

Andrews University

Digital Commons @ Andrews University

Master's Theses

Graduate Research

2012

An Introduction to a Meta-meta-search Engine

Martin Lalnunsanga
Andrews University

Follow this and additional works at: <https://digitalcommons.andrews.edu/theses>

Recommended Citation

Lalnunsanga, Martin, "An Introduction to a Meta-meta-search Engine" (2012). *Master's Theses*. 13.
<https://digitalcommons.andrews.edu/theses/13>

This Thesis is brought to you for free and open access by the Graduate Research at Digital Commons @ Andrews University. It has been accepted for inclusion in Master's Theses by an authorized administrator of Digital Commons @ Andrews University. For more information, please contact repository@andrews.edu.



Seek Knowledge. Affirm Faith. Change the World.

Thank you for your interest in the

**Andrews University Digital Library
of Dissertations and Theses.**

*Please honor the copyright of this document by
not duplicating or distributing additional copies
in any form without the author's express written
permission. Thanks for your cooperation.*

ABSTRACT

AN INTRODUCTION TO A META-META-SEARCH ENGINE

by

Martin Lalnunsanga

Chair: Roy Villafane

ABSTRACT OF GRADUATE STUDENT RESEARCH

Thesis

Andrews University

College of Arts and Sciences

Title: AN INTRODUCTION TO A META-META-SEARCH ENGINE

Name of researcher: Martin Lalnunsanga

Name and degree of faculty chair: Roy Villafane, Ph.D.

Date completed: July 2012

Imagine that all the information in the entire world written in every known language, and every graphic image, video clip, or photograph copied digitally was available at your fingertips. This vast amount of data could then be reduced to digital data packets and stored in miniscule form on computer hard drives that are all connected by several other centrally located larger machines, called servers. However, searching for data in a vast system of inter-connected computers is virtually impossible using human faculties and is a far more intricate process than perusing book titles using the library's Dewey Decimal System. In order to find five or six pieces of information out of a global network of servers, individuals can explore the advantages of meta-search engines, which understand the "language" of each computer on the network and can quickly access global databases to respond to user inquiries, based on certain keywords or phrases.

The advantages of meta-search engines are that they are able to "talk" to other search engines, which contain relevant data. The language that they speak is HTML, or hypertext markup language, a set of electronic codes that enables computers to read, translate, transmit, and store data accessible to the entire world. Every Web page is written in HTML using meta "tags," which are directives to client computers describing the kind of document stored. By reading meta tags, search engines are able to electronically "skim" through vast databases to select data that match a user's inquiry. However, the existing meta-search engines are still facing issues in providing accurate results that match user queries due to the extremely fast growth and the complexity of information that is stored in the Web server.

This thesis proposes a new algorithm that will re-rank the Web search results from some of the best existing meta-search engines. This algorithm can be implemented to form a meta-meta-search engine. As a result, the new search engine will have the capability of listing a more reliable rank list with higher accuracy in comparison to the existing search engines and meta-search engines.

Andrews University
College of Arts and Sciences

AN INTRODUCTION TO A META-META-SEARCH ENGINE

A Thesis
Present in Partial Fulfillment
of the Requirements for the Degree
Master of Science

by
Martin Lalnunsanga

2012

© Copyright by Martin Lalnunsanga 2012

All Rights Reserved

AN INTRODUCTION TO A META-META-SEARCH ENGINE

A thesis
presented in partial fulfillment
of the requirements for the degree
Master of Science

by

Martin Lalnunsanga

APPROVAL BY THE COMMITTEE:

Roy Villafane, Ph.D., Chair

Stephen Thorman, Ph.D.

William Wolfer, M.S.

Date approved

TABLE OF CONTENTS

LIST OF ILLUSTRATIONS	v
LIST OF TABLES	vi
LIST OF ABBREVIATIONS.....	vii
ACKNOWLEDGMENTS	ix
Chapter	
1. INTRODUCTION	1
The Problem.....	3
Purpose of Study	3
Significant of Study	3
2. INFORMATION RETRIEVAL OVERVIEW	4
What Is Information Retrieval?	4
Web Search Background.....	4
Web Browser	5
Brief History	5
The Challenge	7
Basic Design	7
Huge Data/Information	7
Web Characteristics	8
3. SEARCH ENGINE.....	13
What Is a Search Engine?	13
The Purpose of a Search Engine	13
General Search Engine Design	14
Web Crawler	16
Snippet	19
Multiple Search Engine Design	19
What Is the Semantic Web?.....	20
Surface Web vs. Deep Web	21
Understanding the Meta-Search Engine	23

4. SEARCH ENGINE RESULT GROUPING METHOD	27
Result Grouping and Ranking Method	27
Method 1: Fetch Retrieved Document	28
Method 2: Taking the Best Rank	28
Method 3: Weighted Borda-Fuse.....	28
Method 4: Borda's Positional Method.....	29
Method 5: The Original KE (Key Extraction) Algorithm	29
Method 6: Borda Count	30
Method 7: D-WISE Method.....	30
Method 8: Use Top Document to Compute Search Engine Score (TopD)	31
Method 9: Merging Based on Combination Documents Records (SRRs).....	32
Method 10: Use Top Search Result Records (SRRs) to Compute Search Engine Score (TopSRR)	33
Method 11: Compute Simple Similarities Between SRRs and Query (SRRsim).....	34
Method 12: Rank SRRs Using More Features (SRRRank)	35
Method 13: Compute Similarities Between SRRs and Query Using More Features (SRRSimMF)	36
5. META-META SEARCH ENGINE.....	37
MSE Selection Method for MMSE.....	38
Overview of the Existing MSE	39
MSE Selection Criteria	39
Selected MSE.....	41
Proposed Algorithm for MMSE	43
Refined Borda Count	43
6. IMPLEMENTATION.....	46
7. CONCLUSIONS AND FUTURE WORK.....	54
REFERENCE LIST	57

LIST OF ILLUSTRATIONS

1.	Statistics on Number of New Posts and Comments Daily.....	9
2.	Statistics on Number of New Pages Added Daily	10
3.	Statistics on Number of Tweets That Twitter Received Per Day	11
4.	An Example of Crawler-based Search Engine (Google)	15
5.	An Example of Human-Powered Search Engine (Open Directory)	15
6.	An Example of Web Crawler Algorithms	17
7.	Web Crawler Architecture	18
8.	Google Search Result for ‘Burger King’ That Shows Snippets	19
9.	Meta-Search Engine Function.....	24
10.	Architecture of Meta-Search Engine	25
11.	MMSE Architecture.....	38
12.	Pseudo Code for Refined Borda-Count	45
13.	MMSE Ranking System Where Query “Piracy” Is Searched.....	47
14.	The Input and Output Program Screen for MMSE Ranking Algorithm	48

LIST OF TABLES

1.	Brief History About Web Browsers.....	6
2.	Common MSE and Their Features	40
3.	Final Rank Calculation for MMSE.....	49
4.	Calculation for Rank Relation Coefficient Using Sample Data	52

LIST OF ABBREVIATIONS

ADJ	Adjacently
CD	Candidate
CUSI	Configurable Unified Search Index
HTML	Hypertext Markup Language
KE	Key Extraction
MSE	Meta-Search Engine
MMSE	Meta-Meta-Search Engine
NDT	Number of District
QLEN	Query Length
RDF	Resource Description Framework
ROI	Return on Investment
SADJ	Score Adjacently
SD-FN/RNK	Square Difference Between Final Rank and Rank
SE	Search Engine
SRR	Search Result Record
SSRRank	Search Result Record Rank
SSRSim	Search Result Record Similarities
SRRSimMF	Search Result Record Similarities and Query Using More Features
TLoc	The Location of the Occurrence
TNT	Total Number Occurrence of the Query Terms
TopD	Top Document

TITLEN	Title Length
URL	Uniform Resource Locator
WS	Window Size

ACKNOWLEDGMENTS

I could not have completed this research without God who gave me all the strength that I needed and I give thanks to Him for everything. A special thanks to my wife, Babie Lalremsiami, who stood by me through thick and thin, continuously praying for me to accomplish this project. I would like to thank my professor and academic advisor, Mr. William Wolfer, who encouraged me to join this master's program and who is always ready to do whatever it takes to give support. May God bless you.

I also give many thanks to my thesis advisor, Dr. Roy Villafane, for his direction, technical assistance, patience, and support during this research.

CHAPTER 1

INTRODUCTION

Computers have entered almost every arena of human society. They operate in our homes, our workplaces, our schools, our businesses, our social life, and in almost every aspect of our lives. Many of the world's societies depend heavily on computers in the operation of their transportation systems, commerce, utilities, law enforcement, governance, and more. On the other hand, the Internet plays a vital role to successfully and effectively perform various tasks. About 92% of adult Internet users in the U.S. use search engines, with 59% doing so on a regular basis (Young, 2011). However, due to the unstructured nature of the World Wide Web (www), retrieving accurate and relevant information using search engines becomes challenging (Shettar & Bhuptani, 2008, p. 18). A very common issue on the existing search engines is that they return too many unrelated results for users' queries. Web users also play an important role in causing this issue and not just the search engine. For instance, if the query is too general, it is extremely difficult for the search engine to identify the specific documents in which the user was interested. As a result, to find the requested information the user is made to sift through a long list of irrelevant documents. Such a type of search is called a low precision search (Zamir, 1999). While users can be trained on how to use the search engine effectively, the existing search engines make little effort to understand users' intentions, and they retrieve documents that just match query words literally and

syntactically. In this thesis I am proposing a new search engine technique that will give more accurate search results by re-ranking the rank results from five existing meta-search engines, which will be called a Meta-Meta-Search engine (MMSE). There are several existing techniques to solve this problem, but so far there is no absolute remedy that can give a 100% solution.

Chapter 2 opens up our mind to the world of an Information Retrieval (IR) system. It explains the basic definition of an information retrieval system and the challenges faced in retrieving effective information among trillions of information added to the Web daily. Basic information about Web browsers is also discussed with their brief history that is helpful for the reader to understand the importance of retrieving correct data when it is required. In Chapter 3 the reader is introduced to a search engine and how search engines are useful for information retrieval. The main purpose of a search engine and their detail components will also be discussed. This will take us to the understanding of a meta-search engine.

Chapter 4 presents a thorough study on the existing method of grouping search engines. This will educate the reader to understand some of the common algorithms used in grouping search engines to form a meta-search engine. Then Chapter 5 explains meta-meta-search engines and introduces the main core of this research. The meta-search engine (MSE) selection method to develop a MMSE is also discussed. The proposed algorithm is also explained using a demonstration example.

Implementation of my new algorithm is seen in Chapter 6. And finally, conclusions and recommendations appear in Chapter 7.

The Problem

With millions of additional information stored in the Web daily, it is becoming more and more difficult for search engines to generate only the user's expected information or Web pages. Nevertheless, because of users' general-purpose approach, it is very common to receive unnecessary information that is useless to the user. This consumes a lot of time and energy that is worth millions of dollars daily if combined. Although several search engines have been proposed in order to resolve this issue, none of them provide an outstanding solution to it.

Purpose of Study

The purpose for this research was to develop a new concept of search engine that will minimize useless pages for the search result. The new system will utilize the existing information retrieval technique, yet reduce the complexity for general Web users. In order to achieve this goal, extensive research was performed on the existing meta-search engines and the algorithms that are used.

Significance of Study

A meta-meta-search engine provides the layered architecture that possibly will allow overcoming current search engine limitations. Several search engines have been proposed, which allow increasing information retrieval accuracy by exploiting a key component of Semantic Web resources, that is, relations. I believed that this research will not only confirm the validity of the existing unimplemented ideas but also will open a new focus in developing an effective general-purpose search engine.

CHAPTER 2

INFORMATION RETRIEVAL OVERVIEW

What Is Information Retrieval?

Information Retrieval (IR) is the tracing and recovery of specific information from stored data. It is the area of study concerned with searching for documents, for information within documents, and for metadata about documents, as well as that of searching structured storage, relational databases, and the World Wide Web (Definition-of.net, 2012).

Web Search Background

The Web is unprecedented in many ways: unprecedented in scale, unprecedented in the almost-complete lack of coordination in its creation, and unprecedented in the diversity of backgrounds and motives of its participants. Each of these contributes to making Web search different—and generally far harder—than searching “traditional” documents (Manning, Raghavan, & Schütze, 2008).

Manning et al. (2008) also state that the invention of hypertext, envisioned by Vannevar Bush in the 1940s and first realized in working systems in the 1970s, significantly precedes the formation of the World Wide Web in the 1990s. Web usage has shown tremendous growth to the point where it now claims a good fraction of humanity as participants, by relying on a simple, open, client-server design: (a) the server

communicates with the client via a protocol (the http or hypertext transfer protocol) that is lightweight and simple, asynchronously carrying a variety of payloads (text, images and—over time—richer media such as audio and video files) encoded in a simple markup language called HTML (for hypertext markup language); (b) the client—generally a browser, an application within a graphical user environment—can ignore what it does not understand. Each of these seemingly innocuous features has contributed enormously to the growth of the Web, so it is worthwhile to examine them further.

Web Browser

A Web browser is a software application used to locate, retrieve, and display content on the World Wide Web, including Web pages, images, video, and other files. As a client/server model, the browser is the client run on a computer that contacts the Web server and requests information. The Web server sends the information back to the Web browser, which displays the results on the computer or other Internet-enabled device that supports a browser (“Browser,” 2012).

Therefore, the Web browser together with the search engine became the main tool for today’s information retrieval system.

Brief History

Although there are many other notable Web browsers that have evolved today, the following browsers are the most popular ones in 2012. See Table 1.

