2009-06

Detection and Monitoring of Improvised Explosive Device Education Networks Through the World Wide Web.

Stinson, Robert T. III

Monterey, California: Naval Postgraduate School

http://hdl.handle.net/10945/7289

NAVAL POSTGRADUATE SCHOOL

MONTEREY, CALIFORNIA

THESIS

DETECTION AND MONITORING OF IMPROVISED EXPLOSIVE DEVICE EDUCATION NETWORKS THROUGH THE WORLD WIDE WEB

by

Robert T. Stinson III

June 2009

Thesis Advisor: Weilian Su
Second Reader: Douglas Fouts

THIS PAGE INTENTIONALLY LEFT BLANK

REPORT DOCUMENTATION PAGE	*Form Approved OMB No. 0704-0188*

1. AGENCY USE ONLY *(Leave blank)*	**2. REPORT DATE** June 2009	**3. REPORT TYPE AND DATES COVERED** Master's Thesis

4. TITLE AND SUBTITLE Detection and Monitoring of Improvised Explosive Device Education Networks Through the World Wide Web

5. FUNDING NUMBERS

6. AUTHOR(S) Robert T. Stinson III

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
Naval Postgraduate School
Monterey, CA 93943-5000

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING /MONITORING AGENCY NAME(S) AND ADDRESS(ES)
N/A

10. SPONSORING/MONITORING AGENCY REPORT NUMBER

11. SUPPLEMENTARY NOTES The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government.

12a. DISTRIBUTION / AVAILABILITY STATEMENT
Approved for public release; distribution is unlimited

12b. DISTRIBUTION CODE

13. ABSTRACT (maximum 200 words)

As the information age comes to fruition, terrorist networks have moved mainstream by promoting their causes via the World Wide Web. In addition to their standard rhetoric, these organizations provide anyone with an Internet connection the ability to access dangerous information involving the creation and implementation of Improvised Explosive Devices (IEDs). Unfortunately for governments combating terrorism, IED education networks can be very difficult to find and even harder to monitor. Regular commercial search engines are not up to this task, as they have been optimized to catalog information quickly and efficiently for user ease of access while promoting retail commerce at the same time. This thesis presents a performance analysis of a new search engine algorithm designed to help find IED education networks using the Nutch open-source search engine architecture. It reveals which web pages are more important via references from other web pages regardless of domain. In addition, this thesis discusses potential evaluation and monitoring techniques to be used in conjunction with the proposed algorithm.

14. SUBJECT TERMS Improvised Explosive Device, IED, Nutch, WebCrawler

15. NUMBER OF PAGES 123

16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	**18. SECURITY CLASSIFICATION OF THIS PAGE**	**19. SECURITY CLASSIFICATION OF ABSTRACT**	**20. LIMITATION OF ABSTRACT**
Unclassified	Unclassified	Unclassified	UU

NSN 7540-01-280-5500 Standard Form 298 (Rev. 2-89)
Prescribed by ANSI Std. 239-18

THIS PAGE INTENTIONALLY LEFT BLANK

DETECTION AND MONITORING OF IMPROVISED EXPLOSIVE DEVICE EDUCATION NETWORKS THROUGH THE WORLD WIDE WEB

Robert T. Stinson III
Lieutenant, United States Navy
B.S., Maine Maritime Academy, 2003

Submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE IN ELECTRICAL ENGINEERING

from the

NAVAL POSTGRADUATE SCHOOL
June 2009

Author: Robert T. Stinson III

Approved by: Weilian Su
Thesis Advisor

Douglas Fouts
Second Reader

Jeffrey B. Knorr
Chairman, Department of Electrical and Computer Engineering

THIS PAGE INTENTIONALLY LEFT BLANK

ABSTRACT

As the information age comes to fruition, terrorist networks have moved mainstream by promoting their causes via the World Wide Web. In addition to their standard rhetoric, these organizations provide anyone with an Internet connection the ability to access dangerous information involving the creation and implementation of Improvised Explosive Devices (IEDs). Unfortunately for governments combating terrorism, IED education networks can be very difficult to find and even harder to monitor. Regular commercial search engines are not up to this task, as they have been optimized to catalog information quickly and efficiently for user ease of access while promoting retail commerce at the same time. This thesis presents a performance analysis of a new search engine algorithm designed to help find IED education networks using the Nutch open-source search engine architecture. It reveals which web pages are more important via references from other web pages regardless of domain. In addition, this thesis discusses potential evaluation and monitoring techniques to be used in conjunction with the proposed algorithm.

THIS PAGE INTENTIONALLY LEFT BLANK

TABLE OF CONTENTS

I. INTRODUCTION **1**
- **A. PROBLEM OVERVIEW** **1**
- **B. RESEARCH OBJECTIVES** **2**
- **C. THESIS ORGANIZATION** **2**

II. BACKGROUND **3**
- **A. THE IED THREAT** **3**
  - **1. Definition** **3**
  - **2. Generic IED Composition** **4**
  - **3. Brief History of Use** **5**
  - **4. Current Concerns** **6**
- **B. INFORMATION RETRIEVAL** **6**
  - **1. Retrieval Strategies** **7**
    - ***a. Vector Space Model*** ***7***
    - ***b. Language Model*** ***8***
    - ***c. Probabilistic Retrieval*** ***9***
    - ***d. Inference Networks*** ***9***
    - ***e. Extended Boolean Retrieval*** ***10***
    - ***f. Latent Semantic Indexing*** ***10***
    - ***g. Neural Networks*** ***10***
    - ***h. Fuzzy Set Retrieval*** ***10***
  - **2. WebCrawler Algorithms** **11**
    - ***a. Breadth-first*** ***11***
    - ***b. Best-first*** ***12***
    - ***c. Shark-search*** ***14***
    - ***d. Info-spiders*** ***14***
    - ***e. PageRank*** ***16***
- **C. PAGERANK ALGORITHM VARIATIONS** **19**
  - **1. Topic-sensitive** **19**
  - **2. Weighted** **21**
  - **3. Usage-based** **22**
  - **4. TimeRank** **24**
  - **5. DYNA-RANK** **24**

III. NUTCH **27**
- **A. INTRODUCTION** **27**
- **B. ARCHITECTURE** **27**
- **C. LUCENE** **28**
- **D. ADAPTIVE OPIC** **30**

IV. ALGORITHM DEVELOPMENT **33**
- **A. PROBLEM DEFINITION** **33**
- **B. ASSUMPTIONS** **33**

C. NEW ALGORITHM 34

V. PERFORMANCE MEASUREMENTS..37
A. EXPERIMENTAL SETUP 37
1. Hardware & Operating System Configurations 37
2. Simulation Configuration..37
B. BENCHMARKING 37
1. Low Complexity Network 39
2. Medium Complexity Network 45
3. High Complexity Network 51

VI. CONCLUSIONS 57
A. SUMMARY 57
B. CONCLUSIONS 57
C. FUTURE WORK...58

APPENDIX A. NUTCH XML CONFIGURATION FILE 59

APPENDIX B. LUCENE SCORING EXAMPLE 79

APPENDIX C. SIMULATION 3 WEB LINK GRAPH 81

LIST OF REFERENCES 101

INITIAL DISTRIBUTION LIST 105

LIST OF FIGURES

Figure 1. Representation of a generic Explosive Train 4
Figure 2. Generic Improvised Explosive Device Electrical Diagram 5
Figure 3. Representation of documents in a 3-dimensional vector space (From [8]). 8
Figure 4. Breadth-first Crawler Outline. 12
Figure 5. Breadth-first Crawler Tree Diagram Example. 12
Figure 6. Best-first Crawler Outline. 13
Figure 7. Best-first Crawler Tree Diagram Example. 13
Figure 8. Info-Spider Architecture (From [15]). 15
Figure 9. Simplified PageRank Calculation (From [17]). 17
Figure 10. Loop Which Acts as a Rank Sink (From [17]). 18
Figure 11. Nutch search engine high level design (From [25]). 28
Figure 12. Typical application integration with Lucene (From [26]). 29
Figure 13. Simulation 1: Low Complexity Web Link Graph. 40
Figure 14. Simulation 1: Overall OPIC Scores. 41
Figure 15. Simulation 1: Depth Level 2 OPIC Scores. 42
Figure 16. Simulation 1: Depth Level 2 OPIC Score Variations. 42
Figure 17. Simulation 1: Depth Level 2 OPIC Score % Variations. 43
Figure 18. Simulation 1: Depth Level 5 OPIC Scores. 44
Figure 19. Simulation 1: Depth Level 5 OPIC Score Variations. 44
Figure 20. Simulation 1: Depth Level 5 OPIC Score % Variations. 45
Figure 21. Simulation 2: Medium Complexity Web Link Graph. 46
Figure 22. Simulation 2: Overall OPIC Scores. 48
Figure 23. Simulation 2: Depth Level 5 OPIC Scores. 49
Figure 24. Simulation 2: Depth Level 5 OPIC Score Variations. 50
Figure 25. Simulation 2: Depth Level 5 OPIC Score % Variations. 50
Figure 26. Simulation 3: Overall OPIC Scores. 51
Figure 27. Simulation 3: Depth Level 3 OPIC Scores. 52
Figure 28. Simulation 3: Depth Level 3 OPIC Score % Variations. 53
Figure 29. Simulation 3: Depth Level 4 OPIC Scores. 54
Figure 30. Simulation 3: Depth Level 4 OPIC Score % Variations. 54
Figure 31. Simulation 3: Depth Level 5 OPIC Scores. 55
Figure 32. Simulation 3: Depth Level 5 OPIC Score % Variations. 55

THIS PAGE INTENTIONALLY LEFT BLANK

LIST OF TABLES

Table 1. Small term-by-document matrix (From [8]). 7
Table 2. PageRank Recursion Equation Calculations. 17
Table 3. Harvest Rate of Topics (From [20]). 20
Table 4. "scholarship" Query Results (From [21]). 22
Table 5. Original OPIC versus New OPIC Scoring. 35
Table 6. Probability of Creating Specific Document Links. 38
Table 7. Simulation 1, Low Complexity Web Link Graph Data. 40
Table 8. Simulation 2, Medium Complexity Web Link Graph Data. 47

THIS PAGE INTENTIONALLY LEFT BLANK

EXECUTIVE SUMMARY

As the Global War on Terrorism has progressed, the use of Improvised Explosive Devices (IEDs) against coalition forces, governments and civilian populations fighting terrorism has drastically increased. One reason for this is easy access to the World Wide Web [1]. The World Wide Web provides anyone with both a computer and Internet connection access to a plethora of information within the touch of a button; anything from encyclopedias to current news, pictures to movies, basic chemistry to the construction of IEDs. In conjunction with this dangerous information being easily accessible, the users and publishers have the potential to remain anonymous. Complicating things further, terrorist organizations are exploiting this resource by creating IED education networks via the World Wide Web to quickly and efficiently propagate the information to their supporters and operatives.

One possible solution to this problem is an IED specific WebCrawler. An IED WebCrawler has the potential to quickly locate terrorist IED education networks via the World Wide Web. Once found, these networks can be either shutdown, monitored, or infiltrated depending on the objectives of the government or agency employing the search engine. By locating these networks, responsibility for particular attacks can be properly assigned to specific terrorist networks, with particular IED countermeasures deployed to prevent further loss of life and damage to property.

To accomplish this, the Nutch project was selected as the optimum search engine to use. Its versatile plug-in architecture allows for the flexibility needed to design an IED specific WebCrawler while keeping implementation costs low. To improve performance, the original algorithm was modified to dramatically enhance the web-link scores of documents already discovered during a search. Multiple simulations were used to test the new algorithm variations with moderate success.

Overall, the Nutch search engine is well suited for the above task, as well as monitoring the newly discovered networks. Under its current design, Nutch is capable of maintaining a previously found web-link database while updating it with new documents and scores. Inflation issues concerning web-link scores arise depending on the number and frequency of re-crawls conducted but is minor unless looking to discover new networks after an initial crawl. This thesis does not address foreign language issues, robot exclusion protocols or other security measures used to prevent search engines from accessing a web page.

ACKNOWLEDGMENTS

First and foremost, I want to thank my family, Jamie, Elizabeth, Jacob, and Isabel for supporting me through the numerous late nights of reading, writing and simulation. Without their support, this thesis would never have materialized. Second, I would like to thank my parents for all of their years of continuous support and teaching me to chase my dreams.

My thanks to Professor Weilian Su for the numerous hours spent discussing and helping me prepare this thesis. It has been an enlightening and life changing experience. In addition, many thanks to Commander/Professor Alan Shaffer for taking the time to teach me how to properly program Java.

Lastly, I wish to thank Doug Cutting and the open source community for creating and supporting both the Lucene and Nutch projects. Without your insight, dedication to excellence and constant improvements, this thesis would not exist.

THIS PAGE INTENTIONALLY LEFT BLANK

I. INTRODUCTION

A. PROBLEM OVERVIEW

After the terrorist attacks of September 11, 2001, the United States of America was forced to deal with a threat the likes of which had never been seen before. A small network of individuals was able to effectively kill thousands of people with multiple airborne Improvised Explosive Devices (IEDs). Following the attacks, the U.S. launched the Global War on Terrorism; a massive anti-terrorism campaign with the goals of bringing to justice the people responsible for the 9/11 attacks, as well as the terrorist organization that planned it, al-Qaeda. The end state objective of the campaign is to continue to prevent the emergence and sustainment of other terrorist organizations, while permanently degrading the abilities of these organizations to engage in terrorism effectively.

As the Global War on Terrorism has progressed, the use of IEDs against coalition forces, governments and civilian populations fighting terrorism has drastically increased. One reason for this is easy access to the World Wide Web [1]. The World Wide Web provides anyone with both a computer and Internet connection access to a plethora of information within the touch of a button; anything from encyclopedias to current news, pictures to movies, basic chemistry to the construction of IEDs. In conjunction with this dangerous information being easily accessible, the users and publishers have the potential to remain anonymous. Complicating things further, terrorist organizations are exploiting this resource by creating IED education networks via the World Wide Web to quickly and efficiently propagate the information to their supporters and operatives.

One possible solution to this problem is an IED specific WebCrawler. An IED WebCrawler has the potential to quickly locate terrorist IED education networks via the World Wide Web. Once found, these networks can be either shutdown, monitored, or infiltrated depending on the objectives of the government or agency employing the search engine. By locating these networks, responsibility for particular attacks can be properly assigned to specific terrorist networks, with particular IED countermeasures deployed to prevent further loss of life and damage to property.

B. RESEARCH OBJECTIVES

The research objectives of this thesis were to create a random network generator capable of generating a random network to be used in testing the effectiveness of search engine algorithms, while simultaneously developing a new search engine algorithm aimed at identifying IED education networks accessible via the World Wide Web. Additionally, this thesis will briefly mention how an IED WebCrawler could be modified and used as a monitoring device, successfully tracking changes and updates to the IED education networks.

C. THESIS ORGANIZATION

This thesis consists of six chapters. The present chapter states an overview of the problem, objectives, and thesis organization. Chapter II contains a brief description of IEDs, retrieval strategies and a current survey of web crawling algorithms. Chapter III describes the Nutch open-source search engine project. Chapter IV discusses the development of a new search engine algorithm. Chapter V presents the subjective performance measurements, compares different algorithms and determines relative effectiveness. Chapter VI summarizes this thesis, draws conclusions and provides future research recommendations.

II. BACKGROUND

A. THE IED THREAT

1. Definition

In 2008, the United States Department of Defense updated the definition of an Improvised Explosive Device as:

> a device placed or fabricated in an improvised manner incorporating destructive, lethal, noxious, pyrotechnic, or incendiary chemicals and designed to destroy, incapacitate, harass, or distract. [2]

Previously, an IED was only thought to incorporate military stores with non-military components, but this concept is changing. Militaries around the world are incorporating off-the-shelf commercial technology to lower production costs, blurring the line between military and non-military components. What makes an IED special is the fact that some part of the device, generally with regards to the triggering or delivery mechanism, is altered from its original manufactured state to an "improvised" one.

The reason a standard IED definition is hard to agree upon is due to this fact: IEDs are "improvised." For example, there are over 16 commonly used acronyms within the U.S. military to describe different IEDs, with no real consensus on how they are specifically classified: Chemical and Biological IED (CBIED), Command Detonated IED (CDIED), Chemical IED (CIED), Command Wire IED (CWIED), Deep Buried IED (DBIED), Explosively Formed Penetrator (EFP), House-Borne IED (HBIED), Home Made Explosives (HME), Improvised Anti-Armor Grenade (IAAG), Person-Borne IED (PBIED), Radio-Controlled IED (RCIED), Suicide IED (SIED), Suicide Vehicle-Borne IED (SVBIED), Vehicle-Borne IED (VBIED), Victim Operated IED (VOIED), Water-Borne IED (WBIED). Other examples include "sticky" and "flying" IEDs, specifically referencing magnetic and rocket assisted mortars. Overall, there is no easy way to classify all of the different potential types of IEDs.

2. Generic IED Composition

In general, an Improvised Explosive Device works by completing an explosive train from start to finish. An explosive train is defined by the U.S. Department of Defense as "a succession of initiating and igniting elements arranged to cause a charge to function [2]." Figure 1 provides a generic line diagram of an IED explosive train. At the beginning of the chain, a fuse is needed to initiate the reaction, with an accompanying agent being the means of ignition. Fuse examples range greatly from a slow burning piece of twine or cotton to a trail of black powder, etc...; but all require some type of ignition source to start the chain reaction. Next is the primer, which is a container that holds the explosive agent. A detonator, also known as a blasting cap, is then used to create a small explosion which will cause the main charge to ignite. Safety relays and arming leads are usually incorporated in the device in order to prevent early detonation. Booster charges are optional depending on the main charge composition. If the explosive agent being used requires a large amount of energy to ignite its chemical agent, then a booster charge will be required. Multiple booster charges can be used to create a cascade effect if the main charge is in need of the extra energy.

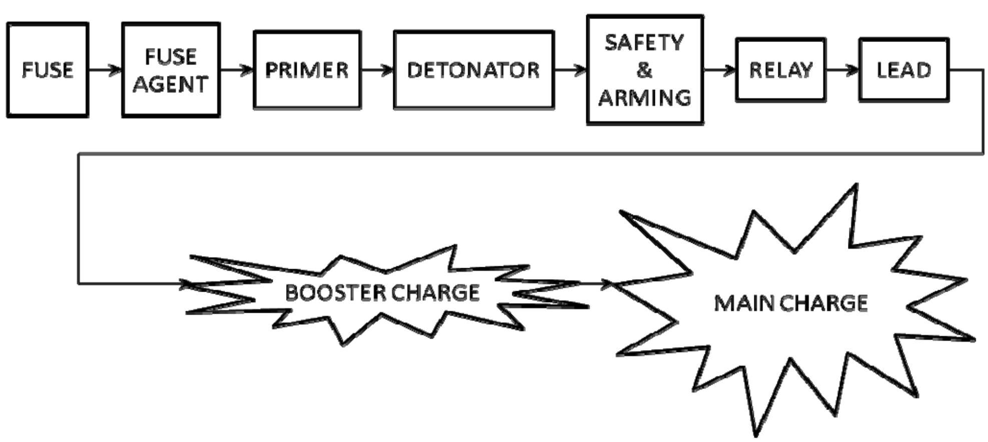

Figure 1. Representation of a generic Explosive Train

Another way to look at IEDs is from an electrical point of view, provided in Figure 2. Initially, a power source is needed to start the reaction. Power sources for such devices range in various sizes, from a small 9V battery to a large car or truck battery.

Essentially, anything can be used as a power source, as long as it has the ability to store a voltage potential and deliver enough current to initiate the explosive reaction. Next, an optional arming switch can be incorporated in the device to prevent premature detonation; otherwise a direct connection would be made. A trigger is then used to complete the circuit, allowing the blasting cap to ignite the main charge.

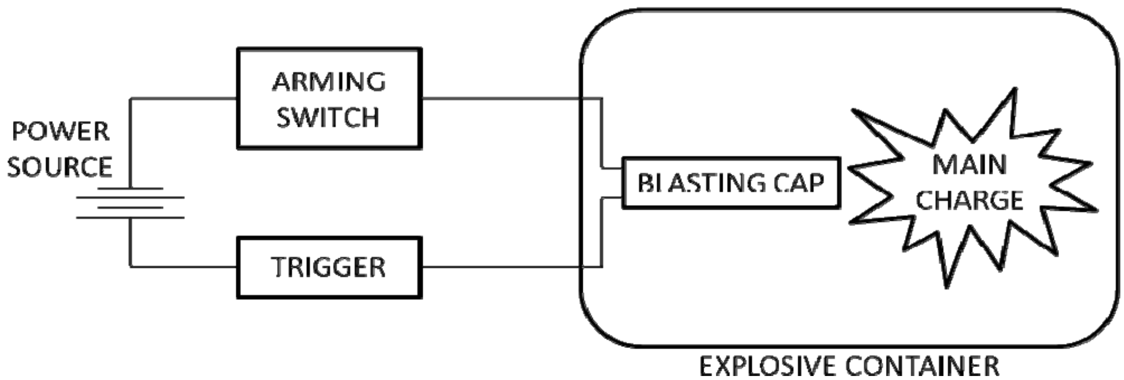

Figure 2. Generic Improvised Explosive Device Electrical Diagram

3. Brief History of Use

Throughout all of mankind's history, many different groups of people have turned to violent means in order to further a cause; whether through formal military measures or small pockets of resistance against a common foe. In general, small groups with minimal amounts of money were forced to become creative in order to effectively attack their enemies, furthering their objectives. The first prominent example of IED use came in the 20th century during the Belarus "Rail War." In 1943, Belarusian partisans waged war with IEDs against the German army; disrupting supply lines and destroying garrisons in order to prevent their advance [3]. During the Vietnam War, Viet Cong soldiers used numerous IEDs against American forces, causing approximately one third of all U.S. casualties [4]. Since then, numerous separatists groups located worldwide have adopted their use, including groups located in areas such as Northern Ireland, Iraq, Afghanistan, Israel, Lebanon and Chechnya.

As the war in Iraq comes to a close, and the U.S. led war in Afghanistan rages on, it has become clear that terrorist groups' weapon of choice is the IED. In response to the high casualty rates in both locations, the United States created the Joint IED Defeat Organization (JIEDDO) to combat the growing epidemic. Since its inception, JIEDDO has effectively assisted in countering IED use; lowering the average number of IED events Coalition forces encounter each month in Iraq and Afghanistan to approximately 900, down from a high of 2,800 in 2007 [5].

4. Current Concerns

Unfortunately, with the advent of the World Wide Web, anyone with a computer and Internet connection can find information on how to create an IED. For example, a well known anarchy book: The Jolly Roger's CookBook can easily be found online within minutes of a Google search involving terms related to IEDs: anarchy, bomb, and explosive [6]. This detailed case-in-point illustrates just how vast the problem has become. Terrorist networks are exploiting the Internet and creating vast IED education networks to further their cause.

B. INFORMATION RETRIEVAL

The science of information retrieval has come to the forefront of Internet research within the last two decades. As more and more people use search engines to find pertinent information, the need to properly classify relevant documents continues to grow and evolve. One success story demonstrating such importance is Google. Their search engine took into account more factors than any other, considering not just term frequencies, but "whether words or phrases on web pages were close together or far apart, what their font size was, whether they were capitalized or in lowercase type [7]." Learning to evaluate what information is important or not is the first step in developing a successful search algorithm. Different methods classifying retrieval strategies and known ranking algorithms are presented below.

1. Retrieval Strategies

a. Vector Space Model

The vector space model is a retrieval strategy widely used in some of today's most successful WebCrawlers. The model works by representing each document as a vector in multiple dimensions, with the number of dimensions dependent on the quantity of terms entered into the query. If a term is found to be in a document, the value of the vector for that document is non-zero. These values or similarity coefficients (SCs) are then compared to determine which documents are the most relevant to a given input query. Specific calculations involving similarity coefficients vary between WebCrawlers and are usually considered proprietary information.

A simple term-by-document matrix example is presented in Table 1 with a document in each column and corresponding term in each row. The value indicated represents the term's frequency within that document. In this specific case, term frequency will be no more than one. For example, Term 3 appears in both Document 2 and Document 3 but not in the other example Documents. To further grasp this concept, Figure 3 demonstrates what Table 1's term-by-document matrix looks like as a vector in 3-dimensional space. If term frequencies were actually considered in this example, an additional normalizing factor would have to be applied to the matrix.

	Document 1	Document 2	Document 3	Document 4
Term 1	1	0	1	0
Term 2	0	0	1	1
Term 3	0	1	1	0

Table 1. Small term-by-document matrix (From [8]).

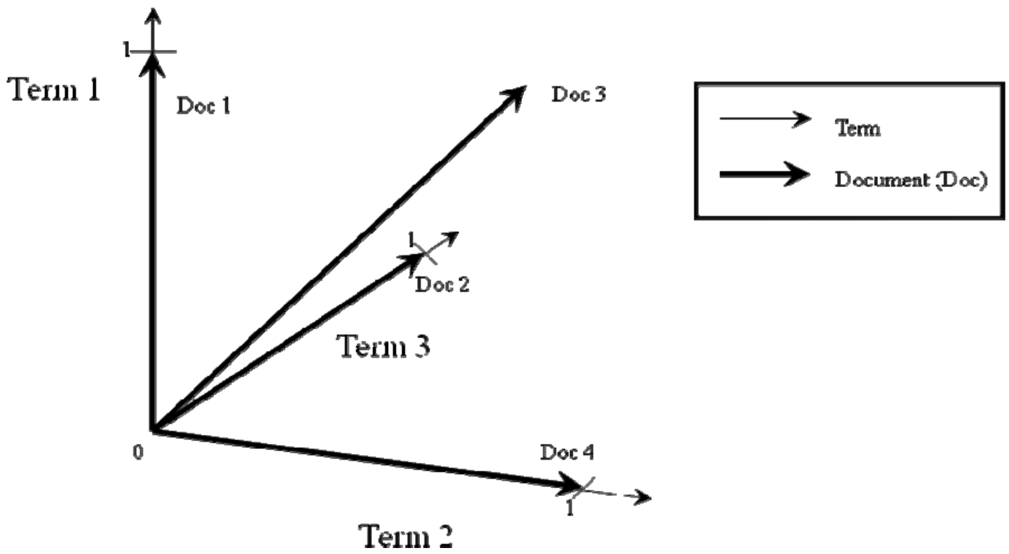

Figure 3. Representation of documents in a 3-dimensional vector space (From [8]).

