The Overview of Web Search Engines

Sunny Lam
Department of Computer Science
University of Waterloo
Waterloo, Ontario
Canada
s6lam@math.uwaterloo.ca

February 9, 2001

Abstract

The World Wide Web allows people to share information globally. The amount of information grows without bound. In order to extract information that we are interested in, we need a tool to search the Web. The tool is called a search engine. This survey covers different components of the search engine and how the search engine really works. It provides a background understanding of information retrieval. It discusses different search engines that are commercially available. It investigates how the search engines find information in the Web and how they rank its pages to the given query. Also, the paper provides guidelines for users on how to use search engines.

Keywords: Search Engines, World Wide Web, Information Retrieval, Page Ranking, Google, Metasearch, Web Crawlers, and Web Directories

1 Introduction

1.1 Motivation

The World Wide Web (also known as "WWW", "Web, or "W3") is the universe of network-accessible information, the embodiment of human knowledge. It allows people to share information globally. That means it allows anyone to read and publish documents freely. The World Wide Web hides all the detail of communication protocols, machine locations, and operating systems from the user. It allows users to point to any other Web pages without any restrictions. Brin defines the Web as follows (1998):

"The Web is a vast collection of completely uncontrolled heterogeneous documents".

The Web is accessible to anyone via a Web browser. Search engines answer tens of millions of queries every day (Brin, 1998). The amount of information on the Web grows exponentially. If we only count the textual data, it is estimated to be in the order of one terabyte (Baeza-Yates, 1999). Of course, the size of multimedia data types (image, audio, and video) is even bigger. Since this data is managed by many individuals, organizations, and companies, the Web is unstructured. Thus, the World Wide Web can be seen as a large unstructured and ubiquitous database. Finding useful information on the Web is a very difficult task. There is a need to develop tools to manage, retrieve, and filter information in the big database. The goal of this paper is to find solutions to search data in the World Wide Web efficiently. In other words, how do we develop a fast, robust, and accurate search engine?

1.2 Difficulties

The Web has become increasingly commercial over time, from 1.5% of .com domain in 1993 to over 60% in 1997. At the same time, search engine development has moved from the academic domain to the commercial domain. Today, most search engine developments take place in companies without technical information to the public. Therefore, it is very difficult to study today's search engines.

The problems for searching information on the Web can be divided into two classes. They are problems with the data itself and problems regarding how users use the information retrieval system (Baeza-Yates, 1999).

Six difficulties are introduced in the first class. They are the following:

- **Distributed data:** data is distributed widely in the world. It is located at different sites and platforms. The communication links between computers vary widely. Plus, there is no topology of data organization.
- **High percentage of volatile data:** documents can be added or removed easily in the World Wide Web. Changes to these documents go unnoticed by others. 40% of Web pages change every month (Baeza-Yates, 1999). There is a very high chance of dangling links.
- **Large volume:** the growth of data is exponential. It poses scaling issues that are difficult to cope with.

- **Unstructured and redundant data:** the Web is not exactly a distributed hypertext. It is impossible to organize and add consistency to the data and the hyperlinks. Web pages are not well structured. 30% of all Web pages are duplicates (Baeza-Yates, 1999). Semantic redundancy can increase traffic.
- **Quality of data:** a lot of Web pages do not involve any editorial process. That means data can be false, inaccurate, outdated, or poorly written.
- **Heterogeneous data:** data on the Web are heterogeneous. They are written in different formats, media types, and natural languages.
- **Dynamic data:** the content of Web document changes dynamically. The content can be changed by a program such as hit counter that keep tracks of number of hits.

The second class of problems deals with interaction between the user and the retrieval system. They are the following:

- **How to specify a query:** the user has to be able to find a way to pose a query, so that the result of the query contains relevant information.
- **How to interpret the answer provided by the system:** the user needs to know how to select the documents from the given query result.

1.4 Scope of this Paper

This paper discusses each component of a search engine. It provides a background understanding, difficulties of creating a search engine, characteristics of the Web, different types of search engines, different search engine architectures, user interfaces, ranking algorithms, web crawlers, metasearchers, indices, guidelines for users to use search engine, and future work.

2 Information Retrieval

Before we can understand search engines, we need to understand information retrieval (IR), because Web searching is within the field of information retrieval. Before the Internet was born, information retrieval was just index searching. For example, searching

authors, title, [1]and subjects in library card catalogs or computers. Today, among other things, IR includes modeling, document classification and categorization, systems architecture, user interfaces, data visualization, filtering, and languages. IR deals with the representation, storage, organization of, and access to information items (Baeza-Yates, 1999). The user should easily retrieve information of what interests him/her.

There is a difference between information retrieval and data retrieval. In data retrieval, the result of a query must be accurate: it should return the exact match tuples of the query, no more and no less. If there is no change to the database, the result of a query executed at different times should be the same. On the other hand, information retrieval can be inaccurate as long as the error is insignificant. The main reason for this difference is that information retrieval usually deals with natural language text which is not always well structured and could be semantically ambiguous. Data retrieval deals with data that has a well-defined structure and semantics (e.g. a relational database). In addition, data retrieval cannot provide a solution given a subject or topic, but information retrieval is able to do so.