Table 1

Brief History About Web Browsers

Browser Name	Release Year/Date	Description
WorldWideWeb	February 26, 1991	It was renamed to Nexus to avoid confusion with the WWW system; was the first graphical Web browser and WYSIWYG HTML editor.
Mosaic	April 22, 1993	Mosaic is credited with popularizing the internet and introducing the Web to the public. Contemporaries such as Internet Explorer and Firefox still use many of the Graphical User Interface (GUI) characteristics of Mosaic such as a top-oriented action bar that provides basic browsing functionalities.
Netscape	October 13, 1994	Marc Andreessen—lead software engineer of Mosaic—ventures out on his own, forming Netscape and releasing the first commercial Web browser: Netscape Navigator.
Internet Explorer (IE)	August 16, 1995	Released by Microsoft answering the release of Navigator with its own browser.
Opera	1996	Released to the public by the largest Norwegian telecommunications company called Telenor. Two years later, it tries to grab hold of the internet-enabled handheld device market, starting a port of Opera to mobile device platforms.
Mozilla Navigator	June 05, 1998	Netscape starts the open source Mozilla project to develop the next generation of Communicator. It becomes evident that a project built around the existing source code was difficult, so focus shifts to building from scratch.
Safari	January 7, 2003	Safari 1.0 was released by Apple and initially worked only on Macintosh. Not until mid-2007 did a version appear for Windows XP, Vista, and 7.
Mozilla Firefox	November 09, 2004	Firefox 1.0, already with a huge following of early adopters via their beta releases, enters the stage. Firefox comprises 7.4% of browsers being used by the end of the year.
Google Chrome	September 2, 2008	It has 43 languages and became a huge success, continuously gaining more users until this time.
RockMelt	November 8, 2010	It is a free social media Web browser developed by Tim Howes and Eric Vishria. The project is backed by Netscape founder Marc Andreessen.

Note. Adapted from “The History of Web Browsers,” by G. Jacob, 2009, retrieved from <http://sixrevisions.com/web-development/the-history-of-web-browsers>, and “The History of Web Browsers,” by P. Daniel, 2010, retrieved from <http://www.instantshift.com/2010/10/15/the-history-of-web-browsers>

The Challenge

The Basic Design

The designers of the first browsers made it easy to view the HTML markup tags on the content of an URL. This simple convenience allowed new users to create their own HTML content without extensive training or experience; rather, they learned from example content that they liked. As they did so, a second feature of browsers supported the rapid proliferation of Web content creation and usage: browsers ignored what they did not understand. This led to the creation of numerous incompatible dialects of HTML. Amateur content creators could freely experiment with and learn from their newly created Web pages without fear that a simple syntax error would bring the system down (Manning et al., 2008).

Huge Data/Information

The mass publishing of information on the Web is essentially useless unless this wealth of information can be discovered and consumed by other users. While continually indexing a significant fraction of the Web, the first generation of Web search engines was largely successful at solving the initial challenges, handling queries with sub-second response times. However, the quality and relevance of Web search results left much to be desired owing to the idiosyncrasies of content creation on the Web, where the number of information files on the Web database daily became increasingly larger with multiple format types. This necessitated the invention of new ranking and spam-fighting techniques in order to ensure the quality of the search results (Manning et al., 2008, p. 422).

WordPress.com users produce about 500,000 new posts and 400,000 new comments on an average day. This means that there are more than 20K posts per hour and three posts every second. Figure 1 presents statistics relating to WordPress.com, which does not include the activity on self-hosted blogs. Figure 2 represents the additional pages of information added to the Internet daily.

According to Mediaistro.com 2012 statistics, Twitter received 1.75 million tweets per day, which means there are 10,000 tweets per second See Figure 3. If this resulted in 10,000 reads per second and each tweet text was equal to 140 characters, that is the equivalent of 200 bytes. We would get

$$\begin{aligned} 10000 \times 200 &= 2000000/1024 = 1953.125 \text{ Kilobytes per second} \\ &\approx 144.44 \text{ Megabytes per minute} \\ &\approx 160 \text{ Gigabytes per day} \end{aligned}$$

Web Characteristics

The essential feature that led to the explosive growth of the Web—decentralized content publishing with essentially no central control of authorship—turned out to be the biggest challenge for Web search engines in their quest to index and retrieve this content.

Web page authors created content in dozens of (natural) languages and thousands of dialects, thus demanding many different forms of stemming and other linguistic operations. Because publishing was now open to tens of millions, Web pages exhibited heterogeneity at a daunting scale, in many crucial aspects. First, content-creation was no longer the privy of editorially trained writers; while this represented a tremendous democratization of content creation, it also resulted in a tremendous variation in grammar and style (and in many cases, no recognizable grammar or style) (Manning et al., 2008).

How many people read blogs on WordPress.com?

Over **324 million people** view more than **2.5 billion pages** each month.

[View weekly pageview stats.](#)



How many posts are published on WordPress.com?

WordPress.com users produce about **500,000 new posts** and **400,000 new comments** on an average day.

[View more posting stats.](#)



Figure 1. Statistics on number of new posts and comments daily. From en.wordpress.com/stats, 2012.

Pages

In addition to blog posts, WordPress.com users can publish web pages as well. Pages are separate from a normal blog chronology but can still be easily managed in the interface (you are reading such a page right now). This allows people to easily create an entire site full of hierarchical content about whatever they like.

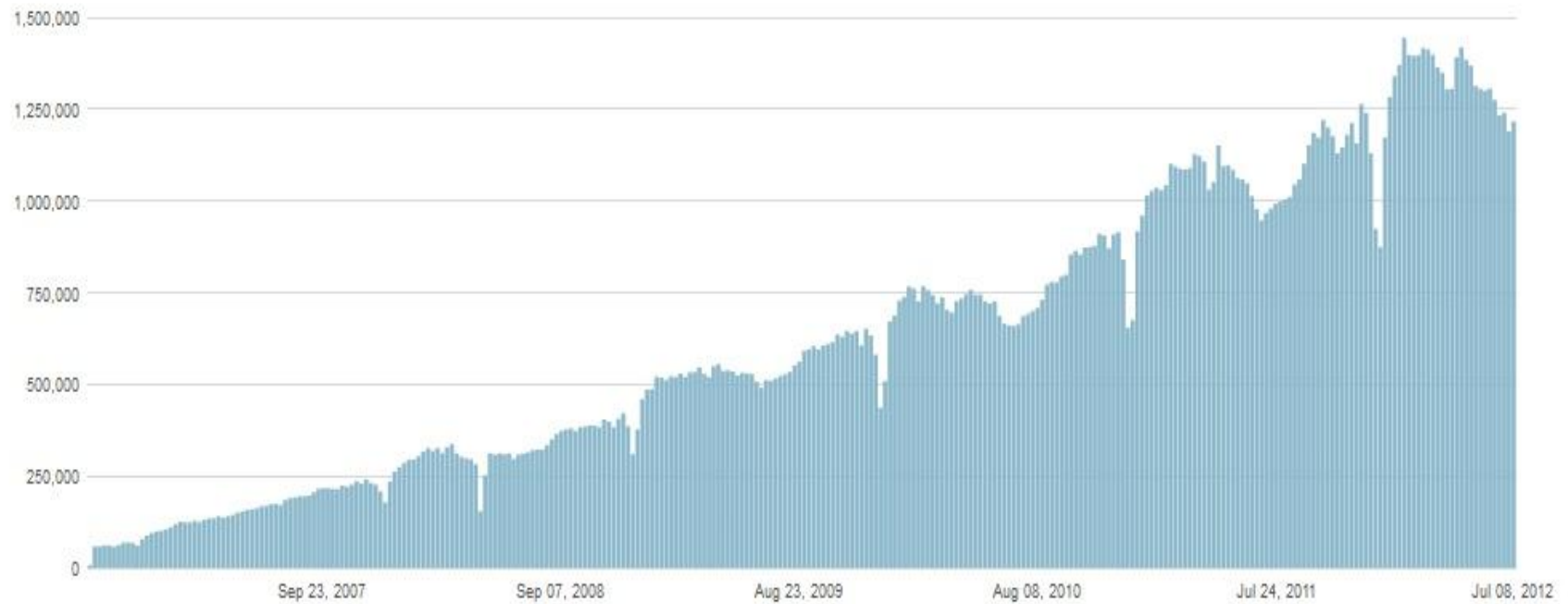


Figure 2. Statistics on number of new pages added daily. From en.wordpress.com/stats, 2012.

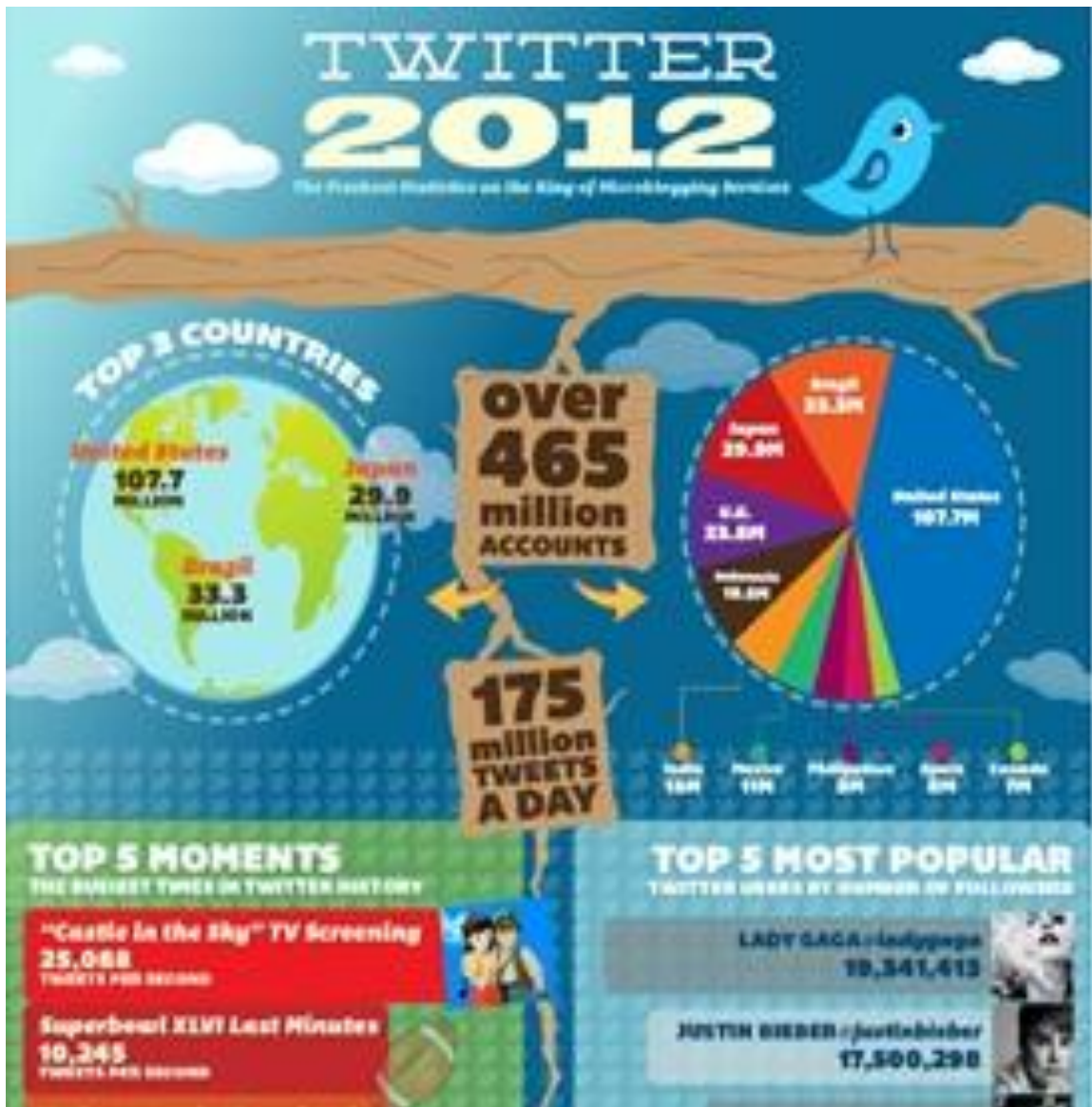


Figure 3. Statistics on number of tweets that Twitter receives per day. From www.mediabistro.com/alltwitter/twitter-statistics-2012_b18914.

More subtly, no universal, user-independent notion of trust may exist. A Web page whose contents are trustworthy to one user may not be so to another. In traditional (non-Web) publishing, this is not an issue: users self-select sources they find trustworthy.

Thus one reader may find the reporting of the *New York Times* to be reliable, while another may prefer the *Wall Street Journal*. But when a search engine is the only viable means for a user to become aware of (let alone select) most content, this challenge becomes significant (Manning et al., 2008).

CHAPTER 3

SEARCH ENGINE

What Is a Search Engine?

A search engine is a World Wide Web application that searches information or Web sites or which users are looking for based on the user's specified keywords typed or inserted in the search box (Freedomscientific.com, 2008). Keywords can be a single word or can be a phrase that is written in the search box of the search engine.

A search engine (SE) is really a general class of programs; however, the term is often used to specifically describe systems like Google, Bing, and Yahoo! Search that enable users to search for documents on the World Wide Web and USENET newsgroups (Webopedia.com, 2011). The users of the World Wide Web find it easy to search the information on a SE, then going through each and every Web site related to their need. Some famous search engines are Google, Bing, Yahoo, Ask, Duckduckgo, Altavista, etc. (Harpreet, 2010).

The Purpose of a Search Engine

For non-technical persons, the common purpose of a search engine is to determine the relevancy of keywords to the content of Web pages. It is done by using software robots to index all the words on billions of pages. Then, they analyze these indexes according to a set of secret algorithms. The order of relevancy will vary among search engines. Different search engines use different methods (algorithms) to index and rank

Web pages. This difference in results is why many searchers prefer one search engine to another. They feel that they get results that are more relevant for their particular needs

(Built-a-Home-Business, 2011):

The most common purpose for developing a search engine from the designer's point of view is for business. Advertisers are major sources of income for Search Engines. If they do not get value (ROI, or return on investment) for their advertising dollars, they stop giving their businesses. Having said that, while trying to develop simple, fast, clean and advertisement free yet effective search engines, we cannot ignore users that are attracted by the advertisements. (p. 1)

General Search Engine Design

The term "search engine" is often used generically to describe both crawler-based search engines and human-powered directories. These two types of search engines gather their listings in radically different ways. Crawler-based search engines, such as Google (Figure 4), create their listings automatically. They "crawl" or "spider" the Web, then people search through what they have found (Latha & Rajaram, 2010).