In general, problems arise with this method due to the fact that the frequency of terms does not always correlate to relevance, nor does the single inclusion of a query term. The order in which terms appear does not factor in as well. Other methods are used in conjunction with the vector space model to enhance the quality of WebCrawler's search results. Relevancy ranks vary among them and are solely dependent on the ranking algorithm.

b. Language Model

The language model is defined as a "probabilistic mechanism for 'generating' a piece of text” [9]. In other words, it generates a distribution for all the possible word patterns and assigns a similarity coefficient based on the likelihood of a document generating a query. Contextual information can be used as well to generate the distribution for more complex algorithms. The difficulty involving this method is that a model is built for each document, making the method extremely computationally intensive.

c. *Probabilistic Retrieval*

Probabilistic retrieval has many variant forms but two fundamental approaches that differ based on usage patterns and query terms. The first method involves usage patterns to predict relevance while the other uses query information to determine relevance. In [10], Fuhr shows that the probability of a document will be relevant given a particular term estimate. Using a binary independence retrieval (BIR) model, he specifically demonstrates that "optimal retrieval quality can be achieved under certain assumptions."

Unfortunately, probabilistic models are not very practical as they must work around two general assumptions: parameter estimations and independence. Parameter estimation refers to obtaining the parameter estimates through the use of training set data. Without an accurate data set, it is very difficult to properly estimate the parameters, which equates directly to their relevance. Independence assumptions on the other hand cause problems as well. For example, it is clear that the presence of the term "big" increases the probability in the English language of the presence of the term "bang" in reference to the "big bang" theory. This assumption is normally required for the model to work, even though the assumption many not be very realistic.

d. *Inference Networks*

Inference networks, also known as Bayesian networks, are networks that take known relationships and "infer" other relationships from the information. By having the ability to infer information from previous relationships, less computation is needed to determine the probability that an event will occur or be relevant. The best known example of an inference network being used to determine search engine results is contained within Google's PageRank algorithm and will be discussed in more detail in section B-2-e of this chapter.

e. Extended Boolean Retrieval

Conventional Boolean retrieval does not work very well when calculating relevance rankings, due to the fact that either the document solely contains the query term, or does not. This problem potentially allows for a lot of documents to be marked as satisfying the input query, but not be relevant, and vice versa. Extended Boolean retrieval adjusts this concept by applying weights to the terms entered in the query, known as term weights. These weights allow for the creation of a vector, with the difference being calculated out from the origin to determine relevance matching. Most modern search engines incorporate extended Boolean retrieval within a part of their ranking algorithm [9].

f. Latent Semantic Indexing

Latent Semantic Indexing is a method recognizing that a single concept can be described by using many different words. Attempting to match only one or a few words with a particular concept will produce many false results. By applying this knowledge, Single Value Decomposition (SVD) is used to generate a similarity coefficient; filtering out the noise and enabling documents with similar lexical semantics to be located closer in multi-dimensional space.

g. Neural Networks

Neural Networks are a set of nodes, composed of importance values. When calculating a value to associate with each node, all of the values from the incoming nodes are used. A portion of or the entire node's value is then passed on through the links going out from it and used to calculate those nodes' values. Training sets are needed to properly modify the weights of the links, ensuring satisfactory importance value calculations.

h. Fuzzy Set Retrieval

Fuzzy set retrieval is a method in which membership in a set is not solely based on having only elements that are in the set, but rather by applying a formula to calculate the SC, or "degree of membership" [9]. Boolean retrieval, union, intersection and complement operations are applied to determine the degree of membership. Another application used within "fuzzy set" retrieval is a spell check function. This function attempts to prevent false results based solely on misspelled pages, as well as allowing misspelled pages to not be penalized within the query results when they are relevant to a particular query.

2. WebCrawler Algorithms

Developing an algorithm to search and properly classify topics throughout the World Wide Web is a difficult task. Early search engines classified information based solely on lexical similarity and frequency [13]. These methods include Breadth-first, Best-first, Shark-search and Info-spiders. With the monolithic rise of Google and subsequent publishing of its PageRank concept, hypertext link structure analysis became the primary tool for Web semantics [7]. Since then, multiple methods have been created using PageRank as their basis, with a survey of such presented within the section. In particular, Google's current algorithm has not been published, as it is considered proprietary information forming the basis of the company's business.

a. Breadth-first

The Breadth-first Search (BFS) algorithm was one of the first and simplest known crawling strategies to be used on the World Wide Web. Developed in 1994 [11], it uses a First-in First-out (FIFO) queue method, crawling links in the order in which they are found. This method uses a single seed, i. e., web pages, and continues crawling until all links are exhausted. An illustration outlining the basic method is shown in Figure 4. Figure 5 presents an example BFS tree diagram containing 15 links; the numbers representing the order in which the web page link is found and processed.

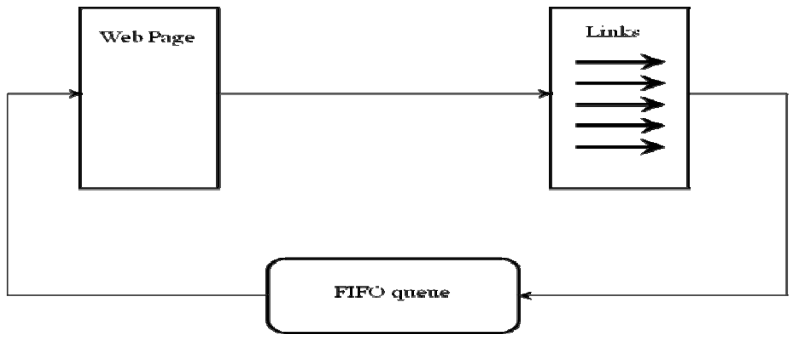

Figure 4. Breadth-first Crawler Outline.

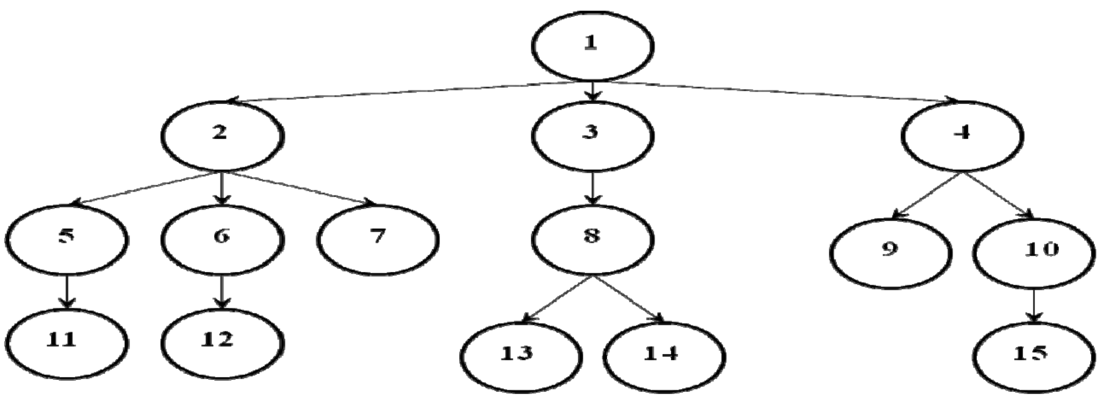

Figure 5. Breadth-first Crawler Tree Diagram Example.

b. *Best-first*

The Best-first algorithm is a method that uses some type of estimation criteria to determine which link to crawl first, given a group of links located on a web page. The idea behind the Best-first algorithm is to efficiently navigate and download relevant pages first, while preventing memory buffer overloads in the server conducting the crawl. An outline of the Best-first Crawler is presented in Figure 6. According to [12], the Uniform Resource Locator (URL) link's name is generally considered the best measure for estimating relevance, given that the name relates to a specific product, device or relevant field. Figure 7 presents an example of a Best-first Tree Diagram.

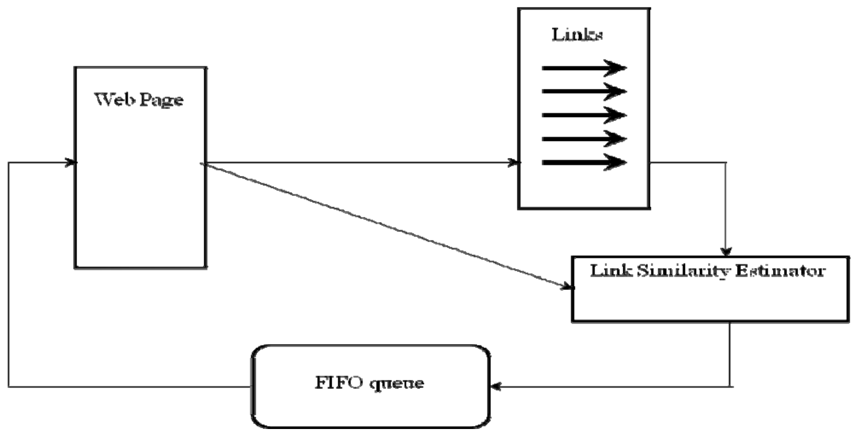

Figure 6. Best-first Crawler Outline.

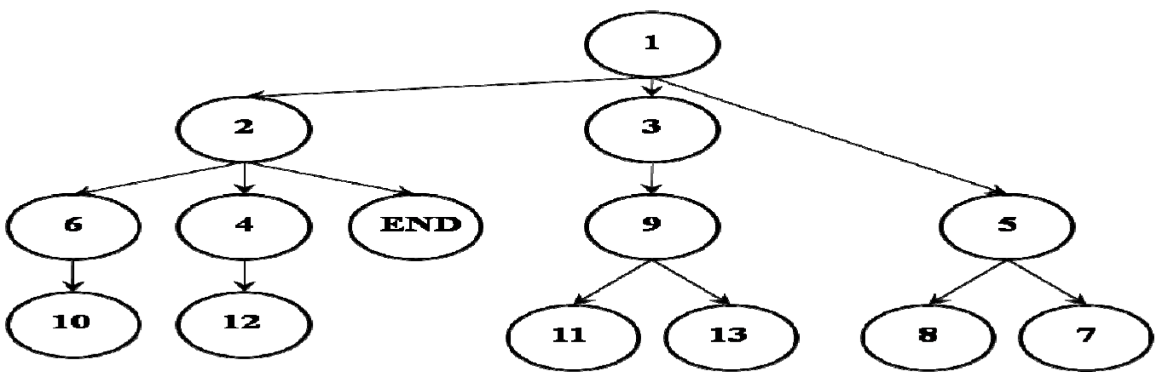

Figure 7. Best-first Crawler Tree Diagram Example.

One example of a generic cosine SC formula used to discriminate relevant web pages is provided below:

$$SC(Q, D_i) = \sum_{j=1}^{t} w_{qj} \times d_{ij} \quad (2.1)$$

where Q is a query weight vector and D is a specific document vector, both of size t, which is the total number of specific terms in the query. d_{ij} is defined as the term weight within the document. w_{qj} is the weight assigned for each specific query term, having treated the query as a document itself. Essentially, this formula takes the anchor text pointing to another web page as a document and compares it to the entered query. The more frequent the terms from the entered query are found in the anchor text, the higher the SC will become.

c. *Shark-search*

The Shark-search algorithm is essentially a hybrid of the Best-first method, using a more complicated function to evaluate relevant links [14]. Scores for links are influenced by more factors than before, including the text surrounding links, anchor text and an inherited score derived from previous page. The value added to a search engine by using the Shark-search algorithm is that link fetching relevance is determined by using a continuously changing value function as opposed to a standard binary function, allowing for a more refined search. Overall, this method saves communication time by obtaining documents that are more likely to be relevant first, leading to other documents that are more relevant later on. Figure 6, shown previously, illustrates the algorithm as well.

d. *Info-spiders*

Info-spiders are defined as independent agents gathering information in parallel over the World Wide Web. Generally speaking, each agent contains a list of key words and evaluates a node or multiple nodes within a network (i.e., web pages within the World Wide Web), looking for new nodes relative to the key words entered. These agents "exhibit an intelligent behavior, being able to evaluate the relevance of the document content with respect to the user's query, and to reason autonomously about future actions that mimic the browsing habits of human users [15]." As the "Spiders" progress to new nodes within a network, the amount of energy, or SC is calculated. Eventually, the value drops below a set threshold, ending the search down a particular linked path. The cycle then repeats itself within different networks determined by the user. An example of such a program found freely on the Internet is MySpiders [15].

Figure 8 is a standard Info-Spider architecture representation, starting and ending the process with a user. To begin, a user enters into the information environment, inputting the key words to be searched out over the World Wide Web. Next, the program fetches each page as a raw html document. After the document is retrieved, it is parsed and saved in a compact format. Meanwhile, the document is weighted for the given key words and its outgoing links processed to determine the likelihood of finding the relevant key words within the next linked page. The process repeats until the energy or SC drops below a set threshold, ending the search. Multiple "Spiders" or paths are taken simultaneously in parallel to speed up the process. At the end of the process, a database has been developed and indexed relative to the entered key words that can be accessed by the user at his or her leisure.

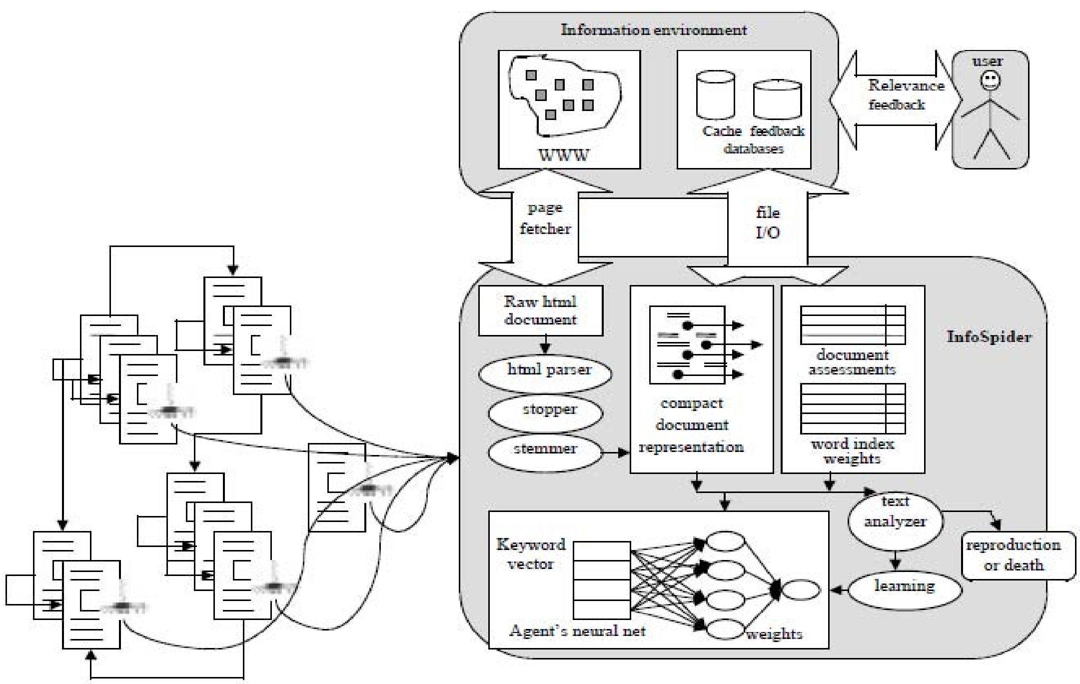

Figure 8. Info-Spider Architecture (From [15]).

e. PageRank

In 1998, Sergey Brin and Lawrence Page forever changed the way the world searches for relevant web pages with the development of Google and the subsequent implementation of the PageRank algorithm. According to [16], PageRank is an algorithm that ranks a web page based solely on its incoming and outgoing hypertext links. In general, pages with more incoming links are viewed as being more "important" than those with less incoming links. The easiest way to envision the concept is as a citation format. Each web page hypertext link is a citation or vote of approval for the web page it points to, with the weight of the citation based on the number of votes of "importance" the page receives. Equation 2.2 defines a slightly simplified PageRank algorithm with R being the ranking, u a web page, F_u as a set of pages u points to and B_u as a set of pages that point to u. The number of links from u is $N_u = |F_u|$ and c is a factor used to normalize all of the rankings.

$$R(u) = c \sum_{v \in B_u} \frac{R(v)}{N_v} \quad [17] \qquad (2.2)$$

The equation is recursive until convergence is reached. Figure 9 presents a visual example of such a simplified calculation reaching an approximate equilibrium. Initially, page A was given a value of 1.0 for its ranking. Having two links, this divides the value in half so that page B and C each have 0.5 ranking. With page B and C only having one outgoing link each, they both pass on their link's value to pages C and A respectively. At this point, page A has a value of 0.5, page B a value of 0.0, and page C a value of 0.5. The Equation is applied recursively until equilibrium is reached, with the results shown in Table 2.

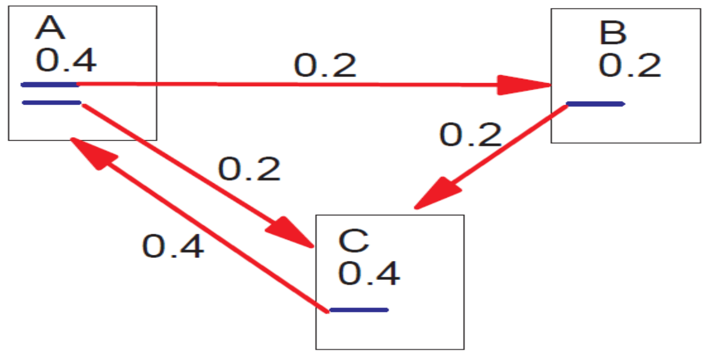

Figure 9. Simplified PageRank Calculation (From [17]).

Recursion #	Page A	Page B	Page C
1	1.0000	0.0000	0.0000
2	0.0000	0.5000	0.5000
3	0.5000	0.0000	0.5000
4	0.5000	0.2500	0.2500
5	0.2500	0.2500	0.5000
6	0.5000	0.1250	0.3750
7	0.3750	0.2500	0.3750
8	0.3750	0.1875	0.4375
9	0.4375	0.1875	0.3750
10	0.3750	0.2188	0.4063
11	0.4063	0.1875	0.4063
12	0.4063	0.2031	0.3906
13	0.3906	0.2031	0.4063
14	0.4063	0.1953	0.3984
15	0.3984	0.2031	0.3984

Table 2. PageRank Recursion Equation Calculations.

Problems can arise with this particular ranking function due to a potential issue known as "rank sink." Simply put, if any pages are fetched and point only to each other, an infinite loop will occur, causing the web page ranks to increase, but never be distributed. An illustration of such an event is given in Figure 10. To solve this problem, a ranking source vector $E(u)$ is introduced in Equation 2.3. The ranking source vector is used as a source of rank to prevent rank sink. Intuitively, it "corresponds to the distribution of web pages that a random surfer periodically jumps to," with E typically equal to 0.15 [17]. R' therefore changes to become an assignment of PageRank to a set of web pages.

$$R'(u) = c\sum_{v \in B_u} \frac{R'(v)}{N_v} + cE(u) \qquad [17] \tag{2.3}$$

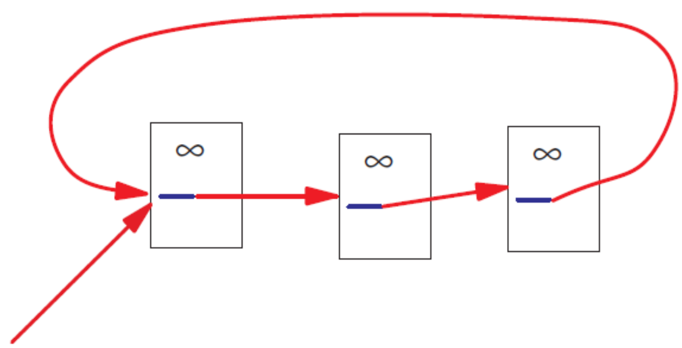

Figure 10. Loop Which Acts as a Rank Sink (From [17]).

The final PageRank formula is developed by going one step further and by replacing c with a dampening factor d in Equation 2.2:

$$PR(u) = (1-d) + d\sum_{v \in B(u)} \frac{R'(v)}{N_v} \qquad [17] \tag{2.4}$$

The dampening factor shown above is a simple means of directly manipulating the PageRank. In general, it should be thought of as the probability that a user will follow the links and $(1-d)$ as the scoring distribution from non-directly linked pages.

One of the biggest issues mentioned by Brin and Page in their research are "dangling links" [17]. Dangling links are defined as any link that points to a page that has no outgoing links. Due to the fact that these links do not have an affect on the ranking, they are removed from the system and added back in after convergence of the PageRank algorithm. Normalization of the other links will change slightly but should not have a large effect on the total population of web pages.

C. PAGERANK ALGORITHM VARIATIONS

Since publishing the generic PageRank algorithm, Google has moved forward to dominate the World Wide Web Search Engine business. Microsoft Network, Yahoo!, Ask, and others still exist and have maintained a significant amount of market share but are nowhere close to that of Google [7]. Google's actual algorithm and code, along with the other companies' mentioned above are still proprietary. Listed below are other known algorithms that attempt to improve upon Google's initial PageRank algorithm with their own variant.

1. Topic-sensitive

A "topic-sensitive," "topic-centric" or "focused" crawler is an algorithm that returns a "local ranking based on each user's preferences as biased by a set of pages they trust or topics they prefer" [18]. This approach differs from PageRank by taking advantage of personalization, tailoring information specific to the search context. It also allows an increase in information relevance at the cost of computational resources. To determine relevance, a similarity score is initially calculated as previously shown in Equation 2.1. This score determines the relevance of the current page and is used as a component to determine the final link score. Equation 2.5 calculates the link score, $Linkscore(j)$ by adding together the URL score, $URLscore(j)$, with the anchor text score, $Anchorscore(j)$ [19]. Linkscore(j) is the score of the hypertext link j; $URLscore(j)$ is the similarity between the current page's hypertext link information of j and the topic specified; and $Anchorscore(j)$ is the similarity between the anchor text and the topic specified.

$$Linkscore(j) = URLscore(j) + Anchorscore(j) \quad (2.5)$$

After the link score is determined, a final score for the link is calculated by combining the current page's similarity score with the previously calculated link score. Equation 2.6 calculates the final score, $Score\_To\_PR(j)$, by adding $TP(j)$ with $Linkscore(j)$ [19]. $Score\_To\_PR(j)$ is defined as the final score of the Topic-PageRank algorithm with respect to link j; $TP(j)$ is the Topic Page similarity score; and $Linkscore(j)$ is the score of the link previously calculated in Equation 2.5.

$$Score\_To\_PR(j) = TP(j) + Linkscore(j) \quad (2.6)$$

Experiments to determine the performance of the above algorithm were conducted by Yuan, Yin, and Liu [20]. Accordingly, a metric called the "harvest ratio" was devised to quantize performance. Equation 2.7 shows the harvest ratio as the percentage of the number of relevant pages divided by the total number of downloaded pages. The topics searched for in this experiment were American History, New Car, China travel and huang shan travel, with their corresponding results are shown in Table 3. Overall, Breadth-first had the worst ranking values with an average ranking of 0.3375 and the largest variation in value. PageRank prefored better with an average ranking value of 0.4625 and had the least variation in value. T-PageRank performed the best with an average ranking value of 0.6225 with only slight variations in value.

$$Harvest\_Ratio = \frac{\#\_of\_Relevant\_Pages}{\#\_of\_Dowloaded\_Pages} \quad (2.7)$$

Topic	*Language*	*Breadth-first*	*PageRank*	*T-PageRank*
American History	English	0.34	0.47	0.64
New Car	English	0.34	0.47	0.65
China travel	Chinese	0.29	0.46	0.59
huang shan travel	Chinese	0.38	0.45	0.61

Table 3. Harvest Rate of Topics (From [20]).

As shown in Table 3, the topic-sensitive algorithm was more effective at providing relevant results when compared to the breadth-first and PageRank algorithms. In a different experiment, according to [18], approximately 70 percent of the pages being returned were the same between a topic-sensitive crawler and that of Google's Global PageRank. The difference between the two results is due to the fact that as more pages are crawled, the results begin to converge. Additionally, seed URLs determine where the search engines look next. If they are the same, the results will be similar.

2. Weighted

The Weighted PageRank (WPR) algorithm is an extension of the original PageRank algorithm, taking into account the importance of both the in and out links by "distributing rank scores based on the popularity of the pages" [21]. Simply put, the algorithm assigns larger rank values to pages that are more popular instead of dividing the rank value assigned to every page evenly among the out links. Equation 2.8 calculates the weighted popularity of the in links as $W^{IN}_{(v,u)}$. This is "based on the number of in-links of page u and the number of in-links of all reference pages of page v" [21]. I_u and I_p represent the number of in-links of pages u and p respectively. $R(v)$ is the reference pages list of page v.

$$W^{IN}_{(v,u)} = \frac{I_u}{\sum_{p \in R(v)} I_p} \quad (2.8)$$

Accordingly, the out links are calculated in a similar way, using Equation 2.9. $W^{OUT}_{(v,u)}$ is the weighted popularity of the out links. This is based on the number of out-links to the page u and the number of out-links of all reference pages of page v. O_u and O_p represent the number of out-links of pages u and p respectively. $R(v)$ is the reference pages list of page v.

$$W^{OUT}_{(v,u)} = \frac{O_u}{\sum_{p \in R(v)} O_p} \quad (2.9)$$

Knowing the above information, the final PageRank formula, Equation 2.4 is then modified to:

$$PR(u) = (1-d) + d \sum_{v \in B(u)} \frac{R(v)}{N_v} W^{IN}_{(v,u)} W^{OUT}_{(v,u)} \quad (2.10)$$

Testing for the Weighted PageRank Algorithm was done using the query "scholarship" in [21]. Table 4 presents the size of the page set obtained, the number of relevant pages and the relevancy value for the given pages. In general, W PR is shown to have higher values for the given relevant pages found, but is st ill finding approximately the same number of relevant pages as the original PageRank algorithm.

