
Technological Educational Institution of Larissa in collaboration with Staffordshire University

***MSc in Computer Science course***

***Postgraduate Thesis***

# Meta Web Crawler for Information Retrieval on the client side

Provatidis Apostolos, September 2007
Instructor: Konstantinos Kokkinos

## Abstract

The Internet is a chaotic source of information that everyone can access. In most cases it is impossible for someone to retrieve specific information by just browsing at random websites. To address this problem many intelligent programs have been built that perform user defined searches and retrieve web pages that contain the information related to the search. These programs are often called web crawlers, spiders, agents, robots, bots, wanderers, worms and various other names. A variety of competing search engines (also called crawlers) uses these programs to catalogue the World Wide Web and provide the best possible results to their users' queries. Each search engine is able to index only a fraction of the www and uses its own algorithms. This has the confusing result that each search engine provides a different result set for the same user query. A different kind of search crawlers called meta crawlers sends user requests to several other search engines simultaneously and re-ranks the results into a more complete and improved result set which can be closer to the ideal model of an information retrieval system. This re-ranking process is the challenging area of research that this document focuses on.

# Table of Contents

# Chapter 1 - Search Engines and Web Crawlers

## 1.1 Introduction

Information retrieval is what the Internet is all about. If there were only a small number of web sites that was known to everyone then searching for information would be a simple task. There are billions of indexable web pages [2] in a continuously growing World Wide Web, having every kind of information imagination can reach. It is clear that a user will need help on conducting a search on the web.

Although the Web facilitates many applications and information services, e.g. E-mail, FTP (File Transfer Protocol), electronic publishing, E-commerce, distance learning, Tele-conferences, etc., the primary use of the web after the E-mail is for finding information. However, finding a specific piece of information among such incredible amount of information would be impossible without powerful tools that automatically browse and search the web.

Four major methods for finding information on the web are identified [57], which include:

- Using a known URL,
- Using Hypertext links to navigate from a web page to another web page,
- Narrowcast services or Portals which push web pages to users according to their particular profiles,
- Search engines which allow users to search the web exploring traditional and advanced information retrieval techniques.

In most cases it is impossible for someone to retrieve specific information by just browsing at websites or following links. Portals can be very helpful but the categories of information that a user can access is very limited. The method that has dominated the information retrieval process is the use of search engines.

A search engine is an information retrieval system designed to help find information stored on a computer system, such as on the World Wide Web, inside a corporate or proprietary network, or in a personal computer. The search engine allows one to ask for content meeting specific criteria (typically those containing a given word or phrase) and retrieves a list of items that match those criteria. This list is often sorted with respect to some measure of relevance of the results. Search engines use regularly updated indexes to operate quickly and efficiently.

## 1.2 Web Crawlers

Each search engine uses intelligent programs, called web crawlers, which browse the World Wide Web in a methodical, automated manner. This process is called Web crawling or spidering. That is the reason why search engines are also called web crawlers. Many sites, in particular search engines, use spidering as a means of providing up-to-date data. Web crawlers are mainly used 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. In general, it starts with a list of URLs to visit, called the seeds. As the crawler visits these URLs, it identifies all the hyperlinks in the page and adds them to the list of URLs to visit, called the crawl frontier. URLs from the frontier are recursively visited according to a set of policies.

At first web crawlers were defined as “software programs that traverse the World Wide Web information space by following hypertext links and retrieving Web documents by standard HTTP protocol” [3]. As research continues web crawlers have been developed that use other methods than following hypertext. An example is the metasearch crawlers. This “species” of crawlers use other search engines, other crawlers, and combine their results [4, 5]. Research in crawlers began in the early 1990’s, the same time that Internet started to attract attention. Many different versions of crawlers have since been developed and studied. The first crawler for the Web is claimed to be Wanderer, written in 1993 [6]. Since then many web crawlers have been developed including the most used search engines today such as the Google search engine [7].

To understand how the web crawlers crawl through the hypertext we have to review the research conducted on the different ways of analyzing and representing the content and

structure of the web. The crawlers rely on such information to navigate them to the results. There are two main categories of the web analysis which are presented below.

The first category is the content based approaches. In this case the web analysis is based on the information extracted of the actual HTML content of a web page. In some approaches the text inside the body tags of an HTML page can be analysed to determine if the page is relevant to a target domain. In other approaches the relativeness of a web page can be determined by the title or the headings of the HTML. In general the text included in the title or the headings is assigned a higher weight in addition to the body [7, 8].

In addition, the URL of a web page can contain useful information about the web page too. For example the URL:

http://en.wikipedia.org/wiki/Web_crawler

can tell us that the URL comes from the domain wikipedia.com, and that it is likely to be related to the topic Web crawler. We also know that this page comes from a .com site, which may be considered less authoritative than pages from a .gov site.

The results of these approaches can be improved by using domain knowledge. The words that are included inside a web page can be compared to a list of domain specific terminology so that the relativity of the web page can be checked.

The second category is the Link based approaches which are used more and more frequently by recent web engines. The links that a web page contains or the links that other pages contain that lead to this page can be used as a manner of ranking the relativity of the web page to a domain. This approach makes one or both of the following assumptions [9]:

**Assumption 1**. A hyperlink from page A to page B is a recommendation to the author of the page B by the author of page A.

**Assumption 2**. If page A and page B are connected by a hyperlink, then they might be relevant or similar.

The higher the number of hyperlinks that lead to a given page, the higher the ranking of this page. For example, in scientific articles, if an article is cited many times is considered better than one that is never cited. Using the same logic, a web page that many hyperlinks pointing to it, is considered better that a web page that no hyperlink point.

We can also analyse the anchor text, the text that we click on a web page and which contain the hyperlink to another web page. It is possible that the anchor text can be a good description of the target web page [10, 11]. The text that appears near a hyper-link can be analysed as well [12]

The link analysis approaches were first used to navigate crawlers on their search [13, 14, 15]. Another two examples of web crawlers that use link analysis are the Focused Crawler [16] which is used for web page classification and the HyPursuit [17] which is used for clustering.

It is reasonable that all the hyperlinks can not be weighted the same. For example a hyperlink from an authoritative source (e.g. Yahoo) must have a higher weight that one from an unimportant personal web page. Two of the most popular algorithms that have been developed to provide the proper weights are the PageRank [7] [18] and the HITS [19] algorithms.

According to PageRank algorithm a web page can have a high PageRank if the page is linked from many other pages, and the rank is higher if these referring pages are also pages that have a high PageRank recursively. PageRank is defined as follows:

"We assume page A has pages $T_1...T_n$ which point to it (i.e., are citations). The parameter d is a damping factor which can be set between 0 and 1. We usually set d to 0.85. Also C(A) is defined as the number of links going out of page A. The PageRank of a page A is given as follows:

$$PR(A) = (1-d) + d\left(\frac{PR(T_1)}{C(T_1)} + ... + \frac{PR(T_n)}{C(T_n)}\right)$$

Note that the PageRanks form a probability distribution over web pages, so the sum of all web pages' PageRanks will be one.

PageRank or PR(A) can be calculated using a simple iterative algorithm, and corresponds to the principal eigenvector of the normalized link matrix of the web. Also, a PageRank for 26 million web pages can be computed in a few hours on a medium size workstation."[7]

"PageRank can be thought of as a model of user behavior. We assume there is a "random surfer" who is given a web page at random and keeps clicking on links, never hitting "back" but eventually gets bored and starts on another random page. The probability that the random surfer visits a page is its PageRank."[7]

The page rank algorithm, although computationally expensive due to its iterative nature [18], has proven to be highly efficient as it is the algorithm that the Google, the most popular search engine today uses [7].

The other popular algorithm is the HITS (hyper-link-induced topic search) algorithm proposed by Kleinberg [19]. The HITS algorithm is similar to the PageRank. The difference is that there are two ranks, a hub and an authority rank. For each page, the authority score for the page is determined by the hub scores of its parents, and the hub score of the page is determined by the authority scores of its children. In other words a page to which many others point should be a good authority, and a page that points to many others should be a good hub. The hub and an authority scores are calculated as follows [1]:

$$AuthorityScore(p) = \sum_{\forall \text{ q pointing to p}} Hubscore(q)$$

$$HubScore(p) = \sum_{\forall \text{ r being pointed by p}} Authorityscore(r)$$

An example of a crawler that uses the HITS algorithm is the Clever search engine [20]. The HITS algorithm has been extended by Bharat and Henzinger [21] and added more scores such as the influence of a page to its neighbours, according to its relevance.

The HITS algorithm is computationally expensive due to the use of iteration when calculating the hub and authority ranks

## *1.3 Use of Web Crawlers*

Web crawlers are widely used in four main areas of web applications [1]:

1. **Personalized search**. These crawlers are used for user defined searches. They allow the user to have more control and personalization options during the search process and they can run on the user's pc which results to more capabilities and functionalities due to the increased computational power [22].
2. **Collection Building**. Most search engines use web crawlers to built collections of web pages and appropriately index them [7, 23, 24, 25]. The collection building can be used for other purposes too, such as URL sampling [26, 27], specific document collections etc.
3. **Archiving**. Web crawlers can be used to archive web sites with all their contents. The number of the archived web pages is restricted mostly by the storage capabilities of the crawler as the World Wide Web is continuously growing.
4. **Web statistics**. Beside collection building and archiving, the web pages collected by web crawlers can be used to provide statistics about the web. There have been many crawlers developed for web statistics [6]. The statistics can be about the number of servers on the web, the average size of a web page, the number of URLs that return a 404 (page not found) respond etc.

## 1.4 Research Topics on web crawlers

Web crawlers have been a research topic since the early 1990's. There are many types of research conducted on web crawlers and some of them are presented in the following lines

The efficiency of web crawlers is clearly a field that researchers will always be interested in especially when the crawlers are used for information retrieval. Improving the efficiency of a web crawler that is used for information retrieval, means that it can retrieve relevant information from the Web more effectively. There are many algorithms as we saw above, and they are continuously improving.

Due to the recursive nature of the algorithms that are used, speed becomes a research topic too. There are studies about applying program-optimization techniques to operations such as I/O procedures and IP address lookup to build fast crawlers that can be scaled up to large collections. Some examples are the Google's crawler [7, 28, 23], and Internet Archive's crawler [29, 30]. Today a web crawler can compute the PageRank for many millions of web pages in a few hours on a medium size workstation.

Policy issues on web crawlers are topic of research too. There can be many reasons that a web site administrator will not want web crawlers, crawling about their site or some parts of it. These reasons can be performance, protected or private content and many others. Also there are web crawlers which are not welcome at all, such as crawlers looking for email addresses to fill up a spam list. That need for restriction has triggered researchers to study ways for controlling the crawler's behaviour. There are two strategies that a web site administrator can follow to control the behaviour of web crawlers on his site. The first is called robot exclusion protocol, and it is implemented by a text file called robot.txt. This file is placed on the root directory of the web site and it indicates to web crawlers which pages it is allowed to visit. The second is called the robots META tag, and is a tag inside the HTML code of the web page. META tags indicate to crawlers whether a document may be indexed, or used to extract more links [31]. There is no way until now to really force web crawlers to comply with these standards but most commercial crawlers do comply.

## 1.5 Graph Search Algorithms

The web can be represented as set of nodes (pages) connected with directed edges (hyper-links) to form a directed graph. There are several graph search algorithms and there are three groups of graph search algorithms that are implemented by web crawlers.

- The first group is called uninformed search algorithms. It consists of simple algorithms like breadth-first and depth-first search. These algorithms are very often used because of their easy implementation but they lack of efficiency. The breadth-first algorithm is very popular among web crawlers and it works by collecting all pages on the current level before proceeding to the next level.
- The second group is called informed search algorithms. In addition to the first group, these algorithms use some information about the next node that they will visit. For example the best-first search algorithm use such information to select the "best" node to be the next visited. The information can be the similarity to the search query, the in-links, the keyword frequency, the PageRank score etc [32].
- The third group is called parallel search. As the name states these algorithms explore different paths in parallel. The Smart Itsy Bitsy Spider [33] uses a genetic algorithm model that falls into this category. Although very powerful these algorithms are difficult to implement and are not used frequently by web crawlers.

## 1.6 Personalization of Web Crawlers

When it comes to information retrieval using web crawlers, crawlers can be personalized and focused to the user needs. A web crawler providing personalisation options to the user can be a very useful tool for an effective search for information on the web. Personalized web crawlers usually run on the client pc. For this reason they can allocate more CPU power and more memory which results to more functionalities and better results.

Many web crawlers have been developed to be personalized and focused. The first personalised crawlers allowed the user to enter the keywords and then specify the depth and width of search (of links in homepages). Other personalised crawlers would search the links of a web page, the "neighbours", to find relevant pages and return a list of the most promising. Another category of crawlers, provide the possibility of sharing relevant search sessions among users. Some crawlers would provide visual graphs of the way that the nodes are connected through hyperlinks. In other examples, the user will provide example pages to start the search or the crawler will download web pages and provide a summary of each page to the user. The personalised web crawlers can have options like scheduled automated search. More sophisticated web crawlers would return results to the user by applying complex algorithms such as genetic algorithms.

Personalisation can be applied to meta crawlers too. As said before, these are web crawlers that connect to multiple search engines and combine their results. In this case the user can choose, among other options, which of the search engines will be included in the search or the weight of each search engine.

Lately, personalised crawlers are used on P2P (peer to peer) networks too. In this case each computer can have its own strategy on how to respond to search requests from other users [34].

There are a lot of personalised and focused web crawlers and some of them will be presented in this section [1].

Some commercial examples are Excalibur's RetrievalWare and Internet Spider that collect, monitor and index information from text documents or graphic files on the Web. Other examples are the WebMiner, the WebRipper and the Teleport crawlers and their job is to select and download files from given websites. Autonomy's products support a wide range of information collection and analysis tasks, such as automatic searching and monitoring information sources in the Internet or the corporate Intranets. One of their tasks is the classification of documents into categories that are predefined by users or domain experts. A similar case is Verity's knowledge-management products which include Intelligent Classifier, Agent Server, and Information Server.

One of the first examples of personalised web crawlers is the trueMosaic [36]. This crawler is used for personalised search and it uses a modified best-first search algorithm. The web crawler is another example of crawler using the best-first search algorithm. The Itsy Bitsy Spider searches the Web using a best-first search too but in combination with and a genetic algorithm approach [35, 36]. The Webnaut web crawler is a more recent crawler that also applies genetic algorithms [37]. Another example of web crawler is based on hybrid simulated annealing [38].

Web crawlers can be integrated into browsers like the TkWWW crawlers [39]. The crawler is designed to search Web neighbourhoods to find logically related homepages and return a list of links that it considers being the most promising. However, the search is limited to one or two hops, or links, from the original homepages. The TkWWW robots can also be run in the background to build HTML indexes, compile WWW statistics, collect a portfolio of pictures, or perform any other function that can be described by the TkWWW Tcl extensions.

SPHINX is a framework for creating personal, site-specific web crawlers [40]. These crawlers perform breadth-first search and view the search results as a 2-dimensional graph. Another example of a personalised web crawler is the CI Spider which uses the search results for linguistic analysis and clustering [41]. This crawler was extended by Collaborative Spider, which is able to share relevant search sessions among users using a multiagent system [42].

Some more examples of personalised web crawlers are the Focused Crawler [16] and the Context Focused Crawler [43]. The user, in the first case, defines some example pages and the crawler locates Web pages that are relevant and analyzes the link structures among them. In the second case the search process is directed by Naive Bayesian classifier.

There are also examples of personalized meta crawlers. The first one was the MetaCrawler on 1995 [5, 44]. Some meta crawlers which followed MetaCrawler are Dogpile [76], Profusion [45] which connects to 6 search engines after selection by the user,

SavvySearch [46] and 37.com which as the name states connects to 37 different search engines.

Grouper, an extension of the MetaCrawler uses a suffix-tree clustering algorithm to categorize the web pages returned from the search engines [47]. In other cases of meta crawlers, users can be assisted to form their queries with the use of domain ontology [48]. Also link analysis can be used to define which are the appropriate search engines for the query [49].

NECI or Inquirus meta spider [50, 51], creates a new summary of the result pages based on the search terms after filtering the Pages that are no longer available (dead lings or 404).

Similar to NECI the MetaSpider also categorize the results and performs a linguistic analysis [4]. TetraFusion on the other hand has the same functionalities as NECI but also performs a hierarchical and graph-based categorization on the result set [52]

## *1.7 Web Crawlers for Specialized Search Engines*

There is a high variety of general search engines on the web but in many occasions the results that these search engines return to the user after a query are too general and too many. There is a need for specialised search engines that focus on specific domain areas. An example of a specialized search engine is the Google Scholar which focuses on the domain of scholarly literature. A Query to the Google scholar will provide more specialized results than a query to the general Google search engine.

These specialised search engines use focused or targeted Web crawlers that look for and provide access to material on a specific subject or domain from material found on the Web. Usually these web crawlers have the relatively easy task to search for web pages that their URL include only a specific range of domain names. For example the web crawlers can be restricted to look for web pages that their URL starts with the string “http://www.staffs.ac.uk/”.

The two main problems that these web crawlers will have to address are that they need to identify from a list of unvisited URLs the ones most likely to contain relevant information because it will be more efficient to visit these URLs first, and that for each downloaded document, the crawlers need to determine its relevance according to a specific purpose by determining the quality and reputation of each document [1].

Although specialized search engines are very promising there can not be a specialised search engine for every possible domain, simply because the domains can be infinite. Only limited domain areas, which are the most popular, have specialised search engines dedicated on them.

There can be many different types of specialised search engines. One type of specialised search engines is to be a part of a general Web engine focusing on a certain subject area or domain, with content often accessible through a specialized interface. Another type can be a focused or targeted Web crawler that looks for material on a specific domain found on the Web. A specialised search engine can also be built only to provide access to material for a specific site or tool.

## *1.8 Implementations of Crawlers on Search Engines*

The first web crawler that was used on a search engine was the World Wide Web Worm on 1994 [53]. This crawler would only index a web page by its title and headers. The Repository-Based Software Engineering or RBSE crawler was the first crawler which could index the whole document [54].

Many web crawlers with full indexing capabilities were developed since then. Some examples are WebCrawler [25], Lycos [24], and Harvest [55].

Most of the web crawlers used on search engines use the simple breadth-first search algorithm as it is easy to implement and fast to execute. It has been shown that using this algorithm high-quality pages can be found early in a searching process. If a URL is relevant to a target domain there is a high probability that the neighbour Web pages are also relevant.

Many focused web crawlers for search engines also use the best-first algorithm. The measurements that define the best next nodes can be the PageRank score, the keyword frequency, the number of in-links, the similarity to starting examples, and others [13]. Google crawlers are known to use variations of PageRank scores [7].

As web crawlers are intelligent agents, machine learning techniques can also be applied. The web crawler that was used in the Cora search engine, a search engine for computer science research papers [56], was traversing the Web based on immediate and future reward as measured in terms of Web page relevance. This crawler was based on reinforcement learning techniques [12].

## *1.9 Dissertation Organization*

This document is introducing a meta search engine model which uses a novel reranking algorithm to sort the result sets of several helper crawlers that are used.

Theoretical material and current research about crawlers and meta crawlers must be presented in order to prepare the reader to understand the introduced model and the reasons that the particular approach to information retrieval is selected. After the theoretical issues are covered, the reranking algorithm must be explained and tested. The proposed model must then be evaluated and compared to existing meta search engines so that conclusions and future research topics can be extracted.

The above tasks are covered and organised in the following way:

- Chapter 1 introduces and explains web crawlers. The categories of the different web crawler types are presented, as well as research topics on web crawlers and issues as personalisation, use and implementation.
- Chapter 2 focuses on meta search crawlers. Meta search is explained, the reasons that meta search is developed are presented and the current research about meta crawlers are covered in this chapter.
- Chapter 3 introduces and explains the proposed meta search model. The reranking algorithm is analysed step by step and the client side proposition is explained
- Chapter 4 presents the evaluation of the proposed model. After the reranking algorithm is tested with both imaginary input and real input on the World Wide Web, the model is compared to other meta search engines.
- Chapter 5 presents the conclusions that can be extracted from the previous chapters and some future research topics that arise.

# Chapter 2 - Meta Search Engines and Current Research

## 2.1 Meta Search Engines

Search engines can be very efficient, providing relevant results to the users query, very fast and in a comprehensive way. If we want to approach an ideal model of a search engine we have to take into consideration the limitations of today search engines. The crawlers that the search engines use are able to access only a fraction of the entire World Wide Web, have different user interfaces and use different ranking algorithms. This leads to the confusing effect that each search engine provides different results. A solution to this problem is the use of meta search engines.

A meta-search engine is a search engine that sends user requests to several other search engines and/or databases and returns the results from each one. Meta search enables users to enter search criteria once and access several search engines simultaneously. Since it is hard to catalogue the entire web, the idea is that by searching multiple search engines you are able to search more of the web in less time and do it with only one click. The ease of use and high probability of finding the desired page(s) make metasearch engines popular with those who are willing to access bigger lists of relevant results. Another use is to get at least some results when no result had been obtained with traditional search engines.
Metasearch engines create what is known as a virtual database. They do not compile a physical database or catalogue of the web. Instead, they take a user's request, pass it to several other heterogeneous databases and then compile the results in a homogeneous manner based on a specific algorithm.

No two metasearch engines are alike. Some search only the most popular search engines while others also search lesser-known engines, newsgroups, 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 search engine or database. Others return results according to relevance, often concealing which search engine returned which results. This benefits the user by eliminating duplicate hits and grouping the most relevant ones at the top of the list.

Search engines frequently have different ways they expect requests submitted. For example, some search engines allow the usage of the word "AND" while others require "+" and others require only a space to combine words. The better metasearch engines try to synthesize requests appropriately when submitting them.

Results can vary between metasearch engines based on a large number of variables. Still, even the most basic metasearch engine will allow more of the web to be searched at once than any one stand-alone search engine.

The reasons for the development of a metasearch engine can be categorized in the following way [58]:

- **Access and search a bigger fraction of the World Wide Web**. The coverage of the Web by individual major general-purpose search engines is decreasing steadily as the World Wide Web is increasing. This is mainly due to the fact that the World Wide Web has been increasing at a much faster rate than the indexing capability of any single search engine. By combining the coverages of multiple search engines through a metasearch engine, a much higher percentage of the World Wide Web can be searched.
- **Solve the scalability of searching the Wold Wide Web**. The approach of employing a single search engine to Search the entire World Wide Web has poor scalability. In contrast, if a metasearch engine on top of all the single search engines can be created as an alternative to search the entire World Wide Web, then the problems associated with employing a single search engine will either disappear or be significantly alleviated. The size of a typical special-purpose search engine is much smaller than that of a major general-purpose search engine. Therefore, it is much easier for it to keep its index data more up to date (i.e., updating of index data to reflect the changes of documents can be carried out more frequently). It is also much easier to build the necessary hardware and software infrastructure for a special-purpose search engine. As a result, the metasearch engine approach for searching the

entire Web is likely to be significantly more scalable than the centralized general-purpose search engine approach.