Figure 5 shows an example of a human-powered directory, such as the Open Directory. Latha and Rajaram (2010) further explain that this type of search engine depends on humans for its listings, which means the search is dependent upon the user's description. When a person submits a short description to the directory for his or her entire site, or editors write one for sites they review, a search looks for matches only in the descriptions submitted.

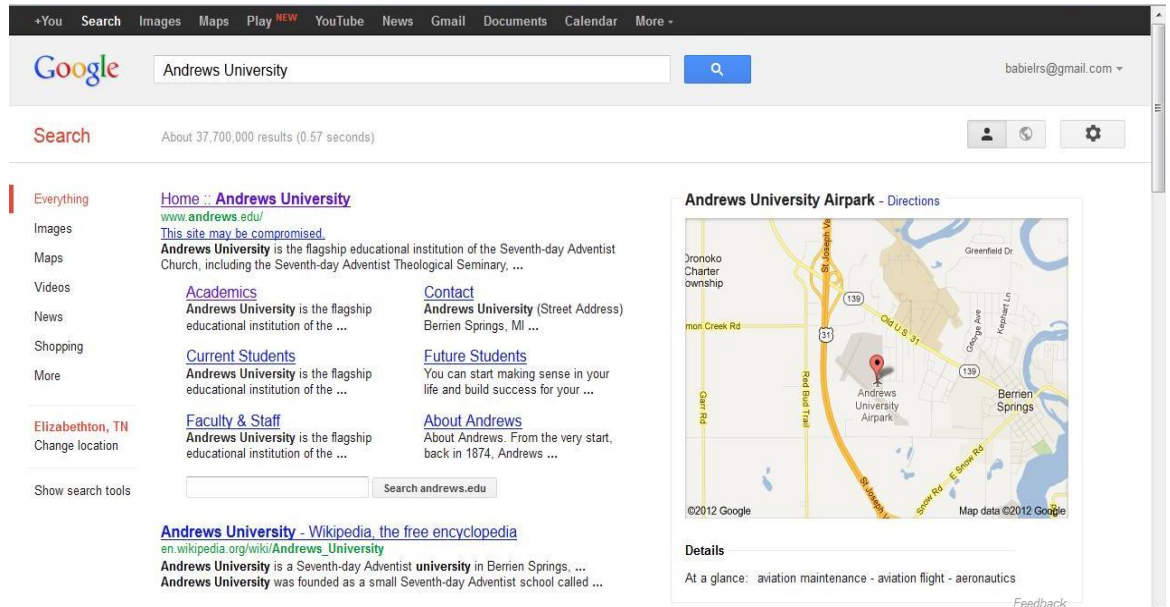


Figure 4. Example of crawler-based search engine (Google) showing Andrews University search results.

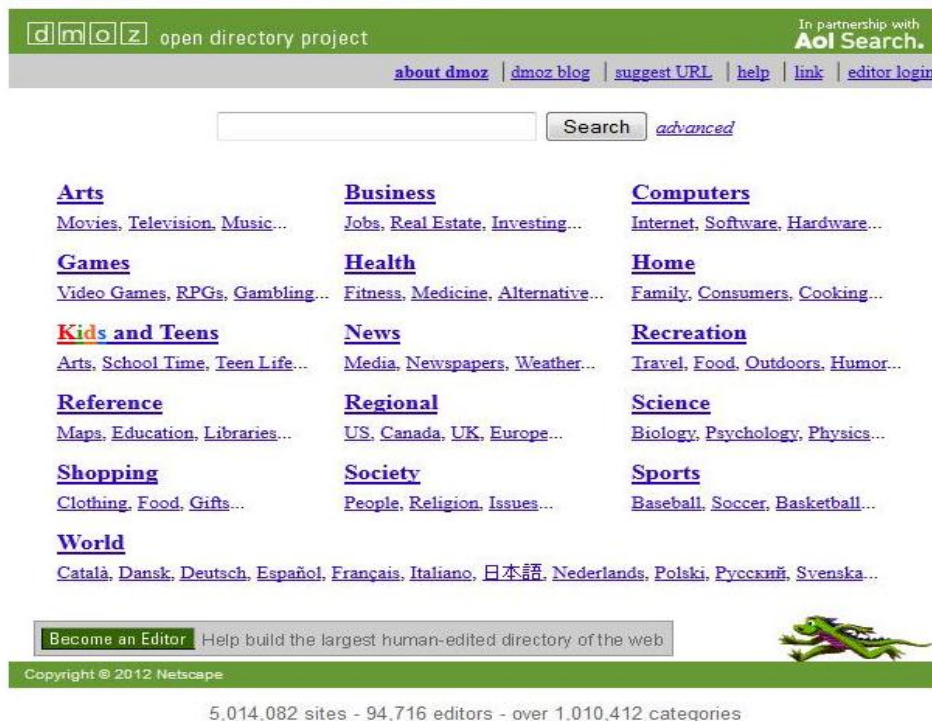


Figure 5. Example of human-powered search engine showing open directory project main categories.

However, while this approach generally works reasonably well, there are of course flaws in the system. If people can work out how a search engine ranks, they can rewrite their individual Web pages to attract higher rankings—often known as ‘gaming’ a search engine (Bradly, 2008). Bradly also uses other reputed human directory-based search engines such as Anoox, Chacha, Colorative Revelance Search Engine, Mahalo, etc., to clarify his point. Bradly states:

While *Anoox* is very straightforward search engine which provides very brief listings of results (title and URL), together with a voting button—to move a result up or down the rankings, or to vote it as spam. I will confess that it does not inspire me with confidence, the first result returned for a search on ‘librarian’ is a placeholder site. Before I can vote results up and down I have to register, declaring my name, address, telephone number, area of expertise and email address. This hiccough takes us straight to the heart of the problem with human-powered search engines—the element of fraud or spam. It’s all too easy to game the engine artificially by voting one particular site higher and another lower

Chaha provides straightforward results in exactly the manner you would expect—sponsored links followed by results with titles, reasonable summaries and URLs. It also provides ‘related searches’ as well. The interesting point with *ChaCha* however is the ‘search with a guide’ option—and you need to register in order to be able to use this.

Collarity learns over time by watching the searches that are performed and matching them to appropriate results. This is best explained by way of an example. However, there is a commercial aspect to this enterprise, so it may not be appropriate for everyone.

Mahalo is taking some of the best elements of existing social networking systems such as Facebook, as well as social bookmarking systems and blending them into a new style network. However, once again there is a problem here, because I have friends and colleagues with widely different areas of interests which do not necessarily overlap. While my contacts may be interested in anything that I find which relates to search engines it doesn’t necessarily follow that they will be equally interested in material on the football team I support or my photography interests.

Web Crawler

The Web crawler is one of the two components directly interacting with the Internet, which is often called a Web spider or robot. Its major role is to automatically discover new resources on the Web in order to make them searchable. Such process is

defined by the Web Discovery Problem (Selberg, 1999). How can all Web pages be located? Web crawlers solve the problem by recursively following hyperlinks obtained from the already visited pages (see Figure 6).

```
/**
 * An example web crawler algorithm
 *
 * urlPool is a set of internet addresses initially containing
 * at least one URL
 * documentIndex is a data structure that stores information about
 * the contents of the crawled pages
 */
webCrawler (UrlPool urlPool, DocumentIndex documentIndex)
{
  while (urlPool not empty)
  {
    url = pick URL from urlPool;
    doc = download url;
    newUrls = extract URLs from doc;
    insert doc into documentIndex;
    insert url into indexedUrls;
    for each u in newUrls
    {
      if (u not in indexedUrls)
      {
        add u to urlPool
      }
    }
  }
}
```

Figure 6. Web crawler algorithm pseudo-code. Adapted from “Towards Comprehensive Web Search” (Unpublished doctoral dissertation), by E. W. Selberg, University of Washington, Seattle, 1999.

The crawler-type search engine has many elements. The first is called the spider, also called the crawler. The spider visits a Web page, reads it, and then follows links to other pages within the site. This is why it refers to a site being "spidered" or "crawled." The spider returns to the site on a regular basis, such as every month or two, to look for changes. Many Websites, in particular search engines, use spidering as a means of providing up-to-date data. Web crawlers are used mainly to create a copy of all the

visited pages for later processing by a search engine that will index the downloaded pages to provide fast searches (see Figure 7).

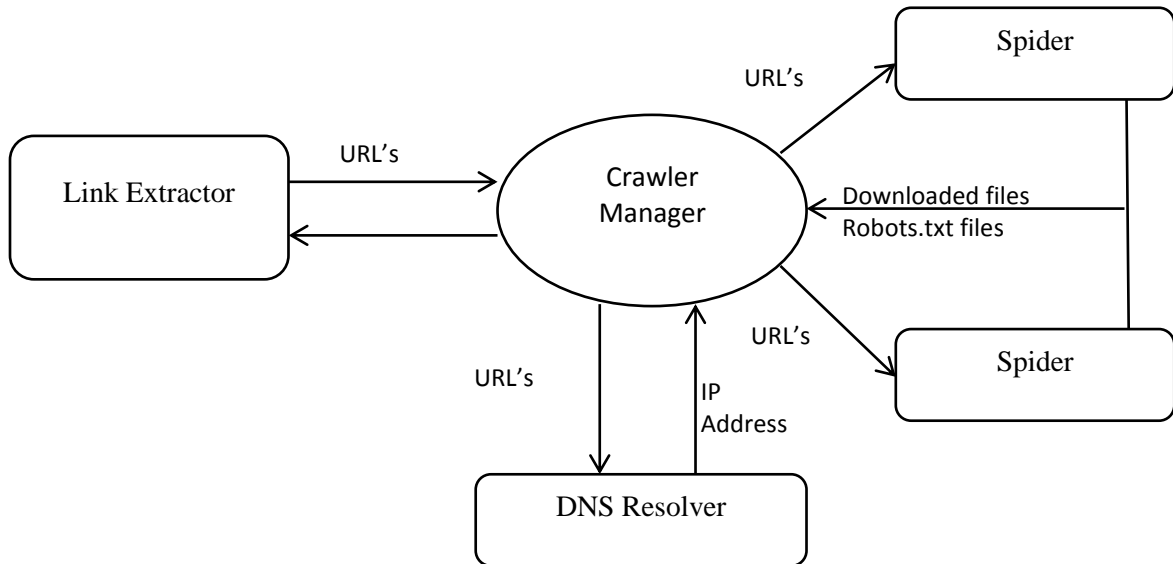


Figure 7. Web crawler architecture.

Crawlers can also be used for automating maintenance tasks on a Website, such as checking links or validating HTML code. Also, crawlers can be used to gather specific types of information from Web pages, such as harvesting e-mail addresses (usually for sending spam) (Shah, 2003). Everything the spider finds goes into the second part of the search engine, the index. The index, sometimes called the catalogue, is like a giant book containing a copy of every Web page that the spider finds. If a Web page changes, then this book is updated with new information.

Search engine software is the third part of a search engine. This is the program that shifts through the millions of pages recorded in the index to find matches to a search and then ranks them in order of what it believes is most relevant. A search engine cannot

work without a proper index where possible searched pages are stored, usually in a compressed format. This index is created by specialized robots, which crawl the Web for new/modified pages (the actual crawlers, or spiders).

Snippet

Cutts (2007), a Google expert, has explained ‘Snippet’ as the components of a search result in the search engine that shows more details about the result. These components may include title, URL, description, history, notes, related site links, similar pages, reviews, address, maps, and more results (see Figure 8).

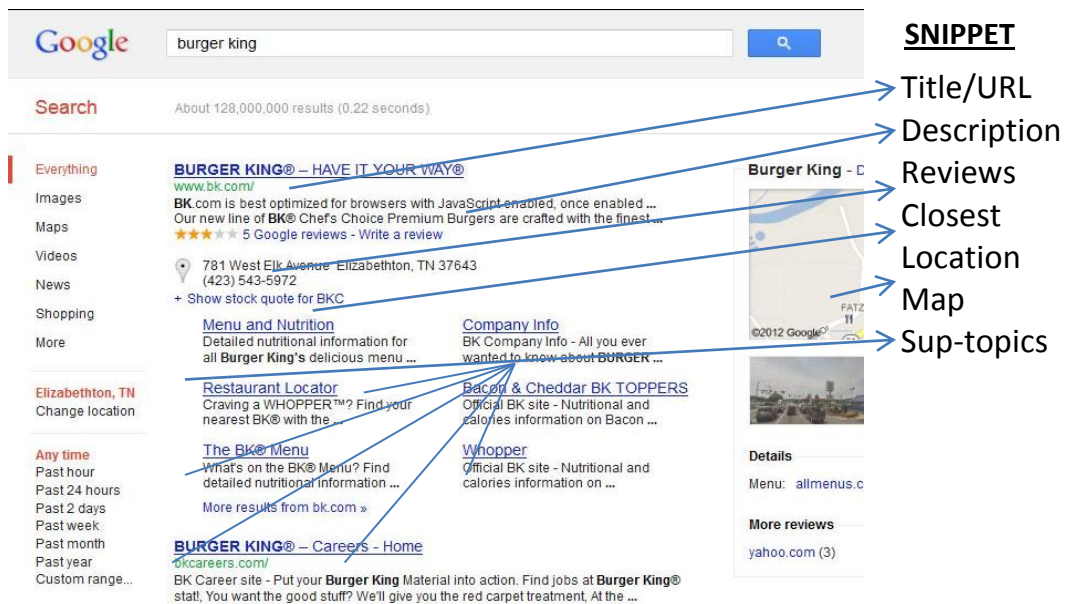


Figure 8. Google search result for “Burger King” that shows snippets.

Multiple Search Engine Design

The most well-known general search engines are Google, Bing, Yahoo, and Ask, but one of the oldest search engines is AltaVista. All existing search engines have

weaknesses. Even Google does not support full Boolean; it only indexes a part of the Web pages, document files, or PDF files, etc. (Latha & Rajaram, 2010). This part represents a real reason for building more search engines. A scalable distributed repository is used to store the crawled collection of Web pages. Strategies for physical organization of pages on the storage devices, distribution of pages across machines, and mechanisms to integrate freshly crawled pages are important issues in the design of this repository. The repository supports both random and stream-based access modes. Random access allows individual pages to be retrieved based on an internal page identifier. Stream-based access allows all or a significant subset of pages to be retrieved as a stream. Query-based access to the pages and the computed features (from the feature repository) is provided via the Web-based query engine (Ding et al., 2004).