In order to satisfy the user's information needs, the IR system must find a way to interpret the contents of the information items and be able to rank them according to a degree of relevance to the user query. This interpretation involves how to extract information in syntactic and semantic ways. The goal of an IR system is to retrieve all the documents, which are relevant to a query while retrieving as few non-relevant documents as possible. To achieve this goal, IR needs users to provide a set of words which convey the semantics of the information need. Also, a document is represented by a set of keywords or index terms for IR to extract. These keywords or index terms can be derived from information experts or a computer through eliminating articles and connectives, the use of stemming (which reduces distinct words to their common grammatical root), and identifying nouns (which eliminates adjectives, adverbs, and verbs).

There is a difference between IR and searching the Web. IR allows access to whole documents, whereas, search engines do not. The reason is that it is too expensive

---

Note: In 1994, the first search engine was developed. It was called World Wide Web Worm (WWWW). It referenced 110,000 pages and it answered 1500 queries per day (Brin, 1998).

to store all the Web pages locally and too slow to access remotely on other Web servers. The paper is going to demonstrate ways to enable searching without access any documents.

3 Characteristics of the Web

As we all know, measuring the World Wide Web is a very difficult task due to its dynamic nature. In 1999, there were over 40 million computers in more than 200 countries connected to the Internet. Within these computers, over 3 million of them are Web servers (NetSizer, 1998). There are two explanations why the number of Web servers is huge. The first one is that many Web sites share the same Web server using virtual hosts and not all of them are fully accessible to the outside world. The second one is that not all Web sites start with the prefix www and by only counting Web sites with this prefix there were only 780,000 in 1998 (Network Wizards, 1998).

As for the total number of Web pages, there were estimated to be 350 million in 1998. Between 1997 and 1998, the size of Web pages was doubled in nine months and was growing at a rate of 20 million pages per month.(Baeza-Yates, 1999).

The most popular formats of Web documents are HTML, followed by GIF and JPG (images format), ASCII files, Postscript and ASP. The most popular compression tools are GNU zip, Zip, and Compress. Most HTML pages are not standard, because they do not comply with HTML specifications. HTML documents seldom start with a document type definition. Also they are typically small with an average of 5 Kb and a median of 2 Kb. On average, each HTML page contains one or two images and five to fifteen hyperlinks. Most of these hyperlinks are local, meaning the associated Web pages are mostly stored in the same Web server (Baeza-Yates, 1999).

The top ten most referenced Web sites such as Microsoft, Netscape, and Yahoo! are referenced in over 100,000 places. In addition, the site containing the most external links is Yahoo!. Yahoo! glues all the isolated Web sites together to form a large Web database. If we assume that the average size of an HTML page is 5 Kb and there are 300 million Web pages, then we have at least 1.5 terabytes of text. This huge size is consistent with other studies done by other organizations (Baeza-Yates, 1999).

4 User Problems

There are some problems when users use the interface of a search engine.

- The users do not exactly understand how to provide a sequence of words for the search.
- The users may get unexpected answers because he/she is not aware of the input requirement of the search engine. For example, some search engines are case sensitive.
- The users have problems understanding Boolean logic: therefore, the user cannot perform advanced searching.
- Novice users do not know how to start using a search engine.
- The users do not care about advertisements, so the search engine lacks funding.
- Around 85% of users only look at the first page of the result, so relevant answers might be skipped.

In order to solve the problems above, the search engine must be easy to use and provide relevant answers to the query.

4.1 Searching Guidelines

Here are some guidelines helping users to search.

- Specify the words clearly, stating which words should be in the page and which words should not be in the page.
- Provide as many particular terms as possible: page title, date, and country.
- If looking for a company, institution, or organization, try to guess the URL by using the www prefix followed by the name, and then (.com, .edu, .org, .gov, or country code).
- Some search engines specialize in some areas. For example, the users can use ResearchIndex (www.researchindex.com) to search research papers.
- If the users use broad queries, try to use Web directories as starting points.
- The user should notice that anyone can publish data on the Web, so information that they get from search engines might not be accurate.

5 Types of Search Engines

Today, most search engines are based in the United States and are specialized for English. The users search documents by keywords. For example, these typical search engines are AltaVista, Excite, and Northern Light. However, there are also other type of search engines that are specialized in other languages such as Chinese, Korean, and Japanese (written Kanji). Examples are Chinese Yahoo! (http://chinese.yahoo.com) and Yahoo! Japan (http://www.yahoo.co.jp). Since these Kanji languages are not written in the Latin alphabet (different data structure), they might need to have different retrieval techniques. Besides keyword search engines, there are search engines like Ask Jeeves! that simulates an interview, DirectHit that ranks the Web pages in the answer in order of their popularity, Yahoo! that searches by categories; Search Broker that searches in specific topics, and DejaNews that searches the USENET archives. (Baeza-Yates, 1999)

6 The Largest Search Engines

The largest search engines in 1998 were AltaVista, HotBot, Northern Light, and Excite, in that order. They cover 28-55% or 14-34% of all Web pages (Baeza-Yates, 1999). The following table lists the popularity of today's search engines.