- **Facilitate the invocation of multiple search engines**. The information needed by a user is frequently stored in the databases of multiple search engines. As an example, consider the case when a user wants to find the best 10 newspaper articles about a special event. It is likely that the desired articles are scattered across the databases of a number of newspapers. The user can send his/her query to every newspaper database and examine the retrieved articles from each database to identify the 10 best articles. This is a formidable task. First, the user will have to identify the sites of the newspapers. Second, the user will need to send the query to each of these databases. Since different databases may accept queries in different formats, the user will have to format the query correctly for each database. Third, there will be no overall quality ranking among the articles returned from these databases even though the retrieved articles from each individual database may be ranked. As a result, it will be difficult for the user, without reading the contents of the articles, to determine which articles are likely to be among the most useful ones. If there are a large number of databases, each returning some articles to the user, then the user will simply be overwhelmed. If a metasearch engine on top of these local search engines is built, then the user only needs to submit one query to invoke all local search engines via the metasearch engine. A good metasearch engine can rank the documents returned from different search engines properly. Clearly, such a metasearch engine makes the user's task much easier.
- **Improve the retrieval effectiveness**. Consider the scenario where a user needs to find documents in a specific subject area. Suppose that there is a special-purpose search engine for this subject area and there is also a general-purpose search engine that contains all the documents indexed by the special-purpose search engine in addition to many documents unrelated to this subject area. It is usually true that if the user submits the same query to both of the two search engines, the user is likely to obtain better results from the special-purpose search engine than the general-purpose search engine. In other words, the existence of a large number of unrelated documents in the general-purpose search engine may hinder the retrieval of desired documents. In text retrieval, documents in the same collection can be grouped into clusters such that the documents in the same cluster are more related than documents across different clusters. When evaluating a query, clusters related to the query can be identified first and then the search can be carried out for these clusters. This method has been shown to improve the retrieval effectiveness of the system [59]. For documents on the Web, the databases in different special-purpose search engines are natural clusters. As a result, if for any given query submitted to the metasearch engine, the search can be restricted to only special purpose search engines related to the query, then it is likely that better retrieval effectiveness can be achieved using the metasearch engine than using a general-purpose search engine. While it may be possible for a general purpose search engine to cluster its documents to improve retrieval effectiveness, the quality of these clusters may not be as good as the ones corresponding to special-purpose search engines. Furthermore, constructing and maintaining the clusters consumes more resources of the general-purpose search engine.

## *2.2 Current Research on Meta Search*

There has been considerable research on information retrieval and on meta searching techniques during the recent years. Merging the results from different search engines is always a challenge. Most search engines provide very little information with which to perform the merging, aside from the document ranking. The document score assigned by a search engine to a document retrieved from a collection may or may not be provided. When scores are not provided, only the document ranking and some a previous knowledge about the

datasets can be used for merging the different result sets. Borda-Fuse [60] and Metacrawler [5] utilize document rank and its appearance in the results list of several engines to perform merging. This is done by summing the ranks of the document in the different ranked lists.

If the document scores are given they can be possibly used as additional information for merging. However, it is difficult to re-rank the documents since document scores are local for each specific dataset and engine combination. An approach is to map the scores returned by each of the search engines to relevance probabilities [61] Another approach can be to compute global scores, based on global statistics of the query terms, as though all datasets were merged to a single collection [59]..

Fox and Shaw [62] proposed several combination techniques, including setting the score of each document to the sum of the scores obtained by the individual search engines (COMBSUM) or by multiplying this sum by the number of engines which have non-zero scores (COMBNZ). Those techniques were extended by Lee [63], by normalizing each engine on a per query basis. He observed that the best combination obtained when systems retrieved similar sets of relevant documents and dissimilar sets of non-relevant documents.

This document focuses on applying a weight to every ranking. There are some suggestions on this topic, mostly by applying a score based on its statistics. The score is used for merging the different rankings by weighting the document scores. Vogt et al. [64] used a linear combination of the document scores, where the linear weights are constant for the search engines, and are learned during a training session. Two very well known and extensively used algorithms that use this approach are CORI [65], which is applicable to the framework of federation, and ProFusion [45], created for metasearch. CORI computes a query dependent score for each dataset. It requires, in addition to document scores, some statistics for each query term from each dataset. ProFusion creates an engine-specific weight by measuring the precision of each search engine over a known set of predefined queries.

Bayes-Fuse [60], introduced an alternative model, which learns the evidence of relevance as a distribution of relevance given the search engines' rankings. The final ranked list is obtained using Bayes optimal decision rule. Another more sophisticated model for combining rankings using conditional probabilities was introduced in [66], where models of permutation were used in order to combine rankings. Cohen et al. [67] show how metasearch can be formulated as an ordering problem, and present an on-line algorithm for learning a weighted combination of ranking systems which is based on an adaptation of Freund and Schapire's Hedge algorithm [68].

Joachims [69] demonstrated a user-driven approach to metasearch. His system learns user preferences based on past activity and assigns weight to individual search engines. Thus, this system is similar to ProFusion, the main difference being that weights are assigned based on the preference of an individual user or group of users rather than search engine precision.

# Chapter 3 – Proposed Meta Search Model

## 3.1 The Proposed model

The Proposed meta search model is a novel approach for information retrieval on the World Wide Web. A simple algorithm re-ranks the initial result sets of each search engine and produces a single final result set. The re-ranking algorithm is based upon citation techniques and takes into consideration the rank that each search engine gave to its results and a gravity factor that each search engine is assigned by the opinion of the other search engines.

The results of a search engine are ranked according to their position in the result set. The first result of a search engine is the higher ranked in its result set. The proposed meta search model firstly finds the average rank that all the search engines gave to each result, using the following expression:

$$R_i = \frac{\sum_{j=1}^{n} G_j \cdot C_j R_i}{n}$$

Where
$R_i$ = Rank of page i,
$C_jR_i$ = Rank of page i, according to crawler j,
$G_j$ = Gravity of crawler j
n = number of Crawlers

Then each crawler (search engine) is given a gravity $G_j$ which is determined by the variations of the ranks that the crawler assigns to its results to the average ranks of all the crawlers to the same results. The extent of similarity of results of each crawler to the results of the other crawlers is used as the gravity factor that each crawler is assigned.

$$CM_i = \frac{\sum_{j=1}^{n} C_j R_i}{n}$$

Where $CM_i$ = Average of Ranks that crawlers gave to page i

$$V_j = \frac{\sum_{i=1}^{z} (C_j R_i - CM_i)^2}{z}$$

$V_j$ = Variation of ranks of crawler j for every page, to the average of Ranks that all crawlers gave this page.
z = number of results.

$$G_j = \frac{1}{V_j}$$

The final result set that this algorithm will provide, will be relatively similar to the results of the best search engines that where used, but it will differ in the fact that it will be bigger and properly reranked according to what all the search engines “voted”.

## 3.2 Into the proposed reranking algorithm

Although the proposed algorithm can be considered relatively simple, it is necessary to be properly analyzed and explained step by step. There is a particular order that each step must be made which allows the algorithm to produce the final result set.

We decide which search engines are to be used and the keywords that are relative to our query. The decision of the search engines to be used is rather easy; the most popular will do as they are bound to produce the most relevant results. The more search engines that are used the better, though it will result to more calculations and make the algorithm slower. There are always technical limitations and the meta search model introduced, takes them into consideration.

The keywords that are relative to our query are a difficult choice and are the decision of the user. There are many efforts in current research as to how a search engine can assist the user to select the proper keywords that are more relevant to what information the user needs. A technique that seems to be helpful is the display of popular keywords as the user is writing the fist letters or the first keywords. The keywords suggested are relevant to what the user writes. Another helpful technique is the display of corrected keywords if the keywords that the user writes can be considered misspelled. Both these techniques are used by very popular search engines such as the Google search engine [70]. Another technique is suggested in Chapter 5, which uses a word pool made from the search results and identifies the words that more frequently appear. These words can be presented as new relevant keywords to the information that the user seeks. The effectiveness of the algorithm is highly based on the keywords used and the selection of these query terms in combination to the faction of the algorithm can be a subject of future research.

### 3.2.1 Gathering the result sets from the search engines.

This is the first step of the algorithm. Let's suppose we have selected which search engines are to be used. We have also selected the keywords that we believe that are most relevant to the information we seek. Both the selection of the search engines and the selection of the keywords have no impact in the function of the algorithm but they have a major impact on the relativeness of the initial result sets, which are used to produce the final result set.

The gathering of the result set of each search engine according to the selected keywords stands no real challenge. All that is needed is good programming skills and good knowledge of the search engines used. For example some search engines will separate keywords by using the word (AND) between them and others will use the symbol (+). It is a fact that the popular search engines are very similar to they way they are used. A simple space between the keywords will do for the function AND. Other logical expressions between the keywords such as OR or NOT are handled exactly the same way in most of the popular search engines.

This algorithm focuses on producing the most relevant result set out of the initial results that the search engines provide, according to the search terms used. There is no effort to correct the search terms and re-search, though something like that can be a topic for a future research. For now our considerations are limited to the proper reranking of all the results already given to produce the most relevant final result set.

The ideal case is to use all the results that the search engines provide to a specific query. This way we can access the biggest possible fraction of the World Wide Web and provide the biggest result set. As we will see in the following paragraphs this will also result to more accurate result set, because the gravity assigned to each search engine will be more accurate. Again technical limitations, such as limited CPU power allocation, memory and connection bandwidth, may force us to use a limited result set from each search engine. Still we have the advantage of accessing the biggest fraction of the WWW to crawl the best results but not all the results will be shown and we loose in accuracy.

### 3.2.2 The average of ranks to each result

The second step of the algorithm includes the calculation of the average rank that all search engines used give to each result.

Let's suppose we use four search engines and we have gathered the first 100 results from each one. It is almost certain that the four result sets will have mutual entries. We must use a ranking system so that every search engine applies a rank to its results. We must also use the same ranking method to every result set in order to be able to compare them. That means that we don't really care about the ranking system of every search engine used. Instead we only care which results are ranked higher from the others. We can be sure that each search engine, using its own rank system has applied the higher rank to the first result, the second

higher rank to the second result and so on. The mutual ranking method that we will use is very simple. The first result of every result set takes the rank 1 the second 2 and so on. Of course the rank 1 is a higher rank than the rank 2.

In the example of four search engines with 100 results each, every search engine will rank its results from 1 to 100 the way described above. If the same result is first in the first search engine, fifth in the second, nineteenth in the third and first in the fourth, then the ranks applied will be "1","5","19"and"1". This will give the average rank of "6,5". The expression of this calculation is shown below:

$$CM_i = \frac{\sum_{j=1}^{n} C_j R_i}{n}$$

Where
$CM_i$ = Average of Ranks that crawlers gave to page i
$C_jR_i$ = Rank of page i, according to crawler j,
n = number of Crawlers

One thing that we must take into consideration when applying the ranking method to the result sets is that some or all of the search engines used will include sponsored links. These results are usually on the top of the result sets but they are not ranked higher that the other results, they are simply advertised. It is obvious that the sponsored links must be ruled out of the ranking system. A good proposition would be that the sponsored links can be showed to the user after the search in a different section than the actual result set.

Another thing that we must take into consideration is the rank that a search engine will apply to the results that the other search engines provide, but are not included in its own result set. For example let's suppose that the first result of the second search engine is not included in the first search engine. The second search engine will give to that result the rank "1" but what will be the rank of the first search engine to that result? The answer is rather difficult.

One solution could be that when a search engine has to rank a result that is not included on its own result set, this result gets a very low rank. In the example of four search engines with 100 results each, the rank could be "1000" or "1000000" (remember that the lowest number is the highest rank). After testing it is realized that a very big number as rank can result in a very low average rank. This will put these results in the very bottom of the result set which is surely wrong.

Another, more elegant solution is to rank these results with the lowest possible rank number, which no other result which is included in the search engine's result set can take. In the example of four search engines with 100 results each, the maximum rank number that a search engine will give to its results is 100. Every result that is not included in its result set will get the rank of 101 (lowest possible rank number, which no other result which is included in the search engine's result set can take). Testing proved that this solution is much more accurate. In fact the more results that a search engine provide (making the lowest possible rank number, which no other result which is included in the search engine's result set can take, bigger) the more accurate the final result set can be. This is due to the fact that the more results a search engine provide, the biggest the possibility that there will be mutual results to other search engines.

### 3.2.3 Variation of results between the search engines

This is the third step of the algorithm. The unique function that differentiates this algorithm from other reranking algorithms is the technique in the form that each search engine used cites all the others. A gravity factor, based on this technique, is applied to every search engine so that it will affect the final ranking process.

One search engine will cite another if it realizes that they have similar result sets. In fact the more similar results, the "better" the citation will be. So we need a way to measure

the similarity between result sets. Mathematics provides us with a powerful function that can make this measurement. We can measure the similarity by calculating the variation between the ranks of the results using the standard deviation method [71]. This function is shown below:

$$V_j = \frac{\sum_{i=1}^{z} (C_j R_i - CM_i)^2}{z}$$

Where
$V_j$ = Variation of ranks of crawler j for every page, to the average of Ranks that all crawlers gave this page.
z = number of results.
$CM_i$ = Average of Ranks that crawlers gave to page i
$C_jR_i$ = Rank of page i, according to crawler j,

The variation parameter V shows us the extend of variation of the ranks of the results of a crawler to the average ranks, which were calculated in the previous step. The variation parameter will be used to decide the gravity parameter of each crawler.

### 3.2.4 Assignment of a gravity factor to each crawler

In the fourth step of the algorithm we try to decide which of the search engines used are producing more relevant results than the others. There is not a sure way to measure relevancy in all the bibliography accessed. The citation technique that this algorithm is based upon suggests that each search engine decides how relevant the results of the other search engines are. This is done by using the standard deviation technique described in the previous step. When two search engines have similar ranking of the results, then these search engines cite each other as more relevant since there results are less varied. That means that the lowest the variation factor the more relevant the ranking of the results can be considered to be.

The gravity factor that will be applied to each crawler will be extracted from the following expression:

$$G_j = \frac{1}{V_j}$$

Where:
$G_j$ = Gravity of crawler j
$V_j$ = Variation of ranks of crawler j for every page, to the average of Ranks that all crawlers gave this page.

By applying a gravity factor to each search engine in the reranking process, we can significantly improve the relativity of the final result set. The ranking of the “better” search engines (the ones with the highest gravity factor) will count more in the calculation of the final result set.

### 3.2.5 The final result set

The fifth end final step is the creation of the final result set. In the previous step we have assigned a gravity factor to every crawler used. We have also gathered the ranks of the results of each search engine in the first step.We can now use the gravity factor and the ranks of the results, to produce the new average ranks of the results. The new average ranks will sort the results to the final result set. The mathematical expression used is shown below:

$$R_i = \frac{\sum_{j=1}^{n} G_j \cdot C_j R_i}{n}$$

where

$R_i$ = Rank of page i,
$C_jR_i$ = Rank of page i, according to crawler j,
$G_j$ = Gravity of crawler j
n = number of Crawlers

The summarization of the complete algorithm is shown below

**Step 1**

- Decide the query terms (keywords) and which crawlers are to be used.
- Gather the results from the crawlers
- Every crawler ranks its own results in the way that the first result takes the rank number 1 the second 2 and so on. Then each crawler ranks the results that is not included in its own result set by using the lowest possible rank number, which no other result which is included in the crawler's result set can take.

**Step 2**

Calculate the average rank number of each result using the mathematical expression:

$$CM_i = \frac{\sum_{j=1}^{n} C_j R_i}{n}$$

**Step 3**

Calculate the extend of variation between the ranks of the results of each crawler and the average ranks using the standard deviation method, with the expression:

$$V_j = \frac{\sum_{i=1}^{z} (C_j R_i - CM_i)^2}{z}$$

**Step 4**

Calculate the gravity factor that will be applied to each crawler using the following expression:

$$G_j = \frac{1}{V_j}$$

**Step 5**

Use the gravity factor and the ranks of the results, to produce the new average ranks of the results and sort the final result set. The lowest rank number is the highest rank. The final average ranks are calculated with the expression:

$$R_i = \frac{\sum_{j=1}^{n} G_j \cdot C_j R_i}{n}$$

## *3.3 Client side proposition*

The meta search model introduced in this document is designed in a way that it is possible to be implemented on the client side. There are many advantages for a meta search engine to be running on the client side and they are described in the following lines. The same centralised implementation, that almost all the popular meta search engines use, can be also easily used. Client side implementation means that the gathering of the initial result sets, the execution of the reranking algorithm and the presentation of the final result set, occurs in the user's personal computer. The centralised implementation means that the gathering of the initial result sets and the execution of the reranking algorithm occurs in dedicated servers used by many users simultaneously, living only the presentation of the final result set to the user's personal computer.

Using the client side implementation results in more CPU power allocation and more memory available for the reranking algorithm to run. The whole process of the meta search

can be executed much faster and it is failure proof. The case of the centralised implementation uses dedicated servers which, although faster than a simple personal computer, have limited capabilities when serving a big number of users simultaneously. Also if the dedicated servers fail, then a big number of users can not be served.

It would not be possible for a search engine to run on the client side. A simple personal computer have too limited capabilities to constantly crawl through the World Wide Web, download web pages, create databases of all the web pages downloaded, rank them, create lexicons e.t.c. A meta search engine on the other hand, and especially the proposed model, can be easily implemented on a simple personal computer because it uses initial result sets of search engines, which consists only by the URL's of pages. It crawls directly only the search engines used not the entire World Wide Web. The most time-consuming part of the proposed model is the execution of the reranking algorithm which is much faster made on the client side. Even if all the web pages contained in the initial result sets have to be downloaded, in future adaptations of the proposed model, it is still a task that a simple personal computer with broadband internet connection can handle.

Also meta search engine implemented on the client side can be very easily and effectively personalised. It can have means of adaptation to the user needs, such as scheduled automated searches, use of the user's local language, predefined use of standard non popular search engines, use of local information sources, user defined representation of the final result set which can include extensive use of graphics, e.t.c.

Another advantage of the client side implementation is that it is considerably less expensive economically. There is no need for dedicated servers with all the expenses that this leads to.

# Chapter 4 – Experimental Results and Evaluation

## 4.1 Testing the algorithm with imaginary initial result sets

Before evaluating the proposed algorithm in the World Wide Web with real initial result sets, we test it with imaginary ones. The purpose of these tests is to see the behavior of the reranking algorithm over extreme possibilities like the case that all the search engines used will provide completely different initial results or the case that all the search engines used provide exactly the same initial results.

The first test includes three search engines giving the same results and one giving the exactly opposite as shown in Table 4.1.1. We use four search engines, crawler A, B, C, and D and we run the algorithm exactly as described in the previous chapter. The results are web pages and they are ranked as already described. In the table we can see the average rank of every page according to the ranking of the crawlers and the final rank that the reranking algorithm assigns. We can also see the gravity assigned by the algorithm to each crawler.

| | Crawler A | Crawler B | Crawler C | Crawler D | Average Rank | Final Rank |
|---|---|---|---|---|---|---|
| gravity | 0,007501 | 0,007501 | 0,007501 | 0,000833 | | |
| page 1 | 1 | 1 | 1 | 80 | 20,75 | 0,022295 |
| page 2 | 2 | 2 | 2 | 79 | 21,25 | 0,027713 |
| page 3 | 3 | 3 | 3 | 78 | 21,75 | 0,03313 |
| page 4 | 4 | 4 | 4 | 77 | 22,25 | 0,038548 |
| page 5 | 5 | 5 | 5 | 76 | 22,75 | 0,043965 |
| page 6 | 6 | 6 | 6 | 75 | 23,25 | 0,049383 |
| page 7 | 7 | 7 | 7 | 74 | 23,75 | 0,0548 |
| page 8 | 8 | 8 | 8 | 73 | 24,25 | 0,060218 |
| page 9 | 9 | 9 | 9 | 72 | 24,75 | 0,065635 |
| page 10 | 10 | 10 | 10 | 71 | 25,25 | 0,071053 |
| page 11 | 11 | 11 | 11 | 70 | 25,75 | 0,07647 |
| page 12 | 12 | 12 | 12 | 69 | 26,25 | 0,081888 |
| page 13 | 13 | 13 | 13 | 68 | 26,75 | 0,087305 |
| page 14 | 14 | 14 | 14 | 67 | 27,25 | 0,092723 |
| page 15 | 15 | 15 | 15 | 66 | 27,75 | 0,09814 |
| page 16 | 16 | 16 | 16 | 65 | 28,25 | 0,103558 |
| page 17 | 17 | 17 | 17 | 64 | 28,75 | 0,108975 |
| page 18 | 18 | 18 | 18 | 63 | 29,25 | 0,114393 |
| page 19 | 19 | 19 | 19 | 62 | 29,75 | 0,11981 |
| page 20 | 20 | 20 | 20 | 61 | 30,25 | 0,125228 |
| page 21 | 21 | 21 | 21 | 60 | 30,75 | 0,130645 |
| page 22 | 22 | 22 | 22 | 59 | 31,25 | 0,136063 |
| page 23 | 23 | 23 | 23 | 58 | 31,75 | 0,14148 |
| page 24 | 24 | 24 | 24 | 57 | 32,25 | 0,146898 |
| page 25 | 25 | 25 | 25 | 56 | 32,75 | 0,152315 |
| page 26 | 26 | 26 | 26 | 55 | 33,25 | 0,157733 |
| page 27 | 27 | 27 | 27 | 54 | 33,75 | 0,16315 |
| page 28 | 28 | 28 | 28 | 53 | 34,25 | 0,168568 |
| page 29 | 29 | 29 | 29 | 52 | 34,75 | 0,173986 |
| page 30 | 30 | 30 | 30 | 51 | 35,25 | 0,179403 |
| page 31 | 31 | 31 | 31 | 50 | 35,75 | 0,184821 |
| page 32 | 32 | 32 | 32 | 49 | 36,25 | 0,190238 |
| page 33 | 33 | 33 | 33 | 48 | 36,75 | 0,195656 |
| page 34 | 34 | 34 | 34 | 47 | 37,25 | 0,201073 |
| page 35 | 35 | 35 | 35 | 46 | 37,75 | 0,206491 |
| page 36 | 36 | 36 | 36 | 45 | 38,25 | 0,211908 |
| page 37 | 37 | 37 | 37 | 44 | 38,75 | 0,217326 |
| page 38 | 38 | 38 | 38 | 43 | 39,25 | 0,222743 |
| page 39 | 39 | 39 | 39 | 42 | 39,75 | 0,228161 |

| | | | | | | |
|---|---|---|---|---|---|---|
| page 40 | 40 | 40 | 40 | 41 | 40,25 | 0,233578 |
| page 41 | 41 | 41 | 41 | 40 | 40,75 | 0,238996 |
| page 42 | 42 | 42 | 42 | 39 | 41,25 | 0,244413 |
| page 43 | 43 | 43 | 43 | 38 | 41,75 | 0,249831 |
| page 44 | 44 | 44 | 44 | 37 | 42,25 | 0,255248 |
| page 45 | 45 | 45 | 45 | 36 | 42,75 | 0,260666 |
| page 46 | 46 | 46 | 46 | 35 | 43,25 | 0,266083 |
| page 47 | 47 | 47 | 47 | 34 | 43,75 | 0,271501 |
| page 48 | 48 | 48 | 48 | 33 | 44,25 | 0,276918 |
| page 49 | 49 | 49 | 49 | 32 | 44,75 | 0,282336 |
| page 50 | 50 | 50 | 50 | 31 | 45,25 | 0,287753 |
| page 51 | 51 | 51 | 51 | 30 | 45,75 | 0,293171 |
| page 52 | 52 | 52 | 52 | 29 | 46,25 | 0,298588 |
| page 53 | 53 | 53 | 53 | 28 | 46,75 | 0,304006 |
| page 54 | 54 | 54 | 54 | 27 | 47,25 | 0,309423 |
| page 55 | 55 | 55 | 55 | 26 | 47,75 | 0,314841 |
| page 56 | 56 | 56 | 56 | 25 | 48,25 | 0,320258 |
| page 57 | 57 | 57 | 57 | 24 | 48,75 | 0,325676 |
| page 58 | 58 | 58 | 58 | 23 | 49,25 | 0,331093 |
| page 59 | 59 | 59 | 59 | 22 | 49,75 | 0,336511 |
| page 60 | 60 | 60 | 60 | 21 | 50,25 | 0,341928 |
| page 61 | 61 | 61 | 61 | 20 | 50,75 | 0,347346 |
| page 62 | 62 | 62 | 62 | 19 | 51,25 | 0,352763 |
| page 63 | 63 | 63 | 63 | 18 | 51,75 | 0,358181 |
| page 64 | 64 | 64 | 64 | 17 | 52,25 | 0,363598 |
| page 65 | 65 | 65 | 65 | 16 | 52,75 | 0,369016 |
| page 66 | 66 | 66 | 66 | 15 | 53,25 | 0,374434 |
| page 67 | 67 | 67 | 67 | 14 | 53,75 | 0,379851 |
| page 68 | 68 | 68 | 68 | 13 | 54,25 | 0,385269 |
| page 69 | 69 | 69 | 69 | 12 | 54,75 | 0,390686 |
| page 70 | 70 | 70 | 70 | 11 | 55,25 | 0,396104 |
| page 71 | 71 | 71 | 71 | 10 | 55,75 | 0,401521 |
| page 72 | 72 | 72 | 72 | 9 | 56,25 | 0,406939 |
| page 73 | 73 | 73 | 73 | 8 | 56,75 | 0,412356 |
| page 74 | 74 | 74 | 74 | 7 | 57,25 | 0,417774 |
| page 75 | 75 | 75 | 75 | 6 | 57,75 | 0,423191 |
| page 76 | 76 | 76 | 76 | 5 | 58,25 | 0,428609 |
| page 77 | 77 | 77 | 77 | 4 | 58,75 | 0,434026 |
| page 78 | 78 | 78 | 78 | 3 | 59,25 | 0,439444 |
| page 79 | 79 | 79 | 79 | 2 | 59,75 | 0,444861 |
| page 80 | 80 | 80 | 80 | 1 | 60,25 | 0,450279 |

Table 4.1.1 – Three search engines gives the same results and one gives the exactly opposite

In Table 4.1.1, we have three crawlers giving exactly the same ranks and the fourth giving the exactly opposite. We expect that the three crawlers will be assigned the same high gravity and the fourth a low gravity. In fact, as we see in the second row of Table 4.1.1 the first tree crawlers get a gravity of 0,007501 and the fourth a gravity of 0,000833. We also expect that the page 1 we get the highest final rank, the page 2 the second highest and so on. That is also true as we see in Table 4.1.1. Remember that the lowest final rank number is the highest rank. Another thing to notice is that if we sort the pages by the final rank or with the average rank we get the same final result set. This is also an indication that the final result set is close to the most relevant one but if the algorithm works as expected, the average ranks and final ranks will differ in more complicated initial result sets. The average rank itself can not be used to sort the final result set in any case because it is very probable and common that two pages get exactly the same average rank. The gravity factor is what makes this algorithm efficient and makes almost sure that two pages cannot get the same final rank.