	Number of Relevant Pages		Relevancy Value(κ)	
Size of the page set	PageRank	WPR	PageRank	WPR
10	0	1	0.1	0.5
20	4	3	13.1	16.8
30	4	4	47.1	49.8
40	4	4	82.1	84.8
50	4	4	117.1	119.8
60	5	5	159.6	162.3
70	7	7	211.7	214.4

Table 4. "scholarship" Query Results (From [21]).

3. Usage-based

According to [22], Usage-based PageRank (UPR) is a modification of the original PageRank algorithm in that it additionally ra nks web pages based on the previous user's navigation behavior. The com putation is esse ntially biased using the infor mation from the previous user's visits that are recorded in the website's log. To do this, a transition matrix m and personalization vector p are both defined in such a way that the pages and paths previously visited by other users are ranked higher.

Following the properties of a Markov theory and the PageRank algorithm , the Usage-based PageRank vector, UPR, is calculated as follows:

$$UPR = (1-\varepsilon)m * UPR + \varepsilon PER \quad (2.11)$$

where ε is the dampening factor, with m as an N x N transition matrix whose elements m_{ij} equal 0 if there does not exist a link from page p_j to p_i. m_{ij} is defined in Equation 2.12 with the personalization vector PER provided in Equation 2.13.

$$m_{ij} = \frac{w_{j \to i}}{\sum_{p_k \in OUT(p_i)} w_{j \to k}} \quad (2.12)$$

$$PER = \left(\frac{w_i}{\sum_{p_j \in WS} w_j} \right)_{Nx1} \quad (2.13)$$

The weight w_i for each node represents the number of times page p_i was visited and the weight $w_{j \to i}$ on each edge represents the number of times p_i was visited after p_j. These equations, when combined, result in the final UPR equation given in Equation 2.14, which was represented previously by Equation 2.11.

$$UPR^n(p_i) = \varepsilon \sum_{p_j \in IN(p_j)} \left(UPR^{n-1}(p_j) \frac{w_{j \to i}}{\sum_{p_k \in OUT(p_j)} w_{j \to k}} \right) + (1-\varepsilon) \frac{w_i}{\sum_{p_j \in WS} w_j} \quad (2.14)$$

In [22], testing for the algorithm was limited, using publically available data from msnbc.com. Comparisons were made showing that UPR performed better than the other two at predicting accuracy. To its advantage, the process of ranking the next possible pages took less than 2 seconds and could be done online without delaying navigation [22].

4. TimeRank

TimeRank is another variant of PageRank in that it uses the web page's record of the last visited time to determine its degree of importance [23]. Essentially, it uses a time factor to improve upon the precision of a given ranking, basing it on the amount of time a user stays on the website. The longer time logged, the more important the page. TimeRank is calculated by Equation 2.15 [23]. $TR(j)$ is the final calculated score; $Score\_To\_PR(j)$ is the same score calculated from Equation 2.6's Topic-Sensitive algorithm and $t(i)$ is the total visiting time of a page related to a topic. $t(i)$ is initially set at 1 to avoid a zero ranking of a relevant topic web page.

$$TR(j) = Score\_To\_PR(j) * t(i) \quad (2.15)$$

Unfortunately, some complications arise with the algorithm due to processing server logs. A rule regarding the use of web proxies is applied to determine a valid source IP. If the source IP is the same in 30 minutes, it is treated as one user, otherwise it is discarded. Another issue not discussed is the fact that a page could be long and contain a lot of information that the reader must sift through. If this is the case, a page may be related to the general topic entered, but not the specific topic searched for and have a higher score due to the $t(i)$ factor.

5. DYNA-RANK

The final PageRank variant discussed is the DYNA-RANK algorithm. DYNA-RANK focuses on "efficiently calculating and updating Google's PageRank vector using 'peer to peer' systems" [24]. Changes in the web structure are handled incrementally amongst peers, requiring less computation time and a fewer number of iterations compared to a centralized approach. The concept uses the fact that changes will only affect up to a certain domain, not requiring a full recalculation of ranking vectors for others outside the domain.

The original PageRank formula is initially used when applying the DYNA-RANK algorithm. Equation 2.16, $new\_weight(K,L)$ is used to calculate the out-link weights for all of the out-link weights within the peer:

$$new\_weight(K,L) = \frac{P_R(K)}{(n(K)_{PEER(i)}) + 1} \quad (2.16)$$

where $new\_weight(K,L)$ is the new edge weight calculated; $P_R(K)$ is the PageRank value of node K and $n(K)_{PEER(i)}$ is the number of out-links of node K on $PEER(i)$. $PEER(i)$ is defined as a specific domain or peer grouping. To figure out which links need to be updated, a relative change value, RC is calculated according to Equation 2.17:

$$RC = \frac{abs(new\_weight - old\_weight)}{(new\_weight)} \quad (2.17)$$

where old_weight was the previously calculated $new\_weight(K,L)$.

Overall, DYNA-RANK performs well in reducing the time to reach relative convergence as well as the number of iterations required [24]. Future work is needed to evaluate this algorithm further with regards to how well it would work given a topic-sensitive PageRank algorithm.

Having now surveyed a variety of algorithms available for use in an IED Education Network WebCrawler, none appear to be specifically tailored or easily capable of discovering hidden networks within the World Wide Web. In order to carry the research forward, a specific WebCrawler must be chosen for future work and implementations; allowing an inside look at the current algorithm being used by the

WebCrawler. Criteria for choosing the WebCrawler was that it must be free, open source software that is scalable and easily deployed. Knowing this, our choice for an IED Education Network WebCrawler was the Nutch project.

III. NUTCH

A. INTRODUCTION

The Nutch project is a Java based open-source search engine, capable of crawling a simple intranet, subset of the Internet, or the entire World Wide Web [25]. Prior to Nutch's development, it was generally not possible to analyze why any random search from a popular search engine would rank a generic web page *y* higher than web page *x* for a given query. This was in part due to the fact that most search engine algorithms are considered proprietary, as well as to prevent spammers from manipulating text and links in order to specifically boost a particular website's rank. The Nutch project attempts to solve the algorithm dilemma by being open-source. Its purpose is two-fold, to bring transparency and a detailed explanation of how the score for a given web page or document is computed in a search engine while providing an alternative search engine for people who are not fully satisfied with the limited number of commercial Internet search engines in existence today. Additionally, Nutch observes robot exclusion protocols to allow administrators the ability to control which parts of their host are collected in this manner.

B. ARCHITECTURE

The Nutch project's architecture is designed to be scalable in both search size and speed, while implementing parallelization retrieval techniques in the process. Its operation can be divided into three parts, a crawler, indexer and a search interface [25]. Figure 11 presents this conceptually from a high level design point of view. The crawler is designed to search through any given file systems, intranet, or the World Wide Web. This information is then stored via a database named WebDB and cached for future use. In addition to storage, the crawler uses a program named Lucene to index the information retrieved. This index is then used to retrieve the data from WebDB via a search interface.

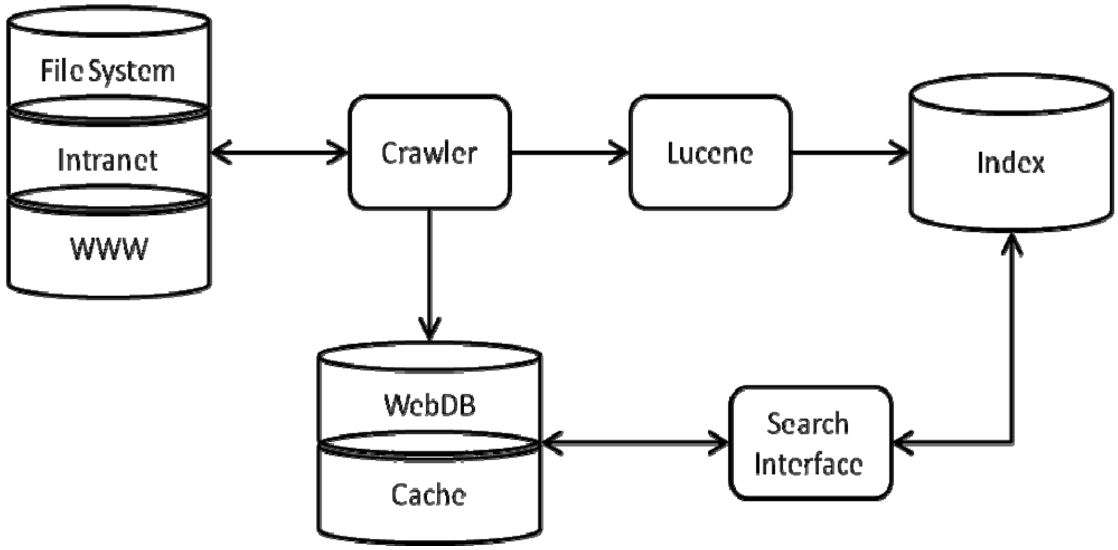

Figure 11. Nutch search engine high level design (From [25]).

The main advantage of using Nutch over other search engines is that the architecture is scalable. Simply put, whether there is a need to index one domain or many, even filter out others, it can handle them all. Nutch accomplishes this by using an extensible markup language (xml) format plug-in architecture that provides the user with the ability to make modifications over a wide range of parameters without having to make any hard coded changes to the Java code. The Nutch default xml configuration file is contained in Appendix A.

C. LUCENE

Lucene is at the heart of the Nutch search engine. Without it, the Nutch crawler would only gather information, storing it into a database void of organization. According to [26], Lucene is a mature, open-source Java program that provides indexing and searching capabilities. It is not an application program like many think, but a Java library that does not make assumptions about what it indexes or searches. Essentially, Lucene can be applied to search and index any type of file that can be converted into a recognizable text format. Figure 12 illustrates this difference between Lucene and an external application using it. Applications using Lucene present an interface to enable the user access Lucene's index while gathering different types of data at the same time, completely dependent upon user input. Lucene differs from this by taking the data obtained through an external application and bringing order to it through indexing. Overall, it provides a means of searching the index generated in order to present the desired information in an application.

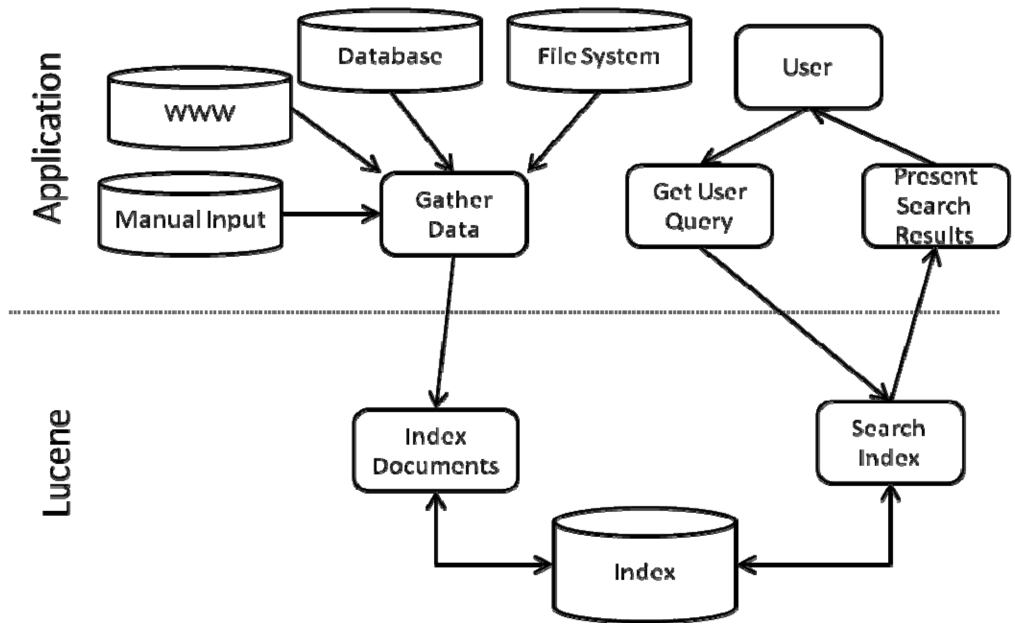

Figure 12. Typical application integration with Lucene (From [26]).

In addition to Lucene's ability to index documents, it has a transparent scoring algorithm which sets it apart from other indexing programs. The formula used by Lucene to score relevant documents d for a given query q is as follows:

$$score(q,d) = \sum_{t\_in\_q} tf(t\_in\_d) \cdot idf(t)^2 \cdot boost(t.field\_in\_d) \cdot lengthNorm(t.field\_in\_d) \quad (3.1)$$

where $tf(t\_in\_d)$ is the term frequency factor for the term t in document d, which allows documents with a higher term frequency obtain a higher score. $idf(t)$ is the inverse document frequency of the term, which allows documents that contain rare search query terms to obtain a higher score. $boost(t.field\_in\_d)$ is a user biasing boost value that can be given to a document set during indexing for a specific $t.field$, being the term field in document d. Finally, $lengthNorm(t.field\_in\_d)$ is the normalization value of a field, given the number of terms contained within the field, allowing a higher score to be assigned to a field that is short and contains a searched query term. The field values discussed above are provided via xml meta tag data, specifically url, anchor text, title, host and phrase. Equation 3.1 can be expanded by multiplying the resulting score by $coord(q,d)$ and $queryNorm(q)$. $coord(q,d)$ is a coordination factor, a score based on how many of the query terms are found in the document while $queryNorm(q)$ is a normalizing factor used to make scores comparable between queries. In Nutch, the formula changes slightly by multiplying the resulting score, $score(q,d)$ by an $Overall\_Boost(d)$ value, shown in below:

$$Overall\_Score(q,d) = Overall\_Boost(d) \cdot coord(q,d) \cdot queryNorm(q) \cdot score(q,d) \quad (3.2)$$

where $Overall\_Boost(d)$ is a boost factor determined by Nutch's page ranking algorithm for document d and $Overall\_Score(q,d)$ is the final score of document d for a given query q. An example calculation for Equations 3.1 and 3.2 is contained in Appendix B.

D. ADAPTIVE OPIC

Nutch is one of the few WebCrawlers to implement the Adaptive On-Line Page Importance Computation, better known as OPIC. Developed in 2003, the algorithm is computed on-line during fetch sequences in order to "focus crawling to the most interesting pages" [27]. The advantage OPIC has over other algorithms is that it does not use a lot of CPU or other disk resources, specifically by not needing to store the actual link matrix, like Page Rank. Essentially, this algorithm can be thought of as a "non-iterative weighted backlink-count strategy," where the ranking value is split evenly among its outgoing links producing a type of greedy algorithm [28].

Nutch implements OPIC by injecting the root node with a specific amount of value or "cash" as it is commonly referred to. The value injected is normally one unless otherwise specified. When discussing cash values within Nutch, there are two specific types: current and historical. Current cash is the amount of cash a document receives from incoming links before or after processing. Typically, this value is the amount of cash value it receives from other documents' out-links having been processed or else was injected with to begin an initial web crawl. Historical cash is the amount of cash a document has after processing and after a search is complete. When a document is processed from the fetch list, the cash is split evenly among the out-going links as shown below:

$$Outlink\_Current\_Cash(d) = \frac{Current\_Cash(d)}{Num\_OutLinks(d)} \quad (3.3)$$

where $Current\_Cash(d)$ is the current cash value of document d being processed and $Num\_OutLinks(d)$ is the number of links coming out from document d. These newly discovered out-links are then added to the web link database, as well as the fetch list database for future processing. Within the fetch list database, the $Outlink\_Current\_Cash(d)$ value is also stored and used as a measure to determine which node is processed next. In general, the search turns into a breadth-first variant where nodes for a specific depth level are not searched in the order found, but rather by their current cash score.

After a WebCrawler search is complete, the final value stored in historical cash is the actual OPIC score for a document, $OPIC\_Score(d)$ defined as:

$$OPIC\_Score(d) = Current\_Cash(d) + Historical\_Cash(d) \quad (3.4)$$

where $Current\_Cash(d)$ is the accumulated current cash of document d at the end of a search and $Historical\_Cash(d)$ is the historical cash value of document d, determined at fetch processing time. This factor affects the final score ranking of a document via the overall boost factor found in Equation 3.2, with the $Overall\_Boost(d)$ defined as:

$$Overall\_Boost(d) = \sqrt{OPIC\_Score(d)} \quad (3.5)$$

Some discussions have taken place in online blogs about why the square root value of the OPIC score is used instead of the straight score or a logarithmic value. Doug Cutting, the creator of both Nutch and Lucene, stated in many of them that the overall boost value was calculated this way to prevent the OPIC score from overly influencing document ranking. Either way, a logarithmic function and a square root function are both types of power functions and can manipulate the score in a similar fashion.

Knowing the above information, a new algorithm can now be developed specifically for IED Education Networks based solely on influencing the OPIC score of Nutch without affecting Lucene's scoring factors, which are based on query terms.

IV. ALGORITHM DEVELOPMENT

A. PROBLEM DEFINITION

When conducting any search over the World Wide Web, the results are only as good as the algorithm linking the database together and the scoring equation used to filter out unwanted documents via content. Initially, this thesis focused on changing the weighted plug-in boost values of the five fields used to score a document, those being url, anchor text, title, host and phrase. These values are calculated at query time and have a mild effect on the final scoring of a document, but are ultimately shaped by the OPIC value calculated during the fetch sequence. IED education networks can easily vary their meta-tag data depending on how visible they would like their information to be.

The Nutch OPIC algorithm assumes that all out-going links are equal. In reality, no link is created equal. To fix this, we chose to change the OPIC algorithm in order to assign a higher OPIC value to the pages which are referred to more, thereby ensuring web pages with more significant importance are ranked accordingly. This will in turn allow an IED focused WebCrawler to appropriately weigh potential root node documents higher, thereby making it easier to discover IED education networks.

B. ASSUMPTIONS

While attempting to develop a new algorithm, it must be assumed that the networks being searched are truly random. IED education networks come in all shapes and sizes and can easily range from just a single web page describing how to make one, to hundreds of web pages with similar information passed among them. Second, all depth levels are considered equal. The reason for this is to have a basis of comparison within a web search. In addition, it is assumed that the education networks being sought are trying to stay hidden within their respective domains and will not be easily located by their domain name, such as www.HowToMakeIEDs.com.

C. NEW ALGORITHM

Given the above criteria and assumptions, the new algorithm developed takes into account the fact that there exist four types of links coming out of a document: self referral links, external domain links, new document links within the domain and previously discovered document links, either external or internal to the domain. Identification of these types of links is critical in properly influencing the value of the OPIC score being given to those documents. Knowing this, the following algorithm was developed where the current cash value or portion a node receives, $Cash\_Portion(d)$ is equal to:

$$Cash\_Portion(d) = \frac{Current\_Cash(d)}{S(d)\cdot Swgt + N(d)\cdot Nwgt + O(d)\cdot Owgt + E(d)\cdot Ewgt} \quad (4.1)$$

where $Current\_Cash(d)$ is the current amount of cash contained within document d, $S(d)$ is the number of self referral links leaving the document, $Swgt$ is the weight assigned to self referral links, $N(d)$ is the number of new document referrals, $Nwgt$ is the weight assigned to new document referrals, $O(d)$ is the number of previously discovered documents referrals, $Owgt$ is the weight assigned to previously discovered document referrals, $E(d)$ is the number of external link referrals and $Ewgt$ is the weight assigned to external link referrals.

For example, a given document that had a current cash value of 0.25 was selected to be the next document processed via the fetch list database. During processing, it is discovered that the document has 8 out-going links: 2 of the 8 links are self referral links, 4 links are new links with one being external and the last 2 out-going links are found to be previously discovered documents. Weights for the different types of links provided are equal to 1, simulating the weighting effect of the original OPIC score. Given this information and applying Equation 4.1 results in the $Cash\_Portion(d)$ for each out-going document link equal to 0.125.

Following the logic given above, the OPIC current cash value for each out-going link is calculated as:

$$Actual\_Cash\_Portion(d) = Cash\_Portion(d) \cdot Assigned\_Wgt \quad (4.2)$$

where $Actual\_Cash\_Portion(d)$ is the portion of docum ent d's current OPIC cash value being given to a specif ic out-going link, either $S(d)$, $N(d)$, $O(d)$, $E(d)$. $Cash\_Portion(d)$ is the value obtained from Equation 4.1 and $Assigned\_Wgt$ is th e weight previously assigned to the type of document link being processed, which can be either $Swgt$, $Nwgt$, $Owgt$ and $Ewgt$. Continuing the pr evious exam ple, the $Actual\_Cash\_Portion(d)$ from Equation 4.2 would be equal to $Cash\_Portion(d)$ calculated from Equation 4.1 because of the weight for each going link being equal to 1.

Now, consider th e sa me docum ent given in the p revious exam ple with th e following weighted scores: $Swgt$ equal to 1, $Nwgt$ equal to 1, $Owgt$ equal to 2 and $Ewgt$ equal to 1. The $Cash\_Portion(d)$ for each of the out-going docum ent links decreases to equal 0.1. This is significantly less than the amount previously calculated. The $Actual\_Cash\_Portion(d)$ is then calcula ted to be 0. 1 for all of the outgoing links except for the previously discovered links, whic h are each now equal to 0.2. This value is now significantly higher than the previously determined value, therefore showing that these nodes are of greate r significance within the overall web link graph, shown in Table 5.

Links	Type	OPIC Score	New Algorithm Score	Difference	% Change
1	Self Referral	0.125	0.1	-0.025	0.2
2	Self Referral	0.125	0.1	-0.025	0.2
3	New	0.125	0.1	-0.025	0.2
4	New	0.125	0.1	-0.025	0.2
5	New	0.125	0.1	-0.025	0.2
6	New	0.125	0.1	-0.025	0.2
7	Old	0.125	0.2	0.075	0.6
8	Old	0.125	0.2	0.075	0.6

Table 5. Original OPIC versus New OPIC Scoring.

Having now developed a new algorithm capable of ranking documents with specific links higher than others, testing was needed to formulate a true understanding of the algorithm's potential and future use against IED Education Networks.

V. PERFORMANCE MEASUREMENTS

The goal of the testing performed below was to establish a preliminary means of judging the effectiveness of the new proposed algorithm's ability to score web pages when compared to the original OPIC algorithm, independent of Nutch. MATLAB code was created to randomly generate networks in order to perform an analysis given three different types of simulations. Multiple simulations were conducted with only three examples discussed herein.

A. EXPERIMENTAL SETUP

1. Hardware & Operating System Configurations

The platform used to conduct the simulation was a single Dell XPS M1330 laptop personal computer. This machine had an Intel Core 2 Duo CPU T9300 at 2.5 GHz, with 4 GB of RAM and a 185 GB hard disk. The operating system used was Microsoft Windows Vista with Service Pack 1.

2. Simulation Configuration

The software used to conduct the random network simulation and algorithm calculations was the MathWorks Matlab R2008a Windows program. Matlab is a private distribution program and requires a license. No special toolboxes or functions outside the original program were needed to perform the simulation. The software used to plot the resulting data was the Microsoft Office Excel Windows program. Microsoft Excel is a private distribution program and requires a license. No special toolboxes or functions outside the original program were needed to plot the results.

B. BENCHMARKING

Benchmarking is the process of characterizing a system as a whole or via its various parts in order to understand the actual or potential performance. In this particular case, three simulations were conducted, varying the random number of potential outgoing links. The first case, simulation 1 contains a low complexity randomly generated network with the maximum number of out-links equal to 5. The second case, simulation 2 is a medium complexity randomly generated network with the maximum number of outgoing links equal to 7. The final case, simulation 3 is a high complexity randomly generated network with the maximum number of out-links equal to 10. All simulations were generated using the following document link probabilities contained below in Table 5. The probabilities shown in Table 6 are not based on any particular network, but were chosen to ensure that the random networks generated will continue to propagate and have the ability to expand. Additionally, the depth level for all simulations was selected to equal 5 in order to visually present the results with clarity.

	Probability	Type of Document
New Document Internal	0.45	1
New Document External	0.05	2
Self Referral Link	0.05	3
Previously Discovered Document	0.45	4

Table 6. Probability of Creating Specific Document Links.

All 3 simulations calculate the original Nutch 0.8.1 OPIC score and 4 variant scores. The original Nutch OPIC is defined in Equation 4.1 as $Swgt$, $Nwgt$, $Owgt$ and $Ewgt$ all equal to 1. Variant 1 is defined as $Swgt$, $Nwgt$ and $Ewgt$ equal to 1 while $Owgt$ is equal to 2. Variant 2 is defined as $Swgt$, $Nwgt$ and $Ewgt$ equal to 1 while $Owgt$ is equal to 4. Variants 3 and 4 are respectively similar to variants 1 and 2 with the exception of $Swgt$ being equal to 0. The reason for using the 4 different variants was to determine if there is any benefit to becoming extremely "greedy" with the algorithm and also to evaluate the effect of removing self referral links from the networks.