Table 1: URLs and estimated size in millions of the largest search engines in May 1998

Search Engine	URL	Web pages indexed
AltaVista	www.altavista.com	140
AOL Search	search.aol.com	N/A
Excite	www.excite.com	55
Google	google.stanford.edu	25
GoTo	goto.com	N/A
HotBot	www.hotbot.com	110
Go	www.go.com	30
Lycos	www.lycos.com	30
Magellan	magellan.excite.com	55
Microsoft	search.msn.com	N/A
Northern Light	www.northernlight.com	67

Open Text	www.opentext.com	N/A
WebCrawler	www.webcrawler.com	2

Note: In 1998, AOL Search was called AOL Netfind and Go was called Infoseek

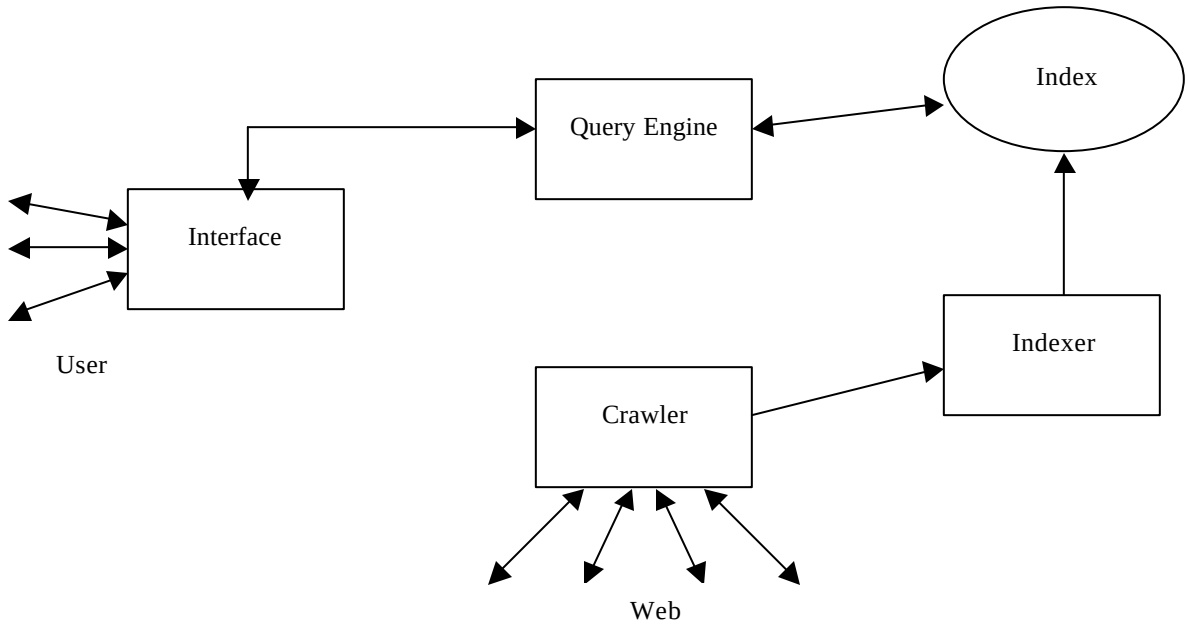
7 Search Engine Architectures

Most search engines use centralized crawler-indexer architecture. As mentioned earlier in Section 1, most implementations of search engines are not available to the public. However, there are still some that can be found. They are AltaVista, Harvest, and Google.

7.1 AltaVista Architecture

This section discusses the AltaVista search engine as an example for demonstrating how this architecture works. The crawler's duty is to run on a local machine and sends requests to remote Web servers. The index is used in a centralized fashion to answer queries from users. The following figure shows AltaVista's software architecture. It can be divided into two parts. The first part consists of the user interface and the query engine. The second part contains the crawler and the indexer (Baeza-Yates, 1999).

Figure 1: The typical crawler-indexer architecture (AltaVista)



