in pattern "AME" as the first character. It recognizes that character "E" differs from "A" and shifts the pointer in text not to the next character (as in Naïve and KMP), but two characters over as in:

```
  AME
I AM AMERICAN
123456789
```

Now the BM algorithm is ready to test character five in text to character three in pattern. The "pre-process" determines that character five in text, a blank, does not exist in pattern. With this knowledge alone, the BM realizes the comparison is invalid and a shift of the length of pattern (3) in text is applicable resulting in:

```
     AME
I AM AMERICAN
123456789
```

The BM algorithm “pre-processes” and compares “E” to “E”. It backs up to compare “M” to “M” and then “A” to “A”. It completes and reports the starting position of “AME” in text. Because of these shifts, BM is often faster than KMP even though it pre-processes and backs-up. Boyer addresses the DM algorithm tests as “Observations” as follows [9]:

Observation 1. If char is known not to occur in pat, then we know we need not consider the possibility of a occurrence of pat starting at string positions 1, 2, ... or patlen: Such an occurrence would require that char be a character of pat.

Observation 2. More generally, if the last (right-most) occurrence of char in pat is delta1 characters from the right end of pat, then we know we can slide pat down delta1 positions without checking for matches.

Observation 3(a). We can use the same reasoning described above - based on the mismatched character char and delta1 - to slide pat down k so as to align the two known occurrences of char.

Observation 3(b). We know that the next m characters of string match the final m characters of pattern.

Most other text searching methods are direct or indirect variations of KMP or BM.

Frakes suggests the fastest overall algorithm to be Boyer-Moore-Horspool (a variation of BM). Six of the above algorithms were tested for speed based on various lengths of the patterns. Based on the results of testing, the Naïve, Shift-OR and KMP algorithms have relatively constant times to complete regardless of the length of the pattern with Shift-OR and KMP being about one-half the time of Naïve. For the three algorithms with declining speeds as the length of the pattern increased, the Boyer-Moore-Horspool was always significantly faster than

the BM and Simplified Boyer-Moore (which were about the same as each other). KMP (constant time) was faster than the Boyer-Moore-Horspool with very small patterns (less than four characters) and faster than BM with patterns less than 11 characters [22].

Watson tested KMP, BM, CW and AC in a textual environment and in a DNA environment. He found that for the most part, KMP, CW and AC were constant costs regardless of the pattern length. BM costs declined as the pattern length increased. CW outperformed AC when tested in the DNA environment where patterns exceeded 100 characters [53].

Breslauer reviews the text searching algorithms with the variation of finding partial patterns instead of whole patterns only [10].

A variation not discussed in much of the literature is adaptations for non-English languages. One paper mentioned French, but only to highlight the fact the French language contains 2.5 times the number of English words when all variants are included. However, Ando and Luk point out some of the problems inherent in working with two-byte alphabets as opposed to one-byte alphabets. Especially since the data is often mixed with one- and two-byte characters, significant changes to the search program may be required [2, 36]. We recommend these differences be considered in the planning of any new routine instead of attempting a change later.

As can be noted above, String Searching is a mature field with lots of published information. Improvements continue to be made to the algorithms, but no significant improvements are expected. A similar area to String Searching is Pattern Matching where speed is even more critical.

Similar to Text Searching, the Pattern Matching routines published and con-

sidered in this research usually have as their goal to find occurrences of short binary strings in packets as efficiently as possible. They expect to have few matches and, when they find a match, to only be concerned with the first match found.

Pattern matching speeds are much more critical than today's string search functions because most pattern matching is focused on Network Intrusion/Detection Devices and inspecting the related network traffic packets. Large volumes of packets could need processing and delays affect many programs and users.

Methods noted in published documents on Pattern Matching that might be applicable to electronic data searching include:

- Use of a Bloom Filter for memory efficient storage of patterns [17]
- Use of Fast String Matching Algorithm called FNP [35]
- Use of Field Programmable Gate Arrays (FPGA) with parallel programming to increase speed of processing [17]
- Use of systolic array-based string matching architecture [5]
- Use of Aho-Corasick multi-pattern matching algorithm in conjunction with the parallel processing [17]
- Use of two-comparator (i.e. parallel processing) variation of the KMP algorithm [5]
- Use of two-tier, cluster-wise matching algorithm named by authors Hierarchical multiple-pattern matching algorithm (HMA) [50]

We suggest that the pattern matching be considered for speed improvements only.

Next we consider a topic area relevant to us as our electronic data could be received from a very large and/or diverse source, yet our focus will be on the data as a whole. In Peer-to-Peer networks, one looks for the data as a whole without consideration of the individual hardware of the machines housing such data.

Zhu discusses techniques similar to some employed in Text Searching (such as Vector Space Model) and relevance ranking techniques to identify relevant

documents. The paper also discusses the use of hashes to verify files transferred match the originals [60].

Of particular interest in [60] related to searching electronic evidence is the Topology Adaptation Algorithm they present which could be adapted to identify the relevant data from the various forms of electronic data captured as if it was from a single evidence source. The Topology Adaptation Algorithm uses nodes for data and applies techniques of neighbor discovery, adaptation and maintenance to improve relevance rankings.

These methods should be considered in addressing the electronic data from various sources. Note that not all relevant electronic data is in the form of text. Zhang defines how concepts between images (pictures) can be conceptually related to one another. Zhang uses a Multiple Bernoulli Relevance Model (MBRM) which is based on the continuous-space relevance model (CRM, also used with Text Context) to identify relevance between images [59]. The work with image contexts is not as mature as that with text.

Similarly, Robles-Kelly suggests converting graph data to text data and then using text search and context routines on the converted graph spectra to find and retrieve relevant data. The techniques suggested include seriation, simulated annealing, mean field annealing, semidefinite programming and eigenvector methods. The work is this case is the conversion to text [46].

The concepts of other forms of data including graphs, pictures, architectural drawings and even executable code should be addressed in the proposed solution of searching all electronic data (not just text data).

2.2 Fuzzy Logic

Fuzzy logic is the like an extension of Boolean searches with variations. Variations of the Boolean search include consideration of plurals, abbreviations and tenses of words. Fuzzy logic can also expand the variation search for minor misspellings and similar words.

Usage of fuzzy logic in practice was identified as occurring only at a few of the larger law firms, and then only for internal use. Its usage is still in its infancy and therefore not fully accepted by the courts as a way to limit discovery.

A basic implementation of fuzzy logic actually starts with a more thorough discussion with the document custodians as to what terms are used in the population of documents. Learning that the product had certain nicknames while in research and development means the keywords should include all such nicknames. Secondly, a Thesaurus is utilized to identify additional keywords from the initial set of keywords. Finally, letter transpositions, common misspellings and common abbreviations of all the keywords are added and searched for similar to the initial keywords.

Patterson focuses the fuzzy logic on the location of the keywords in finding documents with a particular phrase. Instead of searching for "authorize payment to company", Patterson proposes looking for "authorize within 2 words of payment within 10 words of company." Relaxing the strictness of the location of the words greatly increased the finding of documents where memory of the exact phrase was hazy [42].

The algorithms for determining the additional fuzzy logic keywords will result in determining how large the eventual population of total keywords becomes. As this number of keywords exponentially grows due to searching such things as all

possible transpositions in spellings, the processing time and cost will also increase exponentially.

2.3 Context Searching

Just as we want to include in our search results a graph, picture or database about the requested topic, what about text words that aren't there? Context searching identifies the subjects of documents that are not expressly stated otherwise.

As opposed to locating the occurrence of a string in a document, context searching is identifying data in documents that is relevant to your topic, but not necessarily explicitly stated. In addition, the relevance of that context is critical to increasing efficiency of searches of electronic media.

Some context filtering suggestions are based on user inputs from prior searches [24, 43]. We do not believe building such databases for electronic evidence will be relevant as the population of reviewers will be extremely small (and completed by the time most of the reviewers could use this data they themselves created) and that many of the reviewers will not want to share their data with the others (opposing parties to a civil or criminal case).

Sarkar provides a framework for identifying concepts from previous solutions (heuristic learning) [49]. Although we believe the concepts presented have merit, for electronic evidence the application will need significant testing as the data from one case and any prior cases may be very dissimilar.

An alternative is presented by Osinski. A relevance concept from their paper that could have significant value to identifying electronic evidence for further review is to (1) identify the word or concept counts in the electronic evidence. Typically greater weight is given to concepts that appear more often or are very

technical and therefore used rarely. Then (2) identify work or concept counts for the related population, e.g. English documents or peer computers. Using the second data as "Lingo", compare to the first data to identify documents not using the same or similar "Lingo". This technique has the great advantage for the examiner in that it allows the examiner to identify unusual/relevant documents for further examination based on the data itself (versus the historical method of "guessing" what is in the data and providing keywords in hopes of finding it) [41].

Osinski also uses techniques such as stemming (removing word inflections), ignoring stop words (conjunctions, articles, etc.) and text-segmentation heuristics. Findings are clustered and cosines used to identify concept relevancies [41].

Chang suggests the use of vectors to determine relationships noting that, as with most context searching, vectors are used to determine the degree of relationship between documents. Chang proposes the use of Genetic Algorithms to aid in determining the relative strength of those relationships [11]. (The method does use some user responses to learn the relationship strengths which may lessen the value of this particular algorithm for electronic data search purposes in this paper's context.)