Variation for a particular document d is calculated as:

$$Variation(d) = Final\_Cash(d) - Level\_AVG\_Cash \quad (5.1)$$

where $Final\_Cash(d)$ is the final cash value of document d and $Level\_AVG\_Cash$ is the average cash value for the document's level. Following this logic, the percentage variation of document d is calculated as:

$$\%\_Variation(d) = \frac{Variation(d)}{Level\_AVG\_Cash} \quad (5.2)$$

1. Low Complexity Network

The first type of random network to be looked at is one of low complexity. Low complexity is defined here as a network with less than 20 documents in its web-link graph. Figure 13, shown below, is a visual representation of the network's web-link structure. In order to construct Figure 13, Table 7 was used. Table 7 contains the data generated in Matlab to create the network. Column 1 displays the Document Number, which is defined as the number assigned to a document once a link to the document has been discovered and is unrelated to processing order. Column 2 is the depth level the document was found in. Each depth level is separated by a bold line for ease of viewing. Column 3 is an external flag marker, with 0 equal to an internal document and 1 equal to an external. Column 4 is the number of outgoing links. This number is determined randomly with 5 links being the maximum number of out-links possible in this simulation. Column 5 contains the type of out-links for the given number of out-going links in column 4, determined using the probabilities given in Table 5. Column 6 displays the out-link document number corresponding to the link given in column 5. Previously discovered document numbers are randomly determined from the given number of documents in the web-link graph at the time of discovery.

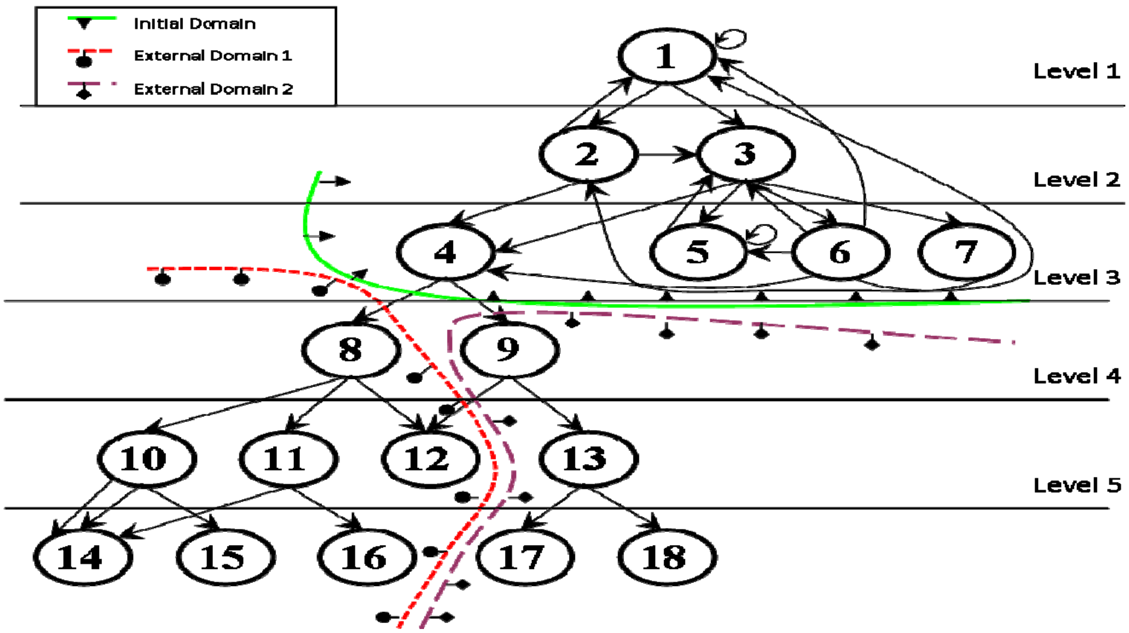

Figure 13. Simulation 1: Low Complexity Web Link Graph.

Doc Num	Depth	Ext Flag	Num Outlinks	Type of Outlink					Outlink Doc Num				
1	1	0	3	3	1	1	0	0	1	2	3	0	0
2	2	0	3	4	4	1	0	0	3	1	4	0	0
3	2	0	4	1	4	1	1	0	5	4	6	7	0
4	3	0	2	2	2	0	0	0	8	9	0	0	0
5	3	0	2	3	4	0	0	0	5	3	0	0	0
6	3	0	5	4	4	4	4	4	1	1	3	4	5
7	3	0	1	4	0	0	0	0	2	0	0	0	0
8	4	1	3	1	1	1	0	0	10	11	12	0	0
9	4	1	2	1	4	0	0	0	13	12	0	0	0
10	5	0	3	1	1	4	0	0	14	15	14	0	0
11	5	0	2	4	1	0	0	0	14	16	0	0	0
12	5	0	0	0	0	0	0	0	0	0	0	0	0
13	5	0	2	1	2	0	0	0	17	18	0	0	0
14	6	0	0	0	0	0	0	0	0	0	0	0	0
15	6	0	0	0	0	0	0	0	0	0	0	0	0
16	6	0	0	0	0	0	0	0	0	0	0	0	0
17	6	0	0	0	0	0	0	0	0	0	0	0	0
18	6	1	0	0	0	0	0	0	0	0	0	0	0

Table 7. Simulation 1, Low Complexity Web Link Graph Data.

Evaluating simulation 1 is very straight forward. Figure 14, shown below, provides an overview of the OPIC score trend, with random spikes representing documents with a higher importance. Depth level 2 document comparisons, contained in Figure 15, demonstrate a significant change in the OPIC scores, but mirror changes with respect to the original OPIC trend. Variant algorithms 3 and 4 continue the trends found in variants 1 and 2, with the increase in score attributed to the removal of document 1's self referral link. Variations with respect to the average cash values within depth level 2 are presented in Figure 16, with Figure 17 showing it as a percentage of the average cash value in the level for a given variant. Both of these figures show that the OPIC score for document 2 drops proportionately with any gain in OPIC score by document 3. This is to be expected as document 2 gives more cash to document 3 based on the network's link structure.

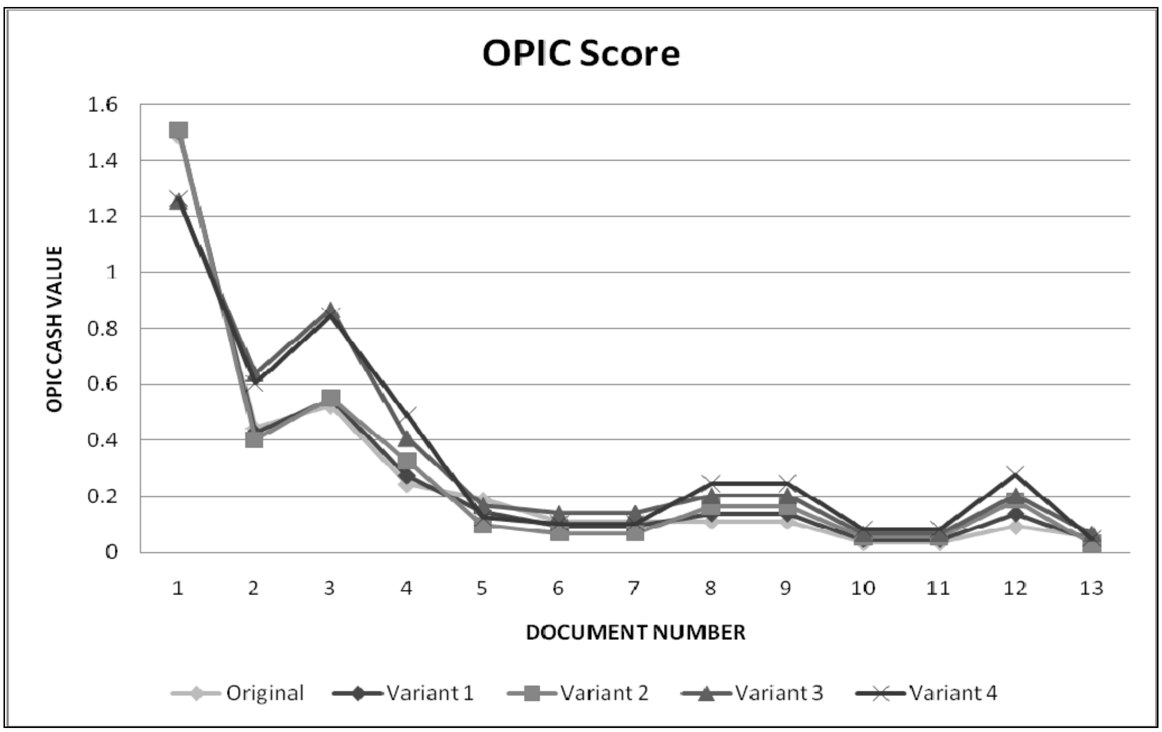

Figure 14. Simulation 1: Overall OPIC Scores.

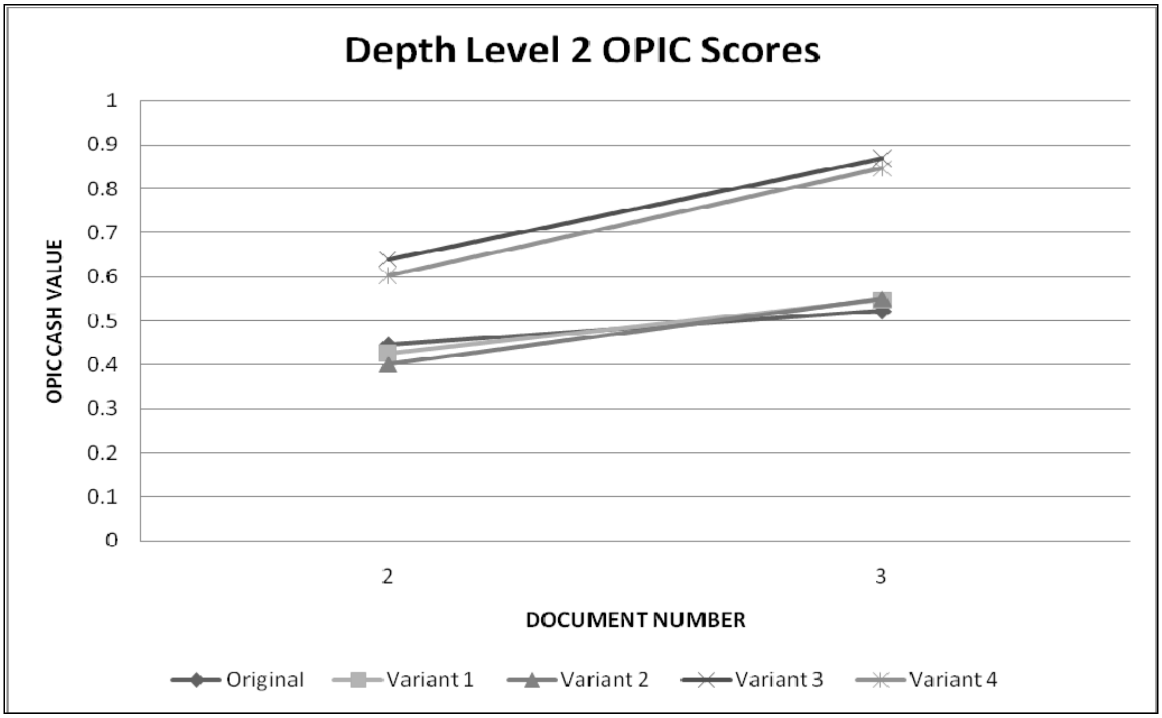

Figure 15. Simulation 1: Depth Level 2 OPIC Scores.

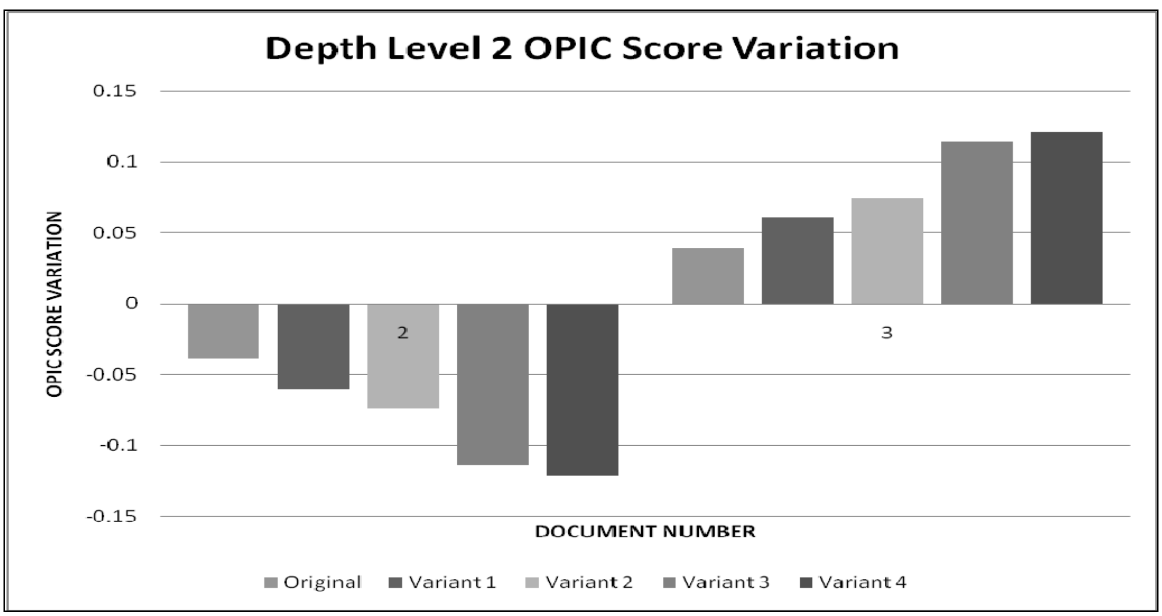

Figure 16. Simulation 1: Depth Level 2 OPIC Score Variations.

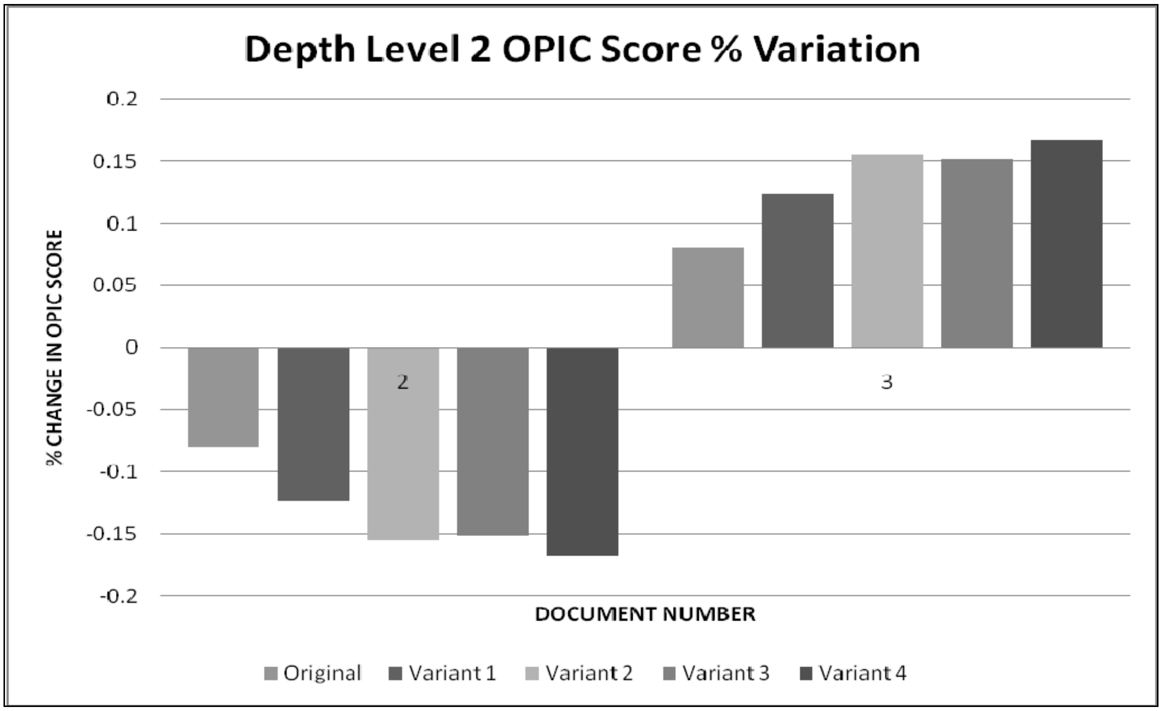

Figure 17. Simulation 1: Depth Level 2 OPIC Score % Variations.

Additionally, depth level 5 also shows a significant change in OPIC scoring trend, shown below in Figure 18; but again, this mirrors the original trend. Variant algorithms 1 and 2 follow previous trends as well, with variants 3 and 4 being in proportion to their respective counterparts. Figures 19 and 20 provide the resulting variations with respect to the average amount of cash within level 5 for a given variant and percentage of such. No new information is gained from these graphs as there are no previously discovered links coming in to any of these documents.

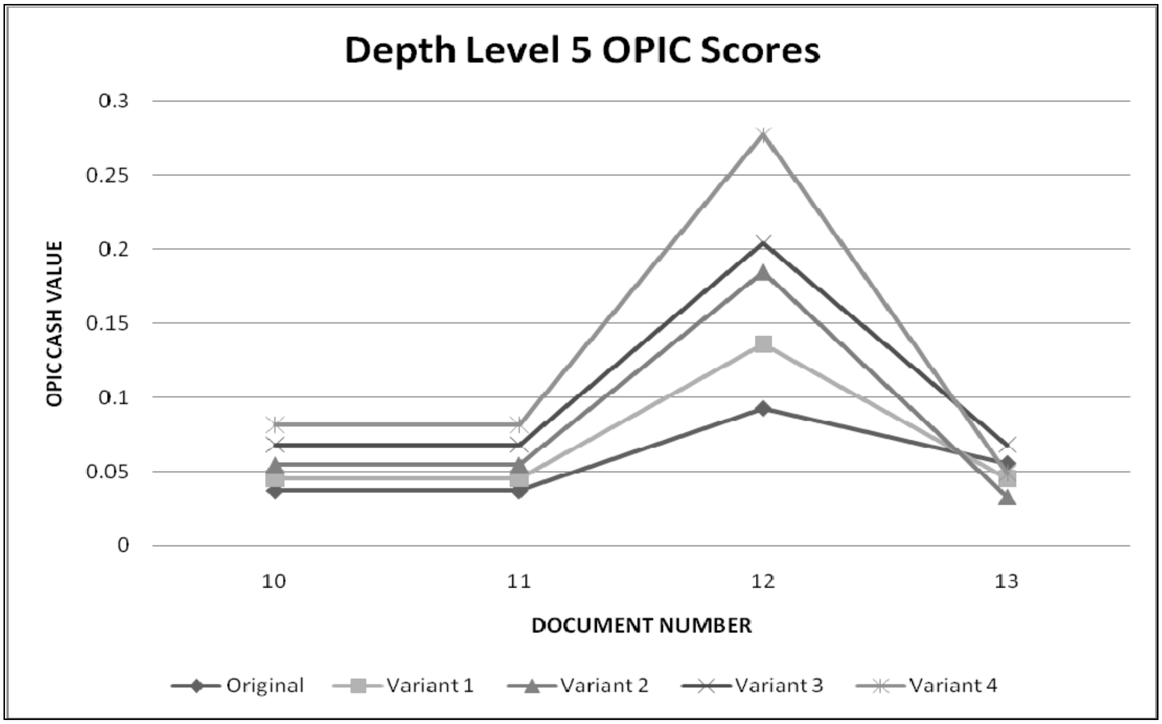

Figure 18. Simulation 1: Depth Level 5 OPIC Scores.

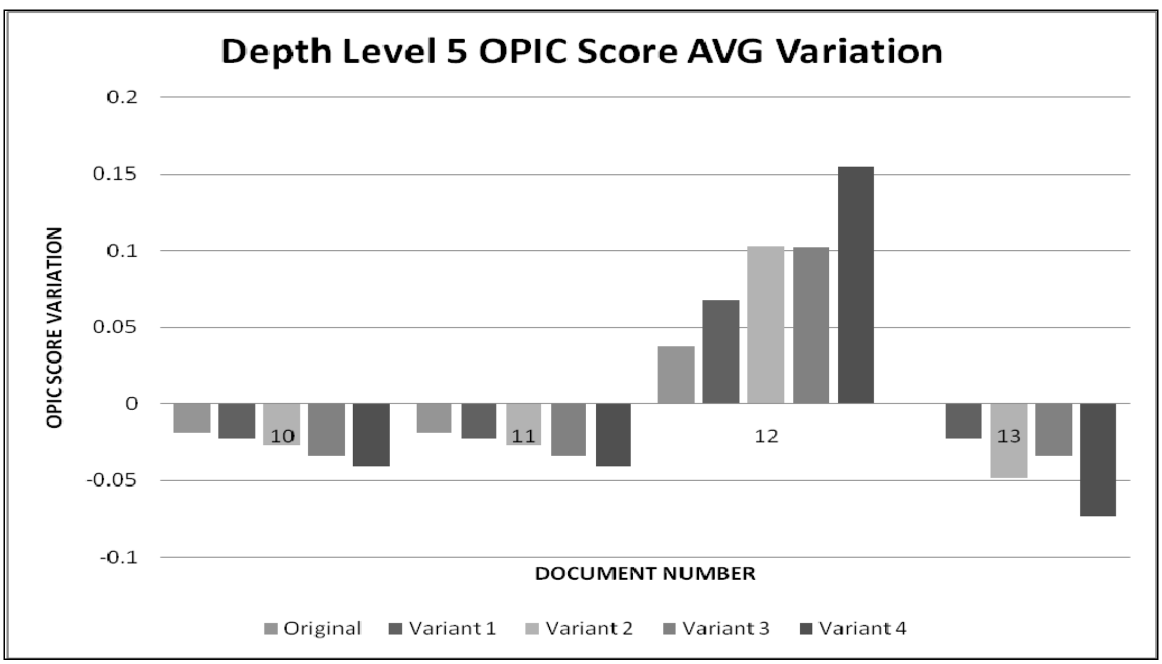

Figure 19. Simulation 1: Depth Level 5 OPIC Score Variations.

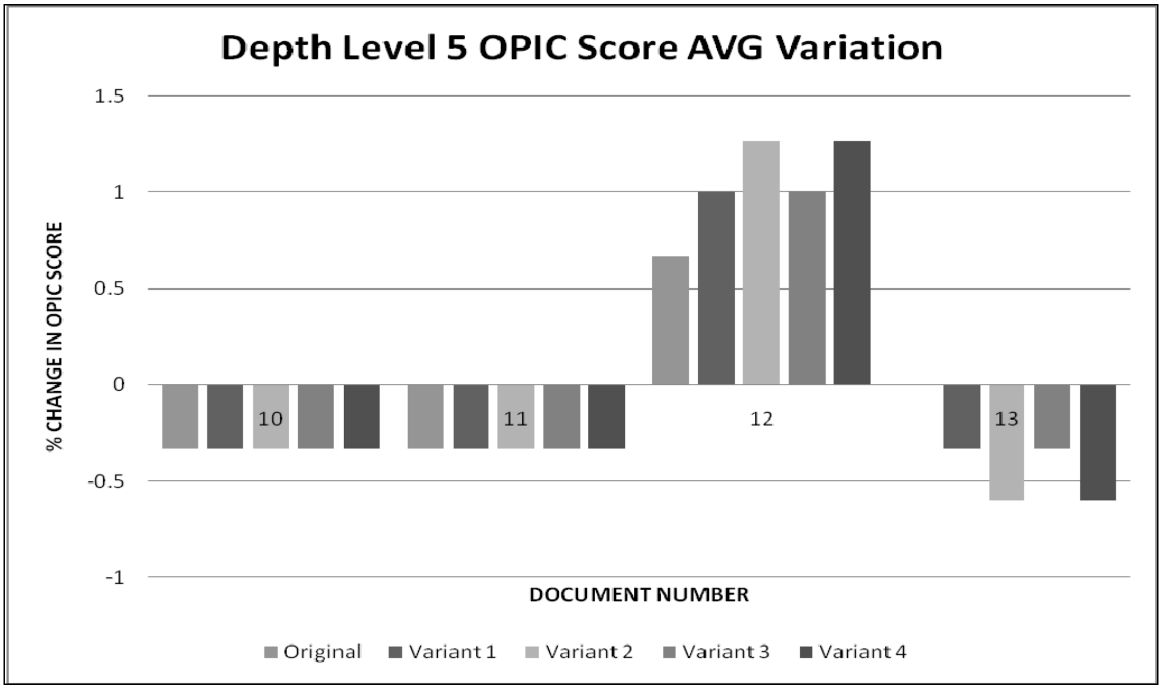

Figure 20. Simulation 1: Depth Level 5 OPIC Score % Variations.

2. Medium Complexity Network

The second type of random network to be looked at is one of medium complexity. Medium complexity is defined here as a network with more than 20, but less than 50 documents in its web-link graph. Figure 21, shown below, is a visual representation of the network's web-link structure. In order to construct Figure 21, Table 8 was used. Table 8 contains the data generated in Matlab to create the network.