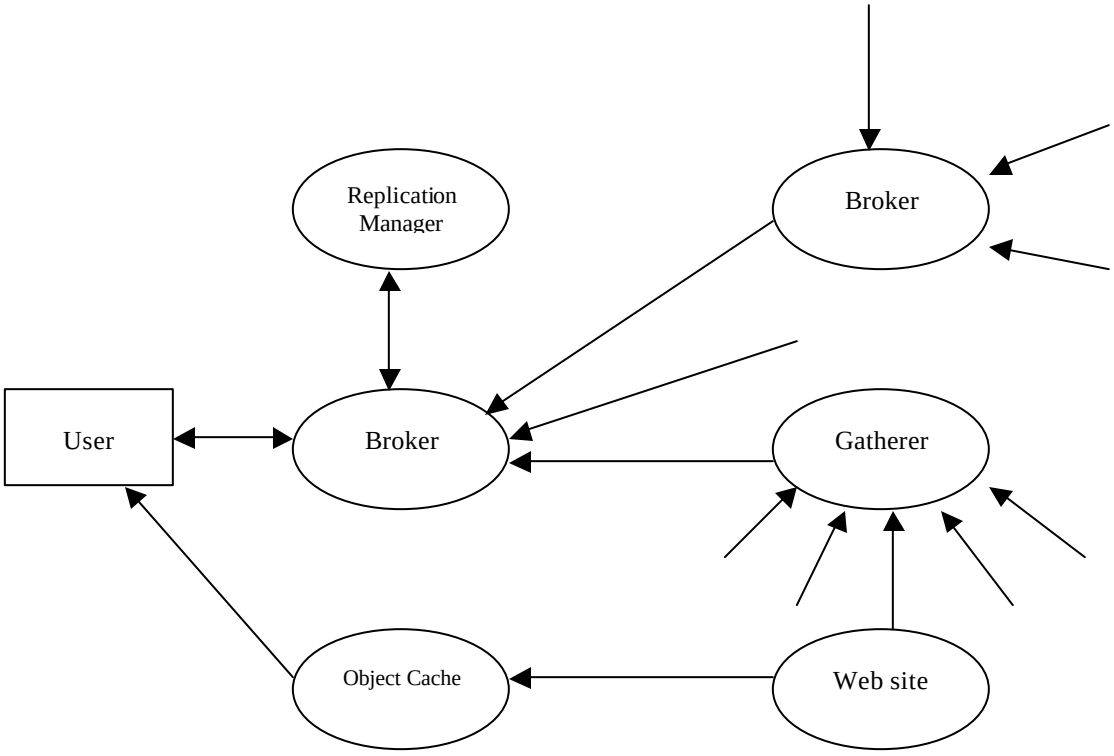
In 1998, AltaVista was running on 20 processors. All processors have 130 Gb of Ram and over 500 Gb of hard disk space. Only the query engine uses more than 75% of these resources. (Baeza-Yates, 1999)

There are two problems with this architecture. The first problem is data gathering in the dynamic Web environment, which uses saturated communication links, and high load at Web servers. The second problem is the volume of the data. The crawler-indexer architecture does not cope with Web growth in the near future. (Baeza-Yates, 1999)

7.2 Harvest Architecture

There are several variants of the crawler-indexer architecture. One of the variants is called Harvest. Harvest is the most important variant that uses distributed architecture to gather data and distribute data. It is used by CIA, NASA, the US National Academy of Sciences, and the US Government Printing Office (Baeza-Yates, 1999). In addition, Netscape's Catalog Server is a commercial version of Harvest and Network Appliances' cache is a commercial version of the Harvest Cache.

Figure 2: Harvest architecture



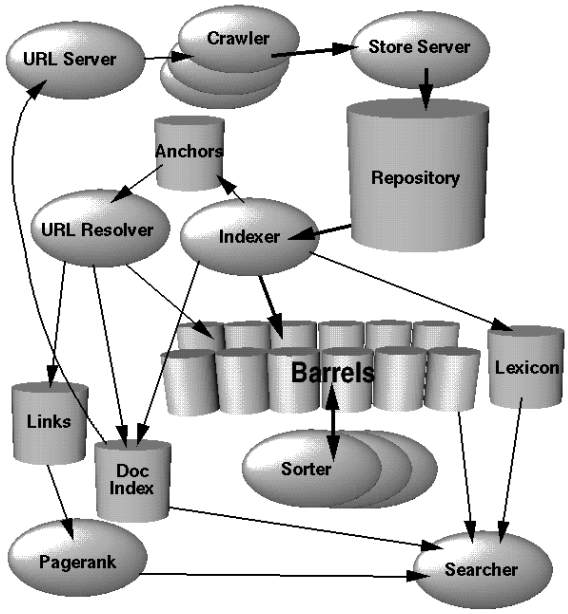
As shown in Figure 2, Harvest introduces two main elements: gatherers and brokers. The job of gatherers is to collect and extract indexing information from one or more Web servers. Gathering times are specified by the Harvest system. The times are periodic as suggested by its name, Harvest. The job of brokers is to provide the indexing mechanism and the query interface to the data gathered. Brokers receive information from gatherers or other brokers to update their indices. Also, brokers can filter information and send it to others, so that other brokers are saved time. Depending on the configuration of gatherers and brokers, server's workload and network traffic can be balanced. The harvest system builds topic-specific brokers and focuses the index contents thereby avoiding many of the vocabulary and scaling problems of generic indices. In addition, the system provides a replication manager (to replicate servers for enhance user-base scalability) and an object cache (to reduce network and server load). For more information, please read (Bowman, 1994).