Harabagiu make a case for using the public database WordNet to identify context not included in any of the words. In their example, they identify that the words "John hit the ball with the bat," should be included in a search for information about baseball. In other words, information unstated in text (context) is important in locating relevant information also. Like many other context algorithms, Harabagiu relies on tries, relationships, relationship strengths (generally defined by cosine values) and parallel processing. The WordNet database they utilize contains both dictionary words and "glosses" which aid in developing the

relationships and strengths of relationships [25].

Feng expands on context by mapping text into linguistic concept space. The four layers of linguistic concept space are: (1) conceptual primitives, (2) semantic category of a sentence, (3) contextual elements, and (4) contexts. The abstract concepts are described from five properties: dynamic(v), static(g), attribute(u), value(z), and result(r). Similar words such as "think" and "idea" are mapped near each other. Words used with different meanings such as "look" in "Look there" versus "Her look at him" are mapped with different properties (v for verb usage and g for noun usage respectively). Use of this method would allow for a search of the keyword and all words within X spots of the keyword in the map [54].

Debnath presents a concept related to web pages that may have applicability to this research's focus on electronic data (albeit not text searching). The tool proposed by Debnath identifies irrelevant parts of Web pages and omits the irrelevant portions from search dictionaries and storage [16]. Although we would not want to omit the irrelevant information from our storage, we should consider how the application of these suggestions might eliminate large duplications of information in database fields that are not relevant (e.g. the context in the document makes them irrelevant), etc. On a simpler basis, perhaps removing known file headers, etc. from the search field criteria (once verified to be valid headers) would remove irrelevant text.

Context searching is not a mature field like text searching. Scholars are making suggestions in a variety of directions and much room for improvement exists. It is also a field that has attracted much interest (including from Google, Yahoo and Microsoft) so many improvements are expected in the coming years.

Some additional methods of determining relevant documents use mathematical

probabilities. These methods are discussed next.

2.4 Methods Involving Mathematical Probabilities

Boolean searches identify relevant documents based on the occurrence or lack of occurrence of a keyword(s) in the document. However, the Boolean search method does not generally weight the relevant documents. At most, the method employed might count the number of keywords in the document to identify those documents with the most keywords. Weighting by keyword count is more efficient than no weighting at all, but it implies that all words are equally important. Expanding on the weighting concept could include: varying weights per keyword, varying weights for the number of occurrences (multiple occurrences are more relevant to a point, then cease relevancy of additional occurrences) and different weightings for keywords in proximity to other keywords (close to, excluded from same document, etc.) Adding this type of weighting greatly increases the complexity of search routines but also greatly increases the value of the results.

Another method involving mathematical probabilities is the analysis of text repetition. This method has more applicability in searching for relevant documents when the custodian is uncooperative or ignorant of the usage of words in the documents themselves. Instead of specific keywords selected in advance, keywords are counted using vector analysis and weighting to determine relevant terms.

An example of searching for the relevant terms involves a company that sells widgets. A Boolean search for "(SALE OR SELL OR SELLING) NEAR WIDGET" might miss many emails by the salespeople who commonly referred to the two types of widgets as REDs and BLUEs. This mathematical method that iden-

tified the frequency of RED and BLUE near SALE, SELL and SELLING would alert the producer to the relevancy of these particular documents.

Another alternative method of searching uses a "baseline" set of data. Using the keywords, the relative occurrence in the baseline data is compared to the relative occurrence in the discovery universe. Using the second data as "Lingo", they compare the occurrences to the first data to identify documents not using the same or similar "Lingo". This technique has the great advantage for the examiner in that it allows the examiner to identify unusual/relevant documents for further examination based on the data itself (versus the historical method of "guessing" what is in the data and providing keywords in hopes of finding it) [41].

Arevian focuses on locating data using a neural network system. Of particular interest to this research is his weighting of keyword finds based on associations with other words and based on the order in a sentence [3].

Some research such as [44] cross the boundaries of the classifications of searching used in this research. Peery discusses the implementation in the Wayfinder File System of document searches that co0sider all three dimensions: (1) content, (2) structure, and (3) metadata.

The Utopian answer would be to allow the reviewer to input "Show me all the documents on the system that discuss being employed by one of our competitors." The concept involves many parts including converting a natural language query to a binary search and expanding terms such as who the competitors are. Although details of the algorithms are not public, Roussinov discusses implementation of just that and tests implementations available on Google and MSN [48]. Identifying that Google and MSN have implemented initial algorithms on their Internet sites is indicative of software improvements that will be available to electronic evidence

also.

Chapter 3

Resources Used For Testing

We implemented the various text searching techniques using Visual Basic and Microsoft SQL Server. We have not attempted to optimize the techniques as the evaluation criteria is the number of potentially relevant documents identified, the relevancy percentage of those documents, and percentage increase in time to find the documents using the particular search technique versus the baseline Boolean search technique. As we are using ratios for comparison instead of absolutes, we believe optimization can be left for the ultimate implementer.

The data used for the testing in this research was obtained from 15 electronic evidence devices actually used in litigation plus one electronic evidence device (hard drive) imaged to use as a non-litigation baseline in the analysis. The electronic evidence from litigation included a thumb drive, an external hard drive from a Macintosh system, a hard drive from a Macintosh system, and 12 hard drives from desktop and laptop computers using various Microsoft operating systems (hereafter referred to as "hard drives" regardless of the original hardware).

The electronic evidence used for testing, after extracting the emails from Microsoft Outlook PST and Lotus Notes NSF files to individual files, contained

1,587,609 individual files in 217,442 folders for a total size of 2,429,029,692,761 bytes (2.20 TB) of data.

Much of the electronic evidence was related to cases where an employee had left one company to join a competitor. The attorney was looking for documents relevant to the employee accepting the new position and the (former) employee taking company materials, including trade secrets, with him. We limit our testing here to the accepting of the new position as it is relevant to the entire class of cases with the electronic evidence and it allows us to present the results with confidentiality of the underlying cases.

For testing, a typical attorney request is the simplest of Boolean searches: identify all documents containing "'{Company}' OR 'Employment' OR 'Interview' OR 'Offer' OR 'Position' OR 'Salary'" and search without case sensitivity. To maintain confidentiality, we tested with the five stated keywords, but not {Company}. All testing was done case insensitive.

All the electronic evidence was imaged using Encase version 6.10.2. The images were viewed through Encase which "unerased" all files from the recycle bin and "unerased" all files identified in NTFS as having been erased, but not yet overwritten by another file. All hard drive areas not assigned to a file, including the unerased files, was designated to a file such as "unallocated clusters".

Encase allows for Boolean searches and was used to comply with the attorney's request for the actual case. However, Encase does not allow for the advanced text search techniques to be performed we analyze in this research. Therefore, all files including the unused space files such as "unallocated clusters" were exported from Encase to a local drive folder. The result we achieved is having folders on our testing machine containing all files from the 16 hard drives including system files,

hidden files, deleted files and new files representing the unused space on the hard drives.

The next process in preparing the test data is to convert the files to searchable text. Several options exist depending on the file type. Text files, including those with common file extensions of TXT, CSV and BAT, convert byte-for-byte to searchable text. HTML and XML similarly convert, but contain formatting information a person viewing the file normally does not consider in searching.

Other files with significant formatting and otherwise hidden program information include Word documents, Excel workbooks, database files, etc. The decision must be made to consider for text searching only the information a user of the native program sees (i.e. the information on the screen of a letter being typed) or all the information (fonts, colors, indentions, etc., some of which is encoded information rather than English), or both. For purposes of the tests in this research, we chose to use the files in their native format (i.e. the formatting was retained).

A third category of files for consideration includes pictures. These file extensions include TIF, GIF, BMP and JPG. Options for these files are to use the data as if it were text (which, for the most part, would be "junk") or to recognize the text in the picture using OCR software (which is not 100% reliable). For purposes of this research, and because most of the picture files did not appear to be relevant, we chose to use the files as if they were text (essentially excluding them from successful searching unless the file extension was erroneous and they actually contained significant text).

The final category of files encountered has no native program to open them and see a text representation of the information on a normal basis. These files include those with extensions COM, DLL, EXE, AVI, MPG, as well as unallocated clusters

and many system files such as SYS. We used these files in their native format on the chance that they either had embedded text that could match keywords or that the extension was misleading and the file was actually another format.

Some files do not fit completely in one category or another. These include Adobe PDF files: some are pictures but others are pictures with embedded text information (which allows searching the file for text, but not necessarily for editing that text). Another is SYS files, some of which appear to be machine code yet others are text information such as found in Config.sys. For this research, such items were converted to text as if they were text. In subsequent work, including if the data were being used in court, each file should be considered on its own merits and many files should be tested as both native (logical) text and as raw text (physical).

One additional pre-process performed to simplify the testing: all file data was converted to text using the following steps (in sequence):

- The file was converted to uppercase.
- Characters except "0" through "9", "A" through "Z", comma, and period were converted to a space character.
- All commas were converted to space characters except those between two digits (i.e. those that might be embedded in a number such as 1,234) in which case the comma was deleted.
- All periods except those between two digits (i.e. those that might be decimal points) were removed.
- All repetitions of space characters were reduced to a single space character.

We saved the information in a Microsoft SQL 2000 Server ("SQL") database. A "word" was considered any sequence of characters separated by a space character. If the number of characters in the word was more than 2, the folder, filename, word, and word sequence in the file ("word order") were stored in the database. The length of the word stored was a maximum of 40 characters in length.