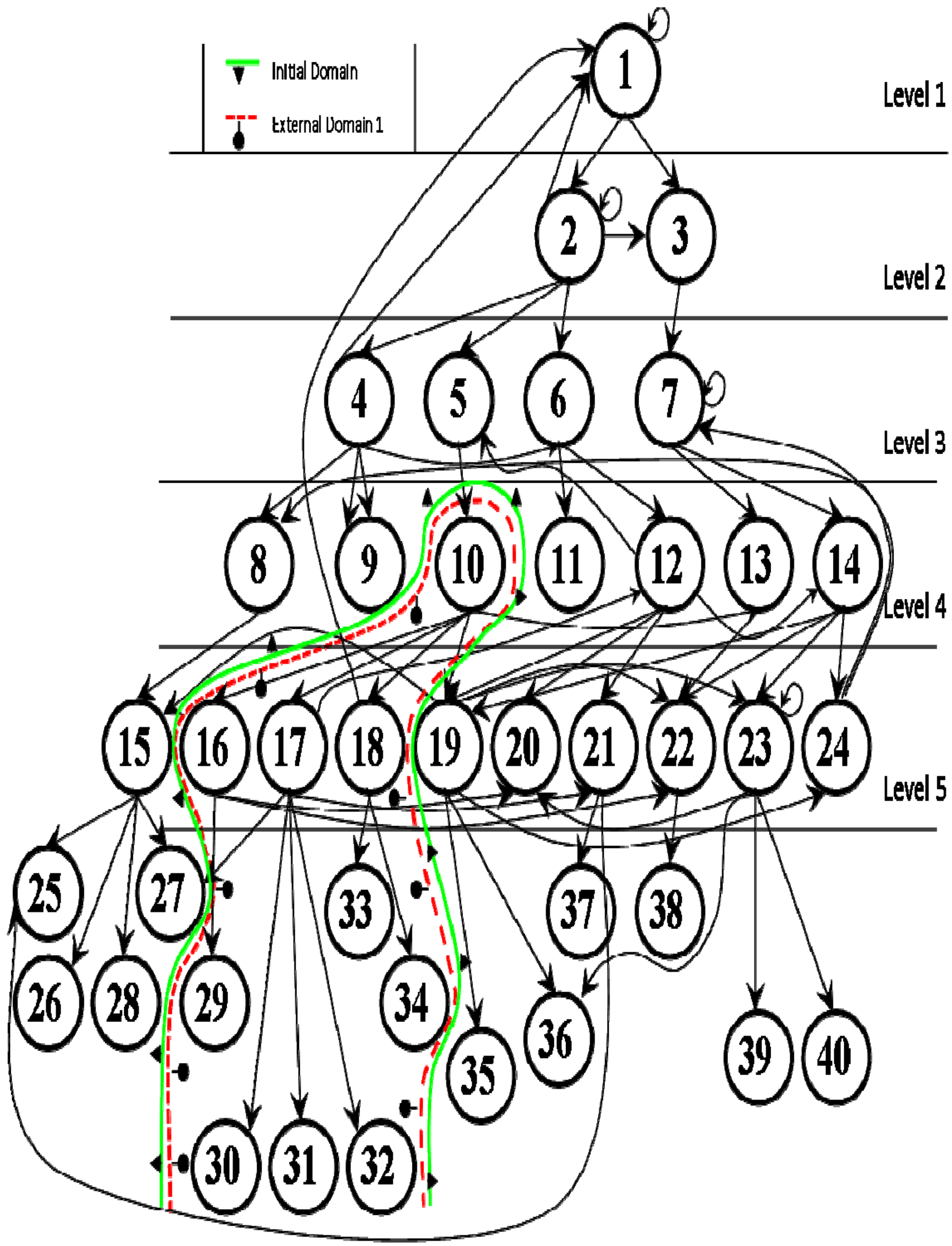

Figure 21. Simulation 2: Medium Complexity Web Link Graph.

Doc Num	Depth	Ext Flag	Num Outlinks	Type of Outlink							Outlink Doc Num						
1	1	0	3	3	1	1	0	0	0	0	1	2	3	0	0	0	0
2	2	0	6	1	4	2	1	3	4	0	4	3	5	6	2	1	0
3	2	0	1	1	0	0	0	0	0	0	7	0	0	0	0	0	0
4	3	0	5	4	1	4	1	4	0	0	6	8	1	9	9	0	0
5	3	1	1	1	0	0	0	0	0	0	10	0	0	0	0	0	0
6	3	0	2	1	1	0	0	0	0	0	11	12	0	0	0	0	0
7	3	0	3	1	1	3	0	0	0	0	13	14	7	0	0	0	0
8	4	0	1	1	0	0	0	0	0	0	15	0	0	0	0	0	0
9	4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
10	4	0	5	1	1	1	4	4	0	0	16	17	18	17	13	0	0
11	4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
12	4	0	5	1	1	4	1	4	0	0	19	20	14	21	5	0	0
13	4	0	1	1	0	0	0	0	0	0	22	0	0	0	0	0	0
14	4	0	4	1	4	1	4	0	0	0	23	19	24	22	0	0	0
15	5	0	4	1	2	1	1	0	0	0	25	26	27	28	0	0	0
16	5	0	3	4	4	1	0	0	0	0	20	22	29	0	0	0	0
17	5	0	6	4	4	1	1	4	1	0	27	21	30	31	12	32	0
18	5	0	3	4	1	1	0	0	0	0	1	33	34	0	0	0	0
19	5	0	6	4	1	4	4	4	1	0	15	35	22	23	24	36	0
20	5	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
21	5	0	2	2	4	0	0	0	0	0	37	25	0	0	0	0	0
22	5	0	1	1	0	0	0	0	0	0	38	0	0	0	0	0	0
23	5	0	6	1	4	4	3	4	1	0	39	36	8	23	20	40	0
24	5	0	1	4	0	0	0	0	0	0	7	0	0	0	0	0	0
25	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
26	6	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
27	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
28	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
29	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
30	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
31	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
32	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
33	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
34	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
35	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
36	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
37	6	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
38	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
39	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
40	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 8. Simulation 2, Medium Complexity Web Link Graph Data.

Due to the increasing complexity of simulation 2's link structure, evaluating a medium complexity simulation is a bit more difficult than the previous. Figure 22, shown below, provides an overview of simulation 2's OPIC scoring trend, with random spikes representing documents suggesting a higher importance. Depth level 2 document comparisons from Figure 22 show that document 3 is more important than document 2 for all of the variant algorithms due to its web-link structure. This is to be expected since document 2 contains a self referral link as well as an outgoing link pointing to document 3. Depth level 4 is also shown to have a significant increase in OPIC value for documents 13 and 14. Again, this is due to the self referral link in document 7 and the incoming link from document 12 to document 14.

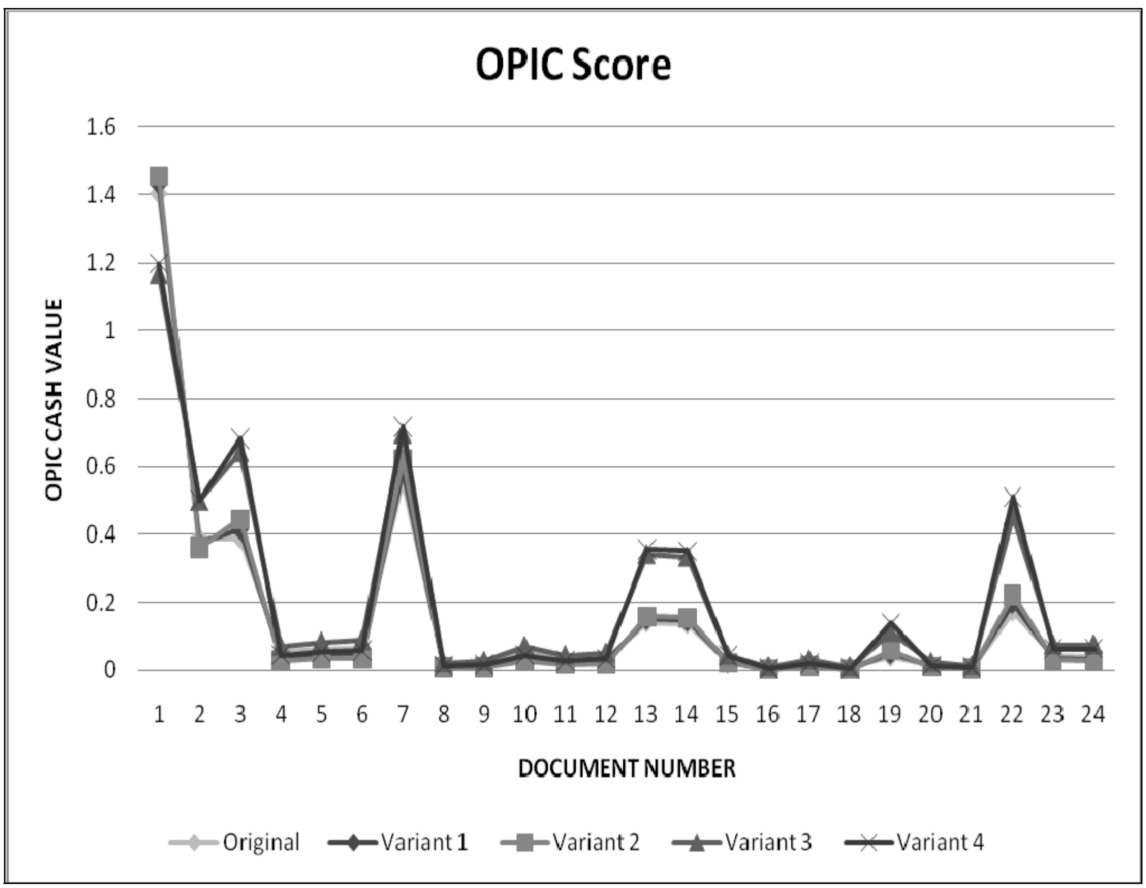

Figure 22. Simulation 2: Overall OPIC Scores.

Depth level 5 provides the most interesting results for the given variant algorithms, provided below in Figure 23. Initially, the OPIC value for document 19 is on par with other documents from within the level. Due to the removal of self referral links and additional value of previously discovered documents pointing to it from within the network, documents 19 significantly increases in value. This is illustrated in Figure 24 as a measure of change from the average cash value within the level. Figure 25 further explains this as an increase, ranging from 120 to 200%. Document 22 also significantly increases in value due to same reasons stated above, with the increase in value ranging from 400 to 1000% when compared to the average cash value contained within the depth level.

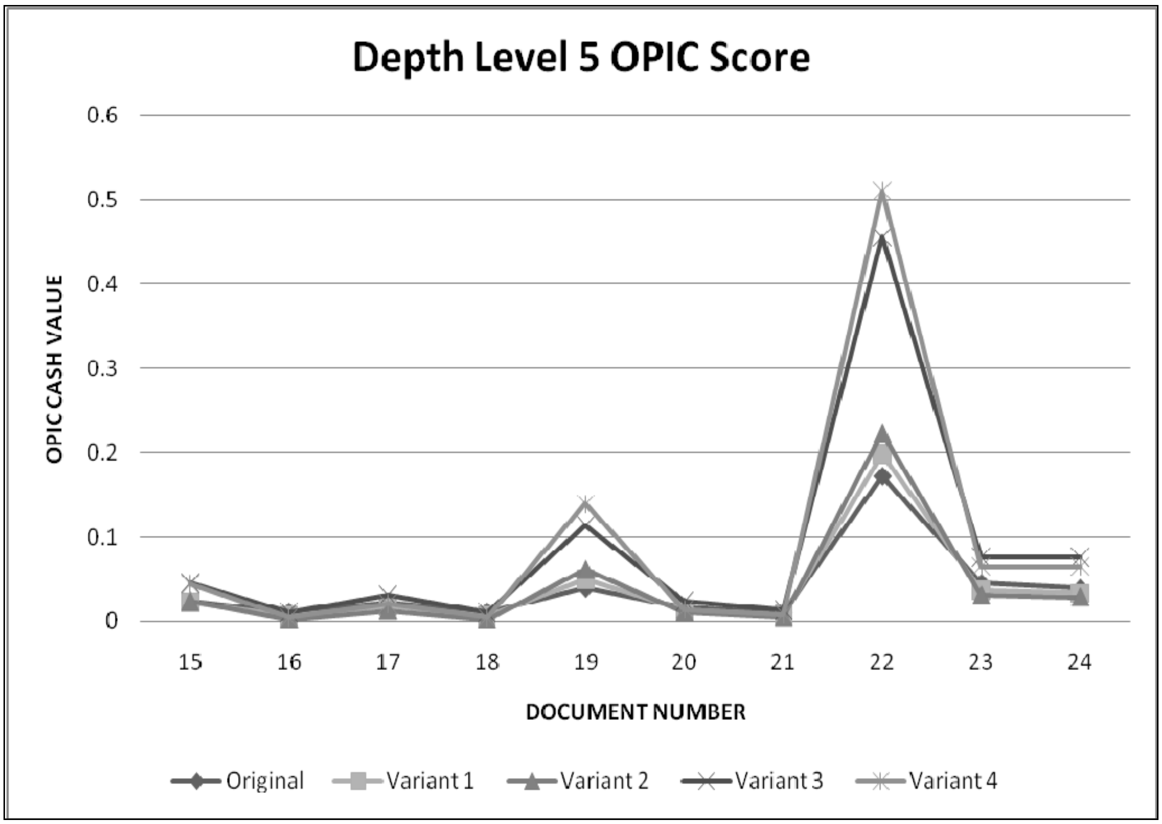

Figure 23. Simulation 2: Depth Level 5 OPIC Scores.

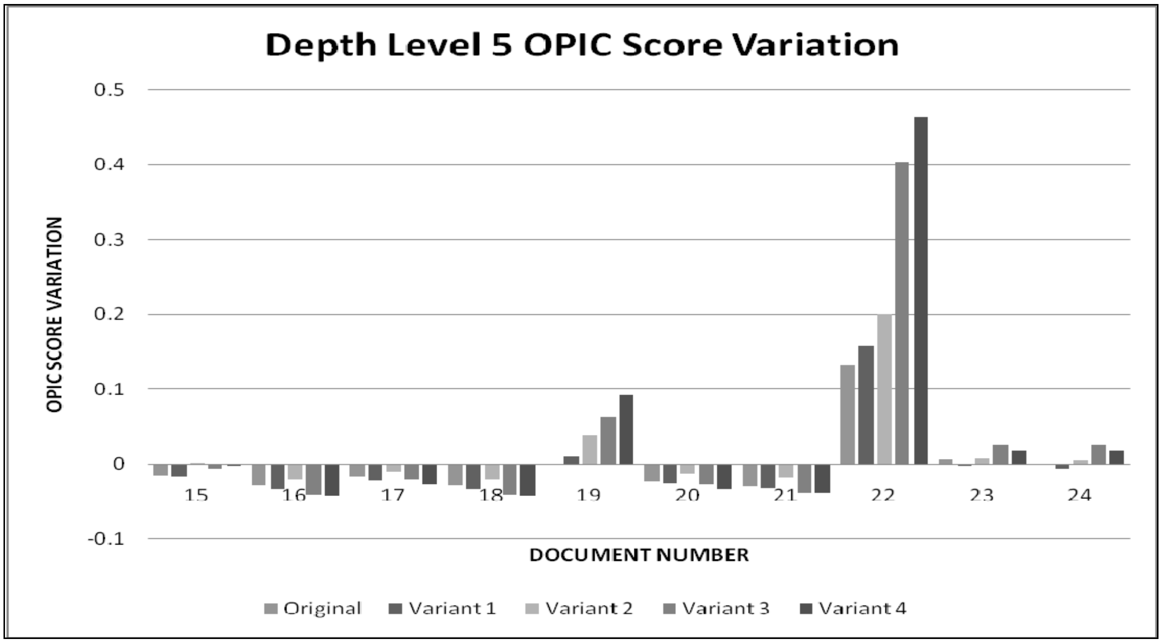

Figure 24. Simulation 2: Depth Level 5 OPIC Score Variations.

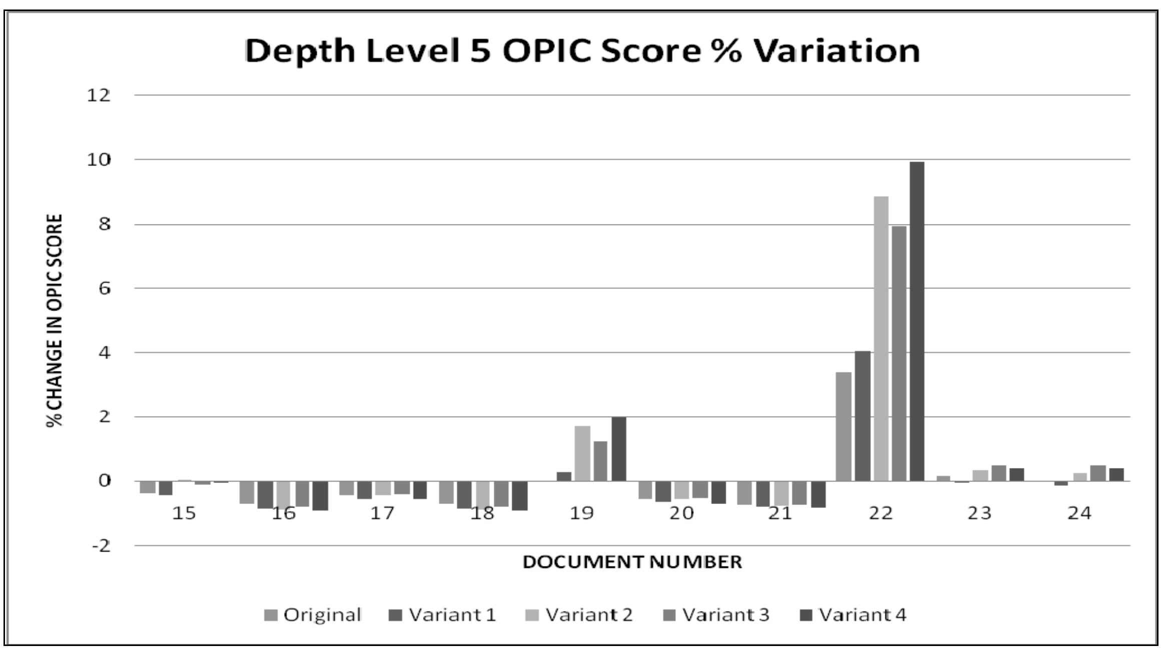

Figure 25. Simulation 2: Depth Level 5 OPIC Score % Variations.

3. High Complexity Network

The final type of random network to be looked at is one of high complexity. High complexity is defined here as a network with more than 50 documents in its web link graph. No figure is provided due to the extreme complexity and length of the network's web-link structure. Appendix B contains the data generated in Matlab to create the given network.

Evaluating a high complexity simulation is very difficult. Figure 26, shown below, provides an overview of simulation 3's OPIC scoring trend, with random spikes representing documents with a higher importance. Due to the high number of documents contained in the network, this graph is only able to show that variations exist within the network, but will need further review within each level.

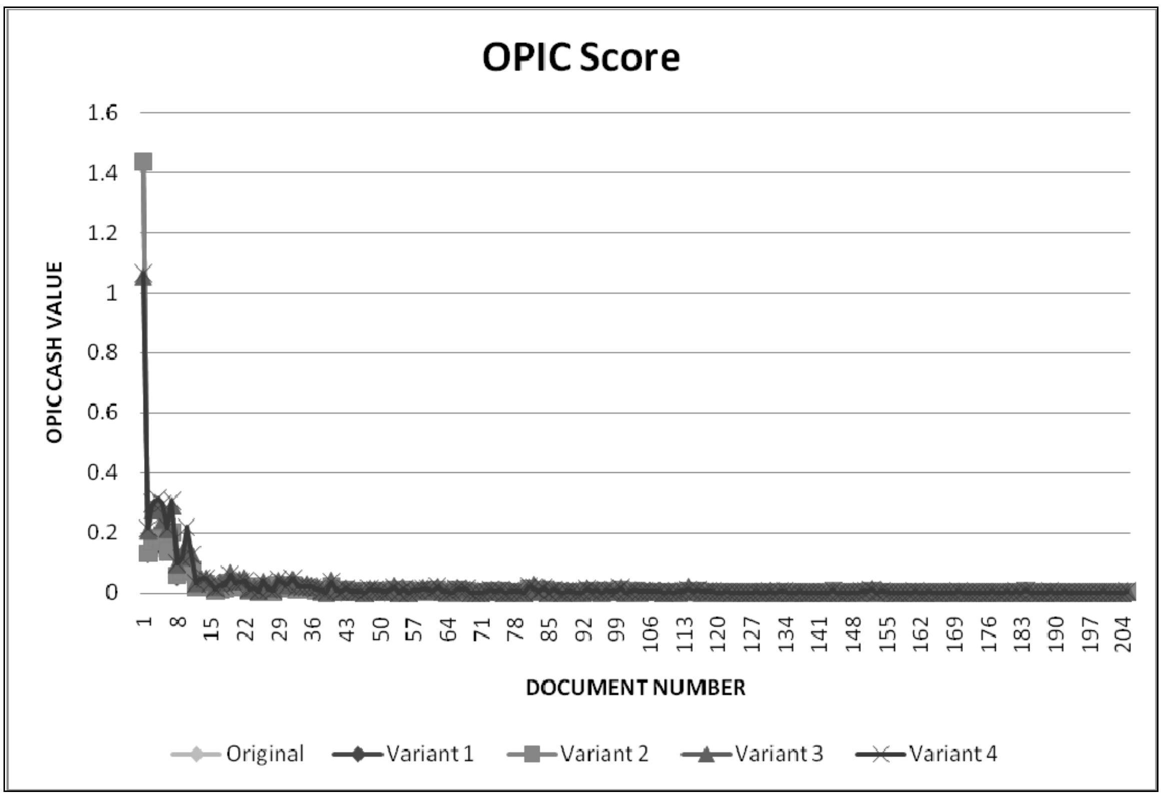

Figure 26. Simulation 3: Overall OPIC Scores.

Depth level 3 document comparisons from Figure 27 show that documents 10 and 19 become significantly more important than other documents in the level for all of the variant algorithms due to the network's web-link structure. Figure 28 shows this variation as a visible increase in the OPIC score for document 10, ranging between 140 to 240%. Document 19 on the other hand is able to maintain its OPIC score while the rest of the documents around it decrease significantly with respect to the average value, therefore maintaining its importance.

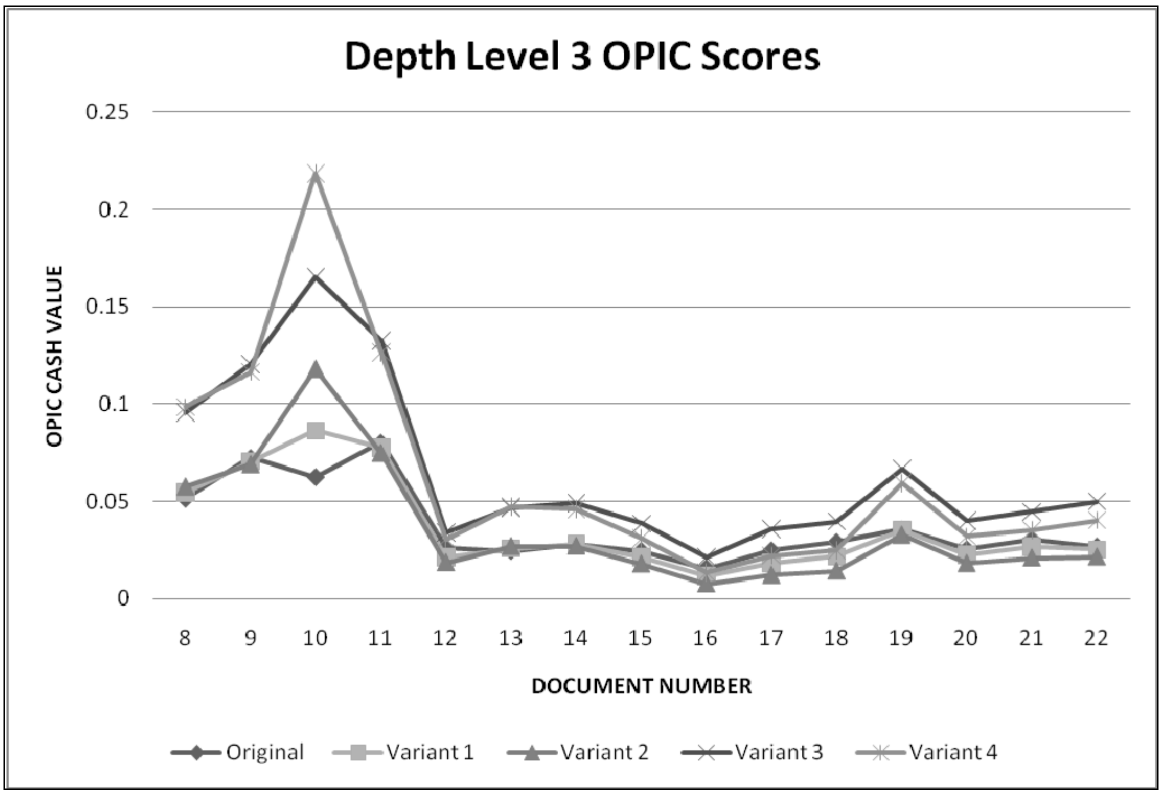

Figure 27. Simulation 3: Depth Level 3 OPIC Scores.

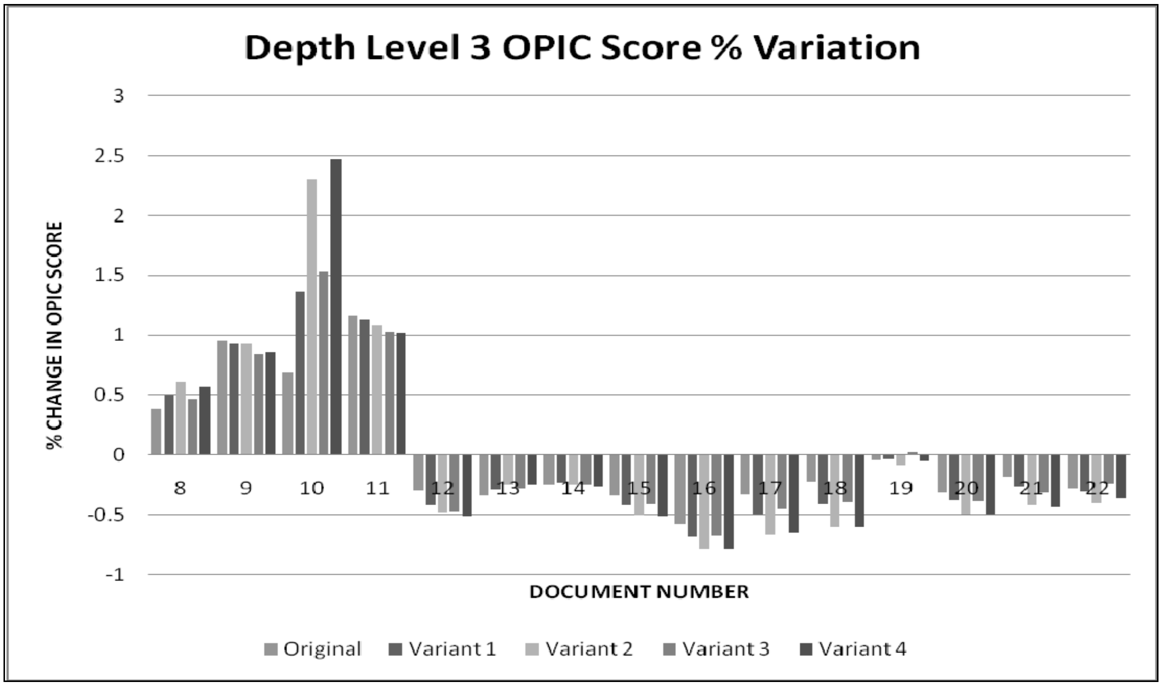

Figure 28. Simulation 3: Depth Level 3 OPIC Score % Variations.