7.3 Google Architecture

The word Google comes from the word googol, which means 10^{100}. The Google search engine (www.google.com) heavily uses the structure present in hypertext. It claims that it produces better results than other search engines today. It references about 24 million pages (Brin, 1998).

Google is mainly written in C/C++ for efficiency reasons. It can run on Solaris or Linux platforms. The architecture is shown in the Figure 3.

Figure 3: The Google Architecture



The URL Server sends lists of URLs to be fetched by the crawlers. The crawlers download pages according to the list and send the downloaded pages to the Store Server. The Store Server compresses the pages and stores them in the repository. Every Web page has an associated ID number called a docID, which is assigned whenever a new URL is parsed out of a Web page. The index performs an indexing function. It reads the repository, uncompresses the documents, and parses them. Each page is converted into a set of word occurrences called hits. The hits contain information about a word: position in document, an approximation of font size, and capitalization. The indexer distributes these hits into a set of "barrels" and creates a partially sorted forward index (like bucket sort). It parses out all the links in every Web page and stores important information about them in an anchors file. The anchors file contains information about where each link points from and to and the text of the link. After that, the URL Resolver reads the anchors file and converts relative URLs into absolute URLs and in turn into docID. It puts the anchor text into the forward index, associated with the docID. It generates a links database for storing links and docIDs. The database is used to compute PageRanks for all the documents. The Sorter takes the barrels and resorts them by wordID instead of

docID in order to generate the inverted index. Also, the Sorter produces a list of wordIDs and offsets into the inverted index.

A program called DumpLexicon takes this list together with the lexicon produced by the indexer and generates a new lexicon to be used by the searcher. The searcher is run by a Web server and uses the lexicon built by DumpLexicon together with the inverted index and the PageRanks to answer queries.

Google had fetched over 24 million Web pages in 1998. Its storage size was 55.2 GB without repository and 108.7 GB with repository. On average, Google answered a query between 1 and 10 seconds. (Brin, 1998)

8 User Interfaces

The user interface of search engines consists of two parts: the query interface and the answer interface. The basic query interface is a box where a sequence of words is entered. The sequence of words entered into different search engines produces different results. For example, AltaVista performs a search by the union of these words, whereas, HotBot performs a search by the intersection of these words (all words must appear in the result documents). Some search engines support complex query interface, which including Boolean operators and other features, such as phrase search, proximity search, URL searches, title search, date range, and data types search.

About the answer interface, search engines usually return pages in the order of relevance to the query. In other words, the most relevant pages appear on the top of the list. Typically, each result entry in the list includes a title of the page, an URL, a brief summary, a size, a date, and a written language.

9 Web Directories

The most popular Web directory is Yahoo! (Brin, 1999). Others are eBLAST, LookSmart, Magellan, and Nacho. Most of them also allow keyword searches. Besides these Web directories, there are also others that provide subject category services. For example, AltaVista Categories, AOL Netfind, Excite Channels, HotBot, Infoseek, Lycos Subjects, and WebCrawler Select. Web directories are also called catalogs, yellow pages,

or subject directories. The following table shows the number of Web sites and the number of categories for each Web directory.

Table 2: The most popular Web directories in 1998

Web directory	URL	Number of Web sites	Categories
eBLAST	www.eblast.com	125	N/A
LookSmart	www.looksmart.com	300	24
Lycos Subjects	www.lycos.com	50	N/A
Magellan	magellan.excite.com	60	N/A
NewHoo	www.newhoo.com	100	23
Netscape	search.netscape.com	N/A	N/A
Search.com	www.search.com	N/A	N/A
Snap	www.snap.com	N/A	N/A
Yahoo!	www.yahoo.com	750	N/A

Note: URL of Lycos Subjects was a2z.lycos.com in 1998. URL of Magellan was www.mckinley.com in 1998. URL of Netscape was www.netscape.com in 1998. Snap is no longer available.

Directories are hierarchical taxonomies that classify human knowledge. The first level of the taxonomies range from 12 to 26 (Baeza-Yates, 1999). Samples of first level categories are education, employment, games, and travel. The largest directory, Yahoo!, has close to one million pages classified. Yahoo! also offers support other languages as well, such as Chinese and Japanese. The second largest is LookSmart, which has about 24,000 categories in total. Usually pages that are submitted to Web directories are reviewed and accepted to more than one category.

10 Ranking

Ranking is the heart of the search engine. In order to produce a good search engine, we need to know how to rank pages properly for the result documents. There is not much information available about this in the public. Today, most search engines use variations

of the Boolean or vector model to do ranking. Recall that search engines do not allow access to the text, but only the indices, because it is too expensive in terms of time and space. So, when searching, ranking must use indices while not accessing the text. Besides that, there are also other difficulties as well. There might be too many relevant pages for a simple query. Also, it is difficult to compare two search engines, because of their continuous improvement.