We acknowledge that the preprocessing could allow us to not find certain information such as a keyword ending more than 40 characters deep within a "word" such as a string "123456789012345678901234567890123456789012345KEYWORD" would be stored as "1234567890123456789012345678901234567890123456789012345KEYWO" and therefore "KEYWORD" would never identified. Another example with numbers would be if a Comma Separated Values file stored the numbers 1, 2, 3, 4 without spaces as in "1,2,3,4", the preprocessing would convert it to "1234" and consider it a single word. We considered these implications and decided the chances of these situations occurring and missing a relevant keyword would be negligible.

We now discuss the tests by category.

Chapter 4

Tests and Analysis

This chapter discusses the tests performed on the electronic evidence and the results obtained. The text was preprocessed as discussed in Chapter 3, Resources Used For Testing (page 34). The preprocessing resulted in 911,861,950 words in the database table. A second table of unique words resulted in 23,168,285 entries. The two tables were linked by indices to facilitate the processing.

The basic Boolean text search finds the word in the unique word table and, via the indices, identifies the list of matching words in the complete table of words. The primary variation of this search, however, is to find every word in the unique word table that contains the text being sought and return all the words in the complete table of words that contain these unique words (e.g. the words "Firetruck" and "Truckstop" are returned when the word "truck" was searched). The routine to find all words containing the search text was used multiple times and therefore was written as a separate routine to run once instead of including it in each applicable test. The routine identified 7,043 unique words containing the text of the five search terms and processed in 6,371 seconds . In our test results, we will add these seconds to the processing time for comparison purposes.

In some tests, we allowed for a single extra character anywhere within the search text (as a means of accommodating a misspelling or typing error). As with the words in the previous paragraph, we located all these word variants in the list of unique words as a subroutine run once. The subroutine found an additional 356 unique words to include in 43,491 seconds . In the applicable test results, we will add these seconds to the processing time for comparison purposes.

Similar to allowing for a single extra character, we also have tests that allow for the exclusion of a single character. The subroutine to find all unique words containing the search text variants of a missing character resulted in 19,118 additional words and took 48,924 seconds to process. In the applicable test results, we will add these seconds to the processing time for comparison purposes.

Another variation allowed in some tests is for a single transposition to have occurred. The subroutine to find all unique words containing the search text variants of a transposition resulted in 670 additional words and took 40,704 seconds to process. In the applicable test results, we will add these seconds to the processing time for comparison purposes.

Some tests include not only the search words, but also related words. One such test includes 27 synonyms of the search words. Processed as a subroutine, 61,098 additional unique words were found matching synonyms of the five search words in 35,923 seconds of processing time. In the applicable test results, we will add these seconds to the processing time for comparison purposes.

Another test that includes not only the search words, but also related words is the tenses of the search words. Processed as a subroutine using three “tenses”, 499 additional unique words were found in 4,865 seconds of processing time. In the applicable test results, we will add these seconds to the processing time for

comparison purposes.

One more test that includes not only the search words, but also related words is the context searching test "WordNet". Processed as a subroutine using 54 "contexts", 359,292 additional unique words were found in 46,103 seconds of processing time. In the applicable test results, we will add these seconds to the processing time for comparison purposes.

We further preprocessed the synonyms, tenses, and WordNets for variants of extra characters, missing characters, and transpositions as done above with the initial five keywords. The number of unique words identified, and the processing time involved for each process, is shown in Table 4.1.

Table 4.1. Unique Words found for Synonym/Tense Variants

Variant	Unique Words	Processing Time
Synonyms		
–Extra Character	63,238	61,933 seconds
–Missing Character	443,595	35,388 seconds
–Transposition	35,008	30,007 seconds
Tenses		
–Extra Character	38	17,277 seconds
–Missing Character	1,109	11,031 seconds
–Transposition	2	8,038 seconds
WordNets		
–Extra Character	346,345	99,810 seconds
–Missing Character	745,491	115,910 seconds
–Transposition	271,849	65,516 seconds

Some tests required that we give weightings to the keywords. We arbitrarily assigned the following weights.

```
Employment   10 points
Interview     3 points
Offer         1 point
Position      1 point
Salary        5 points
```

Other tests required us to assign weights not just to the keywords, but also for how many occurrences of the keyword existed in each document. We arbitrarily assigned the weights listed in Table 4.2.

Table 4.2. Keyword Weights Used in Tests

	Occurrence					
Keyword	**1st**	**2nd**	**3rd**	**4th**	**5th**	**6th+**
Employment	10	6	7	1	1	1
Interview	3	5	10	3	3	1
Offer	1	8	10	12	3	1
Position	1	1	1	1	1	0
Salary	5	5	3	2	2	1

As illustrated in Table 4.2, employment and salary are examples of words with diminishing value. Found one, two, or three times is important; but after that the findings of the words are of much less value.

Interview and offer are examples of words found a few times are the most valuable, but found rarely or many times is of lessor value.

Position is an example of a word with minimal weight for each finding and eventually (six or more times) having no additional weight.

We will now address the individual tests by classification.

4.1 Boolean Searching

In this section, we develope a baseline using Boolean text search and discuss two expansions of the results: Weighting and Intersection of Results. As noted previously, the five keywords used in testing are as listed in Table 4.3.

Table 4.3. Keywords

Keyword
Employment
Interview
Offer
Position
Salary

4.1.1 Boolean Searching - Baseline

For the Boolean text search, we simply find every occurrence of the five search words in the database table of words. The SQL statement to accomplish the task would be:

```
SELECT Word
FROM WordList
WHERE Word='KEYWORD'
```

Since we are identifying the documents related to these words, we modify our SQL statement to retrieve the file names:

```
SELECT DISTINCT Filename
FROM Files JOIN WordList ON Files.FileID = WordList.FileID
WHERE Word='KEYWORD'
```

In processing the Boolean text search, we obtained the results shown in table 4.4. The process identified 50,709 unique documents (out of the 52,375 documents in table 4.4) containing at least one search word in 111 seconds.

Table 4.4. Boolean Text Search, basic

Keyword	Hits	Documents
Employment	2,982	861
Interview	998	401
Offer	7,476	3,715
Position	193,978	46,857
Salary	1,428	541
Total	206,862	52,375

Although we could continue the testing using keywords in this manner, we elected to expand the usage to include "words" for which the keyword was a subset. In particular, in addition to searching for "INTERVIEW" we would also search for "INTERVIEWS", "INTERVIEWING", etc. This results in all possible combinations, including irrelevant additional keywords such as "FILEPOSITION" for "POSITION" when the meaning of what we want is an employment position. At this stage of our testing, we prefer to find too many documents, rather than risk missing any relevant documents.

We therefore modified our SQL statement similar to:

```
SELECT DISTINCT Filename
FROM Files JOIN WordList ON Files.FileID = WordList.FileID
WHERE Word LIKE '%KEYWORD%'
```

In processing the Boolean text search, we obtained the results shown in Table 4.5. The process identified 101,084 unique documents (out of the 104,729 documents in Table 4.5) containing at least one word with the search term in 282 seconds. Including the preprocessing time to identify all the additional words containing the search term, processing time was 6,653 (282 + 6,371) seconds.

Table 4.5. Boolean Text Search, expanded

Keyword	Hits	Documents
Employment	3,557	944
Interview	3,888	809
Offer	26,995	8,147
Position	406,612	94,264
Salary	1,520	565
Total	442,572	104,729

Searching for patterns such as `%KEYWORD%` requires much more processing time and results in many more hits. The test processed in 6,653 seconds (versus 111 seconds before) and identified 101,084 unique documents (versus 50,709 unique

documents before) that contained at least one keyword. The majority of the extra processing time was the chore of finding the words containing the search text (accomplished with the SQL function `PATINDEX()`). We believe that this is a more appropriate test in most instances to assure all documents are considered in the results that may be relevant to the parties.

We use this test as our benchmark for comparison of the additional tests in this research as this particular technique is the most commonly employed search technique of electronic evidence in practice today. In all subsequent tests, we search not only for the keyword itself, but for every word containing the keyword also.

4.1.2 Boolean Searching - Weighting

The Boolean search was relatively fast but provides no weighting of the documents for potential relevancy. Should not the documents containing more keywords, including repetitions of the same keyword, be considered first? Responding to this question results in the first minor alteration of our search technique in that we provide the same listing of 101,084 documents to the reviewer ordered by the number of keyword hits in each document. From the Table 4.5 we noted that there were 442,572 hits (keywords found in a document) for the 101,084 unique documents. From the document listing (not shown in thesis), we see that the number of hits per document ranged from 1 to 883. The document with 883 search hits should be reviewed first.

The resources involved in identifying the number of hits per document is not much more than the costs of finding the unique documents and therefore we consider the costs of this technique to be the same as that of finding the documents

containing the keywords (the benchmark).

4.1.3 Boolean Searching - Intersection of Results

The baseline and weighting discussion above only looked at unions of sets of documents containing a keyword, but consideration can also be given to intersections of the sets [22]. An example of the union of sets is a combination of the documents that have the keyword "Employment" or "Interview" or "Offer" or "Position" or "Salary." The union was done as shown in Table 4.5 above in that the sum of each of the five sets of documents containing any keyword is 104,729 yet the number of unique documents (union) is only 101,084.

An example of an intersection of sets of documents would be the combination of all documents that have the keyword "Offer" and at least one of the keywords: "Employment" or "Interview" or "Position" or "Salary." In processing the test of the intersection example, we obtained the results in Table 4.6.

Table 4.6. Boolean Text Search, Intersection of Results

Keyword	Hits	Documents
Employment	503	181
Interview	318	120
Offer	13,607	2,533
Position	26,023	2,418
Salary	350	85
Total	40,801	5,337

Mathematically the results from an intersection must be equal to or smaller than the results of a union. The logic holds true here. Only 2,533 unique documents were found. The smaller result set was found in only 265 seconds compared to 282 seconds previously (preprocessing time being equal for both routines).