Unlike the traditional keyword-based queries supported by existing search engines, queries to the Web-based query engine can involve predicates on both the content and link structure of the Web pages.

What Is the Semantic Web?

The Semantic Web is the representation of data on the World Wide Web. It is a collaborative effort led by W3C with participation from a large number of researchers and industrial partners. It is based on the Resource Description Framework (RDF), which integrates a variety of applications using XML for syntax and URIs for naming—W3C Semantic Web (Horrocks & Hendler, 2002).

W3schools.com (2012) explains the Semantic Web as follows:

The Semantic Web is a Web that is able to describe things in a way that computers can understand.

For example:

1. Toyota is a popular manufacturer of motor vehicles.

2. Avalon car is a product of Toyota.
3. Toyota is a Japanese car company.

Sentences like the ones above can be understood by people. But how can they be understood by computers? Statements are built with syntax rules. The syntax of a language defines the rules for building the language statements. But how can syntax become semantic? This is what the Semantic Web is all about: describing things in a way that computer applications can understand.

W3schools.com (2012) explains that the Semantic Web is also a web of data. There are lots of data we all use every day, which are not part of the Web. I can check my photographs on Facebook through the Web, see my bank statements, and see my appointments in a calendar. But can I see my photos in a calendar to see what I was doing when I took them? Can I see bank statement lines in a calendar? Why not? Because we don't have a Web of data, and because data are controlled by applications, and each application keeps it to itself. The Semantic Web is about two things: It is about common formats for the integration and combination of data drawn from diverse sources, where the original Web concentrated mainly on the interchange of documents. It is also about language for recording how the data relate to real-world objects. That allows a person, or a machine, to start off in one database, and then move through an unending set of databases that are connected not by wires but by being about the same thing.

Surface Web vs. Deep Web

One may ask: "How many pages are on the Internet?" It seems like an answerable question. But no one really knows how many Websites or individual Web pages make up this seemingly infinite digital universe that is the Internet. John Sutter reported that

“Kevin Kelly, a founder of Wired magazine, has written that there are at least a trillion Web pages in existence, which means the internet's collective brain has more neurons than our actual gray matter that's stuffed between our ears” (Sutter, 2011).

PC Magazine Encyclopedia defined Surface Web: “Content on the World Wide Web that is available to the general public and for indexing by a search engine. Also called the ‘crawlable Web’ and ‘public Web,’ links to the pages on the surface Web are displayed on search engine results pages” (PCmag.com, 2011). On the other hand, the Deep Web is usually defined as the content on the Web not accessible through a search on general search engines. This content is sometimes also referred to as the hidden or invisible Web (Internettutorials.net, 2011).

The Deep Web simply consists of information that’s accessible over the Web but that can’t be found through ordinary search tools such as Google and Yahoo! These search engines can’t find it for two main reasons: It’s stored within databases and can be retrieved only by using a particular site’s search tool, or it resides at sites that require registration or subscription. Deep Web mostly contains sources that are more likely to have been written, developed, or reviewed by experts as identified Resources intended for a specific academic community (Lib.odu.edu, 2010). A video presentation produced by the Office of Scientific and Technical Information (OSTI), U.S. Department of Energy, stated, “A Deep Web does not rely on the store indexes that were built in advance but operates in real time, replicating the query and broadcasting it to multiple databases” (Proz.tumblr.com, 2008). This means that the pre-indexed materials which contain outdated information will not affect the search result if information can be extracted from the Deep Web.

List-Handley (2008) in his book called *Information Literacy and Technology* states that “approximately 80% of the Information on the Web belongs to the ‘invisible Web” (p. 36). According to one popular online magazine called windowssecrets.com. “Total quality content of the deep Web is at least 1,000 to 2,000 times greater than that of the surface Web” (Langa, 2001). In a normal search, the Deep Web pages are beyond the reach of the regular Web crawler due to their dynamic intrinsicality. Langa also states that on an average, Deep Web sites receive about 50% greater monthly traffic than surface sites and are more highly linked to than surface sites. Deep Web sites are not well known to the general public searching the Internet.

Understanding the Meta-Search Engine

A Meta-Search Engine (MSE) means instead of getting results from one search engine, one will be getting the best combined results from a variety of engines as demonstrated in Figure 9, and not just any engines, but industry-leading engines such as Google, Yahoo! Search, and Bing, as well as authority sites Kosmix and Fandango (Dogpile, 2012).

Upon considering each existing search engine, a document or data that a MSE engine can access are known as a “component” of that MSE. From the general user’s point of view, meta-search may look or behave in a similar fashion like any other typical search engine when a search query is submitted on the search box by the user. A list of search-result records that are most relevant with the query will be displayed (Lu, 2011).

Lu (2011) goes on to explain that in the case of MSE, the approach is different as they use a different model. Upon user querying, the MSE forwards the query to the appropriate component search engines through their search interfaces. Only the contents

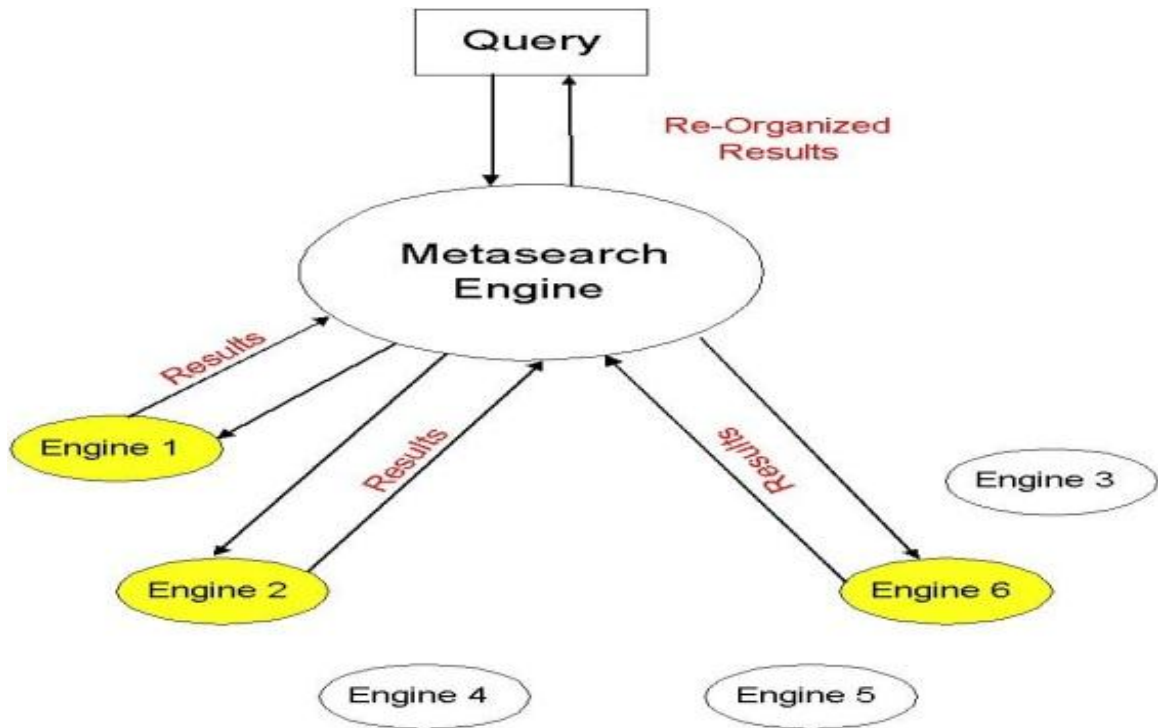


Figure 9. Meta-search engine function.

related to the query are retrieved from all the queried individual search engines, and the results are reorganized into a single ranked list with unified format and then returned back to the user. Since MSEs interact with search engines' query interface directly and there is no need to obtain the content of any search engine in advance, they can reach the Deep Web more naturally than general-purpose search engines. And for the same reason, a MSE can potentially interact with any kind of search engines, including surface Web search engines, Deep Web search engines, traditional general-purpose search engines, and even other MSEs.

If a meta-search system needs only to access a very limited number of search engines, it would be much less challenging since many issues can be resolved manually. However, if the goal is to build a MSE that can potentially connect to hundreds or

thousands of Deep Web sites, we need to develop intelligent techniques that can, to the largest extent, automate the building process (Lu, 2011, p. 5).

To build such highly efficient and large-scale meta-search systems is a complicated process that involves many research areas as illustrated in Figure 10.

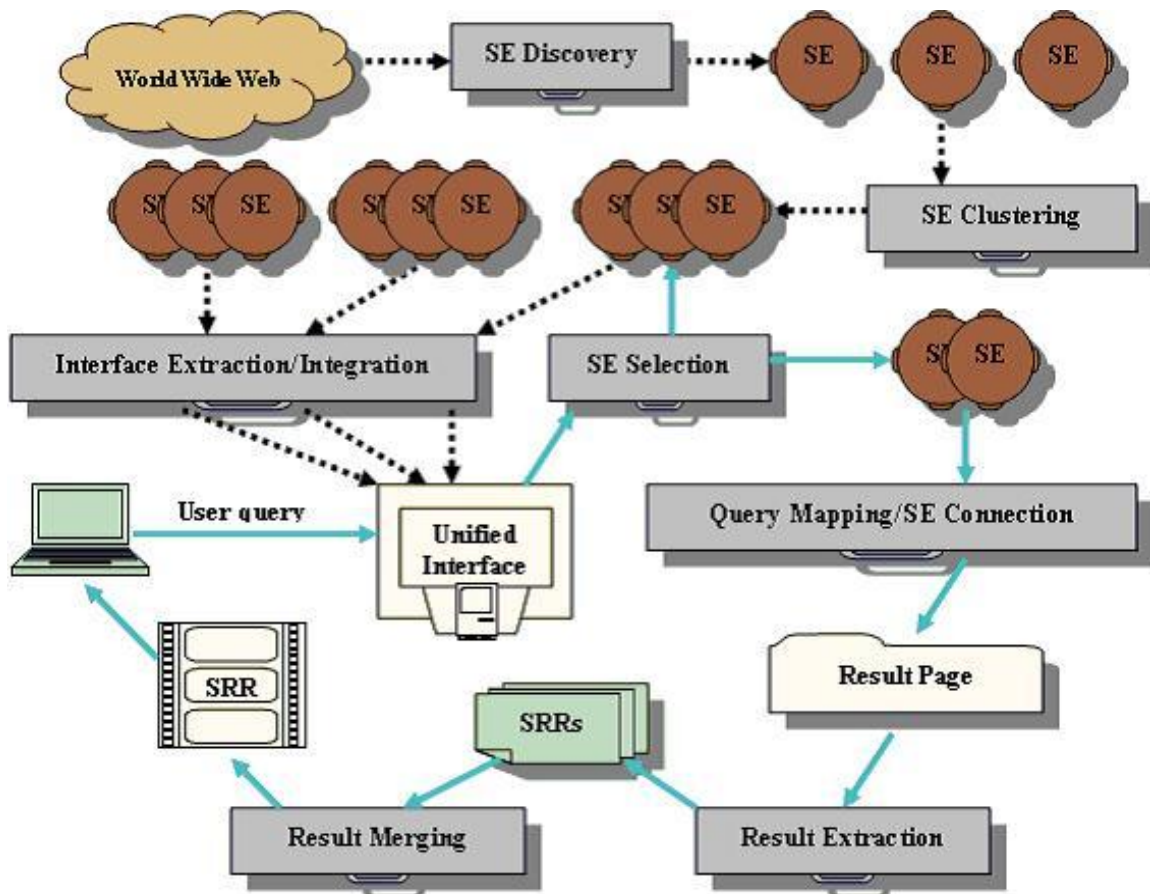


Figure 10. Architecture of meta-search engine. Note. From “Automated Search Interface Clustering and Search Result Processing in Metasearch Engine” (Unpublished doctoral dissertation), p. 5, by Y. Lu, State University of New York-Binghamton, 2011.

The advantage of MSE is that they are able to "talk" to other search engines, which contain relevant data. The language that they speak is HTML, or hypertext markup language, a set of electronic codes which enables computers to read, translate, transmit,

and store data accessible to the entire world. Every Web page is written in HTML using meta "tags," which are directives to client computers, describing the kind of document stored (Christiannet.com, 2012).

Christiannet.com (2012) states that advertisers employ some of the best meta-search engines to measure visitor traffic; assess how much time visitors take to linger on the site, including the number of pages viewed; count the number of CTRs or click-throughs; and, most importantly, assess revenue generated via listings and ads.

CHAPTER 4

SEARCH ENGINE RESULT GROUPING METHOD

For a document meta-search system where every component is a document search engine, the rank is normally determined by the estimated relevancy of the document content to the user query. The document most relevant to the query should have the highest universal rank. In order to discover an effective meta-search grouping algorithms, I will review and discuss various widespread research directions that are related to result merging and ranking strategies along with their algorithms used in them. I will examine their strength and weakness and then introduce a new algorithm by collecting only their positive character.

Result Grouping and Raking Method

The most important methods used for ranking and retrieving search engine results are:

1. Normalizing/uniform the scores of search results (Renda & Stracci, 2003)
2. Some ranking algorithms, which completely ignore the scores assigned by the search engines to the retrieved Web pages, such as bayes-fuse and borda-fuse (Renda & Stracci, 2003)
3. Considering the frequencies of query terms in each SRR, the order, and the closeness of these terms

4. Downloading and analyzing the full document
5. Reliability of each search engine.

Most search engines present more informative search result records (SRRs) of retrieved results to the user. A typical SRR consists of the URL, title, and snippet of the retrieved result (Dwork, Kumar, Naor, & Sivakumar, 2001).

The following are the most common result-merging algorithm strategies for MSEs that were considered for this investigation.