There are three ranking algorithm that are proposed by Yuwono and Lee. They are Boolean spread, vector spread, and most-cited. The first two are the normal ranking algorithms of the Boolean and vector model extended to include pages pointed to by a page in the answer or pages that point to a page in the answer. The third, most-cited, is based only on the terms included in pages having a link to pages in the answer. (Yuwono, 1996)

10.1 PageRank

There are also other approaches as well. WebQuery allows visual browsing of the Web pages. Their ranking algorithm is based on how connected each Web page is. The PageRank algorithm uses the equation defined below:

$$PR(a) = q + (1-q)\sum_{i=1}^{n} PR(p_i) / C(pi)$$

where a is the page that we want to rank; $PR(a)$ is the page rank of page a; q is the probability of the page being accessed; p_1 to p_n is the pages that point to page a. $C(a)$ is the number of out-going link in page a.

The Google search engine uses the PageRank algorithm as well. It simulates users using the search engine and applies the equation to rank the Web pages. It uses a citation graph. The graph contains 518 million links. It allows rapid calculation: 26 million Web pages can be computed in a few hours. The page has high ranking when many other pages point to it or if some other high-ranking pages point to it. (Brin, 1998)

10.2 Anchor Text

Most search engines associate the text of a link with the page that the link is on. However, Google associates it with the page the link points to. There are several advantages for the latter approach. First, anchors often provide more accurate

descriptions of Web pages than the pages themselves. Second, anchors may exist for documents that cannot be indexed by a text-based search engine, such as images, programs, and databases. So, the search engine would not return non-existing pages to users. Google has over 259 million anchors indexed for their crawl of 24 million pages. The idea of anchor text was originated by WWWW (Brin, 1998).

10.3 Other Features

There are also other features provided for ranking Web pages. Google has location information for all hits and so it makes extensive use of proximity in search. Also, Google keeps track of some visual presentation details such as font size of words. It gives a higher ranking to the words in a larger or bolder font. Last, it has full raw HTML of pages available in repository (Brin, 1998).

11 Web Crawlers

Crawlers are also called robots, spiders, worms, wanderers, walkers, and knowbots. The first crawler, Wanderer was developed by Matthew Gray in 1993. Due to the competitive nature of the search engine business, the designs of these crawlers have not been publicly described. There are several crawling techniques available in public. The simplest one is to start with a set of URLs and from that extracts other URLs recursively in breath-first or depth-first manner. So, search engines allow users to submit top Web sites that will be added to the URL set. A variation to this technique is to start with a set of popular URLs, because the pages for these URLs have information frequently requested. Both techniques work well with one crawler. But when we face multiple crawlers, we have to avoid them crawling the same page. One solution is to send crawlers to different country codes or Internet names to avoid duplicated work.

There are problems a crawler needs to cope with. The first problem is that Web pages change dynamically, so the page that the index points to may not exist anymore. That's why the search engine keeps track of date, and shows the date to the query result. Also, the search engine refreshes pages when they are outdated. The fastest crawlers are able to traverse up to 10 million Web pages per day.

As mentioned earlier, there are two policies used to traverse Web pages. The first one is breath-first policy. It looks at all the pages linked by the current page and so on. The coverage will be wide but shallow. This may cause the Web server to have many rapid requests. The second is depth-first policy. We follow the first link of a page and we do the same on that page until we cannot go deeper. After that, it returns recursively. The advantage of using depth-first search is deep and space complexity is cheaper. But the disadvantage of using it is narrow.

11.1 Google Crawler

The Google search engine uses multiple machines for crawling. The crawler works as follows. The crawler consists of five functional components which run in different processes. A URL server process reads URLs out of a file and forwards them to multiple crawler processes. Each crawler process runs on a different machine, which is single threaded. It uses asynchronous I/O to fetch data from up to 300 Web servers in parallel. The crawlers transmit downloaded pages to a single StoreServer process, which compress the pages and store them to disk. Then the indexer process reads pages from disk. It extracts links from the pages and saves them to a different disk file. A URL Resolver process reads the link file, analyzes the URLs contained therein, and saves the absolute URLs to the disk file that is read by the URL server.

11.2 Internet Archive

Another example of crawlers is Internet Archive. It uses multiple machines as well. Each single threaded crawler is assigned up to 64 sites to crawl and no site is assigned to more than one crawler. It reads an assigned list of URLs from disk into per-site queues. Then it uses asynchronous I/O to fetch pages from these queues in parallel. Once a page is downloaded, the crawler extracts the hyperlinks inside the page. The crawler assigns links of the page to appropriate site queues. Periodically, a batch process merges these logged “cross-site” URLs into the site-specific seed sets, filtering out duplicates in the process.

11.3 Mercator

The Web crawler, Mercator is named after the Flemish cartographer Mercator. It is developed by Compaq. It is scalable and extensible, and is entirely written in Java. It is scalable in the sense that it can scale up to the entire Web and has been used to fetch tens of millions of Web documents. By extensible, Mercator is designed in a modular way, so new functionalities can be added by third parties. For example, Mercator can add new protocol modules to support more network protocols. It can also add new processing modules to process documents in different way. Mercator can easily install new components.