4.2 FuzzyLogic

This section contains the results of the tests of fuzzy logic text searching.

In addition to applying the fuzzy logic variants to the Keywords, the fuzzy logic testing included "Keywords & Tenses of the Keywords" and "Keywords & Synonyms of the Keywords". Fuzzy logic variants tested were transpositions, missing characters, additional characters, and combinations of transpositions, missing and additional characters.

We did not specifically test for the fuzzy logic variant of common misspellings since the results would have been equivalent to adding another synonym for each common misspelling word identified.

4.2.1 Fuzzy Logic - Keywords & Tenses

In this variant of the keyword search, we search not only for the keyword, but also various tenses of the keyword. In a basic text search, the tenses would include plurals such as "Offers" and "Positions", but with our searching with words containing the search text, these plurals would not produce different results (i.e. a word containing "Offers" is already considered since it contains "Offer"). As used in this research, the additional searching for tenses resulted in any words containing "Salaried" or "Salaries" since these variants do not also contain "Salary" within them. We also included the tense "Employ" from the keyword "Employment". As our tense is shorter than the keyword, "Employ" will potentially result in more documents being included in the results. Table 4.7 lists the tenses used in testing. Tables 4.8 and 4.9 shows the results of this test of keywords and tenses.

The time to process this test was 1,682 seconds. Including the preprocessing time to identify all the additional words containing the search term, processing

Table 4.7. Tenses

Keyword	Tense
Employment	Employ
Salary	Salaried
Salary	Salaries

Table 4.8. Fuzzy Logic - Keywords & Tenses - Weighted by Word

Keyword	Hits	Documents	Score
Employment	20,299	5,013	52,312
Interview	3,888	809	11,664
Offer	26,995	8,147	26,995
Position	406,612	94,264	406,612
Salary	1,693	606	7,773
Total	459,487	108,839	505,356

time was 12,918 (1,682 + 6,371 + 4,865) seconds. The test resulted in 103,771 unique documents or 2,687 more than searching on the keyword alone.

The tenses resulted in 16,915 additional hits per Table 4.10.

4.2.2 Fuzzy Logic - Keywords & Synonyms

In this variant of the keyword search, we search not only for the keyword, but also various synonyms of the keyword. The synonyms considered for the keywords are listed in Table 4.11.

Tables 4.12 and 4.13 shows the results of this test of keywords & synonyms.

Table 4.9. Fuzzy Logic - Keywords & Tenses - Weighted by Occurrence

Keyword	Hits	Documents	Score
Employment	20,299	5,013	40,492
Interview	3,888	809	13,105
Offer	26,995	8,147	90,945
Position	406,612	94,264	287,795
Salary	1,693	606	5,806
Total	459,487	108,839	438,143

Table 4.10. Hits by Tense

Tense	Hits
Employ	16,742
Salaried	173
Salaries	0
Total	16,915

Table 4.11. Synonyms

Synonym	Keyword	Synonym	Keyword
Service	Employment	Agreement	Offer
Employ	Employment	Deal	Offer
Meeting	Interview	Rank	Position
Talk	Interview	Title	Position
Conference	Interview	Status	Position
Discussion	Interview	Station	Position
Present	Offer	Pay	Salary
Tender	Offer	Income	Salary
Proffer	Offer	Wage	Salary
Bid	Offer	Earning	Salary
Propose	Offer	Money	Salary
Suggest	Offer	Remuneration	Salary
Recommend	Offer	Payment	Salary
Submit	Offer		

Table 4.12. Fuzzy Logic - Keywords & Synonyms - Weighted by Word

Keyword	Hits	Documents	Score
Employment	1,478,399	116,265	1,510,412
Interview	113,275	25,932	121,051
Offer	631,168	119,630	631,168
Position	2,882,878	375,046	2,882,878
Salary	131,216	22,044	137,296
Total	5,236,936	658,917	5,282,805

Table 4.13. Fuzzy Logic - Keywords & Synonyms - Weighted by Occurrence

Keyword	Hits	Documents	Score
Employment	1,478,399	116,265	2,322,769
Interview	113,275	25,932	257,351
Offer	631,168	119,630	2,008,371
Position	2,882,878	375,046	8,897,517
Salary	131,216	22,044	491,935
Total	5,236,936	658,917	13,977,943

The time to process this test was 6,076 seconds. Including the preprocessing time to identify all the additional words containing the search term, processing time was 48,370 (6,076 + 6,371 + 35,923) seconds. The test resulted in 464,317 unique documents or 363,233 more than searching on the keyword alone.

The Synonyms resulted in additional hits as shown in Table 4.14.

4.2.3 Fuzzy Logic - Keywords - Extra Character

In this variant of the keyword search, we search not only for the keyword, but also for any one extra character being included anywhere within the keyword (accomodating for some typing errors and/or misspellings). Table 4.15 shows the results of this test of keywords and extra characters.

The time to process this test was 668 seconds. Including the preprocessing time to identify all the additional words containing the search term, processing time was 106,849 (668 + 6,371 + 99,810) seconds. The test resulted in 101,403 unique documents or 319 more than searching on the keyword alone.

4.2.4 Fuzzy Logic - Keywords - Missing Character

In this variant of the keyword search, we search not only for the keyword, but also for any one missing character being included anywhere within the keyword

Table 4.14. Hits by Synonym

Synonym	Hits
Employ	16,552
Service	1,458,290
Conference	14,053
Discussion	17,152
Meeting	44,794
Talk	33,388
Agreement	39,835
Bid	115,840
Deal	37,736
Present	257,323
Proffer	36
Propose	5,199
Recommend	37,839
Submit	76,377
Suggest	24,583
Tender	9,405
Rank	40,729
Station	73,994
Status	621,775
Title	1,739,768
Earning	4,235
Income	1,909
Money	50,498
Pay	65,221
Payment	-
Remuneration	26
Wage	7,807
Total	5,236,936

Table 4.15. Fuzzy Logic - Keywords & Extra Characters

Keyword	Hits	Documents	Score
Employment	3,563	944	18,606
Interview	4,032	946	13,551
Offer	27,342	8,414	91,380
Position	406,734	94,314	287,917
Salary	1,645	612	6,168
Total	443,316	105,230	417,622

(accomodating for some typing errors and/or misspellings). Table 4.16 shows the results of this test of keywords and missing characters.

Table 4.16. Fuzzy Logic - Keywords & Missing Characters

Keyword	Hits	Documents	Score
Employment	18,719	3,024	98,192
Interview	4,985	1,507	16,453
Offer	508,757	91,468	1,525,104
Position	413,211	95,565	294,017
Salary	3,015	961	10,961
Total	948,687	192,525	1,944,727

The time to process this test was 2,470 seconds. Including the preprocessing time to identify all the additional words containing the search term, processing time was 124,751 (2,470 + 6,371 + 115,910) seconds. The test resulted in 166,360 unique documents or 65,276 more than searching on the keyword alone.

4.2.5 Fuzzy Logic - Keywords - Transposition

In this variant of the keyword search, we search not only for the keyword, but also for any one transposition being included anywhere within the keyword (accomodating for some typing errors and/or misspellings). Table 4.17 shows the results of this test of keywords and transpositions.

Table 4.17. Fuzzy Logic - Keywords & Transpositions

Keyword	Hits	Documents	Score
Employment	3,557	944	18,580
Interview	3,910	825	13,189
Offer	28,814	9,024	96,924
Position	406,667	94,291	287,850
Salary	1,520	565	5,543
Total	444,468	105,649	422,086

The time to process this test was 621 seconds. Including the preprocessing

time to identify all the additional words containing the search term, processing time was 72,508 (621 + 6,371 + 65,516) seconds. The test resulted in 101,680 unique documents or 596 more than searching on the keyword alone.

4.2.6 Fuzzy Logic - Keywords - Combo

In this variant of the keyword search, we search not only for the keyword, but also for missing characters, extra characters, and transpositions being included anywhere within the keyword (accomodating for some typing errors and/or misspellings). Table 4.18 shows the results of this test of keywords, missing characters, extra characters, and transpositions.

Table 4.18. Fuzzy Logic - Keywords & Variants

Keyword	Hits	Documents	Score
Employment	18,725	3,024	98,218
Interview	5,151	1,655	16,983
Offer	509,071	91,712	1,525,460
Position	413,348	95,619	294,154
Salary	3,015	961	10,961
Total	949,310	192,971	1,945,766

The time to process this test was 1,533 seconds. Including the preprocessing time to identify all the additional words containing the search term, processing time was 416,710 (1,533 + 6,371 + 99,810 + 115,910 + 193,086) seconds. The test resulted in 166,642 unique documents or 65,558 more than searching on the keyword alone.

4.3 Context Searching

We used the WordNet public database to generate a test of context searching for this thesis.

The WordNet database takes a word and returns synonyms, antonyms, etc. based on entry by linguists over the past several years. The WordNet database is publicly available at no cost and has numerous different interfaces available. The implementation of the WordNet database used for this thesis passes the keyword to the WordNet database and receives back a list of synonyms in order of most likely usage.

The test implementation takes those synonyms from WordNet and adds them to the list of keywords for searching. The order received from WordNet is used to make the resulting information more valuable by weighting the original keyword a 15, the first line of synonyms from WordNet a 14, the next line 13, etc.