Method 1: Fetch Retrieved Documents

A direct way to perform result merging is to fetch the retrieved documents to the MSE and compute their similarities with the query using a global similarity function. The problem with this approach is that the user has to wait a certain time before the results can be available. Most result-merging techniques utilize the information associated with the search results as returned by component search engines to perform merging. The difficulty lies in the heterogeneities among the component search engines.

Method 2: Taking the Best Rank

This algorithm is based on the URL ranking where the URL will be placed at the best rank it gets in any of the search engine rankings (Dorn & Naz, 2008).

$$\mathit{MetaRank}(x) = \mathit{Min}(\mathit{Rank}1(x), \mathit{Rank}2(x), \dots, \mathit{Rank}n(x));$$

Where, clashes are avoided by search engine popularity.

Method 3: Weighted Borda-Fuse

This algorithm does not treat the search engine equally, but votes are weighed and considered depending on the reliability of each search engine. These weights are set by the users in their profiles.

Thus, the votes that the i result of the j search engine receive are (Dorn & Naz, 2008; Fagin et al., 2003)

$$V(ri, j) = w_j * (\max k(rk) - i + 1);$$

where w_j is the weight of the j search engine and rk is the number of results rendered by search engine k . Retrieved pages that appear in more than one search engines receive the sum of their votes.

Method 4: Borda's Positional Method

In this algorithm, MetaRank of an URL is obtained by computing the L1-Norm of the ranks in different search engines (Fagin et al., 2003)

$$MetaRank(x) = \frac{1}{p} (Rank1(x)^p + Rank2(x)^p + \dots + Rankn(x)^p)^{1/p};$$

Clashes are avoided by search engine popularity.

Method 5: The Original KE (Key Extraction) Algorithm

This algorithm (*Original KE*) is a score-based method (Renda & Straccia, 2003). It exploits the ranking that a result receives by the component engines and the number of its appearances in the component engines' lists. All component engines are considered to be reliable and are treated equally. Each returned ranked item is scored based on the following formula (Souldatos et al., 2006)

$$W_{ke} = \frac{\sum_{i=1}^m r(i)}{n^m \left(\frac{k}{10} + 1 \right)^n}$$

where $\sum_{i=1}^m r(i)$ is the sum of all rankings that the item has taken, n is the number of search engine top- k lists the item is listed in, m is the total number of search engines exploited and K is the total number of ranked items that the *KE Algorithm* uses from each search engine. Therefore, the less weight a result scores, the better ranking it receives.

Method 6: Borda Count

This is a voting-based data fusion method (Akritidis et al., 2008). Each returned result is considered as a candidate and each component search engine as a voter. For each voter, the top ranked candidate is assigned n points, the second top ranked candidate is given $n-1$ points, and so on. For candidates that are not ranked by a voter (i.e., they are not retrieved by the corresponding search engine), the remaining points of the voters are then divided evenly among them. The candidates are then ranked on their received total points in descending order (Akritidis et al., 2008; Aslam & Montague, 2001).

Method 7: D-WISE Method

In D-WISE, the local rank of a document (r_i) returned from search engine j is converted to a ranking score (rs_{ij}); the formula is (Lu et al., 2005)

$$rs_{ij} = 1 - (r_i - 1) * S_{\min} / (m * S_j)$$

where S_j is the usefulness score of the search engine j , S_{\min} is the smallest search engine score among the search engines selected for this query and m is the number of documents

desired across all search engines. This function generates a smaller difference between the ranking scores of two consecutively ranked results from a search engine with a higher search engine score. This has the effect of ranking more results from higher quality search engines higher. One problem of this method is that the highest ranked documents returned from all the local systems will have the same ranking score of 1.

Method 8: Use Top Document to Compute Search Engine Score (TopD)

Assuming S_j denote the score of search engine j with respect to q , this algorithm uses the similarity between q and the top ranked document from search engine j (denoted d_{ij}) (Dwork et al., 2001; Lu et al., 2005. Fetching the top ranked document from its local server has some delay, although more tolerable, as only one document is fetched from each used search engine. The similarity functions using the Cosine function and Okapi function. The formula is (Lu et al., 2005):

$$\sum_{T \in Q} W * \frac{(k_1 + 1) * tf}{K + tf} * \frac{(k_3 + 1) * qtf}{k_3 + qtf},$$

with $w = \log \frac{N - n + 0.5}{n + 0.5}$ and $K = k_1 * ((1 - b) + b * \frac{dl}{avgdl})$, where tf is the frequency of

the query term T within the processed document, qtf is the frequency of T within the query, N is the number of documents in the collection, n is the number of documents containing T , dl is the length of the document, and $avgdl$ is the average length of all the documents in the collection. k_1 , k_3 and b are the constants with values 1.2, 1,000, and 0.75, respectively (Lu et al., 2005). N , n , and $avgdl$ are unknown; some approximations

are used to estimate them. The ranking scores of the top ranked results from all used search engines will be 1 (Lu et al., 2005; Renda & Straccia, 2003). We remedy this problem by computing an adjusted ranking score ars_{ij} by multiplying the ranking score computed by the formula above, namely rs_{ij} , by s_j (Lu et al., 2005), $ars_{ij} = (rs_{ij} * s_j)$. If a document is retrieved from multiple search engines, we compute its final ranking score by summing up all the adjusted ranking scores.

Method 9: Merging Based on Combination Documents Records (SRRs)

Among the proposed merging methods, the most effective one is based on the combination of the evidences of document such as title, snippet, and the search engine usefulness. For each document, the similarity between the query, its title, and its snippet is aggregated linearly as this document's estimated global similarity (Renda & Straccia, 2003). For each query term, I will compute its weight in every component search engine based on the Okapi probabilistic model (Lu et al., 2005). The search engine score is the sum of all the query term weights of this search engine. Finally, the estimated global similarity of each result is adjusted by multiplying the relative deviation of its source search engine's score to the mean of all the search engine scores. It is very possible that for a given query, the same document is returned from multiple component search engines. In this case, their (normalized) ranking scores need to be combined (Renda & Straccia, 2003). A number of linear combination fusion functions have been proposed to solve this problem including min, max, sum, average, etc. (Akritidis et al., 2008).

Method 10: Use Top Search Result Records (SRRs) to Compute Search Engine Score (TopSRR)

In the TopSSR method, when a query q is submitted to a search engine j , the search engine returns the SRRs of a certain number of top ranked documents on a dynamically generated result page. In this algorithm, the SRRs of the top n returned results from each search engine are used to estimate its search engine score instead of the top ranked document (Lu et al., 2005). This could be a reasonable tool for a more useful search engine, for a given query is more likely to retrieve better results, which are usually reflected in the SRRs of these results. All the titles of the top n SRRs from search engine j are merged together to form a title vector TV_j and all the snippets are also merged into a snippet vector SV_j . The similarities between query q and TV_j , and between q and SV_j , are computed separately and then aggregated into the score of search engine j (Lu et al., 2005).

$$S_j = c_1 * Similarity(Q, TV_j) + (1 - c_1) * Similarity(Q, SV_j) ,$$

Here, both the Cosine function and Okapi function are used (Dwork et al., 2001).

Method 11: Compute Simple Similarities Between SRRs and Query (SRRsim)

In this method, SRRs returned from different search engines are ranked, as each SRR can be considered as the representative of the corresponding full document. In the SRRsim algorithm, the similarity between a SRR (R) and a query q is defined as a weighted sum of the similarity between the title (T) of R and q , and the similarity between the snippet (S) of R and q (Dwork et al., 2001; Lu et al., 2005).

$$sim(R, Q) = c_2 * Similarity(Q, T) + (1 - c_2) * Similarity(Q, S) ,$$

Where, C_2 is constant ($C_2 = 0.5$).

Again both the Cosine function and the Okapi function are used. When a document is retrieved from multiple search engines with different SRRs (different search engines employ different ways to generate their SRRs), then the similarity between the query and each of the SRRs will be computed and the largest one will be used as the final similarity for merging.

Method 12: Rank SRRs Using More Features (SRRRank)

The similarity function used in the SRRsim algorithm may not be sufficiently powerful in reflecting the true matches of the SRRs with respect to a given query (Lu et al., 2005). For example, these functions do not take proximity information, such as how close the query terms occur in the title and snippet of a SRR, into consideration, nor does it consider the order of appearances of the query terms in the title and snippet. Sometimes, the order and proximity information have a significant impact on the match of phrases. This algorithm defines five features with respect to the query terms, which are (Dwork et al., 2001; Lu et al., 2005):

1. NDT: The number of distinct query terms appearing in title and snippet
2. TNT: Total number occurrences of the query terms in the title and snippet
3. TLoc: The locations of the occurred query terms
4. ADJ: Whether the occurred query terms appear in the same order as they are in the query and whether they occur adjacently

5. WS: The window size containing distinct occurred query terms. For each SRR of the returned result, the above pieces of information are collected. The SRRRank algorithm works as:

a. All SRRs are grouped based on NDT. The groups having more distinct terms are ranked higher.

b. Within each group, the SRRs are further put into three subgroups based on TLoc. All in the snippet, and scattered in both title and snippet. This feature describes the distribution of the query terms in the SRR. In real applications, the title is more frequently associated with a returned result than the snippet (some search engines provide titles only). Therefore, title is usually given higher priority than the snippet (Lu et al., 2005):

6. Finally, within each subgroup, the SRRs that have more occurrences of query terms (TNT) appearing in the title and the snippet are ranked higher. If two SRRs have the same number of occurrences of query terms, first the one with distinct query terms appearing in the same order, and adjacently (ADJ) as they are in the query, is ranked higher, and then, the one with smaller window size is ranked higher.

If there is a tie, it is broken by the local ranks. The result with the higher local rank will have a higher global rank in the merged list. If a result is retrieved from multiple search engines, we keep only the one with the highest global rank (Fagin et al., 2004).

**Method 13: Compute Similarities
Between SRRs and Query Using More
Features (SRRSimMF)**

This algorithm is similar to SRRRank except that it quantifies the matches based on each feature identified in SRRRank so that the matching scores based on different features can be aggregated into a numeric value (Renda & Straccia, 2003). Consider a given field of a SRR, say title (the same methods apply to snippet).

For the number of distinct query terms (NDT), its matching score is the ratio of NDT over the total number of distinct terms in the query (QLEN), denoted $S_{NDT} = \text{NDT}/\text{QLEN}$. For the total number of query terms (TNT), its matching score is the ratio of TNT over the length of title, denoted $S_{TNT} = \text{TNT}/\text{TITLEN}$. For the query terms order and adjacency information (ADJ), the matching score S_{ADJ} is set to 1 if the distinct query terms appear in the same order and adjacently in the title; otherwise the value is 0. The window size (WS) of the distinct query terms in the processed title is converted into score $S_{WS} = (\text{TITLEN} - \text{WS}) / \text{TITLEN}$. All the matching scores of these features are aggregated into a single value, which is the similarity between the processed title T and q, using this formula (Lu et al., 2005),

$$Sim(T, Q) = S_{NDT} + \frac{1}{QLEN} * (W_1 * S_{ADJ} + W_2 * S_{WS} + W_3 * S_{TNT})$$

For each SRR, the final similarity is,

$$Similarity = \frac{TNDT}{QLEN} * (c_3 * Sim(T, Q) + (1 - c_3) * Sim(S, Q))$$

Where, TNDT is the total number of distinct query terms that appeared in title and snippet (Dwork et al., 2001; Lu et al., 2005).

CHAPTER 5

META-META-SEARCH ENGINE (MMSE)

The new search engine that I am proposing will be called a Meta-Meta-Search Engine. Like many meta-search engines, this search engine will run on top of five selected meta-search engines and will act mainly as an interface to these meta-search engines. It does not maintain its own index on documents. When a MMSE receives a user's query, it first passes the query (with necessary reformatting) to the selected MSE's; the local MSE will then forward the requests to its local search engines and then to several other heterogeneous databases.

The search results that are collected from the local search engines will be compiled by MSE in a homogeneous manner based on a specific algorithm and aggregate the results into a single list or display them according to their source. The MMSE will then collect the rank results from its local MSE's, re-rank those rank results into one final rank according to its specific algorithms, and display the result to the user. This way the MMSE user's task will be drastically simplified, as one can see in Figure 11.

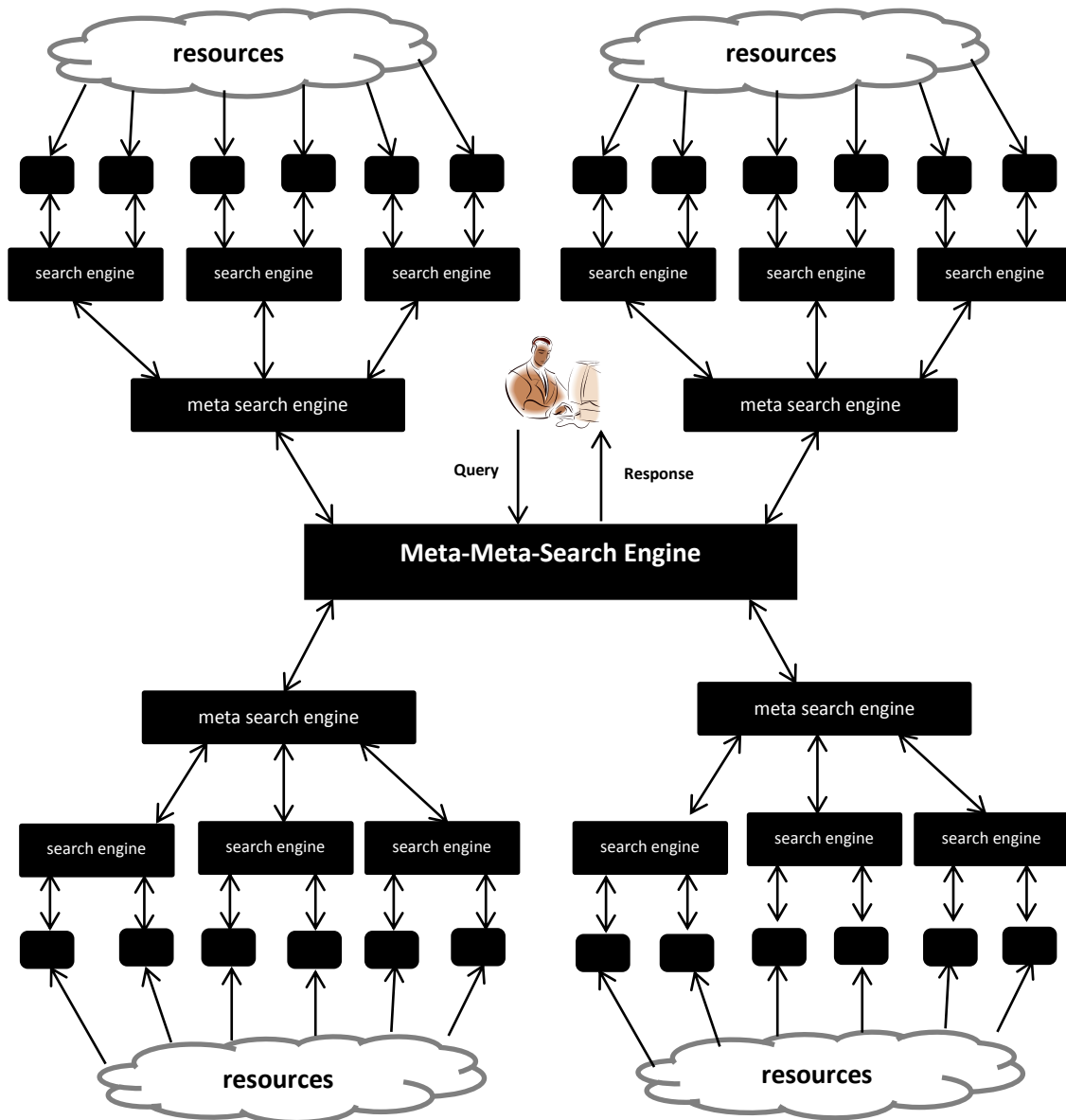


Figure 11. MMSE architecture.