Depth levels 4 and 5 provide the most interesting results for the given variant algorithms, shown below in Figures 29 and 31. Multiple documents increase their given OPIC scores, ranging between 10 to 650% in Figures 30 and 32. These levels demonstrate the effectiveness of this algorithm by significantly increasing the scores of documents 41, 55, 59, 66, 73, 74, 77, 78, 79, 89, 90, 94, 95, 102, 110, 113, 115, 119, 133, 134, 144, 150, 151, 161, 170, 177, 182, 184, 189, and 205 above the average value threshold, while effectively lowering the scores of documents 23, 27, 28, 29, below the average threshold value. These results match the complex link structure that is derived from the data contained in Appendix C.

Overall, having conducted 3 random network simulations, the results clearly indicate moderate success of our newly proposed OPIC algorithm considering results are based solely on the web link graph structure. Comparing a document's OPIC value to the average value contained within the depth level also allowed a measure of comparison regarding effectiveness.

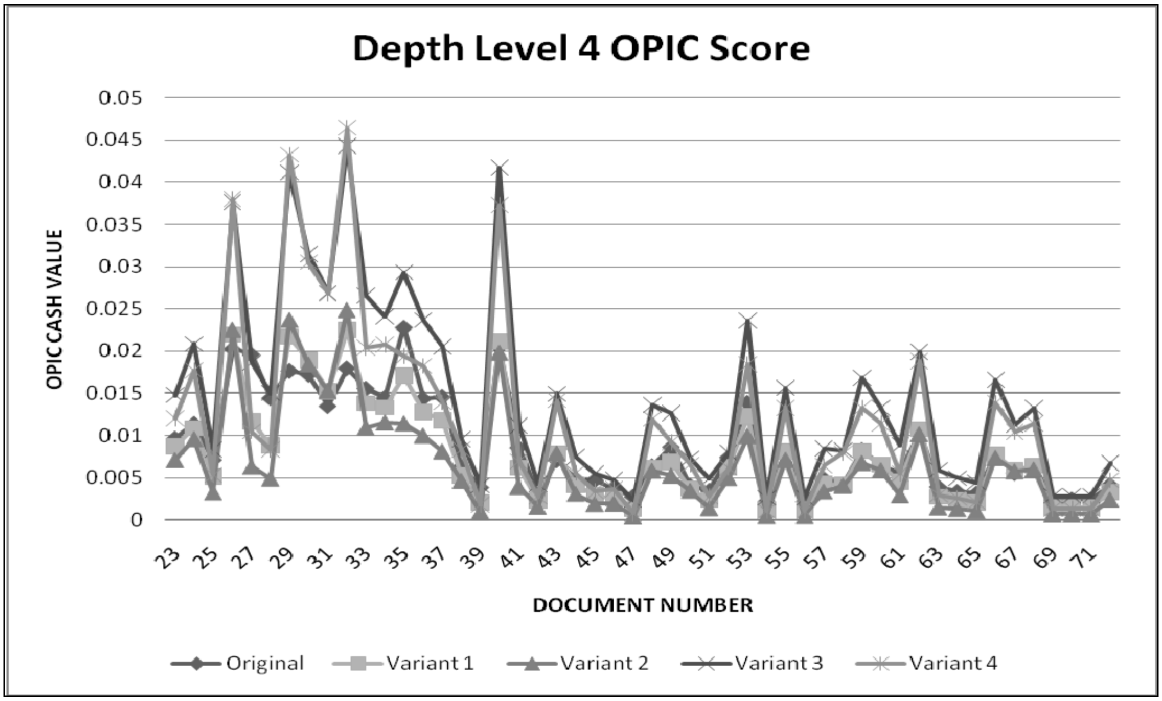

Figure 29. Simulation 3: Depth Level 4 OPIC Scores.

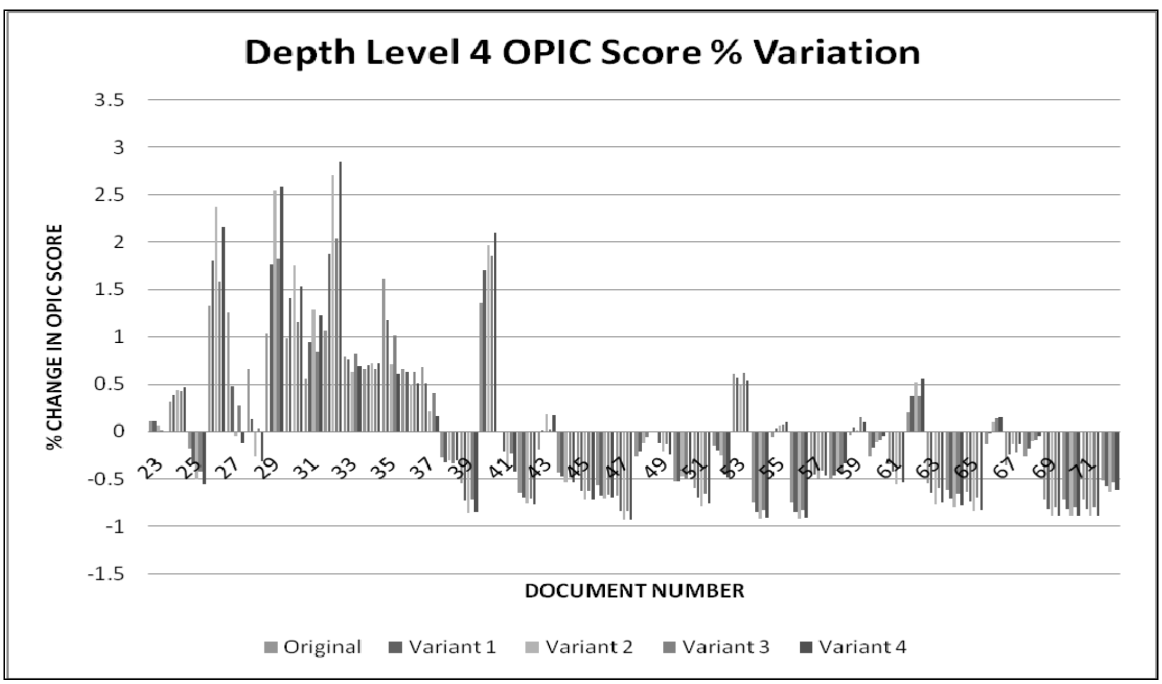

Figure 30. Simulation 3: Depth Level 4 OPIC Score % Variations.

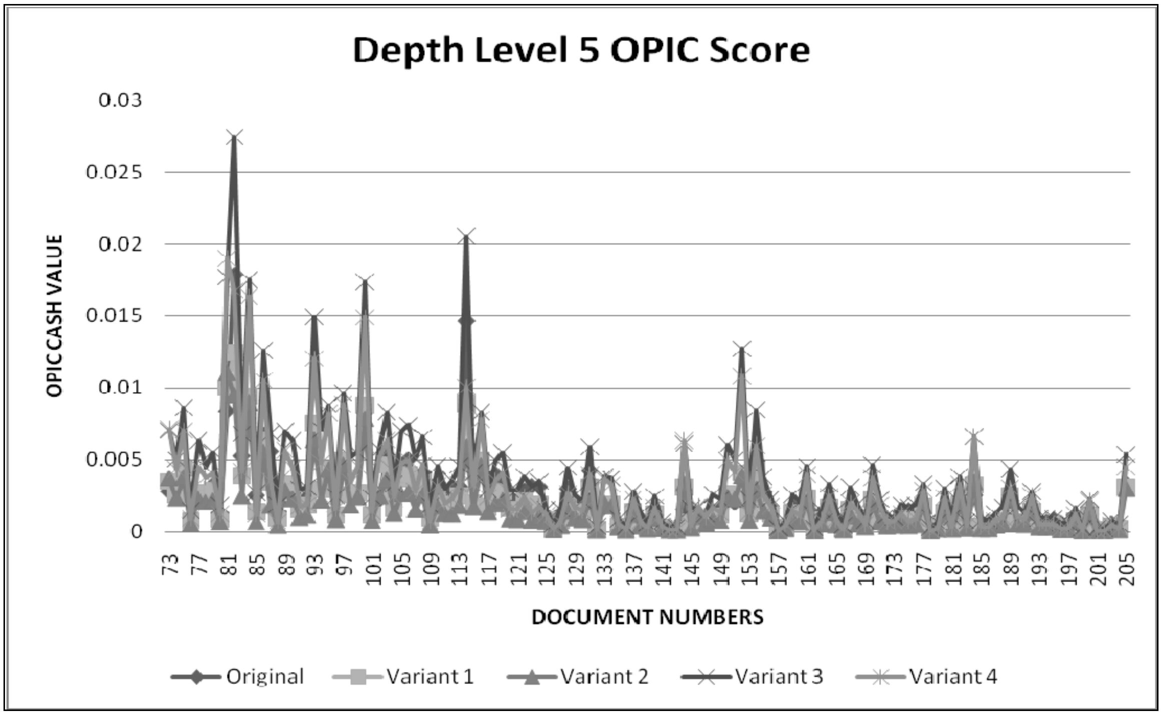

Figure 31. Simulation 3: Depth Level 5 OPIC Scores.

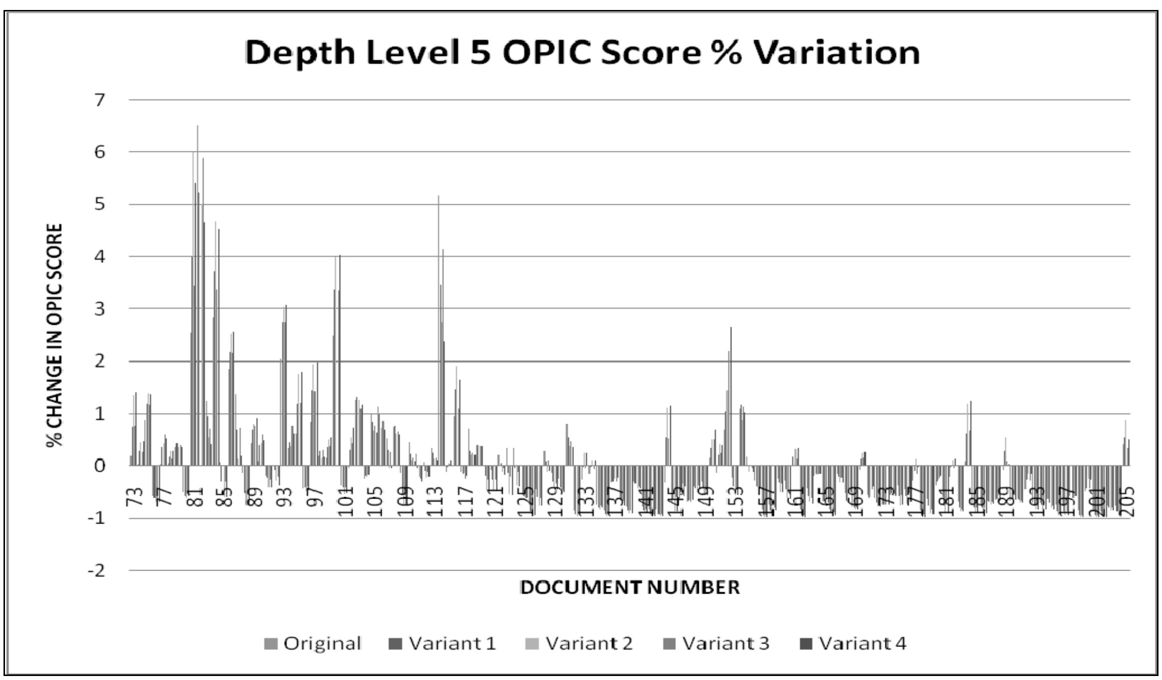

Figure 32. Simulation 3: Depth Level 5 OPIC Score % Variations.

THIS PAGE INTENTIONALLY LEFT BLANK

VI. CONCLUSIONS

A. SUMMARY

The research completed in this thesis showed that when implementing the new OPIC algorithm variations, documents referred to more within a given web graph receive a higher percentage of the overall OPIC cash within that level and throughout the overall web graph, when compared to the original algorithm. This in turn means that the document with a higher OPIC value is more relevant based solely on its link structure. Variants 3 and 4 show the most promise with regards to changing the OPIC score effectively by removing self referral links. We believe that applying this to the Nutch WebCrawler will make it an effective tool in helping to discover, track and monitor IED education networks over the World Wide Web.

B. CONCLUSIONS

Based on the experimental results given in Chapter V, the most important documents within a web graph can be filtered out for a given level via an OPIC threshold score. To do this, a reasonable threshold value for a given level must be set by the user. In these experiments, the average value of a node within the depth level was used with moderate success. Additionally, it was confirmed that the more documents found during a given search increases the chances of another document's OPIC score being influenced, thereby increasing their overall score and the chance that the document will cross the set depth level threshold value.

Overall, this research delivered a random network generator with plug-ins capable of simulating the Nutch OPIC algorithm, as well as a new OPIC variant algorithm. In the end, it must be remembered that no matter how great an algorithm is at ranking, the results will only be as good as the pages indexed by the search engine. A page cannot be ranked if it has not been retrieved. All of these issues and more must be taken into account when attempting to find IED education networks over the World Wide Web.

C. FUTURE WORK

Domain comparison is a serious issue not addressed within the scope of this project. Domains were not separated using this search technique, implying a higher importance to the initial domain searched and less to those found during the search. This will pose significant problems when attempting to search across multiple domains. Additionally, once the cash value given to a node becomes small enough, Java floating point errors have the potential to become a problem for large web-link graphs. It is unknown at this time how big of a web link graph would be needed to make this problem a reality.

Implementation of this new algorithm in searching for IED education networks using Nutch could be accomplished through many different methods. One way might be to use a cluster of different computers with many different addresses and merge their results. Unfortunately for this approach, the domain comparison problem previously mentioned will pose significant challenges. Another would be to use Nutch as a cover; actually knowing an IED education network exists for a given domain and initiating a crawl using the known IED education network root node document to determine the depth of the network's existence. Currently, Nutch is optimized for this by being able to effectively search a single domain knowing that the initial document has significant importance.

Monitoring IED education networks found using this algorithm is the next step in determining the true measure of the new algorithm's effectiveness. Unfortunately, Nutch has inherent flaws implementing OPIC in that the historical cash in the system builds very early and decays slowly over time. This will cause scoring problems for later searches that attempt to monitor changes in OPIC scores concerning sites of interest. Later versions of Nutch have neutralized this problem by resetting the historical cash equal to zero upon re-crawl. Again, this causes another problem in that documents of significant importance are not given any weight for having been previously found to be important. Overall, these problems and concerns will need considerable research conducted to achieve a more effective IED education network web crawler.