Using WordNet is functionally the same as using an expanded set of synonyms. In testing, WordNet matched 359,292 unique words; WordNet with extra characters matched 346,292 unique words; WordNet with missing characters matched 745,491 unique words; and WordNet with transspositions matched 271,849 unique words. The results of this testing would be proportional to the number of WordNet words and characters versus synonym words and characters and is not reproduced in this thesis.

4.4 Methods Involving Mathematical Probabilities

In this section, we look at different methods involving mathematics and mathematical probabilities.

4.4.1 Mathematical - Weighting-Number of Keywords

In this variant of the keyword search, we weight the documents found by Boolean text search by the number of different keywords a document contains.

Table 4.19. WordNet Phrases

.25 **WordNet**	**Keyword**	.25 **WordNet**	**Keyword**
Employ	Employment	View	Position
Work	Employment	Perspective	Position
Engagement	Employment	Posture	Position
Use	Employment	Attitude	Position
Usage	Employment	Status	Position
Utilization	Employment	Post	Position
Utilisation	Employment	Berth	Position
Exercise	Employment	Office	Position
Consultation	Interview	Spot	Position
Audience	Interview	Billet	Position
Question	Interview	Place	Position
Offering	Offer	Situation	Position
Crack	Offer	Spatial Relation	Position
Fling	Offer	Placement	Position
Go	Offer	Location	Position
Pass	Offer	Locating	Position
Whirl	Offer	Positioning	Position
Proffer	Offer	Emplacement	Position
Volunteer	Offer	Situation	Position
Extend	Offer	Place	Position
Bid	Offer	Stance	Position
Tender	Offer	Posture	Position
Put up	Offer	Side	Position
Provide	Offer	Wage	Salary
Extend	Offer	Pay	Salary
Place	Position	Earning	Salary
Military Position	Position	Remuneration	Salary

Table 4.20 shows the results of this weighting. In particular, the test shows three documents contained all five keywords, and 55 documents contained four of the five keywords. Surely these documents should be considered first.

The time to process this test was 309 seconds. Including the preprocessing time to identify all the additional words containing the search term, processing time was 6,680 (309 + 6,371) seconds. The test did not identify any more or

Table 4.20. Mathematical - Weighting-Number of Keywords

No. of Keywords	Hits	Documents
1 Keyword	394,428	97,830
2 Keywords	43,719	2,924
3 Keywords	3,053	272
4 Keywords	1,244	55
5 Keywords	128	3
Total	442,572	101,084

less documents than the baseline. It mearly identified which of the resulting documents was more likely to be important.

4.4.2 Mathematical - Weighting-Value of Keywords

In this variant of the keyword search, we weight the documents found by Boolean text search by a predefined value for each keyword. The weights used in this test were as follows:

```
Employment   10 points
Interview     3 points
Offer         1 point
Position      1 point
Salary        5 points
```

Table 4.21 shows the results of this weighting. Each document has a score based on the weighting and the documents with the highest scores should be considered first.

Table 4.21. Mathematical - Weighting-Value of Keywords

No. of Keywords	Hits	Documents	Score
Employment	3,557	944	35,570
Interview	3,888	809	11,664
Offer	26,995	8,147	26,995
Position	406,612	94,264	406,612
Salary	1,520	565	7,600
Total	442,572	104,729	488,441

The time to process this test was 392 seconds. Including the preprocessing time to identify all the additional words containing the search term, processing time was 6,763 (392 + 6371) seconds. The test did not identify any more or less documents than the baseline. It mearly identified which of the resulting documents was more likely to be important.

4.4.3 Mathematical - Weighting-Value Per Occurrence

In this variant of the keyword search, we weight the documents found by Boolean text search by a predefined value for each keyword based on how many times the keyword occurs in the document. This method requires extensive processing time and therefore was deemed non-comparable to the other methods tested.

4.4.4 Mathematical - Weighting-Value & Proximity

In this variant of the keyword search, we weight the documents found by Boolean text search by a predefined value for each keyword; however, we only consider a keyword as hitting if another keyword also exists within 25 (a chosen value for this test) words of any other keyword. This method requires extensive processing time and therefore was deemed non-comparable to the other methods tested.

4.4.5 Mathematical - Text Repetition

In this variant of the keyword search, we weight the documents found by Boolean text search by a value determined as a ratio to the number of times the word appears in this document to the number of times the word appears in the

entire electronic evidence. This method requires extensive processing time, does not use predetermined keywords, and therefore was deemed non-comparable to the other methods tested.

4.4.6 Mathematical - Baseline - Lingo

In this variant of the keyword search, we weight the documents found by Boolean text search by a value determined as a ratio to the number of times the word appears in the subject electronic evidence to the number of times the word appears in "clean", "comparable" electronic evidence. This method requires extensive processing time, does not use predetermined keywords, and therefore was deemed non-comparable to the other methods tested.

Chapter 5

Evaluation of Results

This chapter compares the processing time and results of various text searching methods. The cost focus is the processing time required to retrieve the results. The benefit focus is on the amount of documents retrieved that could be relevant to the searcher less the amount of superfulous documents. We also point out some lessons learned in writing the test functions.

5.1 Boolean Searching

The basic Boolean search is processed in approximately linear time: Search Time X number of keywords. Based on the method used to tag documents in the `FileNames` table, keywords with more hits will take longer to process than keywords with fewer or no hits.

We decided up-front to limit words to be between three and 40 characters long. Words of one or two characters will be contained in many other words generating false hits. For efficiency, the minimum word length should generally be the length of the smallest keyword, synonym, tense, and/or WordNet. Not recommending

that a short, valid keyword not be used, using only longer keywords will greatly improve the efficiency of the searches (fewer words are stored as the words less than the minimum size are discarded; smaller storage space means faster searches).

Words with meanings common to electronic evidence should be considered for exclusion or used with mathematical text searches. For example, in our tests we used the keyword "POSITION" to identify documents discussing the former employee's new job (position) at his new company. In our search results, "POSITION" accounted for 91.9% of all the hits. This keyword was located in system files, database files, etc. that had no bearing on the term we chosen it for. As an alternative, if we were to limit hits with the work "POSITION" to only those within ten words of "COMPANY", "NEW", or "SALES", we would eliminate the bulk of the false hits.

The cost of identifying the documents with the most hits appears to be minimal compared with it eventual benefit. We recommend the results list always include this (weighted) counter even if the user will be looking at 100% of the listing (at least they can look at the more likely items first).

Using Boolean searches allowing for both unions ("and") and intersections ("or") saves both processing time and review time as the result set is smaller and more precise. In our tests, the intersection was 17 seconds faster (minimal savings) but produced only 9.2% as many hits and 5.1% as many documents (significant savings for the reviewer).

Boolean searches are the staple of the text search world and will continue to be so for the foreseeable future because to their simplicity, familiarity and entrenchment in software.

5.2 Fuzzy Logic

In this section, we discuss our testing of fuzzy logic algorithms. We first compare "keywords" to "keywords with tenses" and "keywords with synonyms". Secondly we discuss fuzzy logic with extra characters, missing characters, and transpositions.

5.2.1 Fuzzy Logic - Keywords, Tenses and Synonyms

Our search for keywords took 6,653 seconds and returned 442,572 hits in 101,084 unique documents. Adding three tenses is similar in processing to adding three additional keywords. The additional processing time should be linearly increased, however; in our test results there was a small decrease to 6,053 seconds. This time variance could be due to a number of non-recurring factors during testing. In our test case, the additional 2,687 documents tagged were probably worth any additional time costs that could have occurred and allowed our goal of completeness to be better served.

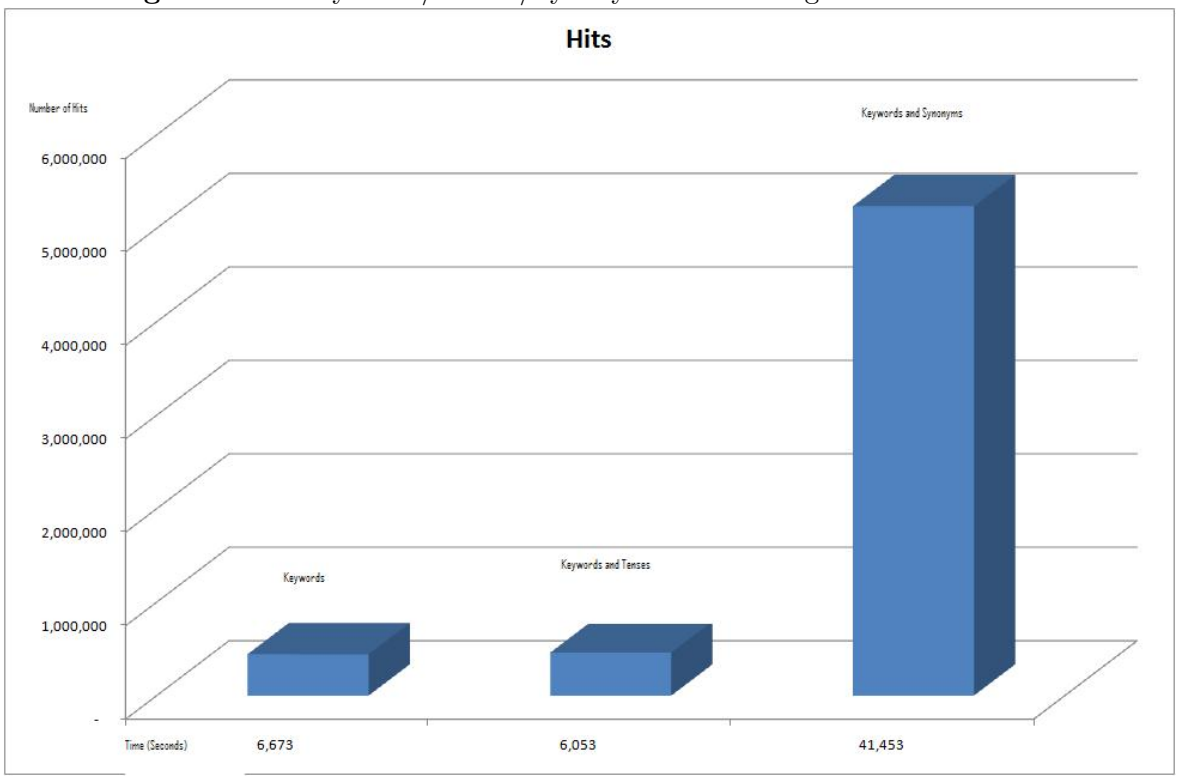
When we included the 27 synonyms, the processing time increased from 6,653 seconds to 48,370 seconds. This increase in time is realatively proportional to the increase in searching for five keywords to searching for 32 keywords. The number of hits increased from 442,572 to 5,236,936. Although this increase in hits (and documents) aids in completeness, it is also too great a percentage of false hits causing greater review time than necessary.