There is a case that we expect to see the same final rank to several pages. If the all the crawlers give completely different and not overlapping result sets, in other words if no page from one crawler is included in another crawler's result set, then all the crawlers will be assigned the same gravity and it is easy to realise that we will get a final result set with pages having mutual final rank. This case is shown in Table 4.1.2. We use 20 pages from every crawler. Every page that is not included in the crawlers initial result set is set to the smallest possible rank number that the crawlers haven't assigned in their own pages, as described in the previous chapter (in this case the rank number is 21).

| | Crawler A | Crawler B | Crawler C | Crawler D | Average Rank | Final Rank |
|---|---|---|---|---|---|---|
| gravity | 0,037166 | 0,037166 | 0,037166 | 0,037166 | | |
| page 1 | 1 | 21 | 21 | 21 | 16 | 0,594657 |
| page 2 | 2 | 21 | 21 | 21 | 16,25 | 0,603949 |
| page 3 | 3 | 21 | 21 | 21 | 16,5 | 0,61324 |
| page 4 | 4 | 21 | 21 | 21 | 16,75 | 0,622532 |
| page 5 | 5 | 21 | 21 | 21 | 17 | 0,631823 |
| page 6 | 6 | 21 | 21 | 21 | 17,25 | 0,641115 |
| page 7 | 7 | 21 | 21 | 21 | 17,5 | 0,650407 |
| page 8 | 8 | 21 | 21 | 21 | 17,75 | 0,659698 |
| page 9 | 9 | 21 | 21 | 21 | 18 | 0,66899 |
| page 10 | 10 | 21 | 21 | 21 | 18,25 | 0,678281 |
| page 11 | 11 | 21 | 21 | 21 | 18,5 | 0,687573 |
| page 12 | 12 | 21 | 21 | 21 | 18,75 | 0,696864 |
| page 13 | 13 | 21 | 21 | 21 | 19 | 0,706156 |
| page 14 | 14 | 21 | 21 | 21 | 19,25 | 0,715447 |
| page 15 | 15 | 21 | 21 | 21 | 19,5 | 0,724739 |
| page 16 | 16 | 21 | 21 | 21 | 19,75 | 0,73403 |
| page 17 | 17 | 21 | 21 | 21 | 20 | 0,743322 |
| page 18 | 18 | 21 | 21 | 21 | 20,25 | 0,752613 |
| page 19 | 19 | 21 | 21 | 21 | 20,5 | 0,761905 |
| page 20 | 20 | 21 | 21 | 21 | 20,75 | 0,771196 |
| page 21 | 21 | 1 | 21 | 21 | 16 | 0,594657 |
| page 22 | 21 | 2 | 21 | 21 | 16,25 | 0,603949 |
| page 23 | 21 | 3 | 21 | 21 | 16,5 | 0,61324 |
| page 24 | 21 | 4 | 21 | 21 | 16,75 | 0,622532 |
| page 25 | 21 | 5 | 21 | 21 | 17 | 0,631823 |
| page 26 | 21 | 6 | 21 | 21 | 17,25 | 0,641115 |
| page 27 | 21 | 7 | 21 | 21 | 17,5 | 0,650407 |
| page 28 | 21 | 8 | 21 | 21 | 17,75 | 0,659698 |
| page 29 | 21 | 9 | 21 | 21 | 18 | 0,66899 |
| page 30 | 21 | 10 | 21 | 21 | 18,25 | 0,678281 |
| page 31 | 21 | 11 | 21 | 21 | 18,5 | 0,687573 |
| page 32 | 21 | 12 | 21 | 21 | 18,75 | 0,696864 |
| page 33 | 21 | 13 | 21 | 21 | 19 | 0,706156 |
| page 34 | 21 | 14 | 21 | 21 | 19,25 | 0,715447 |
| page 35 | 21 | 15 | 21 | 21 | 19,5 | 0,724739 |
| page 36 | 21 | 16 | 21 | 21 | 19,75 | 0,73403 |
| page 37 | 21 | 17 | 21 | 21 | 20 | 0,743322 |
| page 38 | 21 | 18 | 21 | 21 | 20,25 | 0,752613 |
| page 39 | 21 | 19 | 21 | 21 | 20,5 | 0,761905 |
| page 40 | 21 | 20 | 21 | 21 | 20,75 | 0,771196 |
| page 41 | 21 | 21 | 1 | 21 | 16 | 0,594657 |
| page 42 | 21 | 21 | 2 | 21 | 16,25 | 0,603949 |
| page 43 | 21 | 21 | 3 | 21 | 16,5 | 0,61324 |
| page 44 | 21 | 21 | 4 | 21 | 16,75 | 0,622532 |
| page 45 | 21 | 21 | 5 | 21 | 17 | 0,631823 |
| page 46 | 21 | 21 | 6 | 21 | 17,25 | 0,641115 |

| | | | | | | |
|---|---|---|---|---|---|---|
| page 47 | 21 | 21 | 7 | 21 | 17,5 | 0,650407 |
| page 48 | 21 | 21 | 8 | 21 | 17,75 | 0,659698 |
| page 49 | 21 | 21 | 9 | 21 | 18 | 0,66899 |
| page 50 | 21 | 21 | 10 | 21 | 18,25 | 0,678281 |
| page 51 | 21 | 21 | 11 | 21 | 18,5 | 0,687573 |
| page 52 | 21 | 21 | 12 | 21 | 18,75 | 0,696864 |
| page 53 | 21 | 21 | 13 | 21 | 19 | 0,706156 |
| page 54 | 21 | 21 | 14 | 21 | 19,25 | 0,715447 |
| page 55 | 21 | 21 | 15 | 21 | 19,5 | 0,724739 |
| page 56 | 21 | 21 | 16 | 21 | 19,75 | 0,73403 |
| page 57 | 21 | 21 | 17 | 21 | 20 | 0,743322 |
| page 58 | 21 | 21 | 18 | 21 | 20,25 | 0,752613 |
| page 59 | 21 | 21 | 19 | 21 | 20,5 | 0,761905 |
| page 60 | 21 | 21 | 20 | 21 | 20,75 | 0,771196 |
| page 61 | 21 | 21 | 21 | 1 | 16 | 0,594657 |
| page 62 | 21 | 21 | 21 | 2 | 16,25 | 0,603949 |
| page 63 | 21 | 21 | 21 | 3 | 16,5 | 0,61324 |
| page 64 | 21 | 21 | 21 | 4 | 16,75 | 0,622532 |
| page 65 | 21 | 21 | 21 | 5 | 17 | 0,631823 |
| page 66 | 21 | 21 | 21 | 6 | 17,25 | 0,641115 |
| page 67 | 21 | 21 | 21 | 7 | 17,5 | 0,650407 |
| page 68 | 21 | 21 | 21 | 8 | 17,75 | 0,659698 |
| page 69 | 21 | 21 | 21 | 9 | 18 | 0,66899 |
| page 70 | 21 | 21 | 21 | 10 | 18,25 | 0,678281 |
| page 71 | 21 | 21 | 21 | 11 | 18,5 | 0,687573 |
| page 72 | 21 | 21 | 21 | 12 | 18,75 | 0,696864 |
| page 73 | 21 | 21 | 21 | 13 | 19 | 0,706156 |
| page 74 | 21 | 21 | 21 | 14 | 19,25 | 0,715447 |
| page 75 | 21 | 21 | 21 | 15 | 19,5 | 0,724739 |
| page 76 | 21 | 21 | 21 | 16 | 19,75 | 0,73403 |
| page 77 | 21 | 21 | 21 | 17 | 20 | 0,743322 |
| page 78 | 21 | 21 | 21 | 18 | 20,25 | 0,752613 |
| page 79 | 21 | 21 | 21 | 19 | 20,5 | 0,761905 |
| page 80 | 21 | 21 | 21 | 20 | 20,75 | 0,771196 |

Table 4.1.2 – All the crawlers give completely different and not overlapping result sets

As we can see in Table 4.1.2 there are mutual final ranks for every four pages. The pages 1, 21, 41 and 61, the pages 2, 22, 42 and 62, and so on, have the same final ranks. That is due to the fact that these quadruplets have the same average rank in combination to the fact that all the crawlers are assigned the same gravity. There is no similarity, in any way, in the initial result sets, that the algorithm can be based upon to be able to assign different gravities. The slightest change is the initial ranks, the slightest similarity in any way, will lead to slightly different gravities, and there will be no mutual final ranks. In the very rare and extreme possibility that the initial results between the crawlers are completely different, the algorithm is not able to determine different gravities and is unable to sort the final result set.

In this extreme case, the algorithm will have to count on previous searches. Previous searches can be used to extract an average gravity for every crawler. In real life examples this extreme will probably never happen but the gravities must be stored after each search just to be sure.

Another extreme is the case that all the initial result sets are exactly the same (Table 4.1.3). In this case every crawler gives us the same exactly result sets. No variation at all in the pages or the rankings. This results in having a zero standard deviation and to calculate the gravity we divide by zero. In this case the algorithm simply fails to give a final result set. This is logical, due to the fact that there is no need for a meta search engine and for a reranking algorithm if all the crawlers give exactly the same pages and rankings. The answer to this

problem is simple. The initial results sets and the final result set are the same. There is no need for any calculation or reranking.

| | Crawler A | Crawler B | Crawler C | Crawler D | Average Rank | Final Rank |
|---|---|---|---|---|---|---|
| gravity | ∞ | ∞ | ∞ | ∞ | | |
| page 1 | 1 | 1 | 1 | 1 | 1 | ∞ |
| page 2 | 2 | 2 | 2 | 2 | 2 | ∞ |
| page 3 | 3 | 3 | 3 | 3 | 3 | ∞ |
| page 4 | 4 | 4 | 4 | 4 | 4 | ∞ |
| page 5 | 5 | 5 | 5 | 5 | 5 | ∞ |
| page 6 | 6 | 6 | 6 | 6 | 6 | ∞ |
| page 7 | 7 | 7 | 7 | 7 | 7 | ∞ |
| page 8 | 8 | 8 | 8 | 8 | 8 | ∞ |
| page 9 | 9 | 9 | 9 | 9 | 9 | ∞ |
| page 10 | 10 | 10 | 10 | 10 | 10 | ∞ |
| page 11 | 11 | 11 | 11 | 11 | 11 | ∞ |
| page 12 | 12 | 12 | 12 | 12 | 12 | ∞ |
| page 13 | 13 | 13 | 13 | 13 | 13 | ∞ |
| page 14 | 14 | 14 | 14 | 14 | 14 | ∞ |
| page 15 | 15 | 15 | 15 | 15 | 15 | ∞ |
| page 16 | 16 | 16 | 16 | 16 | 16 | ∞ |
| page 17 | 17 | 17 | 17 | 17 | 17 | ∞ |
| page 18 | 18 | 18 | 18 | 18 | 18 | ∞ |
| page 19 | 19 | 19 | 19 | 19 | 19 | ∞ |
| page 20 | 20 | 20 | 20 | 20 | 20 | ∞ |
| page 21 | 21 | 21 | 21 | 21 | 21 | ∞ |
| page 22 | 22 | 22 | 22 | 22 | 22 | ∞ |
| page 23 | 23 | 23 | 23 | 23 | 23 | ∞ |
| page 24 | 24 | 24 | 24 | 24 | 24 | ∞ |
| page 25 | 25 | 25 | 25 | 25 | 25 | ∞ |
| page 26 | 26 | 26 | 26 | 26 | 26 | ∞ |
| page 27 | 27 | 27 | 27 | 27 | 27 | ∞ |
| page 28 | 28 | 28 | 28 | 28 | 28 | ∞ |
| page 29 | 29 | 29 | 29 | 29 | 29 | ∞ |
| page 30 | 30 | 30 | 30 | 30 | 30 | ∞ |
| page 31 | 31 | 31 | 31 | 31 | 31 | ∞ |
| page 32 | 32 | 32 | 32 | 32 | 32 | ∞ |
| page 33 | 33 | 33 | 33 | 33 | 33 | ∞ |
| page 34 | 34 | 34 | 34 | 34 | 34 | ∞ |
| page 35 | 35 | 35 | 35 | 35 | 35 | ∞ |
| page 36 | 36 | 36 | 36 | 36 | 36 | ∞ |
| page 37 | 37 | 37 | 37 | 37 | 37 | ∞ |
| page 38 | 38 | 38 | 38 | 38 | 38 | ∞ |
| page 39 | 39 | 39 | 39 | 39 | 39 | ∞ |
| page 40 | 40 | 40 | 40 | 40 | 40 | ∞ |
| page 41 | 41 | 41 | 41 | 41 | 41 | ∞ |
| page 42 | 42 | 42 | 42 | 42 | 42 | ∞ |
| page 43 | 43 | 43 | 43 | 43 | 43 | ∞ |
| page 44 | 44 | 44 | 44 | 44 | 44 | ∞ |
| page 45 | 45 | 45 | 45 | 45 | 45 | ∞ |
| page 46 | 46 | 46 | 46 | 46 | 46 | ∞ |
| page 47 | 47 | 47 | 47 | 47 | 47 | ∞ |
| page 48 | 48 | 48 | 48 | 48 | 48 | ∞ |
| page 49 | 49 | 49 | 49 | 49 | 49 | ∞ |
| page 50 | 50 | 50 | 50 | 50 | 50 | ∞ |
| page 51 | 51 | 51 | 51 | 51 | 51 | ∞ |

| | | | | | | |
|---|---|---|---|---|---|---|
| page 52 | 52 | 52 | 52 | 52 | 52 | ∞ |
| page 53 | 53 | 53 | 53 | 53 | 53 | ∞ |
| page 54 | 54 | 54 | 54 | 54 | 54 | ∞ |
| page 55 | 55 | 55 | 55 | 55 | 55 | ∞ |
| page 56 | 56 | 56 | 56 | 56 | 56 | ∞ |
| page 57 | 57 | 57 | 57 | 57 | 57 | ∞ |
| page 58 | 58 | 58 | 58 | 58 | 58 | ∞ |
| page 59 | 59 | 59 | 59 | 59 | 59 | ∞ |
| page 60 | 60 | 60 | 60 | 60 | 60 | ∞ |
| page 61 | 61 | 61 | 61 | 61 | 61 | ∞ |
| page 62 | 62 | 62 | 62 | 62 | 62 | ∞ |
| page 63 | 63 | 63 | 63 | 63 | 63 | ∞ |
| page 64 | 64 | 64 | 64 | 64 | 64 | ∞ |
| page 65 | 65 | 65 | 65 | 65 | 65 | ∞ |
| page 66 | 66 | 66 | 66 | 66 | 66 | ∞ |
| page 67 | 67 | 67 | 67 | 67 | 67 | ∞ |
| page 68 | 68 | 68 | 68 | 68 | 68 | ∞ |
| page 69 | 69 | 69 | 69 | 69 | 69 | ∞ |
| page 70 | 70 | 70 | 70 | 70 | 70 | ∞ |
| page 71 | 71 | 71 | 71 | 71 | 71 | ∞ |
| page 72 | 72 | 72 | 72 | 72 | 72 | ∞ |
| page 73 | 73 | 73 | 73 | 73 | 73 | ∞ |
| page 74 | 74 | 74 | 74 | 74 | 74 | ∞ |
| page 75 | 75 | 75 | 75 | 75 | 75 | ∞ |
| page 76 | 76 | 76 | 76 | 76 | 76 | ∞ |
| page 77 | 77 | 77 | 77 | 77 | 77 | ∞ |
| page 78 | 78 | 78 | 78 | 78 | 78 | ∞ |
| page 79 | 79 | 79 | 79 | 79 | 79 | ∞ |
| page 80 | 80 | 80 | 80 | 80 | 80 | ∞ |

Table 4.1.3 – Every crawler gives us the same exactly initial result sets

Again the slightest change in any of the initial result sets will be enough for the algorithm to properly produce a final result set as shown in Table 4.1.4 (see third row). There will be no division by zero in any of the calculations

| | Crawler A | Crawler B | Crawler C | Crawler D | Average Rank | Final Rank |
|---|---|---|---|---|---|---|
| gravity | 640 | 71,11111 | 640 | 640 | | |
| page 1 | 1 | 2 | 1 | 1 | 1,25 | 515,5556 |
| page 2 | 2 | 1 | 2 | 2 | 1,75 | 977,7778 |
| page 3 | 3 | 3 | 3 | 3 | 3 | 1493,333 |
| page 4 | 4 | 4 | 4 | 4 | 4 | 1991,111 |
| page 5 | 5 | 5 | 5 | 5 | 5 | 2488,889 |
| page 6 | 6 | 6 | 6 | 6 | 6 | 2986,667 |
| page 7 | 7 | 7 | 7 | 7 | 7 | 3484,444 |
| page 8 | 8 | 8 | 8 | 8 | 8 | 3982,222 |
| page 9 | 9 | 9 | 9 | 9 | 9 | 4480 |
| page 10 | 10 | 10 | 10 | 10 | 10 | 4977,778 |
| page 11 | 11 | 11 | 11 | 11 | 11 | 5475,556 |
| page 12 | 12 | 12 | 12 | 12 | 12 | 5973,333 |
| page 13 | 13 | 13 | 13 | 13 | 13 | 6471,111 |
| page 14 | 14 | 14 | 14 | 14 | 14 | 6968,889 |
| page 15 | 15 | 15 | 15 | 15 | 15 | 7466,667 |
| page 16 | 16 | 16 | 16 | 16 | 16 | 7964,444 |
| page 17 | 17 | 17 | 17 | 17 | 17 | 8462,222 |
| page 18 | 18 | 18 | 18 | 18 | 18 | 8960 |
| page 19 | 19 | 19 | 19 | 19 | 19 | 9457,778 |

| | | | | | | |
|---|---|---|---|---|---|---|
| page 20 | 20 | 20 | 20 | 20 | 20 | 9955,556 |
| page 21 | 21 | 21 | 21 | 21 | 21 | 10453,33 |
| page 22 | 22 | 22 | 22 | 22 | 22 | 10951,11 |
| page 23 | 23 | 23 | 23 | 23 | 23 | 11448,89 |
| page 24 | 24 | 24 | 24 | 24 | 24 | 11946,67 |
| page 25 | 25 | 25 | 25 | 25 | 25 | 12444,44 |
| page 26 | 26 | 26 | 26 | 26 | 26 | 12942,22 |
| page 27 | 27 | 27 | 27 | 27 | 27 | 13440 |
| page 28 | 28 | 28 | 28 | 28 | 28 | 13937,78 |
| page 29 | 29 | 29 | 29 | 29 | 29 | 14435,56 |
| page 30 | 30 | 30 | 30 | 30 | 30 | 14933,33 |
| page 31 | 31 | 31 | 31 | 31 | 31 | 15431,11 |
| page 32 | 32 | 32 | 32 | 32 | 32 | 15928,89 |
| page 33 | 33 | 33 | 33 | 33 | 33 | 16426,67 |
| page 34 | 34 | 34 | 34 | 34 | 34 | 16924,44 |
| page 35 | 35 | 35 | 35 | 35 | 35 | 17422,22 |
| page 36 | 36 | 36 | 36 | 36 | 36 | 17920 |
| page 37 | 37 | 37 | 37 | 37 | 37 | 18417,78 |
| page 38 | 38 | 38 | 38 | 38 | 38 | 18915,56 |
| page 39 | 39 | 39 | 39 | 39 | 39 | 19413,33 |
| page 40 | 40 | 40 | 40 | 40 | 40 | 19911,11 |
| page 41 | 41 | 41 | 41 | 41 | 41 | 20408,89 |
| page 42 | 42 | 42 | 42 | 42 | 42 | 20906,67 |
| page 43 | 43 | 43 | 43 | 43 | 43 | 21404,44 |
| page 44 | 44 | 44 | 44 | 44 | 44 | 21902,22 |
| page 45 | 45 | 45 | 45 | 45 | 45 | 22400 |
| page 46 | 46 | 46 | 46 | 46 | 46 | 22897,78 |
| page 47 | 47 | 47 | 47 | 47 | 47 | 23395,56 |
| page 48 | 48 | 48 | 48 | 48 | 48 | 23893,33 |
| page 49 | 49 | 49 | 49 | 49 | 49 | 24391,11 |
| page 50 | 50 | 50 | 50 | 50 | 50 | 24888,89 |
| page 51 | 51 | 51 | 51 | 51 | 51 | 25386,67 |
| page 52 | 52 | 52 | 52 | 52 | 52 | 25884,44 |
| page 53 | 53 | 53 | 53 | 53 | 53 | 26382,22 |
| page 54 | 54 | 54 | 54 | 54 | 54 | 26880 |
| page 55 | 55 | 55 | 55 | 55 | 55 | 27377,78 |
| page 56 | 56 | 56 | 56 | 56 | 56 | 27875,56 |
| page 57 | 57 | 57 | 57 | 57 | 57 | 28373,33 |
| page 58 | 58 | 58 | 58 | 58 | 58 | 28871,11 |
| page 59 | 59 | 59 | 59 | 59 | 59 | 29368,89 |
| page 60 | 60 | 60 | 60 | 60 | 60 | 29866,67 |
| page 61 | 61 | 61 | 61 | 61 | 61 | 30364,44 |
| page 62 | 62 | 62 | 62 | 62 | 62 | 30862,22 |
| page 63 | 63 | 63 | 63 | 63 | 63 | 31360 |
| page 64 | 64 | 64 | 64 | 64 | 64 | 31857,78 |
| page 65 | 65 | 65 | 65 | 65 | 65 | 32355,56 |
| page 66 | 66 | 66 | 66 | 66 | 66 | 32853,33 |
| page 67 | 67 | 67 | 67 | 67 | 67 | 33351,11 |
| page 68 | 68 | 68 | 68 | 68 | 68 | 33848,89 |
| page 69 | 69 | 69 | 69 | 69 | 69 | 34346,67 |
| page 70 | 70 | 70 | 70 | 70 | 70 | 34844,44 |
| page 71 | 71 | 71 | 71 | 71 | 71 | 35342,22 |
| page 72 | 72 | 72 | 72 | 72 | 72 | 35840 |
| page 73 | 73 | 73 | 73 | 73 | 73 | 36337,78 |
| page 74 | 74 | 74 | 74 | 74 | 74 | 36835,56 |
| page 75 | 75 | 75 | 75 | 75 | 75 | 37333,33 |
| page 76 | 76 | 76 | 76 | 76 | 76 | 37831,11 |
| page 77 | 77 | 77 | 77 | 77 | 77 | 38328,89 |

| | | | | | | |
|---|---|---|---|---|---|---|
| page 78 | 78 | 78 | 78 | 78 | 78 | 38826,67 |
| page 79 | 79 | 79 | 79 | 79 | 79 | 39324,44 |
| page 80 | 80 | 80 | 80 | 80 | 80 | 39822,22 |

Table 4.1.4 – Every crawler gives similar initial result sets

We can see in the second row of Table 4.1.4 that the crawler B is now assigned a lower gravity than the other crawlers. That is the expected outcome since the only crawler that variate from the others is the second.