MSE Selection Method for MMSE

MSEs that are to be used for my MMSE are selected based on multiple criteria. No two MSE's are alike. Some search only from the most popular search engines such as Google, Yahoo, Bing etc., while others also search lesser-known engines like Bebo, Winzy, Qkport and newsgroups like CNN, NBC, etc., and other databases. They also

differ in how the results are presented and the quantity of engines that are used. Some will list results according to the search engine or database. Others return results and display according to relevance (which are based on number of frequencies that appear in the document or are relevant between the query and the document title and snippet), often concealing which search engine returned which results. Therefore, it is very helpful to select a different variety of MSEs because they will collect information from multiple sources, which is going to improve the quality of the search result as well as the information quality and present what other MSEs may leave out.

Overview of the Existing MSE

There are dozens of useful MSEs that are available today. InfoSpace is the industry gorilla, operating the four arguably best known and most heavily used properties. Dogpile and Metacrawler are the two best known InfoSpace meta-search engines. Less well known is that InfoSpace purchased Excite.

Table 2 shows some of the well-known MSEs and their basic features.

MSE Selection Criteria

The following are some criteria that are used to select the MSE for my MMSE.

1. Good public user rating and among the recent top best award winners for MMSE.
2. Good features overall with strong features like Boolean operation, ability to refine a search, support multiple languages, and collect sources from reputable SEs.
3. Variety of source collection methods. Some MSEs concentrate only on the most common SE like Google, Bing, and Yahoo while leaving out some other possible important SEs.

Table 2

Common MSE and Their Features

MSE	Search Engines	Main Feature	More Options	Advanced Search Abilities	Relevance	Other Search Types
Infospace	Google, Yahoo!, Bing, Ask	Term Cluster	Multiple language support	Boolean Operators (OR, +, -, AND)	High, good number of sponsored links	News, image, video, yellow pages, white pages
Clusty (yippy)	Ask, Gigablast, Live, NY Times, ODP, Shopzilla, Yahoo News, Yahoo Stocks	Term cluster	TLD stats	Boolean Operator (OR, -, +), filetype, and Language Restricted Search	Low, too many “Sponsored Results”	News, images, logs, shops
Dogpile	Google, Yahoo, Ask, Live	Search suggestions (related terms); resent searches		Boolean Operator (“”, -)	High	Images, audio, video, news, Yellow Pages, White Pages
IxQuick	All the Web, Exalead, Qkport, Ask, Gigablast, Wikipedia, Bebo, Bing, Winzy	“Star” system (the more stars = the more search engines agree on the listing rankings).	Multiple language support	Boolean Operator (OR, -, +)	High	Video, Images
IBoogie	AllTheWeb, MSN	Term cluster	“Add your source” option		Low	Images, news
Kartoo	N/A	Search results on a map	Term clustering		Low	Images, Video
SurfWax	CNN, Yahoo, News, HotBot, ODP, Yahoo news, MSN, AllTheWeb	Snapping= displaying the summary of the page containing the search query		No	Moderate	
Mama	Ask.com, About.com, EntireWeb, Business.com, Gigablast, Wisenut, ODP	Add/Exclude any of the search engines	“Refine your query” search suggestions	Boolean operators (“”, -, +)	Moderate	Video, Yellow, Pages, White Pages
Search	Google, Ask.com, MSN		“Narrow/Expand your search”	Boolean operators (OR, -, +, “”), filetype, language, update time, linkdomain restricted search	Moderate	Images, video, people, shopping, music, news, games
Meta Crawler	Google, Yahoo, MSN, Ask		Preferences: Bold search on/off, recent searches on/off	Boolean operators (OR, -, +, “”) “sponsored results”	Moderate, too many “sponsored results”	Images, video, news, yellow pages, white pages
Fuzz Find	Google, Yahoo and MSN and from Del.icio.us	Sort the results based on any of the resources		No	High	No
Infogrid	Google, Yaa, MSN, AOL, Excite, Cnet, Info World	Graph Database, SQL	Object-oriented information, REST-ful	Boolean Operator (+)	Moderate	Image new,

Note. From *Metacrawlers and Metasearch Engines*, by S. Chris, 2005, retrieved from <http://searchenginewatch.com/article/2066974/Metacrawlers-and-Metasearch-Engines>. Infogrid, 2012; Listofsearchengines, 2012; Searchenginewatch.com, 2007.

4. Avoid selecting MSEs from only one or two SE owners: For example, a single company called Infospace owned multiple MSEs like infospace.com, dogpile.com, metacrawler.com, and Webcrawler.com etc. Apparently, the architecture, algorithm, principle and the search engine source that are used in these MSE's are more or less the same. Although, the MSEs mentioned may have a good public ratings, I picked only one or two for my five MSEs in order to have better quality as well as wider variety of search results.

Selected MSE

The following five MSEs are choosen for my new MMSE:

1. Dogpile.com: Dogpile.com is a top aggregator of the most relevant searches from Google, Bing, Yahoo! and Ask. It delivers them conveniently on a single Webpage. Dogpile is owned by InfoSpace (www.listofsearchengines.info, 2012). Dogpile has also received the best search engine award winner by Search Engine Watch for 2003. They also updated their features from time to time to enhance search results (Chris, 2005).

2. InfoSpace.com: Infospace is awarded in 2012 as the best MSE. It was founded by Naveen Jain in early 1996. InfoSpace currently operates one of the Internet's most popular meta-search engines, receiving 4.6 million unique monthly visitors from the U.S. alone according to *Alexa Traffic Rank* (global), 2012. One can perform queries using a specific keyword to get results from Google, Yahoo!, Bing, and Ask (www.listofsearchengines.info, 2012).

3. Yippy.com: Rich (2012), an investor relation of yippy.com, announced in June 12, 2012, that Yippy is an award-winning deep research engine developed out of Carnegie Mellon University. The programs were acquired by Yippy in May 2010 from

Vivisimo, Inc., an industry-leading Enterprise Search company that was recently acquired (May 2012) by International Business Machines (IBM).

When a user's query is entered for a search using Yippy crawl, some special collections, like the AP, the *New York Times*, and Wikipedia, do not maintain an index of the entire Web. Rather, they send the user's query to multiple search engines, gather their results, and present them to the user in a single, intuitive interface, making Yippy a much more powerful research tool than some other search engines.

4. Ixquick.com: Billed as the "World's Most Private Meta-Search Engine," IxQuick lets users search anonymously for images, phone numbers, Websites, and videos. This meta-search engine has a simple and easy-to-use interface, and it returns relevant search results from multiple search engine sources such as all the Web, Exalead, Qkport, Ask, Gigablast, Wikipedia, Bebo, Bing, and Winzy (www.museglobal.com). Ixquick was awarded the first European Privacy Seal (EuroPriSe) for its privacy practices on July 14, 2008. This European Union-sponsored initiative guarantees compliance with EU laws and regulations on data security and privacy through a series of design and technical audits (Kiel, 2008).

5. Mamma.com: Mamma collects information through Ask.com, About.com, Entireweb, Business.com, Gigablast, Wisenut, ODP. Created in 1996 as a master's thesis, Mamma.com helped to introduce meta-search to the Internet as one of the first of its kind. Due to its quality results and the benefits of meta-search, Mamma grew rapidly through word of mouth, and quickly became an established search engine on the Internet. Mamma.com's ability to gather the best search results available from top search sources and to provide useful tools to its users has resulted in its receiving multiple Honorable

Mentions in the Best Metasearch category in the annual SearchEngineWatch Awards (Killerstartups.com, 2011).

Proposed Algorithms for MMSE

Algorithms that are used in MSEs have been extensively studied including the one listed in the previous chapter. Although there are algorithms that are useful for finding the similarities between queries and search results, they might not be very useful in my proposed ranking system. I will not be worrying about whether my search query matches the title and snippet of the document; I would rather focus on the URL ranking where the URL will be placed at the best rank. This is because most of the top ranking strategies are already being applied in the existing MSEs. However, I have carefully selected the MSE that will be used in MMSE.

Refined Borda Count

In a traditional Borda Count system, the returned results are considered as the candidates, and each component search engine (MSE in our case) is a voter. For each voter, the top ranked candidate is assigned n points (n candidates), the second top ranked candidate is given $n-1$ points, and so on. For candidates that are not ranked by a voter (i.e., they are not included among the top 10 in the rank list), the remaining points of the voter will be divided evenly among them. The candidates are then ranked based on their received total points in descending order.

In order to fit Borda count in my new MMSE, I have modified the last section of this algorithm by setting '0' point to the candidates that are not ranked by a voter instead of dividing the remaining points evenly. The reason for setting to '0' is that I stressed the importance more on the first 10 candidates from each MSE than the rest of the un-voted

rank. When dividing the remaining points among the candidates that are not ranked between 1st and 10th, the un-voted candidate will receive more points, but by assigning '0' points to these non-voted candidates, I nullify them. Thus, this has an effect on the final rank.

This algorithm is explained in the following pseudocode (see Figure 12). I used C# to implement this pseudocode.

```

/** Pseudocode for refined borda-count ranking system */

int NUM_LIST = 5; //number of voters
int NUM_ELEMENT_IN_LIST = 10; //number of vote candidates per voter

String rankLists[NUM_LIST][NUM_ELEMENT_IN_LIST]; //allocate the lists of voting
//for voters which store candidates' names
Read in the candidate names voted by each voter into rankLists.
int MAX_POINTS = 0; //this is the max points that a voter can give
for point = 1 to number of candidates //it is 1 + 2 + ... + num candidates
    MAX_POINTS = MAX_POINTS + point; //now, use Refined Borda method to
    //calculate points for each candidate
for each list in rankLists //for each voting list
    {
int totalPointGiven = 0; //total points given by voters for
    //NUM_ELEMENT_IN_LIST candidates
    for each voted candidate in current list //for each voted candidate
    {
        Mark this candidate as processed.
        Calculate his voted point based on his current position in current voting list.
This calculation can be applied with list weight, too.
        Accumulate his points into totalPointGiven
    }

    //for non voted candidates
int pointLeft = MAX_POINTS - totalPointGiven; //out voting point left of this
    //voter

//we can divide the points left evenly among the rest non-voted candidates or simply set to 0
//here
int avgPoint = 0;
for each non-voted candidate
    {
        Set his total points to avgPoint
    }
    //end for each list in rankList

Now, sort all candidates based on total points in descending order.
The candidate order is the final rank order and his total points are now known

```

Figure 12. Pseudocode for refined Borda-count.

CHAPTER 6

IMPLEMENTATION

As part of this thesis I have tested the new MMSE algorithm using a set of five sample ranks in C#. The five sample ranks are the actual URL rank results that are displayed when running a sample query on the five selected MSEs, namely Dogpile.com, Infospace.com, Yippy.com, Ixquick.com, and Mamma.com. The same search query is used for all five MSEs to get the five sets of ranks (Rank1, Rank2, Rank3, Rank4, and Rank5). These five sets of ranks are then re-ranked using my re-ranking algorithm to develop one final rank. This final rank is then displayed to the user.

If you are running a query ‘piracy’ from the MMSE called M2search.com as shown in Figure 13, the MMSE will store the returned rank result in Rank1, Rank2, Rank3, Rank4, and Rank5 up to a list of 10 from the local MSE, namely Dogpile.com, Infospace.com, Yippy.com, Ixquick.com, and Mamma.com, respectively. The implementation of the re-ranking algorithm will be demonstrated by manually entering the URL rank results (e.g., D1, D2, D3,...D18) received from each selected MSE into the program as shown in Figure 14. New and final single rank will then be generated. See Figures 13 and 14. This information is also restated in Table 3.