Adding synonyms to the test is not a problem, but blindly adding every synonym generally will be a problem. We suggest each synonym be selected based on the knowledge of the data and/or the knowledge of what documents are needed. Also, as with Boolean searching for "POSITION", and synonym with too many

hits should be reconsidered with mathematical search techniques to limit its results to more likely documents.

Figure 5.1 shows the great increase in hits found by adding the 27 synonyms to the keywords.

Figure 5.1. Keywords/Tenses/Synonyms - Processing Times vs Hits

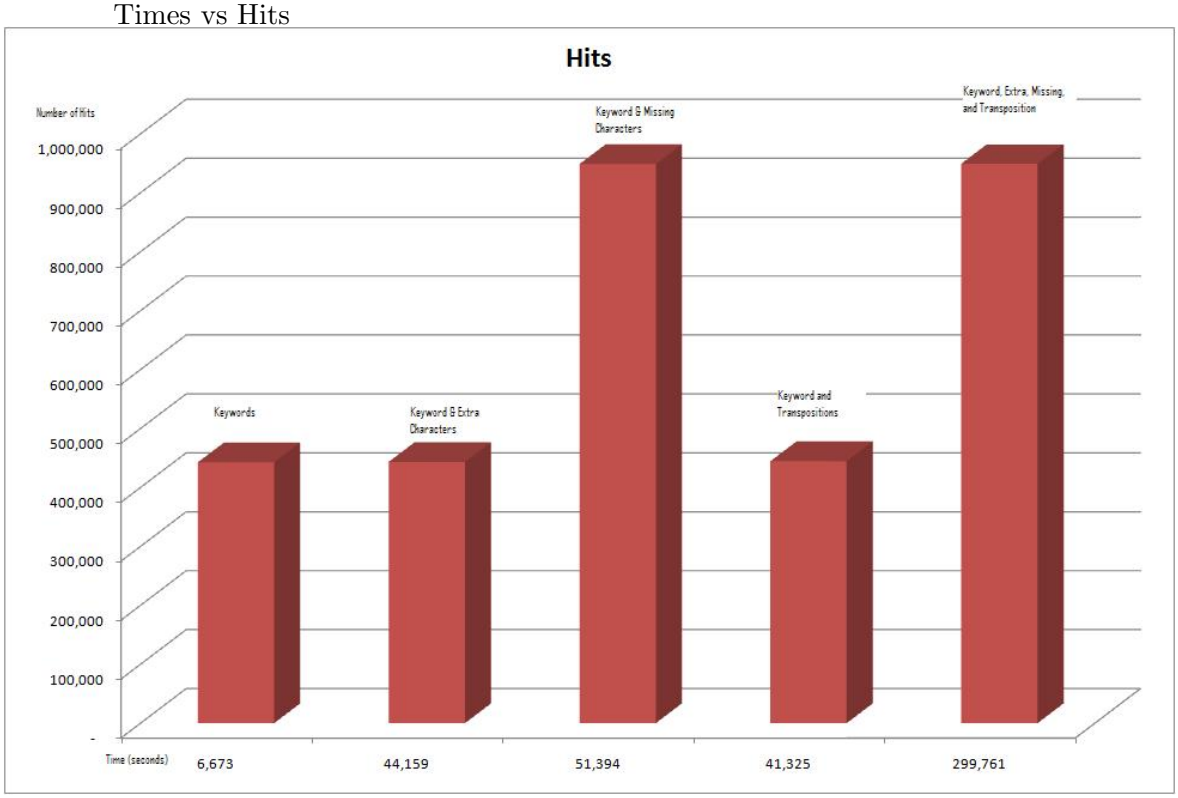


5.2.2 Fuzzy Logic - Extra, Missing & Transposition

Once again, our search for keywords took 6,653 seconds and returned 442,572 hits in 101,084 unique documents. Figure 5.2 shows the increase in hits by searching for extra characters, missing characters, transpositions, and all three at once.

Searching for extra characters within the keywords resulted in an additional

Figure 5.2. Keywords/Transpositions/Extra/Missing - Processing Times vs Hits



100,196 seconds (15 times as long) and found 319 additional documents to consider (0.3% more documents). Although the additional documents may be warranted in some cases, this test is probably better not used based on the law of diminishing returns.

Searching for missing characters identified a flaw in blindly applying the logic: don't look for missing characters in keywords less than or equal to the miniumum length of the the words you selected. With synonyms such as "Bid" and "Pay", a missing character then matches all "BD", "BI", "ID", "PA", "AY", and "PY". A lot of false hits will almost always be found. Prior to fixing the function, a

process of keywords, WordNets, and missing characters resulted in 11,929,808 unique words being tagged as being or containing a keyword or variant. Not searching for missing characters whenever the search word was less than four characters long reduced the number of tagged unique words to a more manageable 745,491. The additional hits were found in 65,276 additional documents with 118,098 additional seconds of processing time.

Finding 65,276 additional documents aids in completeness, but will greatly increase the review time. As previously noted, mathematical routines should also be implemented to reduce the ultimate result set.

Transpositions, similar to extra characters, resulted in 596 additional documents (0.6%) and required an additional 65,855 seconds (almost ten times the processing time). Cost/benefit should be considered here based on the expected value of those 596 documents.

A combination of keywords, missing characters, extra characters and tranpositions resulted in 65,558 additional documents in 416,710 seconds of processing time. As expected, this is not much different than the combined results of the three tests above.

5.3 Context Searching

WordNet did not function as the context finder expected from the research papers. It functioned more as an expanded thesaurus with synonyms and antonyms. All the testing applicable to synonyms above would be applicable to the testing of WordNet also and therefore the tests are not repeated in this thesis (size of the testing was approximately double with 54 WordNets to 27 synonyms, and five keywords in both tests).

Key points with WordNet testing are to not use keywords smaller than the minumum word length and not process missing characters if the keyword equals the minimum word length.

5.4 Methods Involving Mathematical Probabilities

Weighting the value of keywords in any manner - number of keywords in a document, predefined value, or predefined value based on the number of times the keyword occurs in the document; all have little cost in additional processing time and greatly enhance the value of the result set by identifying the most likely documents first (having the highest value). Weighting by itself does not indentify any more or less number of documents.

Weighting of value per occurrence where the weight is determined by the frequency that a keyword occurs in a document may produce less desirable results. The purpose of the document examination needs to be for keywords expected to occur more often, as in code words, rather than a rare occurrence, as in a single email selling company data.

Weighting of value based on proximity to the next or same keyword requires extensive processing time and would only be valuable when multiple words are needed that are not also an exact phrase. For instance, searching for "George" within three words of "Washington" would find "George Washington" regardless of the middle name. For the specific case of this electronic evidence, this type of search would not be beneficial.

Identifying text repetitions where the weight is determined by the frequency that a keyword occurs in a document to the entire electronic evidence involves extensive processing time and may produce less desirable results. The purpose

of the document examination needs to be for keywords expected to occur more often, as in code words, rather than a rare occurrence, as in a single email selling company data.

Baseline - Lingo is an excellent method of fraud investigation when little is known about the documents to begin with. That would be particularly true when examining electronic evidence for fraud or drug dealing. In those cases, you identify code words used frequently to record activity when the code word itself is unknown to you beforehand. This type of search is not comparable to the other text search algorithms explored in this thesis.

Chapter 6

Conclusion, Contributions, and Future Work

6.1 Conclusion

Based on a review of the literature, it is evident that identifying and delivering relevant documents for discovery from the population of all documents are major issues for the courts and costly issues for the parties involved in litigation. The amount of discovery is directly related to the cost of obtaining relevant discovery based on the new rules regarding "reasonably accessible". Thereby, the less expensive we can make the search and extraction of relevant documents, the more documents can be made available to the parties thereby also allowing the parties to better understand their claims and defenses.

Current usage is primarily manual search and Boolean keyword searches. We believe that implementation of advanced technologies using fuzzy logic, context searching and mathematical probabilities could enhance the effectiveness of document relevancy identification (and thus lower costs). These advanced technologies

have individual strengths and weaknesses. Situational usage should be considered based on knowledge of the documents in the database and whether or not fraud is involved which would increase the likelihood code words were utilized. We suggest that rather than relying on a single advanced technology, perhaps two or three could be implemented based on the situation, and the results compared to truly identify the most relevant documents and to rank the documents in order of probable relevancy. The technology should also be used to cluster duplicates and near-duplicates to minimize review time by all parties.