APPENDIX A. NUTCH XML CONFIGURATION FILE

The following text file given below is the standard default Nutch XML configuration file:

```
<?xml version="1.0"?>
<?xml-stylesheet type="text/xsl" href="configuration.xsl"?>

<!-- Do not modify this file directly.  Instead, copy entries that you
-->
<!-- wish to modify from this file into nutch-site.xml and change them
-->
<!-- there.  If nutch-site.xml does not already exist, create it.
-->


<configuration>

<!-- file properties -->

<property>
  <name>file.content.limit</name>
  <value>65536</value>
  <description>The length limit for downloaded content, in bytes.
   If this value is nonnegative (>=0), content longer than it will be
   truncated; otherwise, no truncation at all.
  </description>
</property>

<property>
  <name>file.content.ignored</name>
  <value>true</value>
  <description>If true, no file content will be saved during fetch.
   And it is probably what we want to set most of time, since file://
   URLs are meant to be local and we can always use them directly at
   Parsing and indexing stages. Otherwise file contents will be saved.
   !! NO IMPLEMENTED YET !!
  </description>
</property>

<!-- HTTP properties -->

<property>
  <name>http.agent.name</name>
  <value></value>
  <description>HTTP 'User-Agent' request header. MUST NOT be empty -
   please set this to a single word uniquely related to your
   organization.

   NOTE: You should also check other related properties:
```

```
      http.robots.agents
      http.agent.description
      http.agent.url
      http.agent.email
      http.agent.version

   and set their values appropriately.

  </description>
</property>

<property>
  <name>http.robots.agents</name>
  <value>*</value>
  <description>The agent strings we'll look for in robots.txt files,
   comma-separated, in decreasing order of precedence. You should
   put the value of http.agent.name as the first agent name, and keep
   the default * at the end of the list. E.g.: BlurflDev,Blurfl,*
  </description>
</property>

<property>
  <name>http.robots.403.allow</name>
  <value>true</value>
  <description>Some servers return HTTP status 403 (Forbidden) if
   /robots.txt doesn't exist. This should probably mean that we are
   allowed to crawl the site nonetheless. If this is set to false,
   then such sites will be treated as forbidden.
  </description>
</property>

<property>
  <name>http.agent.description</name>
  <value></value>
  <description>Further description of our bot- this text is used in
   the User-Agent header.  It appears in parenthesis after the agent
   name.
  </description>
</property>

<property>
  <name>http.agent.url</name>
  <value></value>
  <description>A URL to advertise in the User-Agent header.  This will
   appear in parenthesis after the agent name. Custom dictates that
   this should be a URL of a page explaining the purpose and behavior
   of this crawler.
  </description>
</property>

<property>
  <name>http.agent.email</name>
  <value></value>
  <description>An email address to advertise in the HTTP 'From' request
   header and User-Agent header. A good practice is to mangle this
```

```
   address (e.g. 'info at example dot com') to avoid spamming.
  </description>
</property>

<property>
  <name>http.agent.version</name>
  <value>Nutch-0.8.1</value>
  <description>A version string to advertise in the User-Agent
   header.
  </description>
</property>

<property>
  <name>http.timeout</name>
  <value>10000</value>
  <description>The default network timeout, in
   milliseconds.
  </description>
</property>

<property>
  <name>http.max.delays</name>
  <value>100</value>
  <description>The number of times a thread will delay when trying to
   fetch a page.  Each time it finds that a host is busy, it will wait
   fetcher.server.delay.  After http.max.delays attepts, it will give
   up on the page for now.
  </description>
</property>

<property>
  <name>http.content.limit</name>
  <value>65536</value>
  <description>The length limit for downloaded content, in bytes.
   If this value is nonnegative (>=0), content longer than it will be
   truncated; otherwise, no truncation at all.
  </description>
</property>

<property>
  <name>http.proxy.host</name>
  <value></value>
  <description>The proxy hostname.  If empty, no proxy is
   used.
  </description>
</property>

<property>
  <name>http.proxy.port</name>
  <value></value>
  <description>The proxy port.
  </description>
</property>

<property>
```

```
  <name>http.verbose</name>
  <value>false</value>
  <description>If true, HTTP will log more verbosely.
  </description>
</property>

<property>
  <name>http.redirect.max</name>
  <value>3</value>
  <description>The maximum number of redirects the fetcher will follow
   when trying to fetch a page.
  </description>
</property>

<property>
  <name>http.useHttp11</name>
  <value>false</value>
  <description>NOTE: at the moment this works only for protocol-
   Httpclient. If true, use HTTP 1.1, if false use HTTP 1.0 .
  </description>
</property>

<!-- FTP properties -->

<property>
  <name>ftp.username</name>
  <value>anonymous</value>
  <description>ftp login username.
  </description>
</property>

<property>
  <name>ftp.password</name>
  <value>anonymous@example.com</value>
  <description>ftp login password.
  </description>
</property>

<property>
  <name>ftp.content.limit</name>
  <value>65536</value>
  <description>The length limit for downloaded content, in bytes.
   If this value is nonnegative (>=0), content longer than it will be
   truncated; otherwise, no truncation at all. Caution: classical ftp
   RFCs never defines partial transfer and, in fact, some ftp servers
   out there do not handle client side forced close-down very well. Our
   implementation tries its best to handle such situations smoothly.
  </description>
</property>

<property>
  <name>ftp.timeout</name>
  <value>60000</value>
  <description>Default timeout for ftp client socket, in millisec.
   Please also see ftp.keep.connection below.
```

```xml
  </description>
</property>

<property>
  <name>ftp.server.timeout</name>
  <value>100000</value>
  <description>An estimation of ftp server idle time, in millisec.
   Typically it is 120000 millisec for many ftp servers out there.
   Better be conservative here. Together with ftp.timeout, it is used
   to decide if we need to delete (annihilate) current ftp.client
   instance and force to start another ftp.client instance anew. This
   is necessary because a fetcher thread may not be able to obtain next
   request from queue in time (due to idleness) before our ftp client
   times out or remote server disconnects. Used only when
   ftp.keep.connection is true (please see below).
  </description>
</property>

<property>
  <name>ftp.keep.connection</name>
  <value>false</value>
  <description>Whether to keep ftp connection. Useful if crawling same
   host again and again. When set to true, it avoids connection, login
   and dir list parser setup for subsequent urls. If it is set to true,
   however, you must make sure (roughly):
   (1) ftp.timeout is less than ftp.server.timeout
   (2) ftp.timeout is larger than (fetcher.threads.fetch *
   fetcher.server.delay)
   Otherwise there will be too many "delete client because idled too
   long" messages in thread logs.
  </description>
</property>

<property>
  <name>ftp.follow.talk</name>
  <value>false</value>
  <description>Whether to log dialogue between our client and remote
   server. Useful for debugging.
  </description>
</property>

<!-- web db properties -->

<property>
  <name>db.default.fetch.interval</name>
  <value>30</value>
  <description>The default number of days between re-fetches of a page.
  </description>
</property>

<property>
  <name>db.ignore.internal.links</name>
  <value>true</value>
  <description>If true, when adding new links to a page, links from
   the same host are ignored.  This is an effective way to limit the
```

```
    size of the link database, keeping only the highest quality
    links.
  </description>
</property>

<property>
  <name>db.ignore.external.links</name>
  <value>false</value>
  <description>If true, outlinks leading from a page to external hosts
    will be ignored. This is an effective way to limit the crawl to
    include only initially injected hosts, without creating complex
    URLFilters.
  </description>
</property>

<property>
  <name>db.score.injected</name>
  <value>1.0</value>
  <description>The score of new pages added by the injector.
  </description>
</property>

<property>
  <name>db.score.link.external</name>
  <value>1.0</value>
  <description>The score factor for new pages added due to a link from
    another host relative to the referencing page's score. Scoring
    plugins may use this value to affect initial scores of external
    links.
  </description>
</property>

<property>
  <name>db.score.link.internal</name>
  <value>1.0</value>
  <description>The score factor for pages added due to a link from the
    same host, relative to the referencing page's score. Scoring plugins
    may use this value to affect initial scores of internal links.
  </description>
</property>

<property>
  <name>db.score.count.filtered</name>
  <value>false</value>
  <description>The score value passed to newly discovered pages is
    calculated as a fraction of the original page score divided by the
    number of outlinks. If this option is false, only the outlinks that
    passed URLFilters will count, if it's true then all outlinks will
    count.
  </description>
</property>

<property>
  <name>db.max.inlinks</name>
  <value>10000</value>
```

```
  <description>Maximum number of Inlinks per URL to be kept in LinkDb.
   If "invertlinks" finds more inlinks than this number, only the first
   N inlinks will be stored, and the rest will be discarded.
  </description>
</property>

<property>
  <name>db.max.outlinks.per.page</name>
  <value>100</value>
  <description>The maximum number of outlinks that we'll process for a
   page. If this value is nonnegative (>=0), at most
   db.max.outlinks.per.page outlinks will be processed for a page;
   otherwise, all outlinks will be processed.
  </description>
</property>

<property>
  <name>db.max.anchor.length</name>
  <value>100</value>
  <description>The maximum number of characters permitted in an anchor.
  </description>
</property>

<property>
  <name>db.fetch.retry.max</name>
  <value>3</value>
  <description>The maximum number of times a url that has encountered
   recoverable errors is generated for fetch.
  </description>
</property>

<property>
  <name>db.signature.class</name>
  <value>org.apache.nutch.crawl.MD5Signature</value>
  <description>The default implementation of a page signature.
   Signatures created with this implementation will be used for
   duplicate detection and removal.
  </description>
</property>

<property>
  <name>db.signature.text_profile.min_token_len</name>
  <value>2</value>
  <description>Minimum token length to be included in the signature.
  </description>
</property>

<property>
  <name>db.signature.text_profile.quant_rate</name>
  <value>0.01</value>
  <description>Profile frequencies will be rounded down to a multiple
   of QUANT = (int)(QUANT_RATE * maxFreq), where maxFreq is a maximum
   token frequency. If maxFreq > 1 then QUANT will be at least 2, which
   means that for longer texts tokens with frequency 1 will always be
   discarded.
```

```
  </description>
</property>

<!-- generate properties -->

<property>
  <name>generate.max.per.host</name>
  <value>-1</value>
  <description>The maximum number of urls per host in a single
   fetchlist.  -1 if unlimited.
  </description>
</property>

<property>
  <name>generate.max.per.host.by.ip</name>
  <value>false</value>
  <description>If false, same host names are counted. If true,
   hosts' IP addresses are resolved and the same IP-s are counted.

   -+-+-+- WARNING !!! -+-+-+-
   When set to true, Generator will create a lot of DNS lookup
   requests, rapidly. This may cause a DOS attack on
   remote DNS servers, not to mention increased external traffic
   and latency. For these reasons when using this option it is
   required that a local caching DNS be used.
  </description>
</property>

<!-- fetcher properties -->

<property>
  <name>fetcher.server.delay</name>
  <value>5.0</value>
  <description>The number of seconds the fetcher will delay between
   successive requests to the same server.
  </description>
</property>

<property>
 <name>fetcher.max.crawl.delay</name>
 <value>30</value>
 <description>
  If the Crawl-Delay in robots.txt is set to greater than this value
  (in seconds) then the fetcher will skip this page, generating an
  error report. If set to -1 the fetcher will never skip such pages and
  will wait the amount of time retrieved from robots.txt Crawl-Delay,
  however long that might be.
 </description>
</property>

<property>
  <name>fetcher.threads.fetch</name>
  <value>10</value>
  <description>The number of FetcherThreads the fetcher should use.
   This is also determines the maximum number of requests that are
```

```
   made at once (each FetcherThread handles one connection).
  </description>
</property>

<property>
  <name>fetcher.threads.per.host</name>
  <value>1</value>
  <description>This number is the maximum number of threads that
    should be allowed to access a host at one time.
  </description>
</property>

<property>
  <name>fetcher.threads.per.host.by.ip</name>
  <value>true</value>
  <description>If true, then fetcher will count threads by IP address,
   to which the URL's host name resolves. If false, only host name will
   be used. NOTE: this should be set to the same value as
   "generate.max.per.host.by.ip" - default settings are different only
   for reasons of backward-compatibility.
  </description>
</property>

<property>
  <name>fetcher.verbose</name>
  <value>false</value>
  <description>If true, fetcher will log more verbosely.
  </description>
</property>

<property>
  <name>fetcher.parse</name>
  <value>true</value>
  <description>If true, fetcher will parse content.
  </description>
</property>

<property>
  <name>fetcher.store.content</name>
  <value>true</value>
  <description>If true, fetcher will store content.
  </description>
</property>

<!-- indexer properties -->

<property>
  <name>indexer.score.power</name>
  <value>0.5</value>
  <description>Determines the power of link analyis scores.  Each
   pages's boost is set to <i>score<sup>scorePower</sup></i> where
   <i>score</i> is its link analysis score and <i>scorePower</i> is the
   value of this parameter.  This is compiled into indexes, so, when
   this is changed, pages must be re-indexed for it to take
   effect.
```

```
  </description>
</property>

<property>
  <name>indexer.max.title.length</name>
  <value>100</value>
  <description>The maximum number of characters of a title that are
   indexed.
  </description>
</property>

<property>
  <name>indexer.max.tokens</name>
  <value>10000</value>
  <description>
   The maximum number of tokens that will be indexed for a single field
   in a document. This limits the amount of memory required for
   indexing, so that collections with very large files will not crash
   the indexing process by running out of memory.

   Note that this effectively truncates large documents, excluding
   from the index tokens that occur further in the document. If you
   know your source documents are large, be sure to set this value
   high enough to accomodate the expected size. If you set it to
   Integer.MAX_VALUE, then the only limit is your memory, but you
   should anticipate an OutOfMemoryError.
  </description>
</property>

<property>
  <name>indexer.mergeFactor</name>
  <value>50</value>
  <description>The factor that determines the frequency of Lucene
   segment merges. This must not be less than 2, higher values increase
   indexing speed but lead to increased RAM usage, and increase the
   number of open file handles (which may lead to "Too many open files"
   errors).  NOTE: the "segments" here have nothing to do with Nutch
   segments, they are a low-level data unit used by Lucene.
  </description>
</property>

<property>
  <name>indexer.minMergeDocs</name>
  <value>50</value>
  <description>This number determines the minimum number of Lucene
   Documents buffered in memory between Lucene segment merges. Larger
   values increase indexing speed and increase RAM usage.
  </description>
</property>

<property>
  <name>indexer.maxMergeDocs</name>
  <value>2147483647</value>
  <description>This number determines the maximum number of Lucene
   Documents to be merged into a new Lucene segment. Larger values
```

```
   increase batch indexing speed and reduce the number of Lucene
  segments, which reduces the number of open file handles; however,
  this also decreases incremental indexing performance.
  </description>
</property>

<property>
  <name>indexer.termIndexInterval</name>
  <value>128</value>
  <description>Determines the fraction of terms which Lucene keeps in
   RAM when searching, to facilitate random-access.  Smaller values use
   more memory but make searches somewhat faster.  Larger values use
   less memory but make searches somewhat slower.
  </description>
</property>


<!-- analysis properties -->

<property>
  <name>analysis.common.terms.file</name>
  <value>common-terms.utf8</value>
  <description>The name of a file containing a list of common terms
  that should be indexed in n-grams.
  </description>
</property>

<!-- searcher properties -->

<property>
  <name>searcher.dir</name>
  <value>crawl</value>
  <description>
  Path to root of crawl.  This directory is searched (in
  order) for either the file search-servers.txt, containing a list of
  distributed search servers, or the directory "index" containing
  merged indexes, or the directory "segments" containing segment
  indexes.
  </description>
</property>

<property>
  <name>searcher.filter.cache.size</name>
  <value>16</value>
  <description>
   Maximum number of filters to cache.  Filters can accelerate certain
   field-based queries, like language, document format, etc.  Each
   filter requires one bit of RAM per page.  So, with a 10 million page
   index, a cache size of 16 consumes two bytes per page, or 20MB.
  </description>
</property>

<property>
  <name>searcher.filter.cache.threshold</name>
  <value>0.05</value>
```

```
  <description>
   Filters are cached when their term is matched by more than this
   fraction of pages.  For example, with a threshold of 0.05, and 10
   million pages, the term must match more than 1/20, or 50,000 pages.
   So, if out of 10 million pages, 50% of pages are in English, and 2%
   are in Finnish, then, with a threshold of 0.05, searches for
   "lang:en" will use a cached filter, while searches for "lang:fi"
   will score all 20,000 finnish documents.
  </description>
</property>

<property>
  <name>searcher.hostgrouping.rawhits.factor</name>
  <value>2.0</value>
  <description>
   A factor that is used to determine the number of raw hits
   initially fetched, before host grouping is done.
  </description>
</property>

<property>
  <name>searcher.summary.context</name>
  <value>5</value>
  <description>
   The number of context terms to display preceding and following
   matching terms in a hit summary.
  </description>
</property>

<property>
  <name>searcher.summary.length</name>
  <value>20</value>
  <description>
   The total number of terms to display in a hit summary.
  </description>
</property>

<property>
  <name>searcher.max.hits</name>
  <value>-1</value>
  <description>If positive, search stops after this many hits are
   found.  Setting this to small, positive values (e.g., 1000) can make
   searches much faster.  With a sorted index, the quality of the hits
   suffers little.
  </description>
</property>

<property>
  <name>searcher.max.time.tick_count</name>
  <value>-1</value>
  <description>If positive value is defined here, limit search time for
   every request to this number of elapsed ticks (see the tick_length
   property below). The total maximum time for any search request will
   be then limited to tick_count * tick_length milliseconds. When
   search time is exceeded, partial results will be returned, and the
```

```
    total number of hits will be estimated.
  </description>
</property>

<property>
  <name>searcher.max.time.tick_length</name>
  <value>200</value>
  <description>The number of milliseconds between ticks. Larger values
    reduce the timer granularity (precision). Smaller values bring more
    overhead.
  </description>
</property>

<!-- URL normalizer properties -->

<property>
  <name>urlnormalizer.class</name>
  <value>org.apache.nutch.net.BasicUrlNormalizer</value>
  <description>Name of the class used to normalize URLs.
  </description>
</property>

<property>
  <name>urlnormalizer.regex.file</name>
  <value>regex-normalize.xml</value>
  <description>Name of the config file used by the RegexUrlNormalizer
    class.
  </description>
</property>

<!-- mime properties -->

<property>
  <name>mime.types.file</name>
  <value>mime-types.xml</value>
  <description>Name of file in CLASSPATH containing filename extension
    and magic sequence to mime types mapping information
  </description>
</property>

<property>
  <name>mime.type.magic</name>
  <value>true</value>
  <description>Defines if the mime content type detector uses magic
    resolution.
  </description>
</property>

<!-- plugin properties -->

<property>
  <name>plugin.folders</name>
  <value>plugins</value>
  <description>Directories where nutch plugins are located.  Each
    element may be a relative or absolute path.  If absolute, it is used
```

```
   as is.  If relative, it is searched for on the
  classpath.</description>
</property>

<property>
  <name>plugin.auto-activation</name>
  <value>true</value>
  <description>Defines if some plugins that are not activated regarding
   the plugin.includes and plugin.excludes properties must be
   automaticaly activated if they are needed by some actived plugins.
  </description>
</property>

<property>
  <name>plugin.includes</name>
  <value>protocol-http|urlfilter-regex|parse-(text|html|js)|index-
   basic|query-(basic|site|url)|summary-basic|scoring-opic</value>
  <description>Regular expression naming plugin directory names to
   include.  Any plugin not matching this expression is excluded.
   In any case you need at least include the nutch-extensionpoints
   plugin. By default Nutch includes crawling just HTML and plain text
   via HTTP, and basic indexing and search plugins.
  </description>
</property>

<property>
  <name>plugin.excludes</name>
  <value></value>
  <description>Regular expression naming plugin directory names to
   exclude.
  </description>
</property>

<!-- parser properties -->

<property>
  <name>parse.plugin.file</name>
  <value>parse-plugins.xml</value>
  <description>The name of the file that defines the associations
   between content-types and parsers.
  </description>
</property>

<property>
  <name>parser.character.encoding.default</name>
  <value>windows-1252</value>
  <description>The character encoding to fall back to when no other
   information is available
  </description>
</property>

<property>
  <name>parser.html.impl</name>
  <value>neko</value>
  <description>HTML Parser implementation. Currently the following
```

```
   keywords are recognized: "neko" uses NekoHTML, "tagsoup" uses
   TagSoup.
  </description>
</property>

<property>
  <name>parser.html.form.use_action</name>
  <value>false</value>
  <description>If true, HTML parser will collect URLs from form action
   attributes. This may lead to undesirable behavior (submitting empty
   forms during next fetch cycle). If false, form action attribute will
   be ignored.
  </description>
</property>


<!-- urlfilter plugin properties -->

<property>
  <name>urlfilter.regex.file</name>
  <value>regex-urlfilter.txt</value>
  <description>Name of file on CLASSPATH containing regular expressions
   used by urlfilter-regex (RegexURLFilter) plugin.
  </description>
</property>

<property>
  <name>urlfilter.automaton.file</name>
  <value>automaton-urlfilter.txt</value>
  <description>Name of file on CLASSPATH containing regular expressions
   used by urlfilter-automaton (AutomatonURLFilter) plugin.
  </description>
</property>

<property>
  <name>urlfilter.prefix.file</name>
  <value>prefix-urlfilter.txt</value>
  <description>Name of file on CLASSPATH containing url prefixes
   used by urlfilter-prefix (PrefixURLFilter) plugin.</description>
</property>

<property>
  <name>urlfilter.suffix.file</name>
  <value>suffix-urlfilter.txt</value>
  <description>Name of file on CLASSPATH containing url suffixes
   used by urlfilter-suffix (SuffixURLFilter) plugin.</description>
</property>

<property>
  <name>urlfilter.order</name>
  <value></value>
  <description>The order by which url filters are applied.
   If empty, all available url filters (as dictated by properties
   plugin-includes and plugin-excludes above) are loaded and applied in
   system defined order. If not empty, only named filters are loaded
```

```
   and applied in given order. For example, if this property has value:
   org.apache.nutch.net.RegexURLFilter
   org.apache.nutch.net.PrefixURLFilter
   then RegexURLFilter is applied first, and PrefixURLFilter second.
   Since all filters are AND'ed, filter ordering does not have impact
   on end result, but it may have performance implication, depending
   on relative expensiveness of filters.
  </description>
</property>

<!-- scoring filters properties -->

<property>
  <name>scoring.filter.order</name>
  <value></value>
  <description>The order in which scoring filters are applied.
   This may be left empty (in which case all available scoring
   filters will be applied in the order defined in plugin-includes
   and plugin-excludes), or a space separated list of implementation
   classes.
  </description>
</property>

<!-- clustering extension properties -->

<property>
  <name>extension.clustering.hits-to-cluster</name>
  <value>100</value>
  <description>Number of snippets retrieved for the clustering
   extension if clustering extension is available and user requested
   results to be clustered.
  </description>
</property>

<property>
  <name>extension.clustering.extension-name</name>
  <value></value>
  <description>Use the specified online clustering extension. If empty,
   the first available extension will be used. The "name" here refers
   to an 'id' attribute of the 'implementation' element in the plugin
   descriptor XML file.
  </description>
</property>

<!-- ontology extension properties -->

<property>
  <name>extension.ontology.extension-name</name>
  <value></value>
  <description>Use the specified online ontology extension. If empty,
   the first available extension will be used. The "name" here refers
   to an 'id' attribute of the 'implementation' element in the plugin
   descriptor XML file.
  </description>
</property>
```

```
<property>
  <name>extension.ontology.urls</name>
  <value>
  </value>
  <description>Urls of owl files, separated by spaces, such as
   http://www.example.com/ontology/time.owl
   http://www.example.com/ontology/space.owl
   http://www.example.com/ontology/wine.owl
   Or
   file:/ontology/time.owl
   file:/ontology/space.owl
   file:/ontology/wine.owl
   You have to make sure each url is valid.
   By default, there is no owl file, so query refinement based on
   ontology is silently ignored.
  </description>
</property>

<!-- query-basic plugin properties -->

<property>
  <name>query.url.boost</name>
  <value>4.0</value>
  <description> Used as a boost for url field in Lucene query.
  </description>
</property>

<property>
  <name>query.anchor.boost</name>
  <value>2.0</value>
  <description> Used as a boost for anchor field in Lucene query.
  </description>
</property>

<property>
  <name>query.title.boost</name>
  <value>1.5</value>
  <description> Used as a boost for title field in Lucene query.
  </description>
</property>

<property>
  <name>query.host.boost</name>
  <value>2.0</value>
  <description> Used as a boost for host field in Lucene query.
  </description>
</property>

<property>
  <name>query.phrase.boost</name>
  <value>1.0</value>
  <description> Used as a boost for phrase in Lucene query.
   Multiplied by boost for field phrase is matched in.
  </description>
</property>
```

```xml
<!-- creative-commons plugin properties -->

<property>
  <name>query.cc.boost</name>
  <value>0.0</value>
  <description> Used as a boost for cc field in Lucene query.
  </description>
</property>

<!-- query-more plugin properties -->

<property>
  <name>query.type.boost</name>
  <value>0.0</value>
  <description> Used as a boost for type field in Lucene query.
  </description>
</property>

<!-- query-site plugin properties -->

<property>
  <name>query.site.boost</name>
  <value>0.0</value>
  <description> Used as a boost for site field in Lucene query.
  </description>
</property>

<!-- microformats-reltag plugin properties -->

<property>
  <name>query.tag.boost</name>
  <value>1.0</value>
  <description> Used as a boost for tag field in Lucene query.
  </description>
</property>

<!-- language-identifier plugin properties -->

<property>
  <name>lang.ngram.min.length</name>
  <value>1</value>
  <description> The minimum size of ngrams to uses to identify
   language (must be between 1 and lang.ngram.max.length).
   The larger is the range between lang.ngram.min.length and
   lang.ngram.max.length, the better is the identification, but
   the slowest it is.
  </description>
</property>

<property>
  <name>lang.ngram.max.length</name>
  <value>4</value>
  <description> The maximum size of ngrams to uses to identify
   language (must be between lang.ngram.min.length and 4).
   The larger is the range between lang.ngram.min.length and
```

```
   lang.ngram.max.length, the better is the identification, but
   the slowest it is.
  </description>
</property>

<property>
  <name>lang.analyze.max.length</name>
  <value>2048</value>
  <description> The maximum bytes of data to uses to indentify
   the language (0 means full content analysis).
   The larger is this value, the better is the analysis, but the
   slowest it is.
  </description>
</property>

<property>
  <name>query.lang.boost</name>
  <value>0.0</value>
  <description> Used as a boost for lang field in Lucene query.
  </description>
</property>

</configuration>
```

THIS PAGE INTENTIONALLY LEFT BLANK

APPENDIX B. LUCENE SCORING EXAMPLE

The example provided below calculates an $Overall\_Score(q,d)$ from Equation 3.2 given the following information:

A hypothetical query for the phrase "big bang" is conducted and document D1 was selected for analysis. For the word "big", D1 has a term frequency $tf(t\_in\_d)$ equal to 3, an inverse document frequency $idf(t)$ equal to 2, a boost value $boost(t.field\_in\_d)$ equal to 1 (i.e. no boost), and a length normalization value $lengthNorm(t.field\_in\_d)$ equal to 5. For the word "bang", D1 has a term frequency $tf(t\_in\_d)$ equal to 2, an inverse document frequency $idf(t)$ equal to 1.5, a boost value $boost(t.field\_in\_d)$ equal to 1 (i.e. no boost), and a length normalization value $lengthNorm(t.field\_in\_d)$ equal to 5. Applying Equation 3.1, the score value $score(q,d)$ for the query "big bang" in document D1 is equal to 82.5.

Taking this one step further, an overall score value $Overall\_Score(q,d)$ is calculated using an overall boost value $Overall\_Boost(d)$ equal to 0.12, a coordination factor $coord(q,d)$ equal to 0.25 and a query normalization value $queryNorm(q)$ equal to 0.15. Document D1 is then calculated to have an overall score of 0.37125.

THIS PAGE INTENTIONALLY LEFT BLANK

APPENDIX C. SIMULATION 3 WEB LINK GRAPH

The following data is the high complexity random network generated in simulation 3 for Chapter V.

Doc Num	Depth	Ext Flag	Num Outlinks	Type of Outlink										Outlink Doc Num									
1	1	0	10	1	3	1	1	3	3	1	1	1	3	2	1	3	4	1	1	5	6	7	1
2	2	0	10	3	1	4	4	4	4	1	4	4	4	2	8	6	1	8	5	9	8	4	4
3	2	0	6	4	4	1	1	4	1	0	0	0	0	2	7	10	11	7	12	0	0	0	0
4	2	0	2	4	4	0	0	0	0	0	0	0	0	9	11	0	0	0	0	0	0	0	0
5	2	0	9	3	1	1	4	4	4	1	4	1	0	5	13	14	4	4	3	15	13	16	0
6	2	0	5	1	4	4	3	1	0	0	0	0	0	17	3	10	6	18	0	0	0	0	0
7	2	0	6	1	1	1	3	4	1	0	0	0	0	19	20	21	7	10	22	0	0	0	0
8	3	0	8	4	1	1	1	4	4	2	4	0	0	3	23	24	25	19	12	26	6	0	0
9	3	0	5	1	4	2	4	4	0	0	0	0	0	27	26	28	7	14	0	0	0	0	0
10	3	0	5	1	4	1	1	1	0	0	0	0	0	29	5	30	31	32	0	0	0	0	0
11	3	0	7	1	1	1	4	1	4	1	0	0	0	33	34	35	8	36	1	37	0	0	0
12	3	0	5	3	1	4	4	1	0	0	0	0	0	12	38	6	24	39	0	0	0	0	0
13	3	0	8	1	1	4	4	4	1	1	1	0	0	40	41	32	29	4	42	43	44	0	0
14	3	0	4	4	1	4	2	0	0	0	0	0	0	7	45	37	46	0	0	0	0	0	0
15	3	0	7	4	1	4	4	1	4	1	0	0	0	17	47	16	8	48	18	49	0	0	0
16	3	0	6	1	1	4	1	4	4	0	0	0	0	50	51	8	52	1	21	0	0	0	0
17	3	0	3	4	4	4	0	0	0	0	0	0	0	40	35	15	0	0	0	0	0	0	0
18	3	0	10	1	4	2	2	1	1	4	4	3	1	53	11	54	55	56	57	43	19	18	58
19	3	0	5	1	4	1	2	1	0	0	0	0	0	59	40	60	61	62	0	0	0	0	0
20	3	0	7	1	4	1	4	1	4	3	0	0	0	63	40	64	36	65	59	20	0	0	0
21	3	0	8	1	3	4	4	1	4	3	1	0	0	66	21	33	48	67	32	21	68	0	0

Doc Num	Depth	Ext Flag	Num Outlinks	Type of Outlink										Outlink Doc Num									
22	3	0	9	4	4	1	4	1	4	1	4	1	0	9	7	69	23	70	52	71	29	72	0
23	4	0	9	4	3	1	1	2	1	1	1	2	0	55	23	73	74	75	76	77	78	79	0
24	4	0	1	4	0	0	0	0	0	0	0	0	0	53	0	0	0	0	0	0	0	0	0
25	4	0	7	1	4	1	4	3	1	4	0	0	0	80	74	81	33	25	82	61	0	0	0
26	4	1	4	4	4	1	1	0	0	0	0	0	0	15	81	83	84	0	0	0	0	0	0
27	4	0	7	1	4	1	1	4	3	1	0	0	0	85	18	86	87	86	27	88	0	0	0
28	4	1	3	4	4	4	0	0	0	0	0	0	0	8	19	41	0	0	0	0	0	0	0
29	4	0	8	4	4	1	1	4	1	4	1	0	0	58	38	89	90	84	91	68	92	0	0
30	4	0	4	1	4	3	1	0	0	0	0	0	0	93	4	30	94	0	0	0	0	0	0
31	4	0	9	3	1	4	1	1	1	1	4	4	0	31	95	60	96	97	98	99	34	24	0
32	4	0	10	1	4	4	4	4	1	1	1	1	4	100	73	67	11	95	101	102	103	104	50
33	4	0	6	4	1	1	1	1	4	0	0	0	0	100	105	106	107	108	82	0	0	0	0
34	4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
35	4	0	9	4	3	4	1	4	4	4	1	4	0	2	35	105	109	7	75	100	110	81	0
36	4	0	8	4	4	1	1	4	4	1	4	0	0	35	97	111	112	108	103	113	40	0	0
37	4	0	1	1	0	0	0	0	0	0	0	0	0	114	0	0	0	0	0	0	0	0	0
38	4	0	9	1	4	4	1	1	4	4	4	3	0	115	45	19	116	117	34	78	103	38	0
39	4	0	1	1	0	0	0	0	0	0	0	0	0	118	0	0	0	0	0	0	0	0	0
40	4	0	9	1	1	4	1	4	1	4	4	4	0	119	120	116	121	66	122	84	62	7	0
41	4	0	3	1	1	4	0	0	0	0	0	0	0	123	124	55	0	0	0	0	0	0	0
42	4	0	5	4	1	4	1	4	0	0	0	0	0	68	125	2	126	95	0	0	0	0	0
43	4	0	4	1	4	4	1	0	0	0	0	0	0	127	55	75	128	0	0	0	0	0	0
44	4	0	3	1	1	1	0	0	0	0	0	0	0	129	130	131	0	0	0	0	0	0	0
45	4	0	10	1	1	1	1	4	4	3	1	4	4	132	133	134	135	128	82	45	136	33	78
46	4	1	7	4	4	2	1	1	2	4	0	0	0	62	119	137	138	139	140	53	0	0	0
47	4	0	7	2	1	2	1	1	2	1	0	0	0	141	142	143	144	145	146	147	0	0	0
48	4	0	4	1	1	2	4	0	0	0	0	0	0	148	149	150	62	0	0	0	0	0	0
49	4	0	5	4	3	4	1	4	0	0	0	0	0	6	49	77	151	90	0	0	0	0	0
50	4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Doc Num	Depth	Ext Flag	Num Outlinks	Type of Outlink										Outlink Doc Num									
51	4	0	1	4	0	0	0	0	0	0	0	0	0	87	0	0	0	0	0	0	0	0	0
52	4	0	2	3	1	0	0	0	0	0	0	0	0	52	152	0	0	0	0	0	0	0	0
53	4	0	10	4	2	1	4	4	4	1	4	4	2	79	153	154	113	100	106	155	57	110	156
54	4	1	6	1	4	4	1	4	4	0	0	0	0	157	57	59	158	85	10	0	0	0	0
55	4	1	6	1	4	1	4	4	1	0	0	0	0	159	14	160	133	10	161	0	0	0	0
56	4	0	6	4	1	4	1	4	4	0	0	0	0	35	162	61	163	95	40	0	0	0	0
57	4	0	9	1	1	1	1	1	4	4	4	4	0	164	165	166	167	168	154	140	45	153	0
58	4	0	8	1	1	2	2	4	2	1	4	0	0	169	170	171	172	62	173	174	48	0	0
59	4	0	5	4	4	4	4	1	0	0	0	0	0	170	115	72	30	175	0	0	0	0	0
60	4	0	5	4	4	4	4	4	0	0	0	0	0	55	22	117	137	68	0	0	0	0	0
61	4	1	4	4	4	4	4	0	0	0	0	0	0	5	107	128	167	0	0	0	0	0	0
62	4	0	6	1	4	4	4	4	3	0	0	0	0	176	93	89	86	150	62	0	0	0	0
63	4	0	9	4	1	4	1	4	4	2	4	4	0	93	177	123	178	131	114	179	138	40	0
64	4	0	4	1	4	1	4	0	0	0	0	0	0	180	152	181	63	0	0	0	0	0	0
65	4	0	9	1	4	1	1	4	4	1	1	1	0	182	48	183	184	60	180	185	186	187	0
66	4	0	2	4	4	0	0	0	0	0	0	0	0	22	49	0	0	0	0	0	0	0	0
67	4	0	6	2	1	4	3	4	1	0	0	0	0	188	189	97	67	152	190	0	0	0	0
68	4	0	9	1	4	1	4	4	1	1	4	1	0	191	161	192	74	118	193	194	99	195	0
69	4	0	4	1	1	1	4	0	0	0	0	0	0	196	197	198	98	0	0	0	0	0	0
70	4	0	7	4	4	1	4	4	1	1	0	0	0	39	112	199	189	192	200	201	0	0	0
71	4	0	8	4	2	4	1	4	4	4	4	0	0	33	202	185	203	98	81	106	186	0	0
72	4	0	7	2	1	4	4	4	4	4	0	0	0	204	205	38	53	16	6	99	0	0	0
73	5	0	4	1	1	4	4	0	0	0	0	0	0	206	207	67	46	0	0	0	0	0	0
74	5	0	7	1	1	4	4	2	4	4	0	0	0	208	209	147	78	210	161	122	0	0	0
75	5	1	3	4	2	1	0	0	0	0	0	0	0	184	211	212	0	0	0	0	0	0	0
76	5	0	10	1	1	4	4	1	4	4	1	1	3	213	214	1	91	215	25	24	216	217	76
77	5	0	6	1	4	4	1	1	1	0	0	0	0	218	209	145	219	220	221	0	0	0	0
78	5	0	8	1	1	2	3	1	4	4	1	0	0	222	223	224	78	225	41	165	226	0	0
79	5	1	8	4	4	1	1	4	1	4	4	0	0	213	174	227	228	225	229	124	159	0	0