I applied the new ranking algorithm to determine the final rank where all rank lists have the same weight in total and are equally important. What counts will be the position they give to candidates. When a query ‘piracy’ is run, each MSE will return its rank list in ascending order according to their relevancy with the query. Considering the above scenario, 18 different candidates which are D1, D2, D3, D4,... D18 exist. The

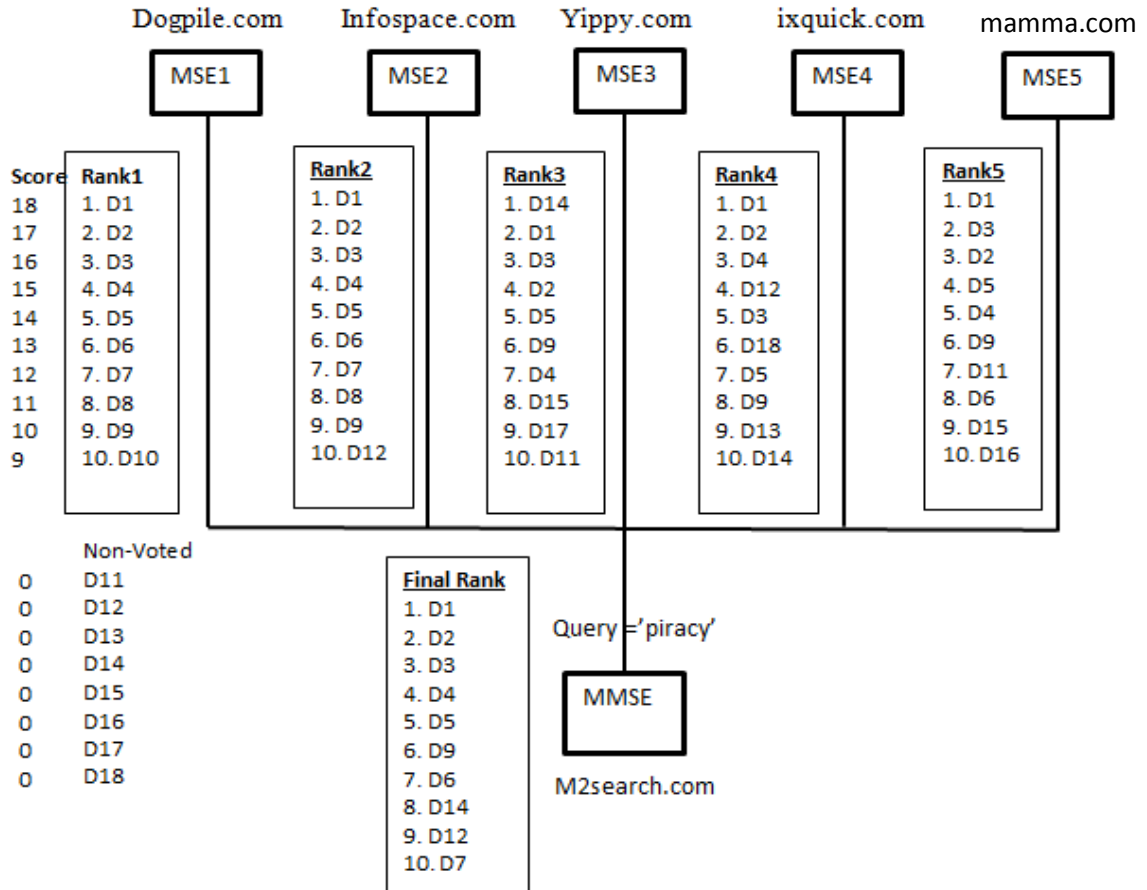


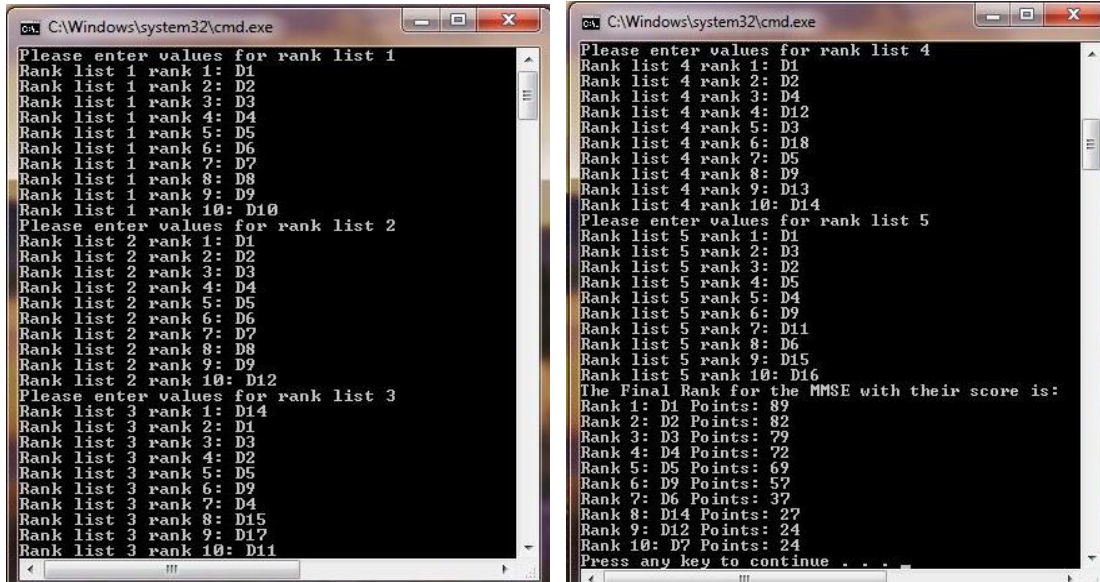
Figure 13. MMSE ranking system where query "piracy" is searched. MME = Meta Search Engine; MMSE= Meta-meta search engine; D=website URL

Where,

- D1= en.wikipedia.org/wiki/Piracy
- D2= en.wikipedia.org/wiki/Copyright_infringement
- D3= dictionary.reference.com/browse/piracy
- D4= www.microsoft.com/piracy
- D5= legal-dictionary.thefreedictionary.com/Piracy
- D6= thepiratebay.se/
- D7= dictionary.reference.com/law/piracy
- D8= www.riaa.com/physicalpiracy.php?content_selector=what-is-online-piracy
- D9= www.newworldencyclopedia.org/entry/Piracy
- D10= www.merriam-webster.com/dictionary/piracy
- D11= planetpiracy.com
- D12= www.noonsite.com/General/Piracy
- D13= www.wired.com/threatlevel/2012/06/uncle-sam-piracy
- D15= www.economist.com/node/18061574?story_id=18061574
- D16= www.siaa.net/index.php?option=com_content&view=article&id=77&Itemid=7
- D17= www.thefreedictionary.com/piracy
- D18= www.chron.com/news/ article/AP-Interview-Wozniak-Dotcom-slam-US-piracy-case-3665669.php

Rank1–Rank3 Lists

Rank4–Rank5 List & Final Rank List with Scores



```
C:\Windows\system32\cmd.exe
Please enter values for rank list 1
Rank list 1 rank 1: D1
Rank list 1 rank 2: D2
Rank list 1 rank 3: D3
Rank list 1 rank 4: D4
Rank list 1 rank 5: D5
Rank list 1 rank 6: D6
Rank list 1 rank 7: D7
Rank list 1 rank 8: D8
Rank list 1 rank 9: D9
Rank list 1 rank 10: D10
Please enter values for rank list 2
Rank list 2 rank 1: D1
Rank list 2 rank 2: D2
Rank list 2 rank 3: D3
Rank list 2 rank 4: D4
Rank list 2 rank 5: D5
Rank list 2 rank 6: D6
Rank list 2 rank 7: D7
Rank list 2 rank 8: D8
Rank list 2 rank 9: D9
Rank list 2 rank 10: D12
Please enter values for rank list 3
Rank list 3 rank 1: D14
Rank list 3 rank 2: D1
Rank list 3 rank 3: D3
Rank list 3 rank 4: D2
Rank list 3 rank 5: D5
Rank list 3 rank 6: D9
Rank list 3 rank 7: D4
Rank list 3 rank 8: D15
Rank list 3 rank 9: D17
Rank list 3 rank 10: D11

C:\Windows\system32\cmd.exe
Please enter values for rank list 4
Rank list 4 rank 1: D1
Rank list 4 rank 2: D2
Rank list 4 rank 3: D4
Rank list 4 rank 4: D12
Rank list 4 rank 5: D3
Rank list 4 rank 6: D18
Rank list 4 rank 7: D5
Rank list 4 rank 8: D9
Rank list 4 rank 9: D13
Rank list 4 rank 10: D14
Please enter values for rank list 5
Rank list 5 rank 1: D1
Rank list 5 rank 2: D3
Rank list 5 rank 3: D2
Rank list 5 rank 4: D5
Rank list 5 rank 5: D4
Rank list 5 rank 6: D9
Rank list 5 rank 7: D11
Rank list 5 rank 8: D6
Rank list 5 rank 9: D15
Rank list 5 rank 10: D16
The Final Rank for the MMSE with their score is:
Rank 1: D1 Points: 89
Rank 2: D2 Points: 82
Rank 3: D3 Points: 79
Rank 4: D4 Points: 72
Rank 5: D5 Points: 69
Rank 6: D9 Points: 57
Rank 7: D6 Points: 37
Rank 8: D14 Points: 27
Rank 9: D12 Points: 24
Rank 10: D7 Points: 24
Press any key to continue . . .
```

Figure 14. The input and output program screen for MMSE ranking algorithm.

assigned scores for each rank where score 18 is the highest for all the rank list, 17 is the second highest score, and so on. For instance, take list 1 where the first position is D1. In list 1, D1 has the highest point which is 18, which also means that it is the most important candidate in list 1. Next row is D2 and has 17 points and has the second importance in the list. The same principle will be applied to the remaining list until the 10th row/rank. Score '0' will be assigned to all non-voted rank. For instance, the candidate D10 has 9 points on Rank1, the remaining candidates D11, D12, D13... D8 will have same score, i.e., '0'. Now, the algorithm will loop through all the lists for each list and then calculate points for all the candidates. The rule is simple. The higher rank a candidate has, the higher points he gets. The scores for all the lists will be sorted in descending order and then displayed in the final rank list as seen in Figure 14.

Table 3

Final Rank Calculation for MMSE

RANK (Final Rank)	Rank1	Rank2	Rank3	Rank4	Rank5	Final Rank RESULT	Scores
1 st	D1	D1	D14	D1	D1	D1	89
2 nd	D2	D2	D1	D2	D3	D2	82
3 rd	D3	D3	D3	D4	D2	D3	79
4 th	D4	D4	D2	D12	D5	D4	72
5 th	D5	D5	D5	D3	D4	D5	69
6 th	D6	D6	D9	D18	D9	D9	57
7 th	D7	D7	D4	D5	D11	D6	45
8 th	D8	D8	D15	D9	D6	D14	39
9 th	D9	D9	D17	D13	D15	D12	24
10 th	D10	D12	D11	D14	D16	D7	24
11 th						D8	34
12 th						D15	21
13 th						D11	21
14 th						D18	29
15 th						D17	10
16 th						D13	10
17 th						D16	9
18 th						D10	9

One thing observed in the final rank list (Figure 14) as well as in the final rank calculation table is that there are candidates that have the same score but received a different rank. In reality, they are supposed to have the same position and the same rank. However, we cannot apply this system in the real Web page, so we can leave them as it is and do not worry about it if the system picks one URL higher than the other. A possible solution can be to place the actual rank number on each line in the corner. This way a user can identify the real URL rank position. In Figure 14 the final rank list displays only up to the 10th rank. To decide how many ranks we should display for the final rank listings can be decided in the future.

Effectiveness of MMSE

In order to prove that my new MMSE system is more effective than any other MSE individually, I used a statistical method called correlation analysis and hypothesis testing. In this case, I am comparing ranks from two data sets. As such, the method of Spearman's Correlation analysis has been used. As per this method, the correlation coefficient is given by:

$$r = 1 - \frac{6 \sum_i^2}{n(n^2 - 1)}$$

Where,

r = Rank Correlation Coefficient

\sum_i^2 = Sum of squared deviations between the sample ranks

n = No. of observations

In this case, the number of observations is 18. Since the sample size is less than 30, the tests have been used for the purposes of hypothesis testing. First, the rank correlation coefficients were computed using the above formula between the following sample pairs:

Final ranks and the rank assigned by MSE1

Final ranks and the rank assigned by MSE2

Final ranks and the rank assigned by MSE3

Final ranks and the rank assigned by MSE4

Final ranks and the rank assigned by MSE5

The values of the correlation coefficient were obtained as 0.6615, 0.8473, 0.4448, 0.6821, and 0.5046 respectively (see Table 4).

The next step is to test the hypothesis that the correlation coefficients are significant. To do this, I framed the hypotheses as:

H₀: Correlation coefficient is not significant and is equal to 0.

H_a: Correlation coefficient is significant and is different from zero.

This means that there is a statistically significant degree of relationship between the two sets of observations. To do this, the *t* statistic is computed by the formula (Wikipedia, 2011)

$$t = r \sqrt{\frac{n-2}{1-r^2}}$$

Here, *n* is the number of observations and *r* is the coefficient of rank correlation. The zone of acceptance of null hypothesis is -1.746 to +1.746 from a standard *t*-test table having 16 degrees of freedom.

The *t* statistic was computed for each of the five rank correlation coefficients obtained, and it was seen that the value was outside of the critical zone of acceptance at 90% significance level and 16 (18–2) degrees of freedom. Thus, it is concluded that the final rank had a statistically significant relationship with each of the set of ranks given by the five individuals.

Table 4

Calculation for Rank Relation Coefficient Using Sample Data

CD	Rank1	Rank 2	Rank3	Rank4	Rank5	Final Ranks	SD-FN/RNK1	SD-FN/RNK2	SD-FN/RNK3	SD-FN/RNK4	SD-FN/RNK5
D1	1	1	2	1	1	1	0	0	1	0	0
D2	2	2	4	2	3	2	0	0	4	0	1
D3	3	3	3	5	2	3	0	0	0	4	1
D4	4	4	7	3	5	4	0	0	9	1	1
D5	5	5	5	7	4	5	0	0	0	4	1
D6	6	6	0	0	8	7	1	1	49	49	1
D7	7	7	0	0	0	10	9	9	100	100	100
D8	8	8	0	0	0	0	64	64	0	0	0
D9	9	9	6	8	6	6	9	9	0	4	0
D10	10	0	0	0	0	0	100	0	0	0	0
D11	0	0	10	0	7	0	0	0	100	0	49
D12	0	10	0	4	0	9	81	1	81	25	81
D13	0	0	0	9	0	0	0	0	0	81	0
D14	0	0	1	10	0	8	64	64	49	4	64
D15	0	0	8	0	9	0	0	0	64	0	81
D16	0	0	0	0	10	0	0	0	0	0	100
D17	0	0	9	0	0	0	0	0	81	0	0
D18	0	0	0	6	0	0	0	0	0	36	0
Sum of Squared Deviations							328	148	538	308	480
Spearman's Coefficient of Correlation							0.6615	0.8473	0.4448	0.6821	0.5046
t statistic of coefficient of correlation							3.5283	6.3804	1.9865	3.7316	2.3381
The critical t value at 90% level of significance for 16 degrees of freedom for two tailed test is 1.746											

Where,

CD= Candidate

SD-FN/RNK1 = Square Difference between Final Rank and Rank1

SD-FN/RNK2 = Square Difference between Final Rank and Rank2

SD-FN/RNK3 = Square Difference between Final Rank and Rank3

SD-FN/RNK4 = Square Difference between Final Rank and Rank4

SD-FN/RNK5 = Square Difference between Final Rank and Rank5

This shows that the final rank is significant compared to the individual ranks since the final ranks have been derived from the set of 90 observations (= 18 * 5). Moreover, it

also has a statistically significant degree of association with each of the sample sets, which means that the final rank was derived after considering all five data sets. This leads to improvement in the accuracy of data and also helps to reduce bias in the data.