There is another very extreme possibility that all the pages get exactly the same average rank and all the crawlers get the same gravity too, as shown in Table 4.1.5

| | Crawler A | Crawler B | Crawler C | Crawler D | Average Rank | Final Rank |
|---|---|---|---|---|---|---|
| gravity | 0,001875 | 0,001875 | 0,001875 | 0,001875 | | |
| page 1 | 1 | 80 | 1 | 80 | 40,5 | 0,075949 |
| page 2 | 2 | 79 | 2 | 79 | 40,5 | 0,075949 |
| page 3 | 3 | 78 | 3 | 78 | 40,5 | 0,075949 |
| page 4 | 4 | 77 | 4 | 77 | 40,5 | 0,075949 |
| page 5 | 5 | 76 | 5 | 76 | 40,5 | 0,075949 |
| page 6 | 6 | 75 | 6 | 75 | 40,5 | 0,075949 |
| page 7 | 7 | 74 | 7 | 74 | 40,5 | 0,075949 |
| page 8 | 8 | 73 | 8 | 73 | 40,5 | 0,075949 |
| page 9 | 9 | 72 | 9 | 72 | 40,5 | 0,075949 |
| page 10 | 10 | 71 | 10 | 71 | 40,5 | 0,075949 |
| page 11 | 11 | 70 | 11 | 70 | 40,5 | 0,075949 |
| page 12 | 12 | 69 | 12 | 69 | 40,5 | 0,075949 |
| page 13 | 13 | 68 | 13 | 68 | 40,5 | 0,075949 |
| page 14 | 14 | 67 | 14 | 67 | 40,5 | 0,075949 |
| page 15 | 15 | 66 | 15 | 66 | 40,5 | 0,075949 |
| page 16 | 16 | 65 | 16 | 65 | 40,5 | 0,075949 |
| page 17 | 17 | 64 | 17 | 64 | 40,5 | 0,075949 |
| page 18 | 18 | 63 | 18 | 63 | 40,5 | 0,075949 |
| page 19 | 19 | 62 | 19 | 62 | 40,5 | 0,075949 |
| page 20 | 20 | 61 | 20 | 61 | 40,5 | 0,075949 |
| page 21 | 21 | 60 | 21 | 60 | 40,5 | 0,075949 |
| page 22 | 22 | 59 | 22 | 59 | 40,5 | 0,075949 |
| page 23 | 23 | 58 | 23 | 58 | 40,5 | 0,075949 |
| page 24 | 24 | 57 | 24 | 57 | 40,5 | 0,075949 |
| page 25 | 25 | 56 | 25 | 56 | 40,5 | 0,075949 |
| page 26 | 26 | 55 | 26 | 55 | 40,5 | 0,075949 |
| page 27 | 27 | 54 | 27 | 54 | 40,5 | 0,075949 |
| page 28 | 28 | 53 | 28 | 53 | 40,5 | 0,075949 |
| page 29 | 29 | 52 | 29 | 52 | 40,5 | 0,075949 |
| page 30 | 30 | 51 | 30 | 51 | 40,5 | 0,075949 |
| page 31 | 31 | 50 | 31 | 50 | 40,5 | 0,075949 |
| page 32 | 32 | 49 | 32 | 49 | 40,5 | 0,075949 |
| page 33 | 33 | 48 | 33 | 48 | 40,5 | 0,075949 |
| page 34 | 34 | 47 | 34 | 47 | 40,5 | 0,075949 |
| page 35 | 35 | 46 | 35 | 46 | 40,5 | 0,075949 |
| page 36 | 36 | 45 | 36 | 45 | 40,5 | 0,075949 |
| page 37 | 37 | 44 | 37 | 44 | 40,5 | 0,075949 |
| page 38 | 38 | 43 | 38 | 43 | 40,5 | 0,075949 |
| page 39 | 39 | 42 | 39 | 42 | 40,5 | 0,075949 |
| page 40 | 40 | 41 | 40 | 41 | 40,5 | 0,075949 |
| page 41 | 41 | 40 | 41 | 40 | 40,5 | 0,075949 |
| page 42 | 42 | 39 | 42 | 39 | 40,5 | 0,075949 |
| page 43 | 43 | 38 | 43 | 38 | 40,5 | 0,075949 |

| | | | | | | |
|---|---|---|---|---|---|---|
| page 44 | 44 | 37 | 44 | 37 | 40,5 | 0,075949 |
| page 45 | 45 | 36 | 45 | 36 | 40,5 | 0,075949 |
| page 46 | 46 | 35 | 46 | 35 | 40,5 | 0,075949 |
| page 47 | 47 | 34 | 47 | 34 | 40,5 | 0,075949 |
| page 48 | 48 | 33 | 48 | 33 | 40,5 | 0,075949 |
| page 49 | 49 | 32 | 49 | 32 | 40,5 | 0,075949 |
| page 50 | 50 | 31 | 50 | 31 | 40,5 | 0,075949 |
| page 51 | 51 | 30 | 51 | 30 | 40,5 | 0,075949 |
| page 52 | 52 | 29 | 52 | 29 | 40,5 | 0,075949 |
| page 53 | 53 | 28 | 53 | 28 | 40,5 | 0,075949 |
| page 54 | 54 | 27 | 54 | 27 | 40,5 | 0,075949 |
| page 55 | 55 | 26 | 55 | 26 | 40,5 | 0,075949 |
| page 56 | 56 | 25 | 56 | 25 | 40,5 | 0,075949 |
| page 57 | 57 | 24 | 57 | 24 | 40,5 | 0,075949 |
| page 58 | 58 | 23 | 58 | 23 | 40,5 | 0,075949 |
| page 59 | 59 | 22 | 59 | 22 | 40,5 | 0,075949 |
| page 60 | 60 | 21 | 60 | 21 | 40,5 | 0,075949 |
| page 61 | 61 | 20 | 61 | 20 | 40,5 | 0,075949 |
| page 62 | 62 | 19 | 62 | 19 | 40,5 | 0,075949 |
| page 63 | 63 | 18 | 63 | 18 | 40,5 | 0,075949 |
| page 64 | 64 | 17 | 64 | 17 | 40,5 | 0,075949 |
| page 65 | 65 | 16 | 65 | 16 | 40,5 | 0,075949 |
| page 66 | 66 | 15 | 66 | 15 | 40,5 | 0,075949 |
| page 67 | 67 | 14 | 67 | 14 | 40,5 | 0,075949 |
| page 68 | 68 | 13 | 68 | 13 | 40,5 | 0,075949 |
| page 69 | 69 | 12 | 69 | 12 | 40,5 | 0,075949 |
| page 70 | 70 | 11 | 70 | 11 | 40,5 | 0,075949 |
| page 71 | 71 | 10 | 71 | 10 | 40,5 | 0,075949 |
| page 72 | 72 | 9 | 72 | 9 | 40,5 | 0,075949 |
| page 73 | 73 | 8 | 73 | 8 | 40,5 | 0,075949 |
| page 74 | 74 | 7 | 74 | 7 | 40,5 | 0,075949 |
| page 75 | 75 | 6 | 75 | 6 | 40,5 | 0,075949 |
| page 76 | 76 | 5 | 76 | 5 | 40,5 | 0,075949 |
| page 77 | 77 | 4 | 77 | 4 | 40,5 | 0,075949 |
| page 78 | 78 | 3 | 78 | 3 | 40,5 | 0,075949 |
| page 79 | 79 | 2 | 79 | 2 | 40,5 | 0,075949 |
| page 80 | 80 | 1 | 80 | 1 | 40,5 | 0,075949 |

Table 4.1.5 – All the pages get exactly the same average rank

In this case it is not possible to sort uniquely the final result set. Every page has the same exactly final rank. Any sorting is equally accurate and equally relative as all the others possible. We can count in previous searches for this case too, to extract average gravities of every crawler and use them to produce a unique final result set. Again the slightest change in any of the initial result sets will be enough for the algorithm to produce a unique final result set, because the gravities will change. This is shown in Table 4.1.6 (see second and third row).

| | Crawler A | Crawler B | Crawler C | Crawler D | Average Rank | Final Rank |
|---|---|---|---|---|---|---|
| gravity | 0,001875309 | 0,001875309 | 0,001875309 | 0,001875266 | | |
| page 1 | 1 | 79 | 1 | 80 | 40,25 | 0,075480328 |
| page 2 | 2 | 80 | 2 | 79 | 40,75 | 0,076417994 |
| page 3 | 3 | 78 | 3 | 78 | 40,5 | 0,075949178 |
| page 4 | 4 | 77 | 4 | 77 | 40,5 | 0,075949189 |
| page 5 | 5 | 76 | 5 | 76 | 40,5 | 0,0759492 |
| page 6 | 6 | 75 | 6 | 75 | 40,5 | 0,07594921 |
| page 7 | 7 | 74 | 7 | 74 | 40,5 | 0,075949221 |

| | | | | | | |
|---|---|---|---|---|---|---|
| page 8 | 8 | 73 | 8 | 73 | 40,5 | 0,075949232 |
| page 9 | 9 | 72 | 9 | 72 | 40,5 | 0,075949243 |
| page 10 | 10 | 71 | 10 | 71 | 40,5 | 0,075949254 |
| page 11 | 11 | 70 | 11 | 70 | 40,5 | 0,075949265 |
| page 12 | 12 | 69 | 12 | 69 | 40,5 | 0,075949276 |
| page 13 | 13 | 68 | 13 | 68 | 40,5 | 0,075949287 |
| page 14 | 14 | 67 | 14 | 67 | 40,5 | 0,075949298 |
| page 15 | 15 | 66 | 15 | 66 | 40,5 | 0,075949309 |
| page 16 | 16 | 65 | 16 | 65 | 40,5 | 0,07594932 |
| page 17 | 17 | 64 | 17 | 64 | 40,5 | 0,075949331 |
| page 18 | 18 | 63 | 18 | 63 | 40,5 | 0,075949342 |
| page 19 | 19 | 62 | 19 | 62 | 40,5 | 0,075949353 |
| page 20 | 20 | 61 | 20 | 61 | 40,5 | 0,075949364 |
| page 21 | 21 | 60 | 21 | 60 | 40,5 | 0,075949375 |
| page 22 | 22 | 59 | 22 | 59 | 40,5 | 0,075949386 |
| page 23 | 23 | 58 | 23 | 58 | 40,5 | 0,075949397 |
| page 24 | 24 | 57 | 24 | 57 | 40,5 | 0,075949408 |
| page 25 | 25 | 56 | 25 | 56 | 40,5 | 0,075949419 |
| page 26 | 26 | 55 | 26 | 55 | 40,5 | 0,07594943 |
| page 27 | 27 | 54 | 27 | 54 | 40,5 | 0,075949441 |
| page 28 | 28 | 53 | 28 | 53 | 40,5 | 0,075949452 |
| page 29 | 29 | 52 | 29 | 52 | 40,5 | 0,075949463 |
| page 30 | 30 | 51 | 30 | 51 | 40,5 | 0,075949474 |
| page 31 | 31 | 50 | 31 | 50 | 40,5 | 0,075949485 |
| page 32 | 32 | 49 | 32 | 49 | 40,5 | 0,075949496 |
| page 33 | 33 | 48 | 33 | 48 | 40,5 | 0,075949507 |
| page 34 | 34 | 47 | 34 | 47 | 40,5 | 0,075949518 |
| page 35 | 35 | 46 | 35 | 46 | 40,5 | 0,075949529 |
| page 36 | 36 | 45 | 36 | 45 | 40,5 | 0,07594954 |
| page 37 | 37 | 44 | 37 | 44 | 40,5 | 0,075949551 |
| page 38 | 38 | 43 | 38 | 43 | 40,5 | 0,075949562 |
| page 39 | 39 | 42 | 39 | 42 | 40,5 | 0,075949573 |
| page 40 | 40 | 41 | 40 | 41 | 40,5 | 0,075949584 |
| page 41 | 41 | 40 | 41 | 40 | 40,5 | 0,075949595 |
| page 42 | 42 | 39 | 42 | 39 | 40,5 | 0,075949606 |
| page 43 | 43 | 38 | 43 | 38 | 40,5 | 0,075949617 |
| page 44 | 44 | 37 | 44 | 37 | 40,5 | 0,075949628 |
| page 45 | 45 | 36 | 45 | 36 | 40,5 | 0,075949639 |
| page 46 | 46 | 35 | 46 | 35 | 40,5 | 0,07594965 |
| page 47 | 47 | 34 | 47 | 34 | 40,5 | 0,075949661 |
| page 48 | 48 | 33 | 48 | 33 | 40,5 | 0,075949672 |
| page 49 | 49 | 32 | 49 | 32 | 40,5 | 0,075949683 |
| page 50 | 50 | 31 | 50 | 31 | 40,5 | 0,075949694 |
| page 51 | 51 | 30 | 51 | 30 | 40,5 | 0,075949705 |
| page 52 | 52 | 29 | 52 | 29 | 40,5 | 0,075949716 |
| page 53 | 53 | 28 | 53 | 28 | 40,5 | 0,075949727 |
| page 54 | 54 | 27 | 54 | 27 | 40,5 | 0,075949738 |
| page 55 | 55 | 26 | 55 | 26 | 40,5 | 0,075949749 |
| page 56 | 56 | 25 | 56 | 25 | 40,5 | 0,07594976 |
| page 57 | 57 | 24 | 57 | 24 | 40,5 | 0,075949771 |
| page 58 | 58 | 23 | 58 | 23 | 40,5 | 0,075949782 |
| page 59 | 59 | 22 | 59 | 22 | 40,5 | 0,075949793 |
| page 60 | 60 | 21 | 60 | 21 | 40,5 | 0,075949804 |
| page 61 | 61 | 20 | 61 | 20 | 40,5 | 0,075949815 |
| page 62 | 62 | 19 | 62 | 19 | 40,5 | 0,075949826 |
| page 63 | 63 | 18 | 63 | 18 | 40,5 | 0,075949837 |
| page 64 | 64 | 17 | 64 | 17 | 40,5 | 0,075949848 |
| page 65 | 65 | 16 | 65 | 16 | 40,5 | 0,075949859 |

| | | | | | | |
|---|---|---|---|---|---|---|
| page 66 | 66 | 15 | 66 | 15 | 40,5 | 0,07594987 |
| page 67 | 67 | 14 | 67 | 14 | 40,5 | 0,075949881 |
| page 68 | 68 | 13 | 68 | 13 | 40,5 | 0,075949892 |
| page 69 | 69 | 12 | 69 | 12 | 40,5 | 0,075949903 |
| page 70 | 70 | 11 | 70 | 11 | 40,5 | 0,075949914 |
| page 71 | 71 | 10 | 71 | 10 | 40,5 | 0,075949925 |
| page 72 | 72 | 9 | 72 | 9 | 40,5 | 0,075949936 |
| page 73 | 73 | 8 | 73 | 8 | 40,5 | 0,075949947 |
| page 74 | 74 | 7 | 74 | 7 | 40,5 | 0,075949958 |
| page 75 | 75 | 6 | 75 | 6 | 40,5 | 0,075949969 |
| page 76 | 76 | 5 | 76 | 5 | 40,5 | 0,07594998 |
| page 77 | 77 | 4 | 77 | 4 | 40,5 | 0,075949991 |
| page 78 | 78 | 3 | 78 | 3 | 40,5 | 0,075950002 |
| page 79 | 79 | 2 | 79 | 2 | 40,5 | 0,075950013 |
| page 80 | 80 | 1 | 80 | 1 | 40,5 | 0,075950024 |

Table 4.1.6 – Not all the pages get exactly the same average rank

In Table 4.1.6 we can see that none of the 80 pages have mutual final ranks. The final ranks are very similar but different.

To summarize, imaginary initial result sets has brought up three extreme cases that the algorithm fails to produce a unique and proper final result set. These extremes are:

- When the all the crawlers have completely different and not overlapping initial result sets (Table 4.1.2). The problem is that all the crawlers will be assigned the same gravity which leads to a final result set with pages having mutual final rank. The solution is to count on previous searches, to extract average gravities for every crawler used, and use them to produce a unique final result set
- When all the initial result sets of the crawlers used are exactly the same (Table 4.1.3). This results in having a zero standard deviation and to calculate the gravity we divide by zero. The solution is to use any of the mutual initial result sets as the final result set.
- When all the pages get exactly the same average rank (Table 4.1.5). In this case all the crawlers get the same gravity and all the pages get the same final rank. There can be no unique final result set. Any sorting of the final result set is equally accurate and equally relevant. We can count in previous searches for this case too, to extract average gravities of every crawler and use them to produce a unique final result set.

## *4.2 Testing the algorithm in the Word Wide Web*

This is the point where the reranking algorithm is tested with real initial result sets and provides real final result sets. We will evaluate the real final result sets and be ready to compare them with result sets from other popular meta search engines.

We decide to use four of the most popular search engines which are google [70], live search [72], yahoo search [73] and ask.com [74].We call them google, live, yahoo and ask from this point forward. The first 20 pages from every crawler will be our initial result sets. All the reranking process takes place exactly as described in chapter 3.

The first search will use the keywords “web” and “crawlers”. The crawlers will be queried with the string “web crawlers” and provide us the twenty first pages each. The query was made in September 10th 2007. Table 4.2.1 shows all the different pages of the four crawlers. The order is of no importance at this point.

| | |
|---|---|
| page 1 | en.wikipedia.org/wiki/Web_crawler |
| page 2 | www.webcrawler.com/ |
| page 3 | java.sun.com/developer/technicalArticles/ThirdParty/WebCrawler/ |
| page 4 | www.manageability.org/blog/stuff/open-source-web-crawlers-java/view |
| page 5 | www.metacrawler.com/ |
| page 6 | www.google.com/support/webmasters/bin/topic.py?topic=8843 |

| | |
|---|---|
| page 7 | www.informationweek.com/news/showArticle.jhtml?articleID=198001674 |
| page 8 | java-source.net/open-source/crawlers |
| page 9 | dir.yahoo.com/Computers_and_Internet/Internet/World_Wide_Web/Searching_the_Web/Crawlers__Robots__and_Spiders/ |
| page 10 | www.robotstxt.org/wc/robots.html |
| page 11 | www.webreference.com/authoring/robots/ |
| page 12 | about.ask.com/en/docs/about/webmasters.shtml |
| page 13 | www.cse.iitb.ac.in/~soumen/focus/ |
| page 14 | www.aaai.org/AITopics/html/webagent.html |
| page 15 | www.codeproject.com/useritems/Web_Crawler.asp |
| page 16 | dollar.biz.uiowa.edu/~pant/Papers/sigir-01.pdf |
| page 17 | http://portal.acm.org/ft_gateway.cfm?id=1031117&type=pdf&coll=GUIDE&dl=GUIDE&CFID=29132182&CFTOKEN=88710268 |
| page 18 | arc.cs.odu.edu:8080/dp9/ |
| page 19 | www.trnmag.com/Stories/2003/102203/Queries_guide_Web_crawlers_102203.html |
| page 20 | www.informatics.indiana.edu/fil/Papers/TOIT.pdf |
| page 21 | www.insecta.com |
| page 22 | forum.statcounter.com/vb/showthread.php?t=14112 |
| page 23 | forum.statcounter.com/vb/archive/index.php/t-14112.html |
| page 24 | www.geocities.com/Heartland/Meadows/5246/babies.html |
| page 25 | publib.boulder.ibm.com/infocenter/db2luw/v8/topic/com.ibm.db2.ii.of.doc/admin/iiysacweb.htm |
| page 26 | www.techweb.com/encyclopedia/defineterm.jhtml?term=crawler |
| page 27 | es.wikipedia.org/wiki/Web_crawler |
| page 28 | allmyeye.blogspot.com/2007/03/persistent-linking-web-crawlers-and.html |
| page 29 | www.psychologytoday.com/articles/PTO-19990901-000040.html |
| page 30 | http://publib.boulder.ibm.com/infocenter/wsiihelp/v8r3/topic/com.ibm.websphere.ii.esearch.ad.doc/administering/iiysacweb.htm |
| page 31 | www.nla.gov.au/padi/topics/333.html |
| page 32 | www.cs.cmu.edu/~rcm/papers/www7 |
| page 33 | www.cs.ucsd.edu/~dboswell/PastWork/WebCrawlingSurvey.pdf |
| page 34 | linkanalysis.wlv.ac.uk/2.htm |
| page 35 | www.fleiner.com/bots |
| page 36 | www.cio.com/xnet/altavista/tsld075.htm |
| page 37 | crawler.archive.org |
| page 38 | www-2.cs.cmu.edu/~rcm/websphinx |
| page 39 | www.answers.com/topic/web-crawler |
| page 40 | webcrawler.com/select |
| page 41 | faculty.cs.byu.edu/~rodham/cs240/crawler/index.html |
| page 42 | java.sun.com/developer/technicalArticles/WebServices/become |
| page 43 | http://searchwebservices.techtarget.com/sDefinition/0,,sid26_gci211854,00.html |
| page 44 | www.noviway.com/Code/Web-Crawler.aspx |
| page 45 | www.almaden.ibm.com/cs/crawler |
| page 46 | longwood.cs.ucf.edu/~hemant/WebCrawlers.htm |
| page 47 | www.hostsun.com/gr/bots_about.php |
| page 48 | www.lycos.com/info/web-crawler.html |
| page 49 | www.answers.com/topic/web-crawler?method=6 |
| page 50 | www.webpronews.com/expertarticles/2006/04/20/truth-about-web-crawlers |
| page 51 | www.humboldt.edu/~tha1/search.html |
| page 52 | www.robotstxt.org/wc/faq.html |
| page 53 | www.searchenginewatch.com/searchday/01/sd1024-robots.html |
| page 54 | en.wikipedia.org/wiki/Surface_web |
| page 55 | citeseer.ist.psu.edu/mclearn02autonomous.html |
| page 56 | citeseer.ist.psu.edu/fiedler98using.html |
| page 57 | www.techmeme.com/070317/p35 |
| page 58 | news.com.com/8300-10784_3-7-0.html?keyword=Web+attack |
| page 59 | www.ibm.com/developerworks/views/xml/libraryview.jsp? |

| | |
|---|---|
| page 60 | cis.poly.edu/tr/tr-cis-2001-03.htm |
| page 61 | info.webcrawler.com/mak/projects/robots/robots.html |
| page 62 | www.cs.cmu.edu/~rcm/websphinx/ |