To provide these two main services, Mercator requires a well-defined interface to each of the components. A mechanism must exist for specifying how the crawler is to be configured from its various components. A sufficient infrastructure must exist to make Mercator easy to write new components. For more detail on Mercator, please read reference (Heydon, 1999).

12 Indices

Most indices use variants of inverted files. An inverted file is a list of sorted words. Each word has pointers to the related pages. A logical view of the text is indexed. Each pointer associates a short description about the page that the pointer points to. There are approximately 500 bytes for storing a URL and a short description. So, search engines require 50Gb for indexing 100 million pages. In addition, search engines store answers in memory in case the user asks the same query again.

This indexing technique helps to reduce the size of inverted files to about 30% of the size of the text. If we have 100 million pages, it saves us 150Gb space. Do not forget that compression can be used to further reduce the size of indices.

In general, a search engine uses a binary search on the inverted files for searching for a single keyword. If the user types multiple keywords in the query, the search engine searches each of the keywords independently and combine all the results to generate the final answer. As for the phrase search, the searching technique is unknown. Baeza-Yates points out that the phrase search is to search words near each other (1999). To do a search by prefix, it requires two binary searches.

13 Metasearchers

A metasearcher is a Web server that takes a given query from the user and sends it to several search engines, Web directories, and other databases. Then the metasearcher collects the answers from these sources and returns a unified result to the user. It is able to sort the result by different attributes such as host, keyword, date, and popularity. In addition, the metasearcher enables the user to pose the same query to various sources through a single common interface. A list of metasearchers is listed in the following table.

Table 2: URLs of metasearchers and number of sources that they use in 1998.

Metasearcher	URL	Sources used
C4	www.c4.com	14
Dogpile	www.dogpile.com	25
Highway61	www.highway61.com	5
InFind	www.infind.com	6
Mamma	www.mamma.com	7
MetaCrawler	www.metacrawler.com	7
MetaMiner	www.miner.uol.com.br	13
Local Find	local.find.com	N/A

Note: C4 was called Cyber 411 (www.cyber411.com) in 1998. InFind was called Inference Find in 1998. Local Find was called MetaSearch (www.metasearch.com) in 1998.

A metasearcher does not necessary run on a Web server. It can run on a client machine as well. Examples are Copernic, EchoSearch, WebFerrret, WebComprass, and WebSeeker (Baeza-Yates, 1999).

A metasearcher is more informative than a single search engine, so browsing is simpler. Although the number of results returned from each search engine can be adjusted, the number is still limited due to efficiency. The whole result may not match the user's query.

In general, less than 1% of the Web is indexed by the most popular search engines such as AltaVista, HotBot, Excite, and Infoseek. Metasearchers can combine search results from each popular search engine and increase the number of indexed Web pages.

13.1 Inquirus

The new metasearchers are expected to have better ranking systems than a single search engine. An example is the NEC Research Institute metasearch engine, Inquirus. Inquirus downloads and analyzes the Web page obtained and then displays each page, highlighting the places where the query terms were found. The results are displayed in a progressive manner in order to reduce access time. It allows non-existent pages or pages that have changed and do not contain the query anymore to be discarded. However, Inquirus is not publicly available. (Lawrence, 1998)

13.2 Savvy Search

Different search engines return different search results same query. No single search engine is likely to return more than 45% of the relevant results (Howe, 1997). Some search engines are good at searching some topics. It is important to query search engines given a set of keywords from the users. The SavvySearch metasearcher has this function. It was available online in 1997, but it is not available today. The SavvySearch is designed to efficiently query other search engines that are likely to return the best result. Its goal is to maximize the likelihood of returning good links while minimizing computational and Web resource consumption. When it processes a query from a user, it determines how many search engines to contact simultaneously and in what order the search engines should be contacted. It queries other search engines concurrently to moderate resource consumption. Search engines rank Web pages. Savvy Search ranks search engines for determining which search engines are worthwhile to contact for a given query. Search engines are ranked based on learned associations between search engines and query terms and recent data on search engine performance. That means the search engines will weight lower when they fail to return for a given query and weight higher when the user explores their links. For more detail, please read (Howe, 1997).

13.3 STARTS

Stanford Protocol Proposal for Internet Retrieval and Search (STARTS) is an emerging protocol for Internet retrieval and search that facilitates the task of querying multiple document sources. It has supported from 11 companies and organizations: Fulcrum, Infoseek, PLS, Verify, WAIS, Microsoft Network, Excite, GILS, Harvest, Hewlett-Packard Laboratories, and Netscape (Gravano, 1997). The goals of STARTS are choosing the best sources (search engines) to evaluate a query, submitting the query at these sources, and merging the query results from these sources.