Using a combination of the search technologies, with a proper consolidation of the results, will increase the relevancy of the findings and lower the costs of manual involvement. The benefit of lower costs achieves the first objective of the FRCP. Having lower costs per relevant document discovered then allows for achieving the second objective: more access to documents as the cost of accessing a relevant document is lower. Thus two benefits are derived: (1) lower cost and (2) more access to relevant documents.

These advanced technologies are still in development. They are primarily being used in fields other than legal, but the legal field has begun taking great interest in applying them for discovery requests and document management.

The costs of processing time increase dramatically, but even the most "expensive" processing time is cheap compared to manual labor time. We believe the benefit of identifying the additional (relevant) documents will generally exceed the cost of the additional processing.

We have identified current topics in several fields related to the electronic data searching being addressed in this paper. We identified which ones could influence a proposed solution for efficient searching and identified some that likely would

not be beneficial. We believe the use of text searching, context searching and relevance identification (Web and Peer-to-Peer concepts as well as those encompassed in context searching literature) will all be part of a complete solution to efficient searching of electronic data in litigation. The additional suggestions for non-English languages and non-textual documents will enhance the solution and increase its life.

6.2 Recommendations

As with many things in life, there is no one right answer to the best text searching algorithm.

In our test scenario, we had a general idea of what data was in the electronic documents and a general idea of what data we were looking for. In this situation, text searching makes the most sense (absent a reliable implementation of context searching). We believe a Boolean search, coupled with consideration of tenses, synonyms, antonyms, and common misspellings provides a beneficial search mechanism for a first pass through the electronic evidence. If finding all documents is critical, all extra characters, missing characters, and transpositions can also be included.

The result set should then be reconsidered. Words that identified the most hits should be reconsidered in usage including possibly reapplying the word with another word or words in a mathematical probability search algorithm to reduce the false hits.

Finally, consideration should be given of using a different set of keywords on the result set to exclude documents that are not relevant or are privileged.

Not every instance is like our test scenario. When little is known about the

electronic documents, or fraud is involved or otherwise evidence is being concealed, mathematical search methods may be the best choice. If information is coded, baseline-lingo will work (regardless of the language). In other cases, text repetition will likely find the more meaningful documents.

Speed in the proposed situation is not as critical as in pattern matching, but incorporating the algorithms with more speed will make the solution more competitive with any other solutions that are proposed.

There are many implementation issues to be resolved such as how documents are to be converted/assumed as text as well as the mathematical implementation of the algorithms.

6.3 Summary of Contributions

We have identified several advanced text search methods and implemented some to identify the benefits and costs. We identified that the advanced text search methods can locate additional documents that may be relevant and can weight the findings to allow a user to focus on the most likely relevant documents first.

We idendified which search techniques work better with known data and which with unknown data.

We identified search techniques that produce additional documents, but a much larger cost than the basic search. We suggest these methods be implemented only with complete coverage is required.

We made recommendations for which techniques to be considered for implementation depending on the circumstances of the electronic evidence.

6.4 Future Work

There a number of issues identified for further research, not the least of which is an implementation of context searching that works closer to theory than WordNet. In particular, if the linguistics of WordNet could be applied using the logic of word processing spell checkers and grammar checkers, a usable result could be derived that would save millions in forensic text searching as well be find more complete results.

Our testing focused solely on files within the electronic evidence. Electronic evidence, however, is located by logically (files) and physically (disk partitions). The testing should be expanded to test the full physical view, its abbreviated physical view and in its logical view.

This thesis focused solely on WordNet for context searching. As this focus area is the area believed most likely to produce significant savings, additional research should be done of other context searching implementation attempts.

Other issues related solely to forensic electronic evidence should be addressed including encrypted files and compressed files. These files, including emails in email databases, should be extracted and included in any search routines to assured completeness of results.

Appendix A

Background Of the FRCP

The Federal Rules of Civil Procedure ("FRCP") dictate the procedural rules that must be followed on all Federal civil cases. These rules are generated by advisory committees using public drafts and public comments with ultimate adoption and enforcement by the United States Supreme Court. The initial FRCP was enacted in 1938. From its initial enactment, the focus was on "just" and the rules related to discovery were "to be applied as broadly and liberally as possible" to allow all parties to a case the opportunity to obtain all relevant information and thereby properly evaluate their claims, defenses, and their potential liability or recovery [40].

Discovery volumes tended to increase, reaching a peak in 1970 [40]. Some abuses to the liberal interpretation involved "fishing expeditions" and requests aimed at increasing costs to opposing parties for the purpose of forcing a settlement. In particular, courts wanted to prevent discovery costs from becoming the case determinant factor.

As such abuses became more commonplace; the courts moved the focus towards "inexpensive". Challenges prohibiting "fishing expeditions" were upheld

and the amount of information a party could obtain became more determinant on the party's ability to prove the need of the specific information for the case. At the same time as courts were becoming concerned about the high cost of discovery, the dynamic changing the workplace was the advent of computers and the tremendous amount of data being stored thereon and on related media (backup tapes in particular). This large amount of data became the focus of additional discovery requests. Courts, not trained in technology, were faced with decisions regarding data access, transfer media, and related costs. Costs of just the electronic portion of discovery were becoming case deterministic in some instances.

In 1983, the first amendment was made to the FRCP with the purpose of limiting electronic discovery:

> "The frequency or extent of use of the discovery methods [otherwise permitted under these rules] shall be limited by the court if it determines that: (i) the discovery sought is unreasonably cumulative or duplicative, or is obtainable from some other source that is more convenient, less burdensome, or less expensive; (ii) the party seeking discovery has had ample opportunity by discovery in the action to obtain the information sought; or (iii) the discovery is unduly burdensome or expensive, taking into account the needs of the case, the amount in controversy, limitations on the parties' resources, and the importance of the issues at stake in the litigation." (FRCP Rule 26(b)(2)(C))

The amended rules were a vast departure from the previous last sentence of Rule 26(a): "Unless ... the court orders otherwise, the frequency and use of discovery is not limited." Like the majority of the FRCP, the rule leaves a lot

of discretion to the judge because every case before the judge is unique to some degree.

The FRCP was amended again in 1993 to further emphasize how courts would be responsible for assuring reasonable cost of litigation. The overarching goal of the amended FRCP was stated in the last sentence of FRCP 1. It states that the FRCP "shall be construed to secure the just, speedy, and inexpensive determination of every action." The other key change in 1993 was the specification that the courts were to be actively involved in administering their cases to assure the overarching goal stated in FRCP 1.

The FRCP amendment in 2000 was equally applicable to all types of discovery. In an attempt to limit costs and promote uniformity between courts, the rules allowing the discoverable material were altered from allowing discovery of all information "relevant to the subject matter involved in the pending action" to information "relevant to the subject matter involved in the pending action, whether it relates to the claim or defense of the party seeking discovery or to the claim or defense of any other party." Although not obvious to the layman, this change drastically reduced the amount of discovery a party had to produce but, at the same time, drastically increased the amount of work a party had to do to determine what was required for production versus not.

The latest amendment to the FRCP related to electronic discovery took effect December 1, 2006. A key change caused by this amendment involved the requirement for an early "discovery meeting" of the parties to discuss what information was available, how it was to be preserved, and how it was to be delivered. The parties must disclose what data is held, in what media, and a determination of whether that media was "reasonably accessible." Media not reasonably accessible,

including backup tapes, need not be provided to the requesting party (unless so ordered by the court after the court determines the burdens of production are less than the expected benefits to be derived, also known as "good cause"). The effect on the court of this amendment is that judges must be much more actively involved in the management of the case. The effect for the requesting party is that they must be much more proactive in determining the information they seek to obtain and limiting its cost of production, thereby increasing their likelihood of receiving it.

Bibliography

[1] T. Y. Allman. The impact of the proposed federal ediscovery rules. In *The Sedona Conference Journal*, volume 7, pages 31–41 vol.7, Fall 2006.

[2] K. Ando, T. Tsuji, M. Fuketa, and J. Aoe. An efficient dictionary access method for morphological analysis. In *Systems, Man, and Cybernetics, 1998. 1998 IEEE International Conference on*, volume 3, pages 2876–2881 vol.3, Oct 1998.

[3] G. Arevian. Recurrent neural networks for robust real-world text classification. In *Web Intelligence, IEEE/WIC/ACM International Conference on*, pages 326–329, Nov. 2007.

[4] S. A. Baase. *Computer Algorithms: Introduction to Design and Analysis, 2nd Edition*. Addison-Wesley Publishing Co., Inc., Reading, MA, USA, 1993.

[5] Z. Baker and V. Prasanna. A computationally efficient engine for flexible intrusion detection. *Very Large Scale Integration (VLSI) Systems, IEEE Transactions on*, 13(10):1179–1189, Oct. 2005.

[6] J. Baron. The sedona conference best practices commentary on the use of search and information retrieval methods in e-discovery. In *The Sedona Conference Journal*, volume 8, pages 189–223 vol.8, Fall 2007.

[7] J. Baron. The trec legal track: Origins and reflections on the first year. In *The Sedona Conference Journal*, volume 8, pages 251–9 vol.8, Fall 2007.

[8] K. Bass. The sedona conference report on the markman process. In *The Sedona Conference Journal*, volume 7, pages 205–39 vol.7, Fall 2006.

[9] R. S. Boyer and J. S. Moore. A fast string searching algorithm. *Commun. ACM*, 20(10):762–772, 1977.