Table 4.2.1 – Correlation between page numbers and URLs on “web crawlers” query

Table 4.2.2 in combination with the correlation made in Table 4.2.1 shows as the calculations of the reranking process.

| | google | live | yahoo | ask | Average Rank | Final Rank |
|---|---|---|---|---|---|---|
| gravity | 0,04824 | 0,035668 | 0,039309 | 0,04281 | | |
| page 1 | 1 | 1 | 1 | 7 | 2,5 | 0,105722 |
| page 2 | 2 | 21 | 2 | 2 | 6,75 | 0,252437 |
| page 3 | 3 | 21 | 6 | 21 | 12,75 | 0,507154 |
| page 4 | 4 | 8 | 18 | 8 | 9,5 | 0,382086 |
| page 5 | 5 | 21 | 21 | 21 | 17 | 0,678683 |
| page 6 | 6 | 21 | 21 | 21 | 17,25 | 0,690743 |
| page 7 | 7 | 21 | 21 | 21 | 17,5 | 0,702803 |
| page 8 | 8 | 21 | 21 | 21 | 17,75 | 0,714862 |
| page 9 | 9 | 21 | 21 | 21 | 18 | 0,726922 |
| page 10 | 10 | 21 | 21 | 5 | 14,25 | 0,567741 |
| page 11 | 11 | 15 | 21 | 1 | 12 | 0,483489 |
| page 12 | 12 | 21 | 21 | 21 | 18,75 | 0,763102 |
| page 13 | 13 | 21 | 21 | 21 | 19 | 0,775162 |
| page 14 | 14 | 20 | 20 | 21 | 18,75 | 0,768478 |
| page 15 | 15 | 21 | 21 | 21 | 19,5 | 0,799282 |
| page 16 | 16 | 21 | 21 | 21 | 19,75 | 0,811342 |
| page 17 | 17 | 21 | 21 | 21 | 20 | 0,823402 |
| page 18 | 18 | 21 | 21 | 21 | 20,25 | 0,835462 |
| page 19 | 19 | 21 | 21 | 21 | 20,5 | 0,847521 |
| page 20 | 20 | 21 | 21 | 21 | 20,75 | 0,859581 |
| page 21 | 21 | 2 | 21 | 21 | 16,25 | 0,702218 |
| page 22 | 21 | 3 | 21 | 21 | 16,5 | 0,711135 |
| page 23 | 21 | 4 | 21 | 21 | 16,75 | 0,720052 |
| page 24 | 21 | 5 | 21 | 11 | 14,5 | 0,621943 |
| page 25 | 21 | 6 | 21 | 21 | 17,25 | 0,737886 |
| page 26 | 21 | 7 | 21 | 21 | 17,5 | 0,746803 |
| page 27 | 21 | 9 | 3 | 21 | 13,5 | 0,587747 |
| page 28 | 21 | 10 | 21 | 21 | 18,25 | 0,773554 |
| page 29 | 21 | 11 | 21 | 21 | 18,5 | 0,782471 |
| page 30 | 21 | 12 | 21 | 21 | 18,75 | 0,791388 |
| page 31 | 21 | 13 | 21 | 21 | 19 | 0,800305 |
| page 32 | 21 | 14 | 21 | 21 | 19,25 | 0,809222 |
| page 33 | 21 | 16 | 21 | 21 | 19,75 | 0,827056 |
| page 34 | 21 | 17 | 21 | 21 | 20 | 0,835973 |
| page 35 | 21 | 18 | 21 | 9 | 17,25 | 0,716459 |
| page 36 | 21 | 19 | 21 | 21 | 20,5 | 0,853807 |
| page 37 | 21 | 21 | 4 | 21 | 16,75 | 0,704578 |
| page 38 | 21 | 21 | 5 | 21 | 17 | 0,714406 |
| page 39 | 21 | 21 | 7 | 21 | 17,5 | 0,73406 |
| page 40 | 21 | 21 | 8 | 21 | 17,75 | 0,743887 |
| page 41 | 21 | 21 | 9 | 21 | 18 | 0,753715 |
| page 42 | 21 | 21 | 10 | 21 | 18,25 | 0,763542 |
| page 43 | 21 | 21 | 11 | 21 | 18,5 | 0,773369 |
| page 44 | 21 | 21 | 12 | 21 | 18,75 | 0,783196 |
| page 45 | 21 | 21 | 13 | 21 | 19 | 0,793023 |
| page 46 | 21 | 21 | 14 | 21 | 19,25 | 0,802851 |

| | | | | | | |
|---|---|---|---|---|---|---|
| page 47 | 21 | 21 | 15 | 21 | 19,5 | 0,812678 |
| page 48 | 21 | 21 | 16 | 21 | 19,75 | 0,822505 |
| page 49 | 21 | 21 | 17 | 21 | 20 | 0,832332 |
| page 50 | 21 | 21 | 19 | 18 | 19,75 | 0,819879 |
| page 51 | 21 | 21 | 21 | 3 | 16,5 | 0,678995 |
| page 52 | 21 | 21 | 21 | 4 | 16,75 | 0,689698 |
| page 53 | 21 | 21 | 21 | 6 | 17,25 | 0,711103 |
| page 54 | 21 | 21 | 21 | 10 | 18,25 | 0,753913 |
| page 55 | 21 | 21 | 21 | 12 | 18,75 | 0,775318 |
| page 56 | 21 | 21 | 21 | 13 | 19 | 0,786021 |
| page 57 | 21 | 21 | 21 | 14 | 19,25 | 0,796723 |
| page 58 | 21 | 21 | 21 | 15 | 19,5 | 0,807426 |
| page 59 | 21 | 21 | 21 | 16 | 19,75 | 0,818128 |
| page 60 | 21 | 21 | 21 | 17 | 20 | 0,828831 |
| page 61 | 21 | 21 | 21 | 19 | 20,5 | 0,850236 |
| page 62 | 21 | 21 | 21 | 20 | 20,75 | 0,860939 |

Table 4.2.2 – Calculations of the reranking process on “web crawlers” query

To extract the final result set, all we have to do is sort the pages according to the final rank number. This is shown in Table 4.2.3. The first column is the Final rank Order (FO), the second column is the Correlation Number (CN) used in Tables 4.2.1 and 4.2.2, the third column is the URL (Universal Resource Locator) of the results-web pages, the fourth column is the average rank and the fifth column is the Final Rank number (FR Number) assigned to each page by the reranking algorithm.

| FO | CN | URL | A.R. | FR Number |
|---|---|---|---|---|
| 1 | page 1 | en.wikipedia.org/wiki/Web_crawler | 2,5 | 0,105722 |
| 2 | page 2 | www.webcrawler.com/ | 6,75 | 0,252437 |
| 3 | page 4 | www.manageability.org/blog/stuff/open-source-web-crawlers-java/view | 9,5 | 0,382086 |
| 4 | page 11 | www.webreference.com/authoring/robots/ | 12 | 0,483489 |
| 5 | page 3 | java.sun.com/developer/technicalArticles/ThirdParty/WebCrawler/ | 12,75 | 0,507154 |
| 6 | page 10 | www.robotstxt.org/wc/robots.html | 14,25 | 0,567741 |
| 7 | page 27 | es.wikipedia.org/wiki/Web_crawler | 13,5 | 0,587747 |
| 8 | page 24 | www.geocities.com/Heartland/Meadows/5246/babies.html | 14,5 | 0,621943 |
| 9 | page 5 | www.metacrawler.com/ | 17 | 0,678683 |
| 10 | page 51 | www.humboldt.edu/~tha1/search.html | 16,5 | 0,678995 |
| 11 | page 52 | www.robotstxt.org/wc/faq.html | 16,75 | 0,689698 |
| 12 | page 6 | www.google.com/support/webmasters/bin/topic.py?topic=8843 | 17,25 | 0,690743 |
| 13 | page 21 | www.insecta.com | 16,25 | 0,702218 |
| 14 | page 7 | www.informationweek.com/news/showArticle.jhtml?articleID=198001674 | 17,5 | 0,702803 |
| 15 | page 37 | crawler.archive.org | 16,75 | 0,704578 |
| 16 | page 53 | www.searchenginewatch.com/searchday/01/sd1024-robots.html | 17,25 | 0,711103 |
| 17 | page 22 | forum.statcounter.com/vb/showthread.php?t=14112 | 16,5 | 0,711135 |
| 18 | page 38 | www-2.cs.cmu.edu/~rcm/websphinx | 17 | 0,714406 |
| 19 | page 8 | java-source.net/open-source/crawlers | 17,75 | 0,714862 |
| 20 | page 35 | www.fleiner.com/bots | 17,25 | 0,716459 |
| 21 | page 23 | forum.statcounter.com/vb/archive/index.php/t-14112.html | 16,75 | 0,720052 |
| 22 | page 9 | dir.yahoo.com/Computers_and_Internet/Internet/World_Wide_Web/Searching_the_Web/Crawlers__Robots__and_Spiders/ | 18 | 0,726922 |
| 23 | page 39 | www.answers.com/topic/web-crawler | 17,5 | 0,73406 |
| 24 | page 25 | publib.boulder.ibm.com/infocenter/db2luw/v8/topic/com.ibm.db2.ii.of.doc/admin/iiysacweb.htm | 17,25 | 0,737886 |
| 25 | page 40 | webcrawler.com/select | 17,75 | 0,743887 |
| 26 | page 26 | www.techweb.com/encyclopedia/defineterm.jhtml?term=crawler | 17,5 | 0,746803 |

| | | | | |
|---|---|---|---|---|
| 27 | page 41 | faculty.cs.byu.edu/~rodham/cs240/crawler/index.html | 18 | 0,753715 |
| 28 | page 54 | en.wikipedia.org/wiki/Surface_web | 18,25 | 0,753913 |
| 29 | page 12 | about.ask.com/en/docs/about/webmasters.shtml | 18,75 | 0,763102 |
| 30 | page 42 | java.sun.com/developer/technicalArticles/WebServices/become | 18,25 | 0,763542 |
| 31 | page 14 | www.aaai.org/AITopics/html/webagent.html | 18,75 | 0,768478 |
| 32 | page 43 | http://searchwebservices.techtarget.com/sDefinition/0,,sid26_gci211854,00.html | 18,5 | 0,773369 |
| 33 | page 28 | allmyeye.blogspot.com/2007/03/persistent-linking-web-crawlers-and.html | 18,25 | 0,773554 |
| 34 | page 13 | www.cse.iitb.ac.in/~soumen/focus/ | 19 | 0,775162 |
| 35 | page 55 | citeseer.ist.psu.edu/mclearn02autonomous.html | 18,75 | 0,775318 |
| 36 | page 29 | www.psychologytoday.com/articles/PTO-19990901-000040.html | 18,5 | 0,782471 |
| 37 | page 44 | www.noviway.com/Code/Web-Crawler.aspx | 18,75 | 0,783196 |
| 38 | page 56 | citeseer.ist.psu.edu/fiedler98using.html | 19 | 0,786021 |
| 39 | page 30 | http://publib.boulder.ibm.com/infocenter/wsiihelp/v8r3/topic/com.ibm.websphere.ii.esearch.ad.doc/administering/iiysacweb.htm | 18,75 | 0,791388 |
| 40 | page 45 | www.almaden.ibm.com/cs/crawler | 19 | 0,793023 |
| 41 | page 57 | www.techmeme.com/070317/p35 | 19,25 | 0,796723 |
| 42 | page 15 | www.codeproject.com/useritems/Web_Crawler.asp | 19,5 | 0,799282 |
| 43 | page 31 | www.nla.gov.au/padi/topics/333.html | 19 | 0,800305 |
| 44 | page 46 | longwood.cs.ucf.edu/~hemant/WebCrawlers.htm | 19,25 | 0,802851 |
| 45 | page 58 | news.com.com/8300-10784_3-7-0.html?keyword=Web+attack | 19,5 | 0,807426 |
| 46 | page 32 | www.cs.cmu.edu/~rcm/papers/www7 | 19,25 | 0,809222 |
| 47 | page 16 | dollar.biz.uiowa.edu/~pant/Papers/sigir-01.pdf | 19,75 | 0,811342 |
| 48 | page 47 | www.hostsun.com/gr/bots_about.php | 19,5 | 0,812678 |
| 49 | page 59 | www.ibm.com/developerworks/views/xml/libraryview.jsp? | 19,75 | 0,818128 |
| 50 | page 50 | www.webpronews.com/expertarticles/2006/04/20/truth-about-web-crawlers | 19,75 | 0,819879 |
| 51 | page 48 | www.lycos.com/info/web-crawler.html | 19,75 | 0,822505 |
| 52 | page 17 | http://portal.acm.org/ft_gateway.cfm?id=1031117&type=pdf&coll=GUIDE&dl=GUIDE&CFID=29132182&CFTOKEN=88710268 | 20 | 0,823402 |
| 53 | page 33 | www.cs.ucsd.edu/~dboswell/PastWork/WebCrawlingSurvey.pdf | 19,75 | 0,827056 |
| 54 | page 60 | cis.poly.edu/tr/tr-cis-2001-03.htm | 20 | 0,828831 |
| 55 | page 49 | www.answers.com/topic/web-crawler?method=6 | 20 | 0,832332 |
| 56 | page 18 | arc.cs.odu.edu:8080/dp9/ | 20,25 | 0,835462 |
| 57 | page 34 | linkanalysis.wlv.ac.uk/2.htm | 20 | 0,835973 |
| 58 | page 19 | www.trnmag.com/Stories/2003/102203/Queries_guide_Web_crawlers_102203.html | 20,5 | 0,847521 |
| 59 | page 61 | info.webcrawler.com/mak/projects/robots/robots.html | 20,5 | 0,850236 |
| 60 | page 36 | www.cio.com/xnet/altavista/tsld075.htm | 20,5 | 0,853807 |
| 61 | page 20 | www.informatics.indiana.edu/fil/Papers/TOIT.pdf | 20,75 | 0,859581 |
| 62 | page 62 | www.cs.cmu.edu/~rcm/websphinx/ | 20,75 | 0,860939 |

Table 4.2.3. – Final ranking results on “web crawlers” query

We see in Table 4.2.3 that the first result, the highest ranked one, is the “en.wikipedia.org/wiki/Web_crawler” web page. In fact this was the highest ranked by google, live, yahoo and the 7th ranked by ask. We would expect it to be the first. The gravities assigned by the algorithm are shown in Table 4.2.2 (second row). The four crawlers agree that google’s ranking must count a little more than others, and ask, yahoo and live ranking (in this order) follows.

Chart 4.2.1 shows us the comparison between the final ranking and the average ranking. We can easily see that these two doesn’t always match. For example the 7th ranked result has a higher average rank than the 6th. As already stated in the previous paragraphs this is the effect of the gravity factors that are used. Also many pages have mutual average ranks but none have mutual final ranks as we can clearly see in Table 4.2.3.

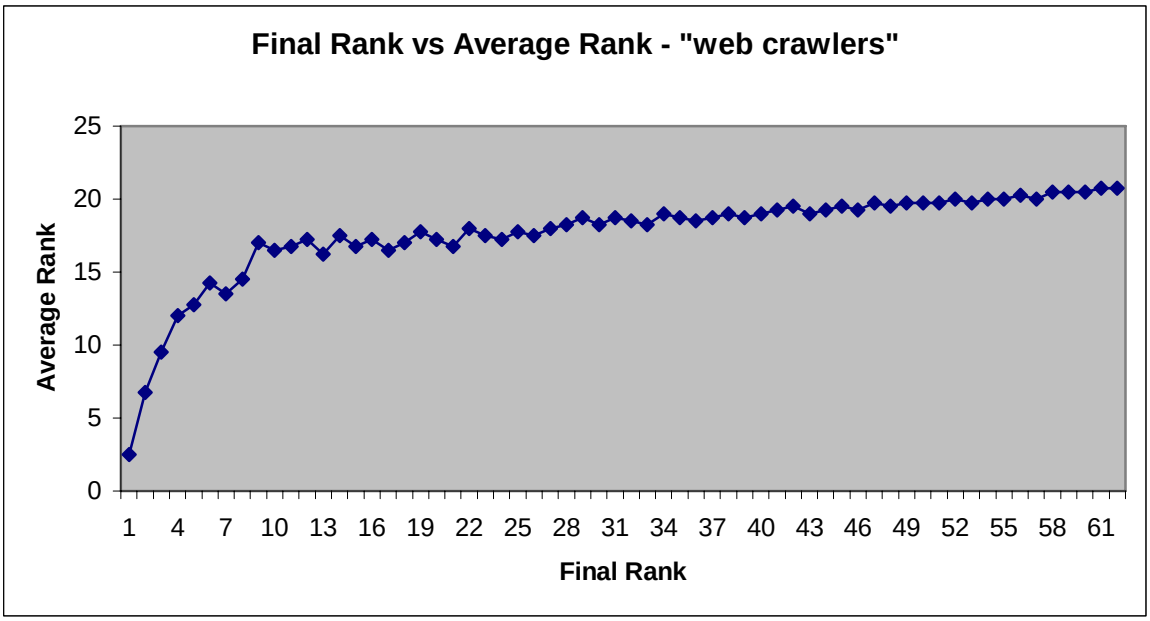


Chart 4.2.1 – Final Ranks versus Average Ranks on “web crawlers” query

We try another search using the same search engine and the keywords “kayak” and “flatwater”. We use the string “kayak flatwater”. Again we use the first 20 pages from each search engine. The query was made in August 22nd 2007. Table 4.2.4 shows us all the different pages.

| | |
|---|---|
| page 1 | www.olympic.org/uk/sports/programme/disciplines_uk.asp?DiscCode=CF |
| page 2 | en.wikipedia.org/wiki/Canoe_racing |
| page 3 | www.ais.org.au/nutrition/documents/FuelCanoe.pdf |
| page 4 | en.beijing2008.cn/cptvenues/sports/canoekayakflatwater/index.shtml |
| page 5 | en.beijing2008.cn/news/sports/headlines/flatwater/n214121834.shtml |
| page 6 | www.canoeicf.com/default.asp?Page=2085 |
| page 7 | pwc.meetup.com/6/?gj=sj5 |
| page 8 | www.anysubject.com/sea-kayak-flatwater-kayak-touring-kayak-sit-on-top-kayak-surf-kayak-canoe.asp |
| page 9 | www.kayakhelp.com/types-of-kayaks/flatwater-kayaks.php |
| page 10 | www.piao.com.cn/en_piao/ticket_1201.html |
| page 11 | http://www.canoeicf.com/site/canoeint/if/downloads/Olympic%20Games/Beijing%202008/Test%20Events%202007%20FWR%20Bulletin108Mar07.pdf |
| page 12 | universalsports.nbcsports.com/articles/show/4810 |
| page 13 | www.cheappremiumtickets.com/Olympics/Canoe-Kayak_Flat_Water_Tickets.cfm |
| page 14 | www.olympic.org/uk/utilities/multimedia/gallery/results_uk.asp?entid=61&MediaType=vid |
| page 15 | corporate.olympics.com.au/index.cfm?p=307 |
| page 16 | www.goodluckbeijing.com.cn/en/accreditation/2007-04-27/4560.html |
| page 17 | www.usboomers.com/kayak.htm |
| page 18 | english.people.com.cn/200408/24/eng20040824_154431.html |
| page 19 | sports.espn.go.com/oly/summer04/medals?discId=11 |
| page 20 | search.ebay.com/kayak-flatwater-touring-touring-kayak-touring-boat_W0QQfsiZ1 |
| page 21 | home.att.net/~paddling/aca_courses/k-fw.html |
| page 22 | www.endlessriveradventures.com/touring.shtm |
| page 23 | www.coreadventures.com/FWkayak.htm |
| page 24 | www.wstickets.com/olympics/games/canoe-kayak-flatwater-racing-tickets.html |
| page 25 | www.olympics.com.au/index.cfm?p=307 |
| page 26 | commissioner.netscape.com/olympics/summer/results/sport/canoekayak-flatwater/0824 |
| page 27 | www.outdoorplay.com/headlines/boat_choose.html |

| | |
|---|---|
| page 28 | www.olympic.org/uk/sports/programme/history_uk.asp?DiscCode=CF&sportCode=CA |
| page 29 | www.canoekayak.gr/worldcup2004/ flatwater/en/default.asp |
| page 30 | www.olympic.org.nz/SportProfile.aspx?print=&id= 3774&SDID=8,6 |
| page 31 | www.southernadventures.com.au/pages/KayakSales.htm |
| page 32 | www.outdoorplay.com |
| page 33 | commissioner.netscape.com/olympics/summer/results/ sport/canoekayak-flatwater/0823 |
| page 34 | en.beijing2008.com/67/90/article212019067.shtml |
| page 35 | www.canoe-europe.org/results/flatwater/CEFlatwater.F200m.Results.pdf |
| page 36 | www.canoe-europe.org/results/flatwater/CEFlatwater.F500m.Results.pdf |
| page 37 | www.naturallysuperior.com/kayak_ courses/flatwater/level_i_ instructor.php |
| page 38 | www.ricka-flatwater.org |
| page 39 | en.wikipedia.org/wiki/Kayak |
| page 40 | www.chinadaily.com.cn/2008/2006-08/21/content_670160.htm |
| page 41 | www.nzoia.org.nz/assets/images/Kayak Flatwater Syllabus.pdf |
| page 42 | www.canoekayak.gr/worldcup2004/flatwater/en/default.asp |
| page 43 | www.kayaksport.net |
| page 44 | www.geneseewaterways.org/courses/kayak/flatwater_race.php |
| page 45 | www.britannica.com/eb/article-9044916/kayak |
| page 46 | http://shop.aldercreek.com/Books-Videos/Flatwater-Books/BCU-Canoe-Kayak-Handbook-C36-i18717.html |
| page 47 | http://shop.aldercreek.com/Books-Videos/Flatwater-Books/Kayak-Puget-Sound--Washburne-C36-i12342.html |
| page 48 | www.ticketcity.com/Summer-Games-Canoe-Kayak-Flatwater-Tickets.html |
| page 49 | http://reviews.ebay.com/Kayak-buying-guide-Touring-and-Recreational-Kayaks_W0QQugidZ10000000001654563?ssPageName=BUYGD:CAT:-1:LISTINGS:4 |
| page 50 | www.usoc.org/11789_35130.htm |
| page 51 | http://www.outdoornewswire.com/v/current/htdocs/etc/sa.php/63617465676f72794e616d653d536561204b6179616b696e672663617465676f72794c6162656c3d5365614b6179616b696e67266c6f636174696f6e3d323030352f30382f313132343732313532342672737349643d31343738 |
| page 52 | www.campearth.org/kayaking.htm |
| page 53 | http://www.athens-olympic-tickets.com/canoe-kayak-flatwater.asp |
| page 54 | http://uk.eurosport.yahoo.com/olympics/rankings/2004/women/313/10181/161908/193307/index.html |
| page 55 | http://uk.eurosport.yahoo.com/olympics/rankings/2004/men/313/10169/index.html |
| page 56 | www.webindia123.com/sports/olymp/2004/canoe.htm |
| page 57 | www.usolympicteam.com/11789_pbimport-25175.htm |
| page 58 | en.beijing2008.com/64/35/article212013564.shtml |
| page 59 | http://www.ibiblio.org/chinesehistory/contents/07spe/specrep02files/specrep02s04f08s01.html |
| page 60 | http://open-site.org/Sports/Olympics/Summer_Games/Canoe_and_Kayak/Flatwater/Medals/Men/ |
| page 61 | http://open-site.org/Sports/Olympics/Summer_Games/Canoe_and_Kayak/Flatwater/Medals/Men/Kayak_1000m_Singles/ |
| page 62 | www.olympic.org/ |
| page 63 | www.lifeinchina.cn/thread-1719-1-2.html |
| page 64 | www.usoc.org/73_35130.htm |
| page 65 | www.ticketsukltd.com/shop/pages/kayakflat.htm |
| page 66 | www.usolympicteam.com/73_21582.htm |
| page 67 | www.orca.on.ca/images/2006Kayak-FI-Accreditation.pdf |
| page 68 | http://www.newsgd.com/specials/athensgames/athensgamesnews/200408240042.htm |