STARTS introduces a protocol for machine-to-machine communication. Its motivation is to keep the protocol simple and easy to implement. It aims to solve the problems below:

- **The Query-Language Problems:** Different search engines use different query languages. The interfaces and capabilities of these search engines may vary over time. For example, some search engines interpret (abstract "databases") that asks for documents that have the keyword databases in the abstract, whereas other search engines do not have this capability.
- **The Rank-Merging Problem:** The proprietary algorithms (ranking pages) of search engines are hidden due to the nature of business, so it is hard to rank the result documents overall. It is still hard to merge query results from different search engines that use the same ranking algorithm, even if we know this algorithm.
- **The Source-Metadata Problem:** If a search engine does not export any information on summary, it becomes hard for a metasearcher to assess what kind of information the source covers.

The protocol mainly deals with what information (i.e. queries, results) needs to be exchanged between search engines and metasearchers. It does not focus on how the information is formatted or transported. It proposes the basic features of the query language that a search engine should support. For more detail, please see (Gravano, 1997).

14 Add-on Tools

There are software tools to help user browsing and searching. One of them is Alexa (Alexa, 2000). It is a free Web navigation service. It appears as a toolbar in the Web browser Internet Explorer 5x. It provides useful information about the sites. It allows the users to browse other related sites. It is able to perform searches within the Web site, related sites, or the whole Web. It allows the user to shop online when the user finds interesting things. More important, it provides popularity, speed of access, freshness overall quality from Alexa users about the Web site.

There are other tools to visualize a subset of the Web by drawing graphs. These tools are Microsoft's SiteAnalyst, MAPA, IBM's Mapuccino, SurfSerf, and Merzscope.

15 Future Works

Search engines have existed for five years. There are many trends and not much research being performed within this area. First, special information retrieval models tailored for the Web are needed in order to provide better information filtering. Second, combining structure and content in the queries are needed as well as new visual metaphors to pose those queries and visualize the answers. Third, new distributed schemes to traverse and search the Web must be devised to cope with its growth. Fourth, new caching techniques are needed to increase efficiency. Fifth, better ranking algorithms are needed so the result pages are more relevant to the query. Sixth, algorithms that choose which pages are indexed are needed. Seventh, techniques that finds dynamic pages, which are created on demand. Eighth, a mechanism to avoid searching for duplicated data. Ninth, techniques that can search multimedia documents on the Web. Tenth, developments of friendly user interfaces are needed. Eleventh, a standard protocol to query search engines might become handful in the near future, for example STARTS. Twelfth, study of Web mining is a good idea. Last, developments of reliable and secure intranet. (Baeza-Yates, 1999)

16 Conclusion

This survey describes the overview of Web search engines. The goal of this paper is to help people perform Web searching easily and effectively. It discusses the different

components of search engines such as architectures, user interfaces, ranking algorithms, Web crawlers, metasearchers and indices. Also, it investigates other issues such as information retrieval, characteristics of the Web, different types of search engines, searching guidelines and possible future research. It provides reasons why we need to study search engines, and it provides relevant references for readers to proceed further. More important, the readers should try out different search engines that are available today.

17 Acknowledgments

I would like to thank Tamer Ozsu for providing me this opportunity to write this survey. I would also like to thank Ian Davis and David Pooley for proof reading this paper. Both Ian Davis and David Pooley's comments are very valuable to me.

18 References

Alexa: Main Page. http://www.alexa.com, 2000.

R. Baeza-Yates and B. Ribeiro-Neto. Modern Information Retrieval. Addison Wesley, New York, NY, USA, 1999

C. Bowman et al. The Harvest Information Discovery and Access System. *Proc. 2nd International World Wide Web Conference*, pages 763-771, 1994.

S. Brin and L. Page. The Anatomy of a Large-Scale Hypertextual Web Search Engine. *Proc. 7th WWW Conference*, 1998.

Google: Main Page. http://www.google.com, 2000.

L. Gravano, K. Chang, H. Garcia-Molina, and A. Paepcke. STARTS: Stanford Proposal for Internet Meta-searching. *Proc. ACM SIGMOD Conference*, pages 207-218, 1997.

L. Gravano and H. Garcia-Molina. Generalizing GIOSS To Vector-Space Databases and Broker Hierarchies. *Proc. VLDB Conference*, 1995.

A. Heydon and M. Najork. Mercator: A Scalable, Extensible Web Crawler. *World Wide Web 2(4)*, pages 219-229, 1999.

A. Howe and D. Dreilinger. SavvySearch: A Meta-Search Engine that Learns which Search Engines to Query. *AI Magazine, 18(2)*, 1997.

J. Kleinberg. Authoritative Sources in a Hyperlinked Environment. *Proc. ACM-SIAM Symp. On Discrete Algorithms*, pages 668-677, 1998.

S. Lawrence and C. L. Giles. Context and Page Analysis for Improved Web Search. *IEEE Internet Computing, 2(4):38-46,* 1998.

NetSizer: Main Page. http://www.netsizer.com, 1998.

Network Wizards: Internet Domain Survey. http://www.nw.com, 1998.

B. Yuwono and D. L. Lee. Search and Ranking algorithms for locating resources on World Wide Web. *Proc. International Conference on Data Engineering (ICDE),* pages 164-171, 1996.