[10] D. Breslauer, L. Colussi, and L. Toniolo. On the comparison complexity of the string prefix-matching problem. *BRICS Report Series RS-95-46*, RS-95-46, Aug 1995.

[11] Y.-C. Chang and S.-M. Chen. A new query reweighting method for document retrieval based on genetic algorithms. *Evolutionary Computation, IEEE Transactions on*, 10(5):617–622, Oct. 2006.

[12] D. Cohan. From the acba: New federal e-discovery rules take effect on december 1. *The Lawyers Journal (Allegheny County Bar Association)*, Nov 2006.

[13] R. Cole. Tight bounds on the complexity of the boyer–moore string matching algorithm. *SIAM Journal on Computing*, 23(5):1075–1091, 1994.

[14] M. Cowper and J. Rosenthal. Not your mother's rule 26(f) conference anymore. In *The Sedona Conference Journal*, volume 8, Fall 2007.

[15] I. De Felipe, V. Hristidis, and N. Rishe. Keyword search on spatial databases. In *Data Engineering, 2008. ICDE 2008. IEEE 24th International Conference on*, pages 656–665, April 2008.

[16] S. Debnath, P. Mitra, N. Pal, and C. Giles. Automatic identification of informative sections of web pages. *Knowledge and Data Engineering, IEEE Transactions on*, 17(9):1233–1246, Sept. 2005.

[17] S. Dharmapurikar and J. Lockwood. Fast and scalable pattern matching for network intrusion detection systems. *Selected Areas in Communications, IEEE Journal on*, 24(10):1781–1792, Oct. 2006.

[18] Bank of Am. Corp. v. SR Int'l Bus. Ins. Co., 2007.

[19] In re. Seroquel Prods. Liab. Litig., 2007.

[20] D. Fisher. The data explosion.(h5 and its services). *Forbes*, 180(6):72, Oct. 1 2007.

[21] M. Fletcher. E-discovery falls hardest on the insurance industry. *BI Industry Focus*, page 13, May 2007.

[22] W. B. Frakes and R. Baeza-Yates. *Information Retrieval: Data Structures and Algorithms*. Prentice Hall, Englewood Cliffs, NJ, USA, 1992.

[23] S. L. Garfinkel. Aff: a new format for storing hard drive images. *Commun. ACM*, 49(2):85–87, 2006.

[24] M. Gordon, W. Fan, and P. Pathak. Adaptive web search: Evolving a program that finds information. *Intelligent Systems, IEEE*, 21(5):72–77, Sept.-Oct. 2006.

[25] S. Harabagiu and D. Moldovan. A parallel system for text inference using marker propagations. *Parallel and Distributed Systems, IEEE Transactions on*, 9(8):729–747, Aug 1998.

[26] C. Hosmer. Digital evidence bag. *Commun. ACM*, 49(2):69–70, 2006.

[27] M.-Y. Kan and Y. F. Tan. Record matching in digital library metadata. *Commun. ACM*, 51(2):91–94, 2008.

[28] J. Krause. What a concept!. *ABA Journal*, 89(8):60, 2003.

[29] J. Krause. The top ten in tech. *ABA Journal*, 90(12):34 – 40, 2004.

[30] J. Krause. Don't try this at home. *ABA Journal*, 91(3):59 – 60, 2005.

[31] J. Krause. The paperless chase. *ABA Journal*, 91(4):48 – 53, 2005.

[32] J. Krause. Staying above the e-flood. *ABA Journal*, 92(5):61, 2006.

[33] J. Krause. E-discovery gets real. *ABA Journal*, 93(2):44 – 51, 2007.

[34] M. Kulekci. Tara: An algorithm for fast searching of multiple patterns on text files. In *Computer and information sciences, 2007. iscis 2007. 22nd international symposium on*, pages 1–6, Nov. 2007.

[35] R.-T. Liu, N.-F. Huang, C.-N. Kao, and C.-H. Chen. A fast pattern matching algorithm for network processor-based intrusion detection system. In *Performance, Computing, and Communications, 2004 IEEE International Conference on*, pages 271–275, 2004.

[36] R. W. P. Luk. Chinese string searching using the kmp algorithm. In *Proceedings of the 16th conference on Computational linguistics*, pages 1111–1114, Morristown, NJ, USA, 1996. Association for Computational Linguistics.

[37] W. Ma, W. Fang, G. Wang, and J. Liu. Concept index for document retrieval with peer-to-peer network. In *Software Engineering, Artificial Intelligence, Networking, and Parallel/Distributed Computing, 2007. SNPD 2007. Eighth ACIS International Conference on*, volume 3, pages 1119–1123, 30 2007-Aug. 1 2007.

[38] A. M. Mason. Recent development: Throwing out the (electronic) trash: True deletion would soothe e-discovery woes. *Minnesota Journal of Law, Science & Technology*, 7:777, May 2006.

[39] M. Mazza, E. K. Quesada, and A. L. Sternberg. In pursuit of frcpa: Creative approaches to cutting and shifting costs of discovery of electronically stored information. *Richmond Journal of Law & Technology*, page 11, Spring 2007.

[40] H. S. Noyes. Good cause is bad medicine for the new e-discovery rules. *Harvard Journal of Law & Technology*, 21:49, Fall 2007.

[41] S. Osinski and D. Weiss. A concept-driven algorithm for clustering search results. *Intelligent Systems, IEEE*, 20(3):48–54, May-June 2005.

[42] K. Patterson, C. Watters, and M. Shepherd. Document retrieval using proximity-based phrase searching. In *Hawaii International Conference on System Sciences, Proceedings of the 41st Annual*, pages 137–137, Jan. 2008.

[43] D. Pavlov, E. Manavoglu, C. Giles, and D. Pennock. Collaborative filtering with maximum entropy. *Intelligent Systems, IEEE*, 19(6):40–47, Nov.-Dec. 2004.

[44] C. Peery, W. Wang, A. Marian, and T. Nguyen. Fuzzy multi-dimensional search in the wayfinder file system. In *Data Engineering, 2008. ICDE 2008. IEEE 24th International Conference on*, pages 1588–1591, April 2008.

[45] J. Pelfrene, S. Abdeddaim, and J. Alesandre. Extracting approximate patterns. In *Combinatorial Pattern matching, 14th Annual Symposium*. Springer, 2003.

[46] A. Robles-Kelly and E. Hancock. Edit distance from graph spectra. In *Computer Vision, 2003. Proceedings. Ninth IEEE International Conference on*, pages 234–241 vol.1, Oct. 2003.

[47] H. L. Roitblat. Search and information retrieval science. In *The Sedona Conference Journal*, volume 8, pages 225–38 vol.8, Fall 2007.

[48] D. Roussinov, W. Fan, and J. Robles-Flores. Beyond keywords: Automated question answering on the web. *Commun. ACM*, 51(9):60–65, 2008.

[49] S. Sarkar, P. Chakrabarti, and S. Ghose. A framework for learning in search-based systems. *Knowledge and Data Engineering, IEEE Transactions on*, 10(4):563–575, Jul/Aug 1998.

[50] T.-F. Sheu, N.-F. Huang, and H.-P. Lee. A novel hierarchical matching algorithm for intrusion detection systems. In *Global Telecommunications Conference, 2005. GLOBECOM '05. IEEE*, volume 3, pages 5 pp.–, Nov.-2 Dec. 2005.

[51] C. G. Shilling. Litigation: Electronic discovery: Litigation crashes into the digital age. *New Hampshire Bar Journal*, page 22, Spring 2006.

[52] R. Venkata and M. A. Geibelson. Computer counselor: Overcoming e-discovery challenges with new technologies. *Los Angeles Lawyer*, page 46, June 2007.

[53] B. Watson. *Taxonomies and Toolkets of Regular Language Algorithms*. PhD thesis, Eindhoven University of Technology, The Netherlands, 1995. Ph.D. Dissertation, Eindhoven University of Technology, The Netherlands.

[54] X. Wei and Q. Zhang. Approach of text search based on semantic parsing model. In *Fuzzy Systems and Knowledge Discovery, 2007. FSKD 2007. Fourth International Conference on*, volume 2, pages 355–359, Aug. 2007.

[55] K. J. Withers. Electronically stored information: The december 2006 amendments to the federal rules of civil procedure. In *The Sedona Conference Journal*, volume 7, pages 1–30 vol.7, Fall 2006.

[56] W. Yi, Y. Sun, S. Zhang, Y. Wu, and Z. Chu. The application and research of ontology construction technology. In *Knowledge Discovery and Data Mining, 2008. WKDD 2008. First International Workshop on*, pages 618–623, Jan. 2008.

[57] Q. Yong, H. Peijie, H. Yu, and Q. Ya-nan. The research and design of the semantic search engine based on ontology. In *Semantics, Knowledge and Grid, Third International Conference on*, pages 610–611, Oct. 2007.

[58] B. Yu, G. Li, K. Sollins, and A. K. H. Tung. Effective keyword-based selection of relational databases. In *SIGMOD '07: Proceedings of the 2007 ACM SIGMOD international conference on Management of data*, pages 139–150, New York, NY, USA, 2007. ACM.

[59] R. Zhang and Z. Zhang. Effective image retrieval based on hidden concept discovery in image database. *Image Processing, IEEE Transactions on*, 16(2):562–572, Feb. 2007.

[60] Y. Zhu and Y. Hu. Enhancing search performance on gnutella-like p2p systems. *Parallel and Distributed Systems, IEEE Transactions on*, 17(12):1482–1495, Dec. 2006.