Table 4.2.4 – Correlation between page numbers and URLs on “kayak flatwater” query

Table 4.2.5 in combination with the correlation made in Table 4.2.4 shows as the calculations of the reranking process

| | google | live | yahoo | ask | Average Rank | Final Rank |
|---|---|---|---|---|---|---|
| gravity | 0,048663 | 0,041657 | 0,044217 | 0,028852 | | |
| page 1 | 1 | 3 | 1 | 21 | 6,5 | 0,205934 |
| page 2 | 2 | 1 | 4 | 21 | 7 | 0,230434 |
| page 3 | 3 | 21 | 15 | 3 | 10,5 | 0,442649 |
| page 4 | 4 | 10 | 7 | 21 | 10,5 | 0,381657 |
| page 5 | 5 | 21 | 21 | 21 | 17 | 0,663138 |
| page 6 | 6 | 21 | 21 | 21 | 17,25 | 0,675304 |
| page 7 | 7 | 21 | 21 | 21 | 17,5 | 0,68747 |
| page 8 | 8 | 21 | 21 | 21 | 17,75 | 0,699635 |
| page 9 | 9 | 21 | 21 | 21 | 18 | 0,711801 |
| page 10 | 10 | 21 | 21 | 21 | 18,25 | 0,723967 |
| page 11 | 11 | 21 | 21 | 21 | 18,5 | 0,736132 |
| page 12 | 12 | 21 | 21 | 21 | 18,75 | 0,748298 |
| page 13 | 13 | 21 | 21 | 21 | 19 | 0,760464 |
| page 14 | 14 | 21 | 6 | 21 | 15,5 | 0,606816 |
| page 15 | 15 | 21 | 21 | 21 | 19,5 | 0,784795 |
| page 16 | 16 | 21 | 21 | 21 | 19,75 | 0,796961 |
| page 17 | 17 | 21 | 21 | 21 | 20 | 0,809126 |
| page 18 | 18 | 21 | 21 | 21 | 20,25 | 0,821292 |
| page 19 | 19 | 21 | 21 | 21 | 20,5 | 0,833458 |
| page 20 | 20 | 21 | 21 | 21 | 20,75 | 0,845623 |
| page 21 | 21 | 2 | 21 | 21 | 16,25 | 0,659918 |
| page 22 | 21 | 4 | 21 | 21 | 16,75 | 0,680746 |
| page 23 | 21 | 5 | 21 | 21 | 17 | 0,691161 |
| page 24 | 21 | 6 | 21 | 21 | 17,25 | 0,701575 |
| page 25 | 21 | 7 | 21 | 21 | 17,5 | 0,711989 |
| page 26 | 21 | 8 | 21 | 21 | 17,75 | 0,722403 |
| page 27 | 21 | 9 | 21 | 21 | 18 | 0,732818 |
| page 28 | 21 | 11 | 21 | 21 | 18,5 | 0,753646 |
| page 29 | 21 | 12 | 9 | 21 | 15,75 | 0,63141 |
| page 30 | 21 | 13 | 21 | 21 | 19 | 0,774475 |
| page 31 | 21 | 14 | 21 | 21 | 19,25 | 0,784889 |
| page 32 | 21 | 15 | 18 | 21 | 18,75 | 0,762141 |
| page 33 | 21 | 16 | 21 | 21 | 19,75 | 0,805718 |
| page 34 | 21 | 17 | 21 | 9 | 17 | 0,729577 |
| page 35 | 21 | 18 | 21 | 21 | 20,25 | 0,826546 |
| page 36 | 21 | 19 | 21 | 21 | 20,5 | 0,83696 |
| page 37 | 21 | 20 | 21 | 21 | 20,75 | 0,847375 |
| page 38 | 21 | 21 | 2 | 21 | 16,25 | 0,647759 |
| page 39 | 21 | 21 | 3 | 21 | 16,5 | 0,658813 |
| page 40 | 21 | 21 | 5 | 21 | 17 | 0,680922 |
| page 41 | 21 | 21 | 8 | 21 | 17,75 | 0,714084 |
| page 42 | 21 | 21 | 9 | 21 | 18 | 0,725138 |
| page 43 | 21 | 21 | 10 | 21 | 18,25 | 0,736193 |
| page 44 | 21 | 21 | 11 | 21 | 18,5 | 0,747247 |
| page 45 | 21 | 21 | 12 | 21 | 18,75 | 0,758301 |
| page 46 | 21 | 21 | 13 | 21 | 19 | 0,769355 |
| page 47 | 21 | 21 | 14 | 21 | 19,25 | 0,78041 |
| page 48 | 21 | 21 | 16 | 21 | 19,75 | 0,802518 |
| page 49 | 21 | 21 | 17 | 21 | 20 | 0,813572 |
| page 50 | 21 | 21 | 19 | 8 | 17,25 | 0,741912 |

| | | | | | | |
|---|---|---|---|---|---|---|
| page 51 | 21 | 21 | 20 | 21 | 20,75 | 0,846735 |
| page 52 | 21 | 21 | 21 | 1 | 16 | 0,71353 |
| page 53 | 21 | 21 | 21 | 2 | 16,25 | 0,720743 |
| page 54 | 21 | 21 | 21 | 4 | 16,75 | 0,735169 |
| page 55 | 21 | 21 | 21 | 5 | 17 | 0,742382 |
| page 56 | 21 | 21 | 21 | 6 | 17,25 | 0,749595 |
| page 57 | 21 | 21 | 21 | 7 | 17,5 | 0,756808 |
| page 58 | 21 | 21 | 21 | 10 | 18,25 | 0,778447 |
| page 59 | 21 | 21 | 21 | 11 | 18,5 | 0,78566 |
| page 60 | 21 | 21 | 21 | 12 | 18,75 | 0,792873 |
| page 61 | 21 | 21 | 21 | 13 | 19 | 0,800086 |
| page 62 | 21 | 21 | 21 | 14 | 19,25 | 0,807298 |
| page 63 | 21 | 21 | 21 | 15 | 19,5 | 0,814511 |
| page 64 | 21 | 21 | 21 | 16 | 19,75 | 0,821724 |
| page 65 | 21 | 21 | 21 | 17 | 20 | 0,828937 |
| page 66 | 21 | 21 | 21 | 18 | 20,25 | 0,83615 |
| page 67 | 21 | 21 | 21 | 19 | 20,5 | 0,843363 |
| page 68 | 21 | 21 | 21 | 20 | 20,75 | 0,850576 |

Table 4.2.5 – Calculations of the reranking process on “kayak flatwater” query

As we see in Table 4.2.5, in this search google gets the higher gravity, and yahoo, live and ask follows. To extract the final result set, we sort the pages according to the final rank number. This is shown in Table 4.2.6. The first column is the Final rank Order (FO), the second column is the Correlation Number (CN) used in Tables 4.2.4 and 4.2.5, the third column is the URL (Universal Resource Locator) of the results-web pages, the fourth column is the average rank and the fifth column is the Final Rank number (FR Number) assigned to each page by the reranking algorithm.

| FO | CN | URL | A.R. | FR Number |
|---|---|---|---|---|
| 1 | page 1 | www.olympic.org/uk/sports/programme/disciplines_uk.asp?DiscCode=CF | 6,5 | 0,205934 |
| 2 | page 2 | en.wikipedia.org/wiki/Canoe_racing | 7 | 0,230434 |
| 3 | page 4 | en.beijing2008.cn/cptvenues/sports/canoekayakflatwater/index.shtml | 10,5 | 0,381657 |
| 4 | page 3 | www.ais.org.au/nutrition/documents/FuelCanoe.pdf | 10,5 | 0,442649 |
| 5 | page 14 | www.olympic.org/uk/utilities/multimedia/gallery/results_uk.asp?entid=61&MediaType=vid | 15,5 | 0,606816 |
| 6 | page 29 | www.canoekayak.gr/worldcup2004/ flatwater/en/default.asp | 15,75 | 0,63141 |
| 7 | page 38 | www.ricka-flatwater.org | 16,25 | 0,647759 |
| 8 | page 39 | en.wikipedia.org/wiki/Kayak | 16,5 | 0,658813 |
| 9 | page 21 | home.att.net/~paddling/aca_courses/k-fw.html | 16,25 | 0,659918 |
| 10 | page 5 | en.beijing2008.cn/news/sports/headlines/flatwater/n214121834.shtml | 17 | 0,663138 |
| 11 | page 6 | www.canoeicf.com/default.asp?Page=2085 | 17,25 | 0,675304 |
| 12 | page 22 | www.endlessriveradventures.com/touring.shtm | 16,75 | 0,680746 |
| 13 | page 40 | www.chinadaily.com.cn/2008/2006-08/21/content_670160.htm | 17 | 0,680922 |
| 14 | page 7 | pwc.meetup.com/6/?gj=sj5 | 17,5 | 0,68747 |
| 15 | page 23 | www.coreadventures.com/FWkayak.htm | 17 | 0,691161 |
| 16 | page 8 | www.anysubject.com/sea-kayak-flatwater-kayak-touring-kayak-sit-on-top-kayak-surf-kayak-canoe.asp | 17,75 | 0,699635 |
| 17 | page 24 | www.wstickets.com/olympics/games/canoe-kayak-flatwater-racing-tickets.html | 17,25 | 0,701575 |
| 18 | page 9 | www.kayakhelp.com/types-of-kayaks/flatwater-kayaks.php | 18 | 0,711801 |
| 19 | page 25 | www.olympics.com.au/index.cfm?p=307 | 17,5 | 0,711989 |
| 20 | page 52 | www.campearth.org/kayaking.htm | 16 | 0,71353 |
| 21 | page 41 | www.nzoia.org.nz/assets/images/Kayak Flatwater Syllabus.pdf | 17,75 | 0,714084 |
| 22 | page 53 | http://www.athens-olympic-tickets.com/canoe-kayak-flatwater.asp | 16,25 | 0,720743 |

| | | | | |
|---|---|---|---|---|
| 23 | page 26 | commissioner.netscape.com/olympics/summer/results/sport/canoe kayak-flatwater/0824 | 17,75 | 0,722403 |
| 24 | page 10 | www.piao.com.cn/en_piao/ticket_1201.html | 18,25 | 0,723967 |
| 25 | page 42 | www.canoekayak.gr/worldcup2004/flatwater/en/default.asp | 18 | 0,725138 |
| 26 | page 34 | en.beijing2008.com/67/90/article212019067.shtml | 17 | 0,729577 |
| 27 | page 27 | www.outdoorplay.com/headlines/boat_choose.html | 18 | 0,732818 |
| 28 | page 54 | http://uk.eurosport.yahoo.com/olympics/rankings/2004/women/313 /10181/161908/193307/index.html | 16,75 | 0,735169 |
| 29 | page 11 | http://www.canoeicf.com/site/canoeint/if/downloads/Olympic%20G ames/Beijing%202008/Test%20Events%202007%20FWR%20Bull etin108Mar07.pdf | 18,5 | 0,736132 |
| 30 | page 43 | www.kayaksport.net | 18,25 | 0,736193 |
| 31 | page 50 | www.usoc.org/11789_35130.htm | 17,25 | 0,741912 |
| 32 | page 55 | http://uk.eurosport.yahoo.com/olympics/rankings/2004/men/313/10 169/index.html | 17 | 0,742382 |
| 33 | page 44 | www.geneseewaterways.org/courses/kayak/flatwater_race.php | 18,5 | 0,747247 |
| 34 | page 12 | universalsports.nbcsports.com/articles/show/4810 | 18,75 | 0,748298 |
| 35 | page 56 | www.webindia123.com/sports/olymp/2004/canoe.htm | 17,25 | 0,749595 |
| 36 | page 28 | www.olympic.org/uk/sports/programme/history_uk.asp?DiscCode= CF&sportCode=CA | 18,5 | 0,753646 |
| 37 | page 57 | www.usolympicteam.com/11789_pbimport-25175.htm | 17,5 | 0,756808 |
| 38 | page 45 | www.britannica.com/eb/article-9044916/kayak | 18,75 | 0,758301 |
| 39 | page 13 | www.cheappremiumtickets.com/Olympics/Canoe- Kayak_Flat_Water_Tickets.cfm | 19 | 0,760464 |
| 40 | page 32 | www.outdoorplay.com | 18,75 | 0,762141 |
| 41 | page 46 | http://shop.aldercreek.com/Books-Videos/Flatwater-Books/BCU- Canoe-Kayak-Handbook-C36-i18717.html | 19 | 0,769355 |
| 42 | page 30 | www.olympic.org.nz/SportProfile.aspx?print=&id= 3774&SDID=8,6 | 19 | 0,774475 |
| 43 | page 58 | en.beijing2008.com/64/35/article212013564.shtml | 18,25 | 0,778447 |
| 44 | page 47 | http://shop.aldercreek.com/Books-Videos/Flatwater-Books/Kayak- Puget-Sound--Washburne-C36-i12342.html | 19,25 | 0,78041 |
| 45 | page 15 | corporate.olympics.com.au/index.cfm?p=307 | 19,5 | 0,784795 |
| 46 | page 31 | www.southernadventures.com.au/pages/KayakSales.htm | 19,25 | 0,784889 |
| 47 | page 59 | http://www.ibiblio.org/chinesehistory/contents/07spe/specrep02file s/specrep02s04f08s01.html | 18,5 | 0,78566 |
| 48 | page 60 | http://open- site.org/Sports/Olympics/Summer_Games/Canoe_and_Kayak/Flat water/Medals/Men/ | 18,75 | 0,792873 |
| 49 | page 16 | www.goodluckbeijing.com.cn/en/accreditation/2007-04- 27/4560.html | 19,75 | 0,796961 |
| 50 | page 61 | http://open- site.org/Sports/Olympics/Summer_Games/Canoe_and_Kayak/Flat water/Medals/Men/Kayak_1000m_Singles/ | 19 | 0,800086 |
| 51 | page 48 | www.ticketcity.com/Summer-Games-Canoe-Kayak-Flatwater- Tickets.html | 19,75 | 0,802518 |
| 52 | page 33 | commissioner.netscape.com/olympics/summer/results/ sport/canoekayak-flatwater/0823 | 19,75 | 0,805718 |
| 53 | page 62 | www.olympic.org/ | 19,25 | 0,807298 |
| 54 | page 17 | www.usboomers.com/kayak.htm | 20 | 0,809126 |
| 55 | page 49 | http://reviews.ebay.com/Kayak-buying-guide-Touring-and- Recreational- Kayaks_W0QQugidZ10000000001654563?ssPageName=BUYG D:CAT:-1:LISTINGS:4 | 20 | 0,813572 |
| 56 | page 63 | www.lifeinchina.cn/thread-1719-1-2.html | 19,5 | 0,814511 |
| 57 | page 18 | english.people.com.cn/200408/24/eng20040824_154431.html | 20,25 | 0,821292 |
| 58 | page 64 | www.usoc.org/73_35130.htm | 19,75 | 0,821724 |
| 59 | page 35 | www.canoe- europe.org/results/flatwater/CEFlatwater.F200m.Results.pdf | 20,25 | 0,826546 |

| | | | | |
|---|---|---|---|---|
| 60 | page 65 | www.ticketsukltd.com/shop/pages/kayakflat.htm | 20 | 0,828937 |
| 61 | page 19 | sports.espn.go.com/oly/summer04/medals?discId=11 | 20,5 | 0,833458 |
| 62 | page 66 | www.usolympicteam.com/73_21582.htm | 20,25 | 0,83615 |
| 63 | page 36 | www.canoe-europe.org/results/flatwater/CEFlatwater.F500m.Results.pdf | 20,5 | 0,83696 |
| 64 | page 67 | www.orca.on.ca/images/2006Kayak-FI-Accreditation.pdf | 20,5 | 0,843363 |
| 65 | page 20 | search.ebay.com/kayak-flatwater-touring-touring-kayak-touring-boat_W0QQfsiZ1 | 20,75 | 0,845623 |
| 66 | page 51 | http://www.outdoornewswire.com/v/current/htdocs/etc/sa.php/63617465676f72794e616d653d536561204b6179616b696e672663617465676f72794c6162656c3d5365614b6179616b696e67266c6f636174696f6e3d323030352f30382f313132343732313532342672737349643d31343738 | 20,75 | 0,846735 |
| 67 | page 37 | www.naturallysuperior.com/kayak_ courses/flatwater/level_i_ instructor.php | 20,75 | 0,847375 |
| 68 | page 68 | http://www.newsgd.com/specials/athensgames/athensgamesnews/200408240042.htm | 20,75 | 0,850576 |

Table 4.2.6 – Final ranking results on “kayak flatwater” query

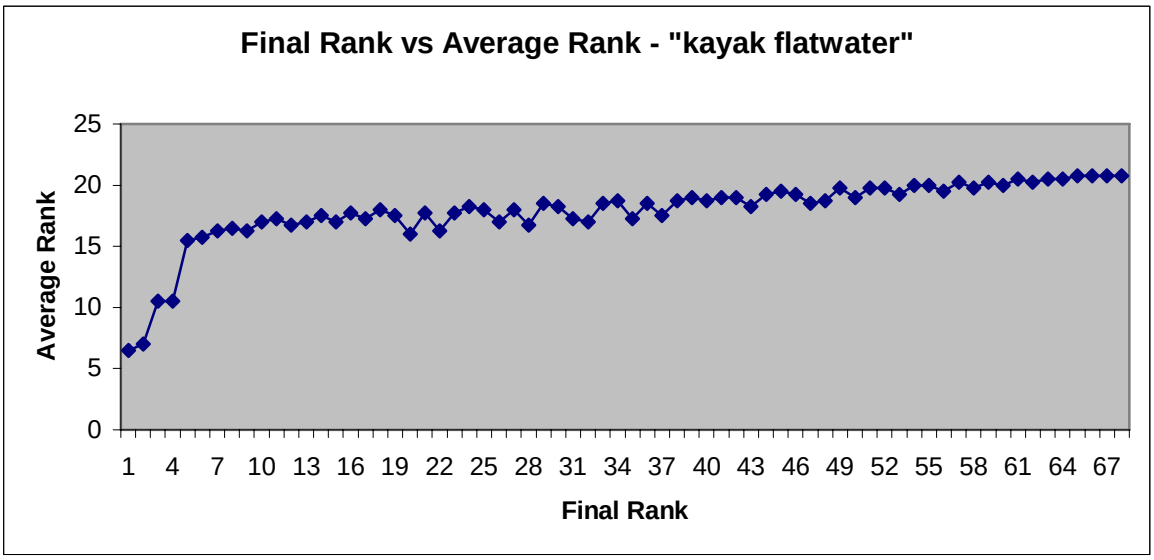


Chart 4.2.2 – Final Ranks versus Average Ranks on “kayak flatwater” query

In Chart 4.2.2 we can see the comparison between the final ranking and the average ranking. Again we see that the final ranks and the average ranks do not match but the final ranks seem to generally follow the average ranks. That means that the gravity factor as well as the average ranking play an important role to the final ranking.

If the gravity factor was too important then Chart 4.2.1 and Chart 4.2.2 would show us that final ranks do not follow average ranks at all. That would be a problem since the average rank is a strong (but not total) indication of the pages relevancy.

If the average ranking was too important then Chart 4.2.1 and Chart 4.2.2 would show as that final ranks match exactly to the average ranks. That would be a problem since the “vote” of every crawler will count the same to the final result set. This is surely wrong because some crawlers are more effective than the others.

Gravity factor makes the reranking algorithm fail safe to the fact that some very ineffective crawlers can be selected to be used. The ineffective crawlers will get a low gravity factor. The effective crawlers will vote each other for a high gravity because their results will variate less. The ineffective crawlers will not vote each other for a high gravity because their results will be completely different. The effective crawlers will also not vote the ineffective crawlers because their results will also greatly differ. In any case the effective crawlers always win and get a higher gravity.

## 4.3 Comparison to other Meta Search Engines

We present another example of the use of the Meta search engine model introduced by this document, and use it to compare the effectiveness of the algorithm used and the relevancy of the results to other popular meta search engines.

We select the keywords meta and search (the string "meta search") to query the crawlers and get the fist 20 pages of their result sets. We use the same 4 crawlers and the same exactly process as in the previous paragraph. The search was made in September 21st 2007.