Doc Num	Depth	Ext Flag	Num Outlinks	Type of Outlink										Outlink Doc Num									
80	5	0	5	1	4	1	4	1	0	0	0	0	0	230	58	231	154	232	0	0	0	0	0
81	5	0	5	1	4	4	1	1	0	0	0	0	0	233	43	225	234	235	0	0	0	0	0
82	5	0	9	1	1	1	1	1	4	1	4	1	0	236	237	238	239	240	205	241	66	242	0
83	5	0	5	4	4	4	4	4	0	0	0	0	0	49	163	44	155	106	0	0	0	0	0
84	5	0	7	4	4	1	4	1	1	4	0	0	0	151	86	243	228	244	245	34	0	0	0
85	5	0	3	1	1	4	0	0	0	0	0	0	0	246	247	164	0	0	0	0	0	0	0
86	5	0	10	1	1	4	1	1	1	4	4	4	4	248	249	24	250	251	252	57	164	119	177
87	5	0	1	1	0	0	0	0	0	0	0	0	0	253	0	0	0	0	0	0	0	0	0
88	5	0	3	1	4	4	0	0	0	0	0	0	0	254	79	103	0	0	0	0	0	0	0
89	5	0	8	1	2	4	1	4	4	1	1	0	0	255	256	155	257	111	112	258	259	0	0
90	5	0	1	1	0	0	0	0	0	0	0	0	0	260	0	0	0	0	0	0	0	0	0
91	5	0	2	1	1	0	0	0	0	0	0	0	0	261	262	0	0	0	0	0	0	0	0
92	5	0	5	1	1	3	1	4	0	0	0	0	0	263	264	92	265	261	0	0	0	0	0
93	5	0	7	1	3	1	4	4	1	1	0	0	0	266	93	267	216	189	268	269	0	0	0
94	5	0	3	1	4	4	0	0	0	0	0	0	0	270	51	100	0	0	0	0	0	0	0
95	5	0	4	1	1	1	1	0	0	0	0	0	0	271	272	273	274	0	0	0	0	0	0
96	5	0	8	1	4	1	4	4	1	1	4	0	0	275	187	276	62	59	277	278	191	0	0
97	5	0	9	1	1	4	4	1	1	4	3	4	0	279	280	264	180	281	282	134	97	222	0
98	5	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
99	5	0	1	4	0	0	0	0	0	0	0	0	0	154	0	0	0	0	0	0	0	0	0
100	5	0	5	1	1	2	1	4	0	0	0	0	0	283	284	285	286	144	0	0	0	0	0
101	5	0	9	1	1	4	4	4	4	1	4	4	0	287	288	98	73	248	160	289	280	268	0
102	5	0	4	2	1	1	4	0	0	0	0	0	0	290	291	292	255	0	0	0	0	0	0
103	5	0	7	4	3	4	1	1	4	4	0	0	0	291	103	66	293	294	281	49	0	0	0
104	5	0	7	1	1	1	1	4	4	4	0	0	0	295	296	297	298	187	73	129	0	0	0
105	5	0	4	4	4	4	2	0	0	0	0	0	0	116	182	232	299	0	0	0	0	0	0
106	5	0	3	4	4	1	0	0	0	0	0	0	0	264	244	300	0	0	0	0	0	0	0
107	5	0	6	4	2	4	1	2	4	0	0	0	0	200	301	260	302	303	131	0	0	0	0
108	5	0	3	1	4	1	0	0	0	0	0	0	0	304	93	305	0	0	0	0	0	0	0

Doc Num	Depth	Ext Flag	Num Outlinks	Type of Outlink										Outlink Doc Num									
109	5	0	2	1	1	0	0	0	0	0	0	0	0	306	307	0	0	0	0	0	0	0	0
110	5	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
111	5	0	3	3	4	1	0	0	0	0	0	0	0	111	206	308	0	0	0	0	0	0	0
112	5	0	1	1	0	0	0	0	0	0	0	0	0	309	0	0	0	0	0	0	0	0	0
113	5	0	8	2	1	1	2	1	4	4	1	0	0	310	311	312	313	314	278	125	315	0	0
114	5	0	1	4	0	0	0	0	0	0	0	0	0	82	0	0	0	0	0	0	0	0	0
115	5	0	7	1	1	1	4	1	1	4	0	0	0	316	317	318	104	319	320	57	0	0	0
116	5	0	5	4	1	3	2	1	0	0	0	0	0	102	321	116	322	323	0	0	0	0	0
117	5	0	5	4	1	4	1	3	0	0	0	0	0	198	324	80	325	117	0	0	0	0	0
118	5	0	4	1	1	4	1	0	0	0	0	0	0	326	327	44	328	0	0	0	0	0	0
119	5	0	1	4	0	0	0	0	0	0	0	0	0	27	0	0	0	0	0	0	0	0	0
120	5	0	10	1	4	2	1	1	2	1	3	1	4	329	83	330	331	332	333	334	120	335	135
121	5	0	1	1	0	0	0	0	0	0	0	0	0	336	0	0	0	0	0	0	0	0	0
122	5	0	7	3	1	4	1	1	1	1	0	0	0	122	337	266	338	339	340	341	0	0	0
123	5	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
124	5	0	3	1	1	1	0	0	0	0	0	0	0	342	343	344	0	0	0	0	0	0	0
125	5	0	1	4	0	0	0	0	0	0	0	0	0	134	0	0	0	0	0	0	0	0	0
126	5	0	5	1	4	4	4	4	0	0	0	0	0	345	30	105	158	188	0	0	0	0	0
127	5	0	4	1	1	2	1	0	0	0	0	0	0	346	347	348	349	0	0	0	0	0	0
128	5	0	1	4	0	0	0	0	0	0	0	0	0	306	0	0	0	0	0	0	0	0	0
129	5	0	9	1	1	4	4	4	1	1	1	4	0	350	351	105	170	140	352	353	354	134	0
130	5	0	5	4	4	1	4	1	0	0	0	0	0	25	211	355	226	356	0	0	0	0	0
131	5	0	3	1	1	1	0	0	0	0	0	0	0	357	358	359	0	0	0	0	0	0	0
132	5	0	7	1	4	1	1	1	4	1	0	0	0	360	281	361	362	363	258	364	0	0	0
133	5	0	8	1	1	4	1	1	1	1	1	0	0	365	366	200	367	368	369	370	371	0	0
134	5	0	9	4	1	3	4	4	1	1	1	1	0	60	372	134	290	361	373	374	375	376	0
135	5	0	4	4	4	1	4	0	0	0	0	0	0	290	103	377	368	0	0	0	0	0	0
136	5	0	5	1	1	1	4	1	0	0	0	0	0	378	379	380	273	381	0	0	0	0	0
137	5	1	5	1	1	1	4	1	0	0	0	0	0	382	383	384	156	385	0	0	0	0	0

Doc Num	Depth	Ext Flag	Num Outlinks	Type of Outlink											Outlink Doc Num									
138	5	0	9	4	4	4	1	4	1	4	1	2	0	221	174	184	386	284	387	227	388	389	0
139	5	0	7	1	4	1	1	4	1	4	0	0	0	390	33	391	392	300	393	306	0	0	0
140	5	1	8	4	1	1	1	4	1	1	4	0	0	230	394	395	396	124	397	398	81	0	0
141	5	1	6	1	1	1	4	4	1	0	0	0	0	399	400	401	306	225	402	0	0	0	0
142	5	0	1	4	0	0	0	0	0	0	0	0	0	172	0	0	0	0	0	0	0	0	0
143	5	1	1	1	0	0	0	0	0	0	0	0	0	403	0	0	0	0	0	0	0	0	0
144	5	0	6	4	1	1	4	1	4	0	0	0	0	146	404	405	192	406	182	0	0	0	0
145	5	0	7	4	2	4	1	4	4	4	0	0	0	374	407	50	408	309	181	362	0	0	0
146	5	1	8	4	4	4	4	1	3	4	3	0	0	148	356	210	240	409	146	287	146	0	0
147	5	0	9	1	4	1	4	4	2	1	1	4	0	410	43	411	212	173	412	413	414	324	0
148	5	0	9	4	4	1	1	1	1	1	4	1	0	312	383	415	416	417	418	419	45	420	0
149	5	0	6	3	1	1	4	1	1	0	0	0	0	149	421	422	355	423	424	0	0	0	0
150	5	1	5	4	1	2	4	2	0	0	0	0	0	184	425	426	334	427	0	0	0	0	0
151	5	0	2	1	4	0	0	0	0	0	0	0	0	428	342	0	0	0	0	0	0	0	0
152	5	0	7	4	4	1	4	1	1	4	0	0	0	171	408	429	366	430	431	168	0	0	0
153	5	1	7	1	1	4	1	4	2	1	0	0	0	432	433	89	434	373	435	436	0	0	0
154	5	0	1	4	0	0	0	0	0	0	0	0	0	217	0	0	0	0	0	0	0	0	0
155	5	0	9	4	4	4	1	1	4	1	1	4	0	154	37	177	437	438	203	439	440	98	0
156	5	1	1	4	0	0	0	0	0	0	0	0	0	131	0	0	0	0	0	0	0	0	0
157	5	0	6	4	1	4	1	3	1	0	0	0	0	378	441	174	442	157	443	0	0	0	0
158	5	0	7	4	4	1	1	4	4	1	0	0	0	356	266	444	445	190	139	446	0	0	0
159	5	0	3	4	4	1	0	0	0	0	0	0	0	279	27	447	0	0	0	0	0	0	0
160	5	0	10	4	4	1	4	4	3	4	1	4	4	184	241	448	129	305	160	182	449	60	261
161	5	0	1	4	0	0	0	0	0	0	0	0	0	324	0	0	0	0	0	0	0	0	0
162	5	0	8	1	4	3	4	1	1	4	4	0	0	450	258	162	206	451	452	200	39	0	0
163	5	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
164	5	0	8	4	1	4	1	1	1	4	4	0	0	93	453	327	454	455	456	167	253	0	0
165	5	0	6	4	1	1	1	4	1	0	0	0	0	383	457	458	459	98	460	0	0	0	0
166	5	0	7	4	4	1	4	1	2	4	0	0	0	43	372	461	125	462	463	318	0	0	0

Doc Num	Depth	Ext Flag	Num Outlinks	Type of Outlink										Outlink Doc Num									
167	5	0	9	1	4	4	4	4	1	1	1	1	0	464	445	92	229	41	465	466	467	468	0
168	5	0	8	3	4	1	4	1	1	4	1	0	0	168	12	469	158	470	471	319	472	0	0
169	5	0	10	1	1	4	1	4	4	4	4	1	1	473	474	198	475	72	386	46	415	476	477
170	5	0	5	1	1	1	4	1	0	0	0	0	0	478	479	480	23	481	0	0	0	0	0
171	5	1	8	4	4	4	4	2	1	4	1	0	0	97	94	410	332	482	483	183	484	0	0
172	5	1	2	1	4	0	0	0	0	0	0	0	0	485	41	0	0	0	0	0	0	0	0
173	5	1	3	1	1	4	0	0	0	0	0	0	0	486	487	262	0	0	0	0	0	0	0
174	5	0	8	1	2	1	2	4	4	1	4	0	0	488	489	490	491	130	85	492	250	0	0
175	5	0	6	1	1	4	4	4	1	0	0	0	0	493	494	430	466	53	495	0	0	0	0
176	5	0	5	1	4	4	2	1	0	0	0	0	0	496	79	189	497	498	0	0	0	0	0
177	5	0	4	4	1	2	1	0	0	0	0	0	0	387	499	500	501	0	0	0	0	0	0
178	5	0	10	1	1	4	4	4	4	4	4	4	1	502	503	87	379	38	37	128	78	96	504
179	5	1	6	3	3	4	3	4	1	0	0	0	0	179	179	462	179	352	505	0	0	0	0
180	5	0	9	4	4	1	4	1	4	1	1	1	0	39	471	506	4	507	64	508	509	510	0
181	5	0	10	1	1	4	2	1	4	2	4	4	3	511	512	486	513	514	25	515	489	99	181
182	5	0	2	1	1	0	0	0	0	0	0	0	0	516	517	0	0	0	0	0	0	0	0
183	5	0	2	3	4	0	0	0	0	0	0	0	0	183	30	0	0	0	0	0	0	0	0
184	5	0	6	1	1	1	4	1	1	0	0	0	0	518	519	520	256	521	522	0	0	0	0
185	5	0	2	4	4	0	0	0	0	0	0	0	0	179	240	0	0	0	0	0	0	0	0
186	5	0	1	1	0	0	0	0	0	0	0	0	0	523	0	0	0	0	0	0	0	0	0
187	5	0	1	4	0	0	0	0	0	0	0	0	0	77	0	0	0	0	0	0	0	0	0
188	5	1	4	4	4	1	1	0	0	0	0	0	0	293	503	524	525	0	0	0	0	0	0
189	5	0	5	1	4	1	4	4	0	0	0	0	0	526	493	527	470	177	0	0	0	0	0
190	5	0	6	1	4	4	1	1	4	0	0	0	0	528	326	43	529	530	141	0	0	0	0
191	5	0	6	4	1	1	2	2	2	0	0	0	0	75	531	532	533	534	535	0	0	0	0
192	5	0	3	1	1	4	0	0	0	0	0	0	0	536	537	289	0	0	0	0	0	0	0
193	5	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
194	5	0	5	4	4	1	4	1	0	0	0	0	0	138	505	538	214	539	0	0	0	0	0
195	5	0	4	4	4	3	4	0	0	0	0	0	0	237	59	195	45	0	0	0	0	0	0

Doc Num	Depth	Ext Flag	Num Outlinks	Type of Outlink										Outlink Doc Num									
196	5	0	6	1	1	4	1	4	4	0	0	0	0	540	541	471	542	529	335	0	0	0	0
197	5	0	3	2	4	4	0	0	0	0	0	0	0	543	508	123	0	0	0	0	0	0	0
198	5	0	2	4	4	0	0	0	0	0	0	0	0	260	442	0	0	0	0	0	0	0	0
199	5	0	4	1	2	1	4	0	0	0	0	0	0	544	545	546	413	0	0	0	0	0	0
200	5	0	8	4	4	4	4	4	1	1	4	0	0	24	439	372	450	72	547	548	30	0	0
201	5	0	8	4	1	1	1	4	4	1	1	0	0	489	549	550	551	165	64	552	553	0	0
202	5	1	1	4	0	0	0	0	0	0	0	0	0	347	0	0	0	0	0	0	0	0	0
203	5	0	1	4	0	0	0	0	0	0	0	0	0	140	0	0	0	0	0	0	0	0	0
204	5	1	9	1	4	2	1	4	1	4	4	1	0	554	364	555	556	290	557	23	377	558	0
205	5	0	3	1	4	3	0	0	0	0	0	0	0	559	516	205	0	0	0	0	0	0	0
206	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
207	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
208	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
209	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
210	6	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
211	6	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
212	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
213	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
214	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
215	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
216	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
217	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
218	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
219	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
220	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
221	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
222	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
223	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
224	6	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Doc Num	Depth	Ext Flag	Num Outlinks	Type of Outlink										Outlink Doc Num									
225	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
226	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
227	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
228	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
229	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
230	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
231	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
232	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
233	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
234	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
235	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
236	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
237	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
238	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
239	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
240	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
241	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
242	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
243	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
244	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
245	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
246	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
247	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
248	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
249	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
250	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
251	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
252	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
253	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Doc Num	Depth	Ext Flag	Num Outlinks	Type of Outlink										Outlink Doc Num									
254	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
255	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
256	6	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
257	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
258	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
259	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
260	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
261	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
262	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
263	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
264	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
265	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
266	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
267	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
268	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
269	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
270	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
271	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
272	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
273	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
274	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
275	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
276	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
277	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
278	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
279	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
280	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
281	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
282	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Doc Num	Depth	Ext Flag	Num Outlinks	Type of Outlink										Outlink Doc Num									
283	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
284	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
285	6	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
286	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
287	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
288	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
289	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
290	6	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
291	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
292	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
293	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
294	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
295	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
296	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
297	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
298	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
299	6	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
300	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
301	6	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
302	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
303	6	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
304	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
305	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
306	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
307	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
308	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
309	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
310	6	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
311	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Doc Num	Depth	Ext Flag	Num Outlinks	Type of Outlink										Outlink Doc Num									
312	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
313	6	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
314	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
315	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
316	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
317	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
318	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
319	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
320	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
321	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
322	6	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
323	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
324	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
325	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
326	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
327	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
328	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
329	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
330	6	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
331	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
332	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
333	6	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
334	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
335	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
336	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
337	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
338	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
339	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
340	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Doc Num	Depth	Ext Flag	Num Outlinks	Type of Outlink										Outlink Doc Num									
341	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
342	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
343	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
344	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
345	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
346	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
347	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
348	6	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
349	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
350	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
351	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
352	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
353	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
354	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
355	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
356	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
357	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
358	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
359	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
360	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
361	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
362	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
363	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
364	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
365	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
366	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
367	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
368	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
369	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Doc Num	Depth	Ext Flag	Num Outlinks	Type of Outlink										Outlink Doc Num									
370	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
371	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
372	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
373	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
374	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
375	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
376	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
377	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
378	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
379	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
380	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
381	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
382	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
383	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
384	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
385	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
386	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
387	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
388	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
389	6	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
390	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
391	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
392	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
393	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
394	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
395	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
396	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
397	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
398	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Doc Num	Depth	Ext Flag	Num Outlinks	Type of Outlink										Outlink Doc Num									
399	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
400	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
401	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
402	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
403	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
404	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
405	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
406	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
407	6	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
408	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
409	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
410	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
411	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
412	6	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
413	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
414	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
415	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
416	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
417	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
418	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
419	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
420	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
421	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
422	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
423	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
424	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
425	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
426	6	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
427	6	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Doc Num	Depth	Ext Flag	Num Outlinks	Type of Outlink										Outlink Doc Num									
428	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
429	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
430	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
431	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
432	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
433	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
434	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
435	6	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
436	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
437	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
438	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
439	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
440	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
441	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
442	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
443	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
444	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
445	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
446	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
447	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
448	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
449	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
450	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
451	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
452	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
453	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
454	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
455	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
456	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Doc Num	Depth	Ext Flag	Num Outlinks	Type of Outlink										Outlink Doc Num									
457	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
458	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
459	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
460	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
461	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
462	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
463	6	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
464	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
465	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
466	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
467	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
468	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
469	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
470	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
471	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
472	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
473	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
474	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
475	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
476	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
477	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
478	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
479	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
480	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
481	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
482	6	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
483	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
484	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
485	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Doc Num	Depth	Ext Flag	Num Outlinks	Type of Outlink										Outlink Doc Num									
486	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
487	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
488	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
489	6	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
490	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
491	6	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
492	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
493	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
494	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
495	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
496	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
497	6	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
498	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
499	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
500	6	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
501	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
502	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
503	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
504	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
505	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
506	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
507	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
508	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
509	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
510	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
511	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
512	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
513	6	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
514	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Doc Num	Depth	Ext Flag	Num Outlinks	Type of Outlink										Outlink Doc Num									
515	6	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
516	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
517	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
518	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
519	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
520	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
521	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
522	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
523	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
524	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
525	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
526	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
527	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
528	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
529	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
530	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
531	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
532	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
533	6	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
534	6	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
535	6	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
536	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
537	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
538	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
539	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
540	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
541	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
542	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
543	6	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Doc Num	Depth	Ext Flag	Num Outlinks	Type of Outlink										Outlink Doc Num									
544	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
545	6	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
546	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
547	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
548	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
549	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
550	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
551	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
552	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
553	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
554	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
555	6	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
556	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
557	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
558	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
559	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

LIST OF REFERENCES

[1] National Science Foundation, Scientists Use the "Dark Web" to Snag Extremists and Terrorists Online. Retrieved January 9, 2009 from http://www.nsf.gov/news/news_summ.jsp?cntn_id=110040

[2] Department of Defense, *Joint Publication 1-02: Department of Defense Dictionary of Military and Associated Terms*, October 2008.

[3] Belarus During the Great Patriotic War. Retrieved January 9, 2009 from http://www.belarus.by/en/belarus/history/11/index3.php

[4] IEDs: the insurgent's deadliest weapon. Retrieved January 9, 2009 from http://www.thefirstpost.co.uk/46075,features,ieds-the-insurgents-deadliest-weapons

[5] G. Grant, 900 IED Attacks a Month in Iraq and Afghanistan: Metz. Retrieved December 16, 2008 from http://www.dodbuzz.com/2008/12/12/900-ied-attacks-a-month-in-iraq-and-afghanistan-metz/

[6] The Jolly Roger's Cookbook III. Retrieved January 9, 2009 from http://www.textfiles.com/anarchy/JOLLYRODGER

[7] D. Vise and M. Malseed, *The Google Story*, New York: Bantam Dell, November 2005.

[8] M. Berry and M. Brown, *Understanding Search Engines: Mathematical Modeling and Text Retrieval*, Ed. 2, p.5, Philadelphia: Society for Industrial and Applied Mathematics, 2005.

[9] D. Grossman and O. Frieder, *Information Retrieval: Algorithms and Heuristics*, Ed. 2, pp. 9-92, Netherlands: Springer, 2004.

[10] N. Fuhr, Probabilistic Models in Information Retrieval. The Computer Journal, 35(3): 243-255, 1992.

[11] B. Pinkerton, Finding What People Want: Experiences with the WebCrawler. Retrieved November 15, 2008 from http://thinkpink.com/bp/WebCrawler/WWW94.html.

[12] J. Cho, H. Garcia-Molina and L. Page, Efficient Crawling Through URL Ordering. Retrieved November 11, 2008 from http://infolab.stanford.edu/pub/papers/efficient-crawling.ps

[13] F. Menczer, G. Pant and P. Srinivasan, Topical Web Crawlers: Evaluating Adaptive Algorithms. ACM Transactions on Internet Technology, 4(4): 378-419, 2004.

[14] M. Hersovici, M. Jacovi, Y. Maarek, D. Pelleg, M. Shtalhaim and S. Ur, The shark-search algorithm. "An application: tailored Web site mapping." Computer Networks and ISDN Systems, vol. 30, pp. 317-326, 1998.

[15] M. Degeratu and F. Menczer, Complementing Search Engines with Online Web Mining Agents. July 26, 2000. Retrieved February 3, 2009 from http://dollar.biz.uiowa.edu/~ fil/Papers/dm-dss.pdf

[16] S. Brin and L. Page, The Anatomy of a Large-Scale Hypertextual Web Search Engine. Retrieved March 5, 2009 from http://ilpubs.stanford.edu:8090/422/1/1999-66.pdf

[17] S. Brin and L. Page, The PageRank Citation Ranking: Bringing Order to the Web. January 29, 1998. Retrieved March 5, 2009 from http://infolab.stanford.edu/~backrub/pageranksub.ps

[18] S. Al-Saffar and G. Heileman, "Experimental Bounds on the Usefulness of Personalized and Topic-Sensitive PageRank," in ACM International Conference on Web Intelligence, 2007, pp. 671-675.

[19] Y. Zhang, C. Yin and F. Yuan, "An Application of Improved PageRank in Focused Crawler," in Fourth International Conference on Fuzzy Systems and Knowledge Discovery, 2007.

[20] F. Yuan, C. Yin and J. Liu, "Improvement of PageRank for Focused Crawler," in Eighth ACIS International Conference on Software Engineering, Artificial Intelligence, Networking, and Parallel/Distributed Computing, 2007, pp.797-802.

[21] W. Xing and A. Ghorbani, "Weighted PageRank Algorithm," in Proceedings of the Second Annual Conference on Communication Networks and Services Research, 2004.

[22] M. Eirinaki and M. Vazirgiannis, "Usage-based PageRank for Web Personalization," in Proceedings of the Fifth IEEE International Conference on Data Mining, 2005.

[23] H. Jiang, Y. GE, D. Zuo and B. Han, "TimeRank: A Method of Improving Ranking Scores by Visited Time," in Proceedings of the Seventh International Conference on Machine Learning and Cybernetics, 2008.

[24] M. Kale and P. Thilagam, "DYNA-RANK: Efficient calculation and updation of PageRank," in International Conference on Computer Science and Information Technology, 2008.

[25] M. Konchady, *Building Search Applications: Lucene, LingPipe, and Gate*, p 321-336, Oakton: Mustru Publishing, 2008.

[26] O. Gospodnetic and E. Hatcher, *Lucene In Action*, Greenwich: Manning Publications Co., 2005.

[27] S. Abiteboul, M. Preda and G. Cobena, "Adaptive On-Line Page Importance Computation" WWW2003 Conference, 2003.

THIS PAGE INTENTIONALLY LEFT BLANK

INITIAL DISTRIBUTION LIST

1. Defense Technical Information Center
   Ft. Belvoir, Virginia

2. Dudley Knox Library
   Naval Postgraduate School
   Monterey, California