CHAPTER 7

CONCLUSIONS AND FUTURE WORK

In this project I addressed the common problem faced by search engine users: innumerable unrelated results with an unreliable rank list displayed for users' queries. Although many attempts have been made to solve this problem, the solution is very difficult due to the unstructured architecture of the Web and the increasingly rapid speed of new information stored in the Web daily. One of the most effective solutions is the development of MSEs. Since MSEs are designed to dig deeper into the Web with the help of the combination of existing search engines, the search results have a higher accuracy with more variety than the regular search engines. However, every MSE is not the same. For example, using the same query, when one focuses on retrieving information regarding the latest news, others may be concentrating on science subject-related information.

In order to have a general-purpose information retrieval system, I proposed a Meta-MSE system where common users may find helpful information in comparison to other existing MSEs. As far as my research is concerned, this method is a new idea that none of the scientific research news and professional journals has published in the past.

Five MSEs were carefully selected for the new MMSE. A ranking algorithm was designed to collect the search rank result from the individual MSE and then re-rank it into a final and single rank. The final rank list is more refined and is found to have a much

higher variety of information in comparison to the existing MSE, especially in terms of general usage. At the same time, it is more reliable because it is a collection of the best out of the bests. Therefore, statistically the new system proved to be more reliable than all the existing MSE available today.

However, after performing an extensive study about the available information retrieval systems and search engines as a whole, I conclude that there cannot be a search engine that gives a 100% solution and can satisfy different types of users, unless and until a human mind's readable computer system is developed which can extract all available data in the Web. It is due to the complexity of the human mind that one person expects one result and another person may expect different results from the same query. Another reason is that the rapid speed of the growth of information or data stored in the Web daily surpasses the invention speed of a new data extractor system for our search engines, a million times faster.

In order to decide which is the best among the available search engines, one may choose the search engine that best fits his or her requirement for that day, based on the type of algorithm and sources which are used in the search engine. The next day the requirement may switch to something else for a best fit. This is why the general purpose type search engine comes in handy. Another possible advantage of this new system is the ability to convert to a search engine where users can select what combination of MSEs they like to use to search. This can be a powerful search engine that concentrates the search focus in a particular direction rather than being used for general purposes. The proposed ranking algorithm is implemented using C# and was tested with at least 20 sample user queries.

The full implementation of this new system in the live Web is left for future work. A couple of items were not discussed in this research. To have a fully functional MMSE, a written Web crawler for each MSE used in this MMSE is needed. A query parsing mechanism that will translate the user string of query into a specific instruction, which the MSE and search engine can understand, is also needed.

Expanding the search option, refining the query, and sorting results based on conditions may be added features. These are only minor features, yet they contribute to the MMSE becoming an efficient search engine. Whether one should include the sponsored pages can be decided later. Another feature is making my MMSE a user selectable MSE, as discussed earlier, which means that users can choose which MSE they like to combine as their search source. This way the search can be more focused, and the MMSE will become a very powerful information retrieval tool.

REFERENCE LIST

- Akritidis, L., Katsaros, D., & Bozanis, P. (2008, August). Effective ranking fusion methods for personalized metasearch engines. In S. Katsikas (Chair), *Proceedings of the 2008 Panhellenic Conference on Informatics* (pp. 39-43). Washington, DC: IEEE Computer Society. doi:10.1109/PCI.2008.31 Retrieved from <http://delab.csd.auth.gr/papers/PCI08akb.pdf>
- Aslam, J. A., & Montague, M. (2001, September). Models for metasearch. In D. H. Kraft (Chair), *Proceedings of the 24th annual international ACM SIGIR conference* (pp. 276-284). New York, NY: ACM. doi:10.1145/383952.384007 Retrieved from www.ccs.neu.edu/home/jaa/IS4200.10X1/.../borda_metas.pdf
- Bradly, P. (2008, January). Human-powered search engines: An overview and roundup. *Ariadne*, 54. Retrieved from <http://www.ariadne.ac.uk/issue54/search-engines/>
- Built-a-Home-Business. (2011). *The purpose of search engines*. Retrieved from <http://www.build-a-home-business-website.com/purpose-of-search-engines.html>
- Chris, S. (2005, March 22). *Metacrawlers and metasearch engines*. Retrieved from <http://searchenginewatch.com/article/2066974/Metacrawlers-and-Metasearch-Engines>
- Christianet.com. (2012). *Advantages of metasearch engines*. Retrieved from <http://www.christianet.com/searchengines/advantagesofmetasearchengines.htm>
- Cutts, M. (2007, November). *What is the Google Snippet?* Retrieved from http://www.youtube.com/watch?v=vS1Mw1Adrk0&feature=player_embedded
- Daniel, P. (2010). *The history of web browsers*. Retrieved from <http://www.instantshift.com/2010/10/15/the-history-of-web-browsers/>
- Ding, L., Finin, T., Joshi, A., Pan, R., Cost, R. S., Peng, Y., Reddivari, P. et al. (2004). Swoogle: A search and metadata engine for the semantic web. In D. Grossman (Chair), *Proceedings of the 13th ACM International Conference Information and Knowledge Management (CIKM)* (pp. 652-659). Washington, DC: CIKM.
- Dogpile. (2012). *Metasearch 101: Or what makes Dogpile better than the rest!* Retrieved from <http://www.dogpile.com/support/metasearch>

- Dorn, J., & Naz, T. (2008, April). *Structuring meta-search research by design patterns*. Institute of Information Systems, Technical University Vienna, Austria. Presentation at the meeting of the International Computer Science and Technology Conference; San Diego, CA. Retrieved from <http://www.ec.tuwien.ac.at/~dorn/Papers/ICSTC08.pdf>
- Dwork, C., Kumar, R., Naor, M., & Sivakumar, D. (2001). Rank aggregation methods for the Web. In V. Y. Shen (Chair), *Proceedings of ACM Conference on World Wide Web (WWW)* (pp. 613-622). Hong Kong: Hypermedia Track of the 10th International World Wide Web Conference. Retrieved from www.eecs.harvard.edu/~michaelm/CS222/rank.pdf
- Fagin, R., Kumar, R., Sivakumar, D., Mahdian, M., & Vee, E. (2004, June). Comparing and aggregating rankings with ties. In D. Suci (Chair), *Proceedings of the 23rd ACM SIGMOD-SIGACT-SIGART symposium on principles of database systems: PODS 2004* (pp. 47-58). Paris, France. Retrieved from <http://www.cs.uiuc.edu/class/fa05/cs591han/sigmodpods04/pods/pdf/P-06.pdf>
- Freedom Scientific. (2008). *Using search engines to find information on the web*. Retrieved from http://www.freedomscientific.com/training/Surfs-Up/Search_Engines.htm
- Harpreet. (2010). *What is a search engine?* Retrieved from <http://www.trivology.com/articles/307/what-is-a-search-engine.html>
- Horrocks, I., & Hendler, J. (Eds.). (2002, June). *Proceedings: The semantic web*. First International Semantic Web Conference, Sardinia, Italy.
- Jacob, G. (2009, September 30). *The history of web browsers*. Retrieved from <http://sixrevisions.com/web-development/the-history-of-web-browsers/>
- Keil. (2008, July). *European Privacy Seal awarded to privacy-friendly search engines Ixquick, Startpage and Startpage*. Retrieved from <https://www.ixquick.com/eng/press/pr-ixquick-europrise.html>
- Killer Startups. (2011). *Mamma.com: The mother of all search engines*. Retrieved from <http://www.killerstartups.com/review/mamma-com-the-mother-of-all-search-engines/>
- Langa, F. (2001, January). *Surface Vs. Deep Web*. Retrieved from <http://windowssecrets.com/langalist-plus/surface-vs-deep-web%C2%A0/>
- Latha, S., & Rajaram, M. (2010). Semantic-based multiple web search engine. *International Journal of Human-Computer Science & Engineering Studies*, 2(5), 1722-1728.
- List-Handley, C. J. (2008). *Information literacy and technology* (4th ed.). Dubuque, IA: Kendall/Hunt, p. 36.

- List of Search Engines. (2012). *Best meta-search engines*. Retrieved from <http://www.listofsearchengines.info/meta-search-engines>
- Lu, Y. (2011). *Automatic search interface clustering and search result processing in metasearch engine* (Unpublished doctoral dissertation). State University of New York, Binghamton.
- Lu, Y., Meng, W., Shu, L., Yu, C., & Liu, K. (2005, November). Evaluation of result merging strategies for metasearch engines. In A. H. H. Ngu (Ed.), *Web information systems engineering: WISE 2005* (pp. 53-66). Sixth International Conference on Web Information Systems Engineering. New York, NY: WISE. Retrieved from www.cs.binghamton.edu/~meng/pub.d/Lu_p211.pdf
- Manning, C. D., Raghavan, P., & Schütze, H. (2008). *An introduction to information retrieval*. Cambridge, England: Cambridge University Press.
- Office of Scientific and Technical Information. (2008). *Searching the deep web*. Retrieved from <http://proz.tumblr.com/post/65657477/science-accelerator-widget-science-accelerator>
- Old Dominion University Library. (2010). *Open web vs. deep (invisible) web*. Retrieved from http://www.lib.odu.edu/genedinfolit/3searching/open_web_vs_deep_invisible_web.html
- PC Magazine Encyclopedia*. (2011). Definition of: Surface web. Retrieved from http://www.pcmag.com/encyclopedia_term/0,2542,t=surface+Web&i=52273,00.asp
- Renda, M. E., & Stracci, U. (2003). Web metasearch: Rank vs. score based rank aggregation methods. In G. Lamount (Chair), *Proceedings of the 2003 ACM symposium on applied computing* (pp. 841-846). New York, NY: ACM.
- Rich, G. (2011, June). Yippy, Inc. (YIPI) and MuseGlobal to merge, offering unified access to a data cloud of curated content. Retrieved from <http://www.museglobal.com/news/press.php?content=2012/20120611.html>
- Search Engine Watch. (2007, March). *How search engines work*. Retrieved from <http://searchenginewatch.com/article/2065173/How-Search-Engines-Work>
- Selberg, E. W. (1999). *Towards comprehensive web search* (Unpublished doctoral dissertation). University of Washington, Seattle.
- Shah, S. (2003). *Implementing an effective web crawler: Devbistro.com*. Retrieved from <http://www.devbistro.com/articles/Misc/Implementing-Effective-Web-Crawler>
- Shea, B. (2012, February). *How big is twitter in 2012?* Retrieved from http://www.mediabistro.com/alltwitter/twitter-statistics-2012_b18914

- Shettar, R., & Bhuptani, R. (2008). A vertical search engine based on domain classifier. *International Journal of Computer Science & Security*, 2(4), 18.
- Signorini, A. (2010). *Average query length on major search engines*. Retrieved from <http://blog.alessiosignorini.com/2010/02/average-query-length-february-2010/>
- Souldatos, S., Dalamagas, T., & Sellis, T. (2006, November). Captain Nemo: A metasearch engine with personalized hierarchical search space. *Informatica Slovenia*, 30(2), 173-181. Retrieved from <http://web.imis.athena-innovation.gr/~dalamag/pub/sds-ssi06.pdf>
- Sullivan, D. (2002, October). *Intro to search engine optimization: Searchenginewatch.com*, Retrieved from http://www.msit2005.mut.ac.th/msit_media/1_2549/ITEC5611/Materials/Optimizing%20for%20Crawlers.pdf
- Sutter, J. D. (2011, September). *How many pages are on the internet?* Retrieved from http://articles.cnn.com/2011-09-12/tech/web.index_1_internet-neurons-human-brain?_s=PM:TECH
- Van Eesteren, A. (2008, July). *First European Privacy Seal awarded to search engine Ixquick*. Retrieved from <https://www.ixquick.com/eng/press/eu-privacy-seal.html>
- Webopedia. (2012). *Browser*. Retrieved from <http://www.webopedia.com/TERM/B/browser.html>
- Webopedia. (2011). *Search engine*. Retrieved from http://www.webopedia.com/TERM/S/search_engine.html
- Web3school. (1999-2012). *The semantic web: What is semantic web?* Retrieved from http://www.w3schools.com/web/web_semantic.asp
- Wikipedia. (2011). *Spearman's rank correlation coefficient*. Retrieved from http://en.wikipedia.org/wiki/Spearman%27s_rank_correlation_coefficient
- Wordpress. (2012). *A live look at activity across WordPress.com*. Retrieved from <http://en.wordpress.com/stats/>
- Young, R. D. (2011, August). *Who uses search engines? 92% of adult U.S. internet users [Study]: Search engine watch*. Retrieved from <http://searchenginewatch.com/article/2101282/Who-Uses-Search-Engines-92-of-Adult-U.S.-Internet-Users-Study>
- Zamir, O. & Etzioni, O. (1999). Grouper: A dynamic clustering interface to web search results. *Computer Networks: The International Journal of Computer & Telecommunications Networking*, 31(11), 1361-1374. doi:10.1016/S1389-1286(99)00054-7