Table 4.3.1 shows us all the different pages included in the initial result sets. Table 4.3.2 shows us the results of the algorithm. The final result set is sorted by the final rank in Table 4.3.3. In Chart 4.3.1 we can see the graphical representation of the final ranks versus the average ranks of the pages.

| | |
|---|---|
| page 1 | www.metacrawler.com/ |
| page 2 | www.mamma.com/ |
| page 3 | www.dogpile.com/ |
| page 4 | www.lib.berkeley.edu/TeachingLib/Guides/Internet/MetaSearch.html |
| page 5 | www.ixquick.com/ |
| page 6 | en.wikipedia.org/wiki/Metasearch_engine |
| page 7 | http://searchenginewatch.com/showPage.html?page=2156241 |
| page 8 | metasearch.langenberg.com/ |
| page 9 | www.widow.com/ |
| page 10 | www.pandia.com/metasearch/index.html |
| page 11 | www.kartoo.com/ |
| page 12 | kids.ithaki.net/ |
| page 13 | searchenginewatch.com/showPage.html?page=2160791 |
| page 14 | www.one2seek.com/ |
| page 15 | www.searchengineshowdown.com/multi/ |
| page 16 | www.zworks.com/ |
| page 17 | vivisimo.com/ |
| page 18 | www.stpt.com/ |
| page 19 | www.ithaki.net/ |
| page 20 | www.etools.ch/ |
| page 21 | www.metasearch.com |
| page 22 | resources.rootsweb.com/cgi-bin/metasearch |
| page 23 | en.wikipedia.org/wiki/Meta_search |
| page 24 | www.lemmefind.com |
| page 25 | www.faganfinder.com/meta |
| page 26 | www.jobbankusa.com/search.html |
| page 27 | www.learnwebskills.com/search/parallel.html |
| page 28 | www.searchengines.com/generalMeta.html |
| page 29 | www.torrentini.com |
| page 30 | www.russiansabroad.com/MetaSearch |
| page 31 | www.internettutorials.net/meta.html |
| page 32 | www.metagopher.com |
| page 33 | www.doogate.com |
| page 34 | www.lemmefind.com/advanced_search |
| page 35 | www.mectronic.com |
| page 36 | www.searchallinone.com |
| page 37 | www.metaresearch.org/home.asp |
| page 38 | http://searchwebservices.techtarget.com/sDefinition/0,,sid26_gci542231,00.html |
| page 39 | www.search.com |
| page 40 | www.ibm.com/search |
| page 41 | www.metawebsearch.com |
| page 42 | websearch.about.com/od/m/g/meta_search_eng.htm |
| page 43 | meta.wikimedia.org/wiki/Help:Searching |

| | |
|---|---|
| page 44 | meta.openpkg.org/global-search.php |
| page 45 | www.zdnet.com/searchiq |
| page 46 | www.searchenginewatch.com/sereport/01/05-metasearch.html |
| page 47 | www.profusion.com/ |
| page 48 | www.expertsavenue.com/ |
| page 49 | searchenginewatch.com/ |
| page 50 | searchengineland.com/070516-143312.php |
| page 51 | www.altavista.com/ |
| page 52 | www.oneseek.com/ |
| page 53 | www.surfwax.com/ |
| page 54 | www.jux2.com/ |
| page 55 | http://www.theage.com.au/news/technology/giant-leap-for-the-search-unknown/2007/09/10/1189276633425.html |
| page 56 | www.queryserver.com/web.htm |
| page 57 | www.copernic.com/ |

Table 4.3.1 – Correlation between page numbers and URLs on “meta search” query

| | google | live | yahoo | ask | Average Rank | Final Rank |
|---|---|---|---|---|---|---|
| gravity | 0,051859 | 0,032029 | 0,046659 | 0,049517 | | |
| page 1 | 1 | 2 | 21 | 3 | 6,75 | 0,311078 |
| page 2 | 2 | 21 | 4 | 5 | 8 | 0,302638 |
| page 3 | 3 | 21 | 9 | 1 | 8,5 | 0,32441 |
| page 4 | 4 | 16 | 2 | 2 | 6 | 0,228064 |
| page 5 | 5 | 21 | 3 | 11 | 10 | 0,404143 |
| page 6 | 6 | 21 | 21 | 10 | 14,5 | 0,614695 |
| page 7 | 7 | 8 | 21 | 21 | 14,25 | 0,659736 |
| page 8 | 8 | 21 | 21 | 21 | 17,75 | 0,776796 |
| page 9 | 9 | 21 | 21 | 21 | 18 | 0,789761 |
| page 10 | 10 | 21 | 21 | 21 | 18,25 | 0,802726 |
| page 11 | 11 | 21 | 19 | 21 | 18 | 0,792361 |
| page 12 | 12 | 21 | 21 | 21 | 18,75 | 0,828655 |
| page 13 | 13 | 21 | 21 | 21 | 19 | 0,84162 |
| page 14 | 14 | 21 | 21 | 21 | 19,25 | 0,854585 |
| page 15 | 15 | 11 | 21 | 21 | 17 | 0,787477 |
| page 16 | 16 | 21 | 8 | 21 | 16,5 | 0,728873 |
| page 17 | 17 | 7 | 6 | 7 | 9,25 | 0,433097 |
| page 18 | 18 | 21 | 21 | 21 | 20,25 | 0,906445 |
| page 19 | 19 | 21 | 21 | 21 | 20,5 | 0,919409 |
| page 20 | 20 | 21 | 21 | 21 | 20,75 | 0,932374 |
| page 21 | 21 | 1 | 1 | 21 | 11 | 0,551897 |
| page 22 | 21 | 3 | 21 | 21 | 16,5 | 0,801208 |
| page 23 | 21 | 4 | 5 | 21 | 12,75 | 0,622578 |
| page 24 | 21 | 5 | 21 | 21 | 17 | 0,817222 |
| page 25 | 21 | 6 | 21 | 21 | 17,25 | 0,82523 |
| page 26 | 21 | 9 | 21 | 21 | 18 | 0,849251 |
| page 27 | 21 | 10 | 21 | 21 | 18,25 | 0,857259 |
| page 28 | 21 | 12 | 21 | 21 | 18,75 | 0,873273 |
| page 29 | 21 | 13 | 21 | 21 | 19 | 0,881281 |
| page 30 | 21 | 14 | 21 | 21 | 19,25 | 0,889288 |
| page 31 | 21 | 15 | 21 | 21 | 19,5 | 0,897295 |
| page 32 | 21 | 17 | 21 | 8 | 16,75 | 0,75238 |
| page 33 | 21 | 18 | 21 | 21 | 20,25 | 0,921317 |
| page 34 | 21 | 19 | 21 | 21 | 20,5 | 0,929325 |
| page 35 | 21 | 20 | 21 | 21 | 20,75 | 0,937332 |
| page 36 | 21 | 21 | 7 | 21 | 17,5 | 0,782032 |
| page 37 | 21 | 21 | 10 | 21 | 18,25 | 0,817026 |

| | | | | | | |
|---|---|---|---|---|---|---|
| page 38 | 21 | 21 | 11 | 21 | 18,5 | 0,828691 |
| page 39 | 21 | 21 | 12 | 21 | 18,75 | 0,840356 |
| page 40 | 21 | 21 | 13 | 21 | 19 | 0,852021 |
| page 41 | 21 | 21 | 14 | 21 | 19,25 | 0,863686 |
| page 42 | 21 | 21 | 15 | 21 | 19,5 | 0,87535 |
| page 43 | 21 | 21 | 16 | 21 | 19,75 | 0,887015 |
| page 44 | 21 | 21 | 17 | 21 | 20 | 0,89868 |
| page 45 | 21 | 21 | 18 | 21 | 20,25 | 0,910345 |
| page 46 | 21 | 21 | 20 | 15 | 19,25 | 0,859399 |
| page 47 | 21 | 21 | 21 | 4 | 16,75 | 0,734893 |
| page 48 | 21 | 21 | 21 | 6 | 17,25 | 0,759651 |
| page 49 | 21 | 21 | 21 | 9 | 18 | 0,796789 |
| page 50 | 21 | 21 | 21 | 12 | 18,75 | 0,833926 |
| page 51 | 21 | 21 | 21 | 13 | 19 | 0,846306 |
| page 52 | 21 | 21 | 21 | 14 | 19,25 | 0,858685 |
| page 53 | 21 | 21 | 21 | 16 | 19,75 | 0,883443 |
| page 54 | 21 | 21 | 21 | 17 | 20 | 0,895822 |
| page 55 | 21 | 21 | 21 | 18 | 20,25 | 0,908202 |
| page 56 | 21 | 21 | 21 | 19 | 20,5 | 0,920581 |
| page 57 | 21 | 21 | 21 | 20 | 20,75 | 0,93296 |

Table 4.3.2 – Calculations of the reranking process on “meta search” query

| FO | CN | URL | A.R. | FR Number |
|---|---|---|---|---|
| 1 | page 4 | www.lib.berkeley.edu/TeachingLib/Guides/Internet/MetaSearch.html | 6 | 0,228064 |
| 2 | page 2 | www.mamma.com/ | 8 | 0,302638 |
| 3 | page 1 | www.metacrawler.com/ | 6,75 | 0,311078 |
| 4 | page 3 | www.dogpile.com/ | 8,5 | 0,32441 |
| 5 | page 5 | www.ixquick.com/ | 10 | 0,404143 |
| 6 | page 17 | vivisimo.com/ | 9,25 | 0,433097 |
| 7 | page 21 | www.metasearch.com | 11 | 0,551897 |
| 8 | page 6 | en.wikipedia.org/wiki/Metasearch_engine | 14,5 | 0,614695 |
| 9 | page 23 | en.wikipedia.org/wiki/Meta_search | 12,75 | 0,622578 |
| 10 | page 7 | http://searchenginewatch.com/showPage.html?page=2156241 | 14,25 | 0,659736 |
| 11 | page 16 | www.zworks.com/ | 16,5 | 0,728873 |
| 12 | page 47 | www.profusion.com/ | 16,75 | 0,734893 |
| 13 | page 32 | www.metagopher.com | 16,75 | 0,75238 |
| 14 | page 48 | www.expertsavenue.com/ | 17,25 | 0,759651 |
| 15 | page 8 | metasearch.langenberg.com/ | 17,75 | 0,776796 |
| 16 | page 36 | www.searchallinone.com | 17,5 | 0,782032 |
| 17 | page 15 | www.searchengineshowdown.com/multi/ | 17 | 0,787477 |
| 18 | page 9 | www.widow.com/ | 18 | 0,789761 |
| 19 | page 11 | www.kartoo.com/ | 18 | 0,792361 |
| 20 | page 49 | searchenginewatch.com/ | 18 | 0,796789 |
| 21 | page 22 | resources.rootsweb.com/cgi-bin/metasearch | 16,5 | 0,801208 |
| 22 | page 10 | www.pandia.com/metasearch/index.html | 18,25 | 0,802726 |
| 23 | page 37 | www.metaresearch.org/home.asp | 18,25 | 0,817026 |
| 24 | page 24 | www.lemmefind.com | 17 | 0,817222 |
| 25 | page 25 | www.faganfinder.com/meta | 17,25 | 0,82523 |
| 26 | page 12 | kids.ithaki.net/ | 18,75 | 0,828655 |
| 27 | page 38 | http://searchwebservices.techtarget.com/sDefinition/0,,sid26_gci542231,00.html | 18,5 | 0,828691 |
| 28 | page 50 | searchengineland.com/070516-143312.php | 18,75 | 0,833926 |
| 29 | page 39 | www.search.com | 18,75 | 0,840356 |
| 30 | page 13 | searchenginewatch.com/showPage.html?page=2160791 | 19 | 0,84162 |
| 31 | page 51 | www.altavista.com/ | 19 | 0,846306 |
| 32 | page 26 | www.jobbankusa.com/search.html | 18 | 0,849251 |
| 33 | page 40 | www.ibm.com/search | 19 | 0,852021 |

| | | | | |
|---|---|---|---|---|
| 34 | page 14 | www.one2seek.com/ | 19,25 | 0,854585 |
| 35 | page 27 | www.learnwebskills.com/search/parallel.html | 18,25 | 0,857259 |
| 36 | page 52 | www.oneseek.com/ | 19,25 | 0,858685 |
| 37 | page 46 | www.searchenginewatch.com/sereport/01/05-metasearch.html | 19,25 | 0,859399 |
| 38 | page 41 | www.metawebsearch.com | 19,25 | 0,863686 |
| 39 | page 28 | www.searchengines.com/generalMeta.html | 18,75 | 0,873273 |
| 40 | page 42 | websearch.about.com/od/m/g/meta_search_eng.htm | 19,5 | 0,87535 |
| 41 | page 29 | www.torrentini.com | 19 | 0,881281 |
| 42 | page 53 | www.surfwax.com/ | 19,75 | 0,883443 |
| 43 | page 43 | meta.wikimedia.org/wiki/Help:Searching | 19,75 | 0,887015 |
| 44 | page 30 | www.russiansabroad.com/MetaSearch | 19,25 | 0,889288 |
| 45 | page 54 | www.jux2.com/ | 20 | 0,895822 |
| 46 | page 31 | www.internettutorials.net/meta.html | 19,5 | 0,897295 |
| 47 | page 44 | meta.openpkg.org/global-search.php | 20 | 0,89868 |
| 48 | page 18 | www.stpt.com/ | 20,25 | 0,906445 |
| 49 | page 55 | http://www.theage.com.au/news/technology/giant-leap-for-the-search-unknown/2007/09/10/1189276633425.html | 20,25 | 0,908202 |
| 50 | page 45 | www.zdnet.com/searchiq | 20,25 | 0,910345 |
| 51 | page 19 | www.ithaki.net/ | 20,5 | 0,919409 |
| 52 | page 56 | www.queryserver.com/web.htm | 20,5 | 0,920581 |
| 53 | page 33 | www.doogate.com | 20,25 | 0,921317 |
| 54 | page 34 | www.lemmefind.com/advanced_search | 20,5 | 0,929325 |
| 55 | page 20 | www.etools.ch/ | 20,75 | 0,932374 |
| 56 | page 57 | www.copernic.com/ | 20,75 | 0,93296 |
| 57 | page 35 | www.mectronic.com | 20,75 | 0,937332 |

Table 4.3.3 – Final ranking results on “meta search” query

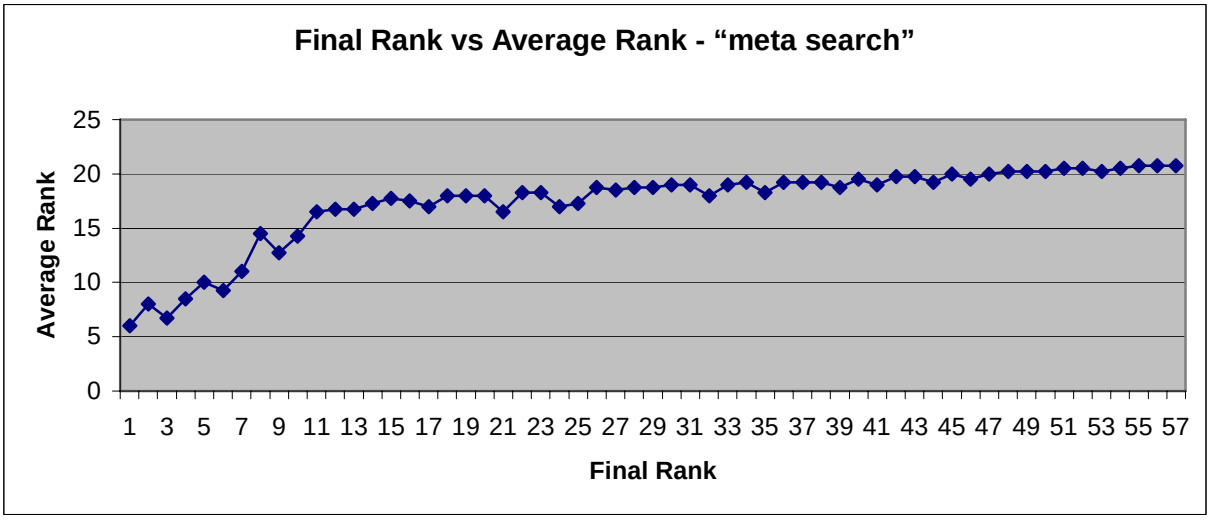


Chart 4.3.1 – Final Ranks versus Average Ranks on “meta search” query

To evaluate our search engine model we must compare the final result set to the result sets of other meta search engines. We select some of the most popular ones. Metacrawler [5, 44, 75], Dogpile [76], Ixquick [77] are some widely used and trusted meta search engines at the time. We use the 20 first pages from each result set and we can see them in Table 4.3.4. The search was made the same date for all the metasearch engines.

| Rank | our model | metacrawler | dogpile | ixquick |
|---|---|---|---|---|
| 1 | www.lib.berkeley.edu/TeachingLib/Guides/Internet/MetaSearch.html | www.metacrawler.com/ | http://www.lib.berkeley.edu/TeachingLib/Guides/Internet/MetaSearch.html | www.lib.berkeley.edu/TeachingLib/Guides/Internet/MetaSearch.html |
| 2 | www.mamma.com/ | http://www.lib.berkeley.edu/TeachingLib/Guides/Internet/MetaSearch.html | www.metacrawler.com/ | www.metacrawler.com |
| 3 | www.metacrawler.com/ | www.mamma.com/ | www.mamma.com/ | www.vivisimo.com |
| 4 | www.dogpile.com/ | www.dogpile.com/ | www.dogpile.com/ | www.metasearch.com |
| 5 | www.ixquick.com/ | www.metasearch.com/ | www.ixquick.com/ | www.mamma.com |
| 6 | vivisimo.com/ | www.ixquick.com/ | vivisimo.com/ | www.dogpile.com |
| 7 | www.metasearch.com | en.wikipedia.org/wiki/Metasearch_engine | http://websearch.about.com/od/enginesanddirectories/a/dogpile.htm | www.searchengineshowdown.com/multi/ |
| 8 | en.wikipedia.org/wiki/Metasearch_engine | en.wikipedia.org/wiki/Meta-search | www.metasearch.com/ | www.expertsavenue.com |
| 9 | en.wikipedia.org/wiki/Meta_search | http://searchenginewatch.com/showPage.html?page=2156241 | http://websearch.about.com/od/enginesanddirectories/a/clusty.htm | en.wikipedia.org/wiki/Meta_search |
| 10 | http://searchenginewatch.com/showPage.html?page=2156241 | www.searchallinone.com/ | en.wikipedia.org/wiki/Metasearch_engine | www.lemmefind.com/about/ |
| 11 | www.zworks.com/ | metasearch.langenberg.com/ | websearch.about.com/od/m/g/meta_search_eng.htm | www.kartoo.com |
| 12 | www.profusion.com/ | resources.rootsweb.com/cgi-bin/metasearch | www.lemmefind.com/ | www.searchenginewatch.com/links/article.php/2156241 |
| 13 | www.metagopher.com | vivisimo.com/ | www.metagopher.com/ | www.3mnetwork.com |
| 14 | www.expertsavenue.com/ | www.widow.com/ | websearch.about.com/b/a/218187.htm | www.haabaa.com/metasearch.php |
| 15 | metasearch.langenberg.com/ | www.profusion.com/ | www.searchengineshowdown.com/multi/ | www.searchalot.com |
| 16 | www.searchallinone.com | www.lemmefind.com/ | websearch.about.com/b/a/218183.htm | www.ixquick.com |
| 17 | www.searchengineshowdown.com/multi/ | www.pandia.com/metasearch/index.html | resources.rootsweb.com/cgi-bin/metasearch | www.search-engines-links.com/metasearch/ |
| 18 | www.widow.com/ | www.metaresearch.org/home.asp | www.profusion.com/ | www.mycgiserver.com/~overload/ |
| 19 | www.kartoo.com/ | websearch.about.com/b/a/218183.htm | en.wikipedia.org/wiki/Meta-search | www.all4one.com/tips.htm |
| 20 | searchenginewatch.com/ | www.lemmefind.com/advanced_search/ | http://websearch.about.com/od/enginesanddirectories/a/dogpile_2.htm | |

Table 4.3.4 – The four result sets of the competing meta search engines

As we can see in Table 4.3.4 all metasearch engines provide very similar result sets. The result sets contain many overlapping pages, which is expectable considering the fact that they came from meta search engines which are using about the same crawlers. We can see that our meta search model is competing with the other metasearch engines but we need a way to evaluate the relevancy of the result sets.

In this document we have already introduced a way to compare the relevancy of different information sources to the same topic. The reranking algorithm uses the standard deviation method to measure the extend of variation of the crawlers ranking to the average ranking of all the crawlers and assigns a gravity factor to each crawler. We can use the same method to evaluate our meta search model, by letting all the meta search engines decide which of them gets the bigger gravity.

We run the algorithm again, but this time the initial result sets are the results sets of the competing meta search engines and we do not care about a final result set. The relevancy of the result sets will be decided by the gravities assigned to each meta search engine by all the meta search engines. Table 4.3.5 shows the ranks of each result set, the average ranks and the gravities.

| | our model | metacrawler | dogpile | ixquick | Average Rank |
|---|---|---|---|---|---|
| gravity | 0,173816 | 0,100289 | 0,067841 | 0,058176 | |
| www.lib.berkeley.edu/TeachingLib/Guides/Internet/MetaSearch.html | 1 | 2 | 1 | 1 | 1,25 |
| www.mamma.com/ | 2 | 3 | 3 | 5 | 3,25 |
| www.metacrawler.com/ | 3 | 1 | 2 | 2 | 2 |
| www.dogpile.com/ | 4 | 4 | 4 | 6 | 4,5 |
| www.ixquick.com/ | 5 | 6 | 5 | 16 | 8 |
| vivisimo.com/ | 6 | 13 | 6 | 3 | 7 |
| www.metasearch.com | 7 | 5 | 8 | 4 | 6 |
| en.wikipedia.org/wiki/Metasearch_engine | 8 | 7 | 10 | 21 | 11,5 |
| en.wikipedia.org/wiki/Meta_search | 9 | 8 | 19 | 9 | 11,25 |
| http://searchenginewatch.com/showPage.html?page=2156241 | 10 | 9 | 21 | 12 | 13 |
| www.zworks.com/ | 11 | 21 | 21 | 21 | 18,5 |
| www.profusion.com/ | 12 | 15 | 18 | 21 | 16,5 |
| www.metagopher.com | 13 | 21 | 13 | 21 | 17 |
| www.expertsavenue.com/ | 14 | 21 | 21 | 8 | 16 |
| metasearch.langenberg.com/ | 15 | 11 | 21 | 21 | 17 |
| www.searchallinone.com | 16 | 10 | 21 | 21 | 17 |
| www.searchengineshowdown.com/multi/ | 17 | 21 | 15 | 7 | 15 |
| www.widow.com/ | 18 | 14 | 21 | 21 | 18,5 |
| www.kartoo.com/ | 19 | 21 | 21 | 11 | 18 |
| searchenginewatch.com/ | 20 | 21 | 21 | 21 | 20,75 |

| | | | | | |
|---|---|---|---|---|---|
| resources.rootsweb.com/cgi-bin/metasearch | 21 | 12 | 17 | 21 | 17,75 |
| www.lemmefind.com/ | 21 | 16 | 12 | 21 | 17,5 |
| www.pandia.com/metasearch/index.html | 21 | 17 | 21 | 21 | 20 |
| www.metaresearch.org/home.asp | 21 | 18 | 21 | 21 | 20,25 |
| websearch.about.com/b/a/218183.htm | 21 | 19 | 16 | 21 | 19,25 |
| www.lemmefind.com/advanced_search/ | 21 | 20 | 21 | 21 | 20,75 |
| http://websearch.about.com/od/enginesanddirectories/a/dogpile.htm | 21 | 21 | 7 | 21 | 17,5 |
| http://websearch.about.com/od/enginesanddirectories/a/clusty.htm | 21 | 21 | 9 | 21 | 18 |
| websearch.about.com/od/m/g/meta_search_eng.htm | 21 | 21 | 11 | 21 | 18,5 |
| websearch.about.com/b/a/218187.htm | 21 | 21 | 14 | 21 | 19,25 |
| http://websearch.about.com/od/enginesanddirectories/a/dogpile_2.htm | 21 | 21 | 20 | 21 | 20,75 |
| www.lemmefind.com/about/ | 21 | 21 | 21 | 10 | 18,25 |
| www.3mnetwork.com | 21 | 21 | 21 | 13 | 19 |
| www.haabaa.com/metasearch.php | 21 | 21 | 21 | 14 | 19,25 |
| www.searchalot.com | 21 | 21 | 21 | 15 | 19,5 |
| www.search-engines-links.com/metasearch/ | 21 | 21 | 21 | 17 | 20 |
| www.mycgiserver.com/~overload/ | 21 | 21 | 21 | 18 | 20,25 |
| www.all4one.com/tips.htm | 21 | 21 | 21 | 19 | 20,5 |
| www.sitepros2000.com/meta.html | 21 | 21 | 21 | 20 | 20,75 |

Table 4.3.5 – Calculation of the gravities of the meta search engines

Table 4.3.5 shows us that our model gets the gravity factor 0,173816, and metacrawler, dogpile and ixquick gets 0,100289, 0,067841 and 0,058176 respectively. The fact that our model gets the highest gravity is a sign of the high relevancy of its results. It is also a sing of the high effectiveness of the algorithm because we used only 20 pages of each initial result set to get the final result set (Table 4.3.3), and still it can compete and prove more effective that the other meta search engines.

In Chart 4.3.2 we can see the page versus rank graphical representation off all the meta search engines and the average ranks. We can compare the ranks of each meta search engine to the average ranks.

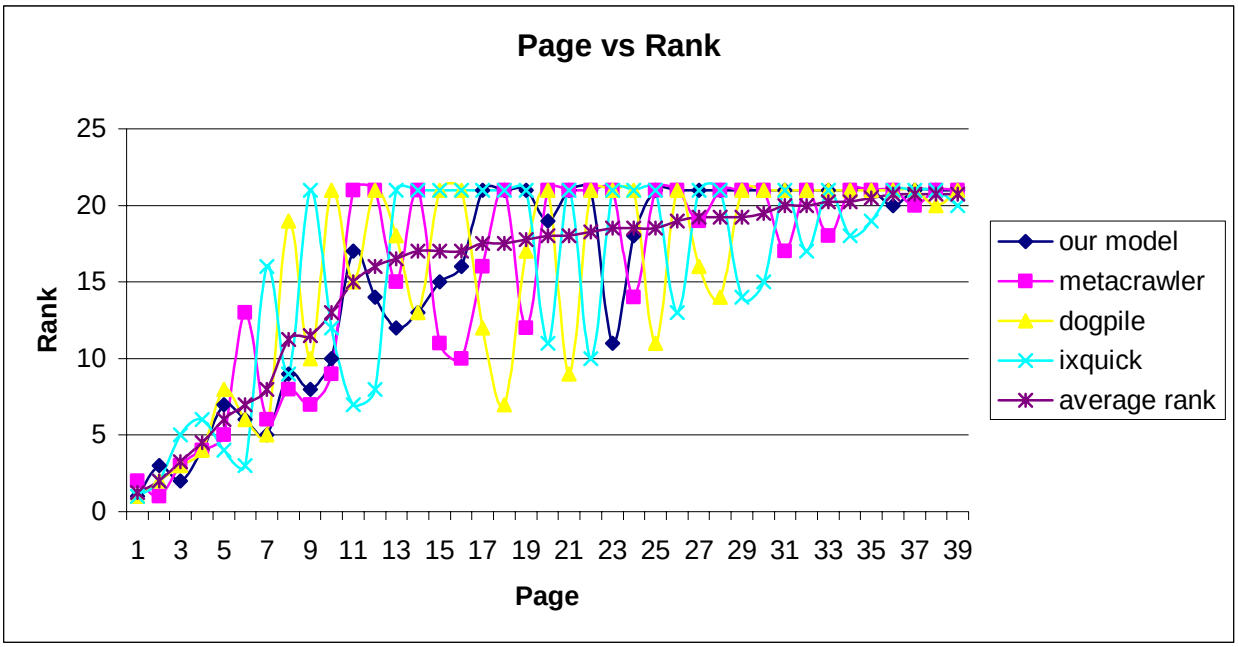


Chart 4.3.2 Page vs rank off all the meta search engines and the average ranks

# Chapter 5 - Conclusions and Future Research

This document is introducing a novel meta search engine model, which is using a method, to give the search engines used the ability to cite each other and assign a gravity factor. The method that is used is the standard deviation method, which can be translated as the extend of variation of the ranks of the pages assigned by a crawler to the average ranks of the pages given by all the crawlers.

In Chapter 3 the meta search reranking algorithm is fully explained and in Chapter 4 some experimental results are shown. There are some extreme cases that the algorithm fails to produce a unique and proper final result set and these are

- When the all the crawlers have completely different and not overlapping initial result sets. The solution is to count on previous searches, to extract average gravities for every crawler used, and use them to produce a unique final result set
- When all the initial result sets of the crawlers used are exactly the same. The solution is to use any of the mutual initial result sets as the final result set.
- When all the pages get exactly the same average rank. In this case all the crawlers get the same gravity and all the pages get the same final rank. There can be no unique final result set. Any sorting of the final result set is equally accurate and equally relevant. We can count in previous searches for this case too, to extract average gravities of every crawler used and use them to produce a unique final result set.

In all of these cases the smallest change in any of the initial result sets will be enough for the meta search algorithm to function as expected.

In Chapter 4 we compare the final result set of the model introduced to the final result set of other popular meta search engines and we use the same citation method to see which of the meta search engine gets the best gravity factor. In our surprise our model gets the best gravity factor despite the fact that in the example our model used only 20 results from four crawlers, while the other meta search engines used their full capabilities and more crawlers. In our attempt to be fair we had to use only the 20 first entries of each final result set to get the gravity factor and this may results in low accuracy in our evaluation.

To be able to fully and fairly evaluate our model, it has to be used in a large scale in the World Wide Web, as the other competing meta search engines do. Comparing millions of results and repeating the evaluation process several times will improve the accuracy of our claim that our model is producing more relevant result sets, or cancel it.

It is a fact though that the first results of the initial results sets are the most important ones. This makes the fact that our model got the better gravity in the evaluation example all the more promising.

Of course there can be many ways that this model can be further improved by future research. One idea is to try to make the first results count more in producing a gravity factor and see if the final result set gets to be even more relevant. Another Idea is to use the average gravity each crawler got from previous searches, and combine it with the current gravity to produce the final gravity for the particular search.

To go even further, we could use a word pool extracted by the contents of all the results and compare the content of each result to that pool. The words in that pool that are more often used can be assigned a bigger gravity. The pages that contain the words with the bigger gravity can be considered more relevant. This measurement can be used in the reranking algorithm in many ways. One way is a second gravity factor, this time for the pages. Another way is to use this measurement to farther tweak the the gravity of the crawlers. We can even re-rerank the final result set.

The pool described in the previous paragraph can be used to sagest corrections to initial keywords too. It is very likely that the words in that pool that get the biggest gravity could be used as keywords for the same search. This way we can get more relevant initial result sets, which lead to more relevant final result set, and more relevant word pool. Thinking recursively, new high gravity words can be added as new keywords, and repeat this circle as many times as necessary.

Any of the candidate improvements above can result in making the meta search process more computationally expensive. The client side implementation is already proposed

and it offers the solution this potential problem. Furthermore, client side implementation helps to make the proposed model adaptive to personalization.

Personalization is certainly a topic of future research to the proposed meta search model. Automatic selection of which search engines are to be used according to the users preferences, extra gravity factors to the search engines that the user wishes to apply or extra gravity factors to the language or the geographical location of the results, are just some of the adaptations to the user needs that can be taken into consideration.

# References

[1] Michael Chau and Hsinchun Chen, “Personalized and Focused Web Spiders”, Tucson, Arizona 85721, USA (2003)

[2] P. Lyman, H.R. Varian, “How Much Information”, [Online]. Available at http://www2.sims.berkeley.edu/research/projects/how-much-info-2003/execsum.htm (2003)

[3] F.C. Cheong, “Internet Agents: Spiders, Wanderers, Brokers, and Bots” (New Riders Publishing, Indianapolis, Indiana, USA, 1996)

[4] H. Chen, H. Fan, M. Chau, D. Zeng, “MetaSpider: Meta-searching and Categorization on the Web”, Journal of the American Society of Information Science & Technology, 52 (13), 1134-1147 (1998)

[5] E. Selberg, O. Etzioni, “Multi-service Search and Comparison using the MetaCrawler”, *Proc. the 4th World-Wide Web Conference* (Boston, MA USA, December 1995)

[6] M. Gray, “Internet Growth and Statistics: Credits and Background”, [Online]. Available at http://www.mit.edu/people/mkgray/net/background.html (1993)

[7] S. Brin, L. Page, “The Anatomy of a Large-scale Hypertextual Web Search Engine”, *Proc. the 7th International World-Wide Web Conference* (Brisbane, Australia, 1998)

[8] S. Chakrabarti, M. Joshi, V. Tawde, “Enhanced Topic Distillation using Text, Markup Tags, and Hyperlinks”, *Proc. the 24th ACM-SIGIR Conference on Research and Development in Information Retrieval* (New Orleans, Louisiana, USA, Sep. 2001)

[9] M. R. Henzinger, “Hyperlink Analysis for the Web”, IEEE Internet Computing, 5 (1), 45-50 (2001).

[10] Amitay, E., “Using Common Hypertext Links to Identify the Best Phrasal Description of Target Web Documents”, *Proc. the 21st ACM-SIGIR Post-Conference Workshop on Hypertext Information Retrieval for the Web* (Melbourne, Australia, 1998)

[11] R. Armstrong, D. Freitag, T. Joachims, T. Mitchell, “WebWatcher: A Learning Apprentice for the World-Wide Web”, *Proc. the AAAI-95 Spring Symposium on Information Gathering from Heterogenous, Distributed Environments* (Stanford, California, USA,1995)

[12] J. Rennie, A.K. McCallum, “Using Reinforcement Learning to Spider the Web Efficiently”, *Proc. the 16th International Conference on Machine Learning (ICML-99)* (Bled, Slovenia, 1999) pp. 335-343

[13] J. Cho, H. Garcia-Molina, L. Page, “Efficient Crawling through URL Ordering”, Proc. the 7th International World-Wide Web Conference (Brisbane, Australia, Apr 1998)

[14] P. Pirolli, J. Pitkow, R. Rao, “Silk from a Sow’s Ear: Extracting Usable Structures from the Web”, *Proc. the ACM Conference on Human Factors in Computing Systems* (Vancouver, Canada, Apr 1996)

[15] E. Spertus, “ParaSite: Mining Structural Information on the Web”, *Proc. the 6th International World-Wide Web Conference* (Santa Clara, California, USA, Apr 1997)

[16] S. Chakrabarti, M. van den Berg, B. Dom, "Focused Crawling: A New Approach to Topic-specific Web Resource Discovery", *Proceedings of the 8th International World-Wide Web Conference* (Toronto, Canada, May 1999)

[17] R.Weiss, B. Velez, M.A. Sheldon, "HyPursuit: A Hierarchical Network Search Engine that Exploits Content-link Hypertext Clustering", *Proceedings of the ACM Conference on Hypertext* (Washington, DC, USA, 1996)

[18] T.H. Haveliwala, "Efficient Computation of PageRank", Stanford University Technical Report. Available at: http://dbpubs.stanford.edu:8090/pub/1999-31 (1999)

[19] J. Kleinberg, "Authoritative Sources in a Hyperlinked Environment", *Proc. the 9th ACMSIAM Symposium on Discrete Algorithms* (San Francisco, California, USA, Jan 1998) pp. 668–677.

[20] S. Chakrabarti, B. Dom, S.R. Kumar, P. Raghavan, S. Rajagopalan, A. Tomkins, D. Gibson, J. Kleinberg, "Mining the Web's Link Structure", IEEE Computer, 32 (8), 60-67 (1999)

[21] K. Bharat, M.R. Henzinger, "Improved Algorithms for Topic Distillation in a Hyperlinked Environment", *Proc. the 21st ACMSIGIR Conference on Research and Development in Information Retrieval, Melbourne* (Australia, 1998)

[22] M. Chau, D. Zeng, H. Chen, "Personalized Spiders forWeb Search and Analysis", *Proc. the 1st ACM-IEEE Joint Conference on Digital Libraries* (Roanoke, Virginia, USA, Jun 2001) pp. 79-87.

[23] A. Heydon, M. Najork, "Mercator: A Scalable, Extensible Web Crawler", World-Wide Web, 219-229 (Dec 1999)

[24] M.L. Mauldin, "Lycos: Design Choices in an Internet Search Service", IEEE Expert, 12 (1) 8-11 (1997)

[25] B. Pinkerton, "Finding What People Want: Experiences with the WebCrawler", *Proc. the 2nd International World-Wide Web Conference* (Chicago, Illinois, USA, 1994)

[26] K. Bharat, A. Broder, "A Technique for Measuring the Relative Size and Overlap of Public Web Search Engines", *Proc. the 7th International World-Wide Web Conference* (Brisbane, Australia, 1998)

[27] M.R. Henzinger, A. Heydon, M. Mitzenmacher, M. Najork, "On Near-uniform URL Sampling", *Proc. the 9th International World-Wide Web Conference* (Amsterdam, Netherlands, May 2000)

[28] A. Heydon, M. Najork, "Performance Limitations of the Java Core Libraries", *Proc. The 1999 ACM Java Grande Conference*, (Jun 1999) pp. 35-41.

[29] M. Burner, "Crawling Towards Eternity: Building an Archive of the World-Wide Web", Web Techniques, 2 (5) (1997)

[30] B. Kahle, "Preserving the Internet. Scientific America", (Mar 1997).

[31] M.L. Mauldin, "Spidering BOF Report", *Report of the Distributed Indexing/Searching Workshop*, (Cambridge, Massachusetts, USA, May 1996)

[32] J. Cho, H. Garcia-Molina, L. Page, “Efficient Crawling through URL Ordering”, Proc. the 7th International World-Wide Web Conference (Brisbane, Australia, Apr 1998)

[33] H. Chen, Y. Chung, M. Ramsey, C.C. Yang, “A Smart Itsy-Bitsy Spider for the Web”, Journal of the American Society for Information Science, Special Issue on AI Techniques for Emerging Information Systems Applications, 49 (7), 604-618 (1998)

[34] S.Waterhouse, D.M. Doolin, G. Kan, Y. Faybishenko, “Distributed Search in P2P Networks”, IEEE Internet Computing, 6 (1) 68-72 (2002)

[35] P. DeBra, R. Post, “Information Retrieval in the World-Wide Web: Making Client-based Searching Feasible”, *Proc. the First International World-Wide Web Conference* (Geneva, Switzerland, 1994)

[36] H. Chen, Y. Chung, M. Ramsey, C.C. Yang, “An Intelligent Personal Spider (Agent) for Dynamic Internet/Intranet Searching”, Decision Support Systems, 23, 41-58 (1998)

[37] Z.Z. Nick, P. Themis, “Web Search Using a Genetic Algorithm”, IEEE Internet Computing, 5 (2) 18-26 (2001)

[38] C.C. Yang, J. Yen, H. Chen, “Intelligent Internet Searching Agent Based on Hybrid Simulated Annealing”, Decision Support Systems, 28, 269-277 (2000)

[39] S. Spetka, “The TkWWW Robot: Beyond Browsing”, *Proc. the 2nd International World-Wide Web Conference* (Chicago, Illinois, USA, 1994)

[40] R.C. Miller, K. Bharat, “SPHINX: A Framework for Creating Personal, Site-specific Web Crawlers”, *Proceedings of the 7th International World-Wide Web Conference* (Brisbane, Australia, Apr 1998)

[41] H. Chen, M. Chau, D. Zeng, “CI Spider: A Tool for Competitive Intelligence on the Web”, Decision Support Systems (2002) in press.

[42] M. Chau, D. Zeng, H. Chen, M. Huang, D. Hendriawan, “Design and Evaluation of a Multi-agent Collaborative Web Mining System”, Decision Support Systems (2002) in press.

[43] M. Diligenti, F. Coetzee, S. Lawrence, C.L. Giles, M. Gori, “Focused Crawling using Context Graphs”, *Proc. the 26th International Conference on Very Large Databases (VLDB 2000)* (Cairo, Egypt, 2000) pp. 527-534

[44] E. Selberg, O. Etzioni, “The MetaCrawler Architecture for Resource Aggregation on the Web”, IEEE Expert, Jan-Feb, 11-14 (1997)

[45] S. Gauch, G. Wang, M. Gomez, “Profusion: Intelligent Fusion from Multiple Different Search Engines”, Journal of Universal Computer Science, 2 (9) (1996)

[46] A.E. Howe, D. Dreilinger, “SavvySearch: A Meta-search Engine that Learns which Search Engines to Query”, AI Magazine, 18 (2) 19-25 (1997)

[47] O. Zamir, O. Etzioni, “Grouper: A Dynamic Clustering Interface to Web Search Results”, *Proc. the 8th World-Wide Web Conference* (Toronto, May 1999)

[48] Y.J. Chen, V.W. Soo, “Ontology-based Information Gathering Agent”, *Proc. the 1st Asia-Pacific Conference on Web Intelligence* (Maebashi City, Japan, Oct 2001) pp. 423-427.

[49] C. Yu, W. Meng, K.L. Liu, “Efficient and Effective Metasearch for Text Databases Incorporating Linkages among Documents”, *Proc. the 2001 ACM SIGMOD International Conference on Management of Data* (Dallas, Texas, May 2001) pp. 187-198

[50] S. Lawrence, C.L. Giles, “Inquirus, the NECI Meta Search Engine”, *Proc. the 7th International World-Wide Web Conference* (Brisbane, Australia, Apr 1998)

[51] S. Lawrence, C.L. Giles, “Context and Page Analysis for Improved Web Search”, IEEE Internet Computing, Jul-Aug, 38-46 (1998).

[52] F. Crimmins, A.F. Smeaton, T. Dkaki, J. Mothe, “TetraFusion: Information Discovery on the Internet”, IEEE Intelligent System, Jul-Aug, 55-62 (1999)

[53] O.A. McBryan, “GENVL and WWWW: Tools for Taming the Web”, *Proc. the 1st International World Wide Web Conference* (Geneva, Switzerland, 1994)

[54] D. Eichmann, “The RBSE Spider Balancing Effective Search AgainstWeb Load”, *Proc. the 1st International World-Wide Web Conference* (Geneva, Switzerland, 1994)

[55] C. Bowman, P. Danzig, U. Manber, M. Schwartz, “Scalable Internet Resource Discovery: Research Problems and Approaches”, Communications of the ACM, 37 (8) 98-107 (1994)

[56] A. McCallum, K. Nigam, J. Rennie, K. Seymore, “A Machine Learning Approach to Building Domain-specific Search Engines”, *Proc. the International Joint Conference on Artificial Intelligence (IJCAI-99)* (1999) pp. 662-667

[57] Michael Gordon, Praveen Pathak, “Finding information on the World Wide Web: the retrieval effectiveness of search engines”, *Information Processing and Management: an International Journal*, Volume 35 , Issue 2 (March 1999), Pergamon Press, Inc. Tarrytown, NY, USA

[58] Weiyi Meng, Clement Yu, King-Lup Liu “Building Efficient and Effective Metasearch Engines”, ACM Computing Surveys, Vol. 34, No. 1, March 2002, pp. 48–89.

[59] Xu, J. and Croft, B, “Cluster-based language models for distributed retrieval”, In Proceedings of the ACM SIGIR Conference (Berkeley, CA, Aug. 1999), 254–261.

[60] J. A. Aslam and M. Montague, “Models for metasearch”, In Proceedings of the 24th annual international ACM SIGIR conference on Research and development in information retrieval, pages 276–284. ACM Press, 2001.

[61] R. Manmatha, T. Rath, and F. Feng, “Modeling score distributions for combining the outputs of search engines”, In Proceedings of the 24th annual international ACM SIGIR conference on Research and development in information retrieval, pages 267–275. ACM Press, 2001.

[62] E. A. Fox and J. A. Shaw, “Combination of multiple searches”, In E. M. Voorhees and D. Harman, editors, Overview of the Second Text REtrieval Conference
(TREC-2), pages 243–249, 1994.

[63] J. H. Lee, “Analyses of multiple evidence combination”, In Proceedings of the 20th annual international ACM SIGIR conference on Research and development in information retrieval, pages 267–276. ACM Press, 1997.

[64] C. Vogt, G. W. Cottrell, R. Belew, and B. Bartell, “Using relevance to train a linear mixture of experts”, In E. M. Voorhees and D. Harman, editors, Overview of the Fifth Text REtrieval Conference (TREC-5), pages 503–515, 1997.

[65] J. Callan, Z. Lu, and W. Croft, “Searching distributed collections with inference networks”, In Proceedings of the 18th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, pages 21 – 28, Seattle, Washington, 1995.

[66] G. Lebanon and J. D. Lafferty, “Cranking: Combining rankings using conditional probability models on permutations”, In Proceedings of the 19th International Conference on Machine Learning, Morgan Kaufmann Publishers, 2002.

[67] W. W. Cohen, R. E. Schapire, and Y. Singer, “Learning to order things”, In NIPS ’97: Proceedings of the 1997 conference on Advances in neural information processing systems 10, pages 451–457. MIT Press, 1998.

[68] Y. Freund and R. E. Schapire, “A decision-theoretic generalization of on-line learning and an application to boosting”, In EuroCOLT ’95: Proceedings of the Second European Conference on Computational Learning Theory, pages 23–37. Springer-Verlag, 1995.

[69] T. Joachims, “Optimizing search engines using clickthrough data”, In Proceedings of the ACM Conference on Knowledge Discovery and Data Mining (KDD). Association of Computer Machinery, 2002.

[70] http://www.google.com

[71] http://en.wikipedia.org/wiki/Standard_deviation

[72] http://www.live.com/

[73] http://search.yahoo.com/

[74] http://www.ask.com/

[75] http://www.metacrawler.com

[76] http://www.dogpile.com/

[77] http://eu.ixquick.com/

# Appendix

## *Open-source crawlers*

**DataparkSearch** is a crawler and search engine released under the GNU General Public License.

**GNU Wget** is a command-line operated crawler written in C and released under the GPL. It is typically used to mirror web and FTP sites.

**Heritrix** is the Internet Archive's archival-quality crawler, designed for archiving periodic snapshots of a large portion of the Web. It was written in Java.

**ht://Dig** includes a Web crawler in its indexing engine.

**HTTrack** uses a Web crawler to create a mirror of a Web site for off-line viewing. It is written in C and released under the GPL.

**Larbin** by Sebastien Ailleret Webtools4larbin by Andreas Beder

**Methabot** is a speed-optimized web crawler and command line utility written in C and released under a 2-clause BSD License. It features a wide configuration system and has support for targeted crawling through local filesystem, HTTP or FTP.

**Nutch** is a crawler written in Java and released under an Apache License. It can be used in conjunction with the Lucene text indexing package.

**WebVac** is a crawler used by the Stanford WebBase Project.

**WebSPHINX** (Miller and Bharat, 1998) is composed of a Java class library that implements multi-threaded Web page retrieval and HTML parsing, and a graphical user interface to set the starting URLs, to extract the downloaded data and to implement a basic text-based search engine
.
**WIRE** (Baeza-Yates and Castillo, 2002) is a web crawler written in C++ and released under the GPL, including several policies for scheduling the page downloads and a module for generating reports and statistics on the downloaded pages so it has been used for Web characterization.

**LWP::RobotUA** (Langheinrich , 2004) is a Perl class for implementing well-behaved parallel web robots distributed under Perl5's license.

**Web Crawler** Open source web crawler.

**Sherlock Holmes** Sherlock Holmes gathers and indexes textual data (text files, web pages, ...), both locally and over the network. Holmes is sponsored and commercially used by the Czech web portal Centrum.

**YaCy** YaCy is a web crawler, indexer, web server with user interface to the application and the search page, and implements a peer-to-peer protocol to communicate with other YaCy

installations. YaCy can be used as stand-alone crawler/indexer or as a distributed search engine. (licensed under GPL)

**Ruya** Ruya is an Open Source, high performance breadth-first, level-based web crawler. It is used to crawl English, Japanese websites in a well-behaved manner. It is released under GPL and was purely developed in Python language. A SingleDomainDelayCrawler implementation obeys robots.txt, meta-robots with a crawl delay during crawl of